HepMC

2.06.09

Generated by Doxygen 1.8.5

Fri Jun 28 2019 12:15:27

Contents

Chapter 1

Todo List

Member filterEvent (p. **319) (HepMC::GenEvent (**p. **72) ∗ge)**

Have to build a list, since the GV::add_particle_out method modifies the end vertex!

Why does this cause an error?

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 CLHEP Namespace Reference

6.1.1 Detailed Description

CLHEP (p. 15) Vector classes are used in one of the examples

6.2 detail Namespace Reference

6.2.1 Detailed Description

internal namespace

6.3 HepMC Namespace Reference

Namespaces

- **detail**
- **Units**

Classes

- class **Flow**

  The flow object.
- class **GenCrossSection**

  *The **GenCrossSection** (p. 67) class stores the generated cross section.*
- class **GenEvent**

  *The **GenEvent** (p. 72) class is the core of **HepMC** (p. 15).*
- class **GenParticle**

  *The **GenParticle** (p. 106) class contains information about generated particles.*
- class **GenEventVertexRange**

  ***GenEventVertexRange** (p. 104) acts like a collection of vertices.*
- class **ConstGenEventVertexRange**

  ***ConstGenEventVertexRange** (p. 34) acts like a collection of vertices.*
- class **GenEventParticleRange**

GenEventParticleRange *(*p. *103) acts like a collection of particles.*

- class **ConstGenEventParticleRange**

  ConstGenEventParticleRange *(*p. *33) acts like a collection of particles.*

- class **GenVertexParticleRange**

  GenVertexParticleRange *(*p. *144) acts like a collection of particles.*

- class **GenParticleProductionRange**

  GenParticleProductionRange *(*p. *123) acts like a collection of particles.*

- class **ConstGenParticleProductionRange**
- class **GenParticleEndRange**

  GenParticleEndRange *(*p. *122) acts like a collection of particles.*

- class **ConstGenParticleEndRange**
- class **GenVertex**

  GenVertex *(*p. *125) contains information about decay vertices.*

- class **HeavyIon**

  The ***HeavyIon*** *(*p. *146) class stores information about heavy ions.*

- class **HEPEVT_Wrapper**

  Generic Wrapper for the fortran HEPEVT common block.

- class **IO_AsciiParticles**

  event input/output in ascii format for eye and machine reading

- class **IO_BaseClass**

  all input/output classes inherit from ***IO_BaseClass*** *(*p. *180)*

- class **IO_Exception**

  IO exception handling.

- class **IO_GenEvent**

  IO_GenEvent *(*p. *186) also deals with* ***HeavyIon*** *(*p. *146) and* ***PdfInfo*** *(*p. *233).*

- class **IO_HEPEVT**

  HEPEVT IO class.

- class **IO_HERWIG**

  IO_HERWIG *(*p. *199) is used to get Herwig information.*

- class **PdfInfo**

  The ***PdfInfo*** *(*p. *233) class stores PDF information.*

- class **Polarization**

  The ***Polarization*** *(*p. *245) class stores theta and phi for a* ***GenParticle*** *(*p. *106).*

- class **FourVector**

  FourVector *(*p. *54) is a simple representation of a physics 4 vector.*

- class **ThreeVector**

  ThreeVector *(*p. *271) is a simple representation of a position or displacement 3 vector.*

- class **StreamInfo**

  StreamInfo *(*p. *260) contains extra information needed when using streaming IO.*

- class **TempParticleMap**

  TempParticleMap *(*p. *267) is a temporary GenParticle∗ container used during input.*

- class **WeightContainer**

  Container for the Weights associated with an event or vertex.

Enumerations

- enum **IteratorRange** {
  parents, **children**, **family**, **ancestors**,
  descendants, **relatives**, **parents**, **children**,
  family, **ancestors**, **descendants**, **relatives** }

  type of iteration
- enum **known_io** {
  gen =1, **ascii**, **extascii**, **ascii_pdt**,
  extascii_pdt, **gen** =1, **ascii**, **extascii**,
  ascii_pdt, **extascii_pdt** }

  The known_io enum is used to track which type of input is being read.
- enum **IteratorRange** {
  parents, **children**, **family**, **ancestors**,
  descendants, **relatives**, **parents**, **children**,
  family, **ancestors**, **descendants**, **relatives** }

  type of iteration
- enum **known_io** {
  gen =1, **ascii**, **extascii**, **ascii_pdt**,
  extascii_pdt, **gen** =1, **ascii**, **extascii**,
  ascii_pdt, **extascii_pdt** }

  The known_io enum is used to track which type of input is being read.

Functions

- **GenCrossSection getHerwigCrossSection** (int ngen)

  calculate the Herwig cross section and statistical error
- bool **compareGenEvent** (**GenEvent** ∗, **GenEvent** ∗)
- bool **compareSignalProcessVertex** (**GenEvent** ∗, **GenEvent** ∗)
- bool **compareBeamParticles** (**GenEvent** ∗, **GenEvent** ∗)
- bool **compareWeights** (**GenEvent** ∗, **GenEvent** ∗)
- bool **compareVertices** (**GenEvent** ∗, **GenEvent** ∗)
- bool **compareParticles** (**GenEvent** ∗, **GenEvent** ∗)
- bool **compareVertex** (**GenVertex** ∗v1, **GenVertex** ∗v2)
- std::ostream & **operator**<< (std::ostream &os, **GenCrossSection** &xs)
- std::istream & **operator**>> (std::istream &is, **GenCrossSection** &xs)
- template<class InputIterator , class OutputIterator , class Predicate >
  void **copy_if** (InputIterator first, InputIterator last, OutputIterator out, Predicate pred)

  define the type of iterator to use
- std::ostream & **operator**<< (std::ostream &, **GenEvent** &)

  standard streaming IO output operator
- std::istream & **operator**>> (std::istream &, **GenEvent** &)

  standard streaming IO input operator
- std::istream & **set_input_units** (std::istream &, **Units::MomentumUnit**, **Units::LengthUnit**)

  set the units for this input stream
- std::ostream & **write_HepMC_IO_block_begin** (std::ostream &)

  *Explicitly write the begin block lines that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **write_HepMC_IO_block_end** (std::ostream &)

  *Explicitly write the end block line that **IO_GenEvent** (p. 186) uses.*
- **GenEvent** & **convert_units** (**GenEvent** &evt, **Units::MomentumUnit** m, **Units::LengthUnit** l)
- std::ostream & **operator**<< (std::ostream &, **HeavyIon** const ∗)

  *Write the contents of **HeavyIon** (p. 146) to an output stream.*
- std::istream & **operator**>> (std::istream &, **HeavyIon** ∗)

*Read the contents of **HeavyIon** (p. 146) from an input stream.*

- std::ostream & **operator**<< (std::ostream &, **PdfInfo** const *)
- std::istream & **operator**>> (std::istream &, **PdfInfo** *)
- **GenCrossSection getPythiaCrossSection** ()

  calculate the Pythia cross section and statistical error
- bool **not_in_vector** (std::vector< **HepMC::GenParticle** * > *, **GenParticle** *)

  returns true if it cannot find GenParticle in the vector*
- std::vector
  < **HepMC::GenParticle** * >
  ::iterator **already_in_vector** (std::vector< **GenParticle** * > *__v__, **GenParticle** *__p__)

  *returns true if **GenParticle** (p. 106) is in the vector*
- void **version** (std::ostream &os=std::cout)

  *print **HepMC** (p. 15) version*
- void **writeVersion** (std::ostream &os)

  *write **HepMC** (p. 15) version to os*
- std::string **versionName** ()

  *return **HepMC** (p. 15) version*
- std::ostream & **operator**<< (std::ostream &ostr, const **Flow** &f)

  *send **Flow** (p. 44) informatin to ostr for printing*
- void **HepMCStreamCallback** (std::ios_base::event e, std::ios_base &b, int i)
- template<class IO >
  StreamInfo & **get_stream_info** (IO &iost)
- std::ostream & **establish_output_stream_info** (std::ostream &os)
- std::istream & **establish_input_stream_info** (std::istream &is)
- std::ostream & **operator**<< (std::ostream &ostr, const **GenParticle** &part)

  Dump this particle's full info to ostr.
- std::ostream & **operator**<< (std::ostream &ostr, const **GenVertex** &vtx)

  send vertex information to ostr for printing
- std::ostream & **operator**<< (std::ostream &ostr, const **Polarization** &polar)

  write theta and phi to the output stream

Variables

- static const double **HepMC_pi** = 3.14159265358979323846
- static const double **HepMC_pi** = 3.14159265358979323846

6.3.1 Detailed Description

All classes in the **HepMC** (p. 15) packages are in the **HepMC** (p. 15) namespace

6.3.2 Enumeration Type Documentation

6.3.2.1 enum HepMC::IteratorRange

type of iteration

Enumerator

parents

children

family

ancestors
descendants
relatives
parents
children
family
ancestors
descendants
relatives

Definition at line 17 of file IteratorRange.h.

6.3.2.2 enum HepMC::IteratorRange

type of iteration

Enumerator

parents
children
family
ancestors
descendants
relatives
parents
children
family
ancestors
descendants
relatives

Definition at line 17 of file 2.06.09/HepMC/IteratorRange.h.

6.3.2.3 enum HepMC::known_io

The known_io enum is used to track which type of input is being read.

Enumerator

gen
ascii
extascii
ascii_pdt
extascii_pdt
gen
ascii
extascii
ascii_pdt
extascii_pdt

Definition at line 17 of file 2.06.09/HepMC/StreamInfo.h.

6.3.2.4 enum HepMC::known_io

The known_io enum is used to track which type of input is being read.

Enumerator

gen

ascii

extascii

ascii_pdt

extascii_pdt

gen

ascii

extascii

ascii_pdt

extascii_pdt

Definition at line 17 of file StreamInfo.h.

6.3.3 Function Documentation

6.3.3.1 std::vector< HepMC::GenParticle * >::iterator HepMC::already_in_vector (std::vector< HepMC::GenParticle * > * *v*, GenParticle * *p*)

returns true if **GenParticle** (p. 106) is in the vector

Returns the index of a GenParticle* within a vector. Returns -1 if GenParticle* is not in the vector.

Definition at line 18 of file HepMC-2.06.09/src/SearchVector.cc.

References p.

Referenced by not_in_vector(), HepMC::GenVertex::remove_particle_in(), and HepMC::GenVertex::remove_particle_out().

6.3.3.2 bool HepMC::compareBeamParticles (GenEvent * *e1*, GenEvent * *e2*)

Definition at line 77 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenEvent::beam_particles().

Referenced by compareGenEvent().

6.3.3.3 bool HepMC::compareGenEvent (GenEvent * *e1*, GenEvent * *e2*)

Examples:

fio/testHerwigCopies.cc, **fio/testPythiaCopies.cc**, and **testMultipleCopies.cc.in**.

Definition at line 16 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenEvent::alphaQCD(), HepMC::GenEvent::alphaQED(), compareBeamParticles(), compareParticles(), compareSignalProcessVertex(), compareVertices(), compareWeights(), HepMC::GenEvent::event_number(), HepMC::GenEvent::event_scale(), HepMC::GenEvent::heavy_ion(), HepMC::GenEvent::mpi(), HepMC::GenEvent::pdf_info(), HepMC::GenEvent::random_states(), and HepMC::GenEvent::signal_process_id().

Referenced by main().

6.3.3.4 bool HepMC::compareParticles (GenEvent * *e1,* GenEvent * *e2*)

Definition at line 98 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), and HepMC::GenEvent::particles_size().

Referenced by compareGenEvent().

6.3.3.5 bool HepMC::compareSignalProcessVertex (GenEvent * *e1,* GenEvent * *e2*)

Definition at line 64 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenEvent::signal_process_vertex().

Referenced by compareGenEvent().

6.3.3.6 bool HepMC::compareVertex (GenVertex * *v1,* GenVertex * *v2*)

Definition at line 141 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::particles_in_const_begin(), HepMC::GenVertex::particles_in_const_end(), HepMC::GenVertex::particles_in_size(), HepMC::GenVertex::particles_out_const_begin(), HepMC::GenVertex::particles_out_const_end(), HepMC::GenVertex::particles_out_size(), and HepMC::GenVertex::position().

Referenced by compareVertices().

6.3.3.7 bool HepMC::compareVertices (GenEvent * *e1,* GenEvent * *e2*)

Definition at line 120 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenEvent::barcode_to_vertex(), compareVertex(), v, HepMC::GenEvent::vertices_begin(), HepMC::GenEvent::vertices_end(), and HepMC::GenEvent::vertices_size().

Referenced by compareGenEvent().

6.3.3.8 bool HepMC::compareWeights (GenEvent * *e1,* GenEvent * *e2*)

Definition at line 92 of file HepMC-2.06.09/src/CompareGenEvent.cc.

References HepMC::GenEvent::weights().

Referenced by compareGenEvent().

6.3.3.9 GenEvent & HepMC::convert_units (GenEvent & *evt,* Units::MomentumUnit *m,* Units::LengthUnit *l*) [inline]

Definition at line 665 of file GenEvent.h.

References HepMC::GenEvent::use_units().

6.3.3.10 template<class InputIterator , class OutputIterator , class Predicate > void HepMC::copy_if (InputIterator *first,* InputIterator *last,* OutputIterator *out,* Predicate *pred*)

define the type of iterator to use

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 50 of file GenEvent.h.

Referenced by main(), and simpleIter2().

6.3.3.11 std::istream & HepMC::establish_input_stream_info (std::istream & *is*)

Definition at line 667 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), and get_stream_info().

6.3.3.12 std::ostream & HepMC::establish_output_stream_info (std::ostream & *os*)

Definition at line 653 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), and get_stream_info().

6.3.3.13 template<class IO > StreamInfo & HepMC::get_stream_info (IO & *iost*)

A custom iomanip that allows us to store and access user data (**StreamInfo** (p. 260)) associated with the stream. This method creates the **StreamInfo** (p. 260) object the first time it is called.

Definition at line 51 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMCStreamCallback().

Referenced by HepMC::detail::establish_input_stream_info(), establish_input_stream_info(), HepMC::detail::establish_output_stream_info(), establish_output_stream_info(), HepMC::GenEvent::read(), HepMC::detail::read_particle(), set_input_units(), HepMC::GenEvent::write(), write_HepMC_IO_block_begin(), and write_HepMC_IO_block_end().

6.3.3.14 GenCrossSection HepMC::getHerwigCrossSection (int *ngen*)

calculate the Herwig cross section and statistical error

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 24 of file fio/HerwigWrapper.cc.

References hwevnt, and HepMC::GenCrossSection::set_cross_section().

Referenced by main().

6.3.3.15 GenCrossSection HepMC::getPythiaCrossSection ( ) [inline]

calculate the Pythia cross section and statistical error

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 28 of file PythiaWrapper.h.

References pyint5, and HepMC::GenCrossSection::set_cross_section().

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

6.3.3.16 void HepMC::HepMCStreamCallback (std::ios_base::event *e*, std::ios_base & *b*, int *i*)

This method is called by the stream destructor. It does cleanup on stored user data (**StreamInfo** (p. 260)) and is registered by the first call to **get_stream_info()** (p. 22).

Definition at line 29 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::stream_id().

Referenced by get_stream_info().

6.3.3.17 bool HepMC::not_in_vector (std::vector< **HepMC::GenParticle** * > * *v*, GenParticle * *p*)

returns true if it cannot find GenParticle* in the vector

Definition at line 11 of file HepMC-2.06.09/src/SearchVector.cc.

References already_in_vector().

Referenced by HepMC::Flow::connected_partners(), and HepMC::Flow::dangling_connected_partners().

6.3.3.18 std::ostream & HepMC::operator<< (std::ostream & *os*, GenCrossSection & *xs*) [inline]

Definition at line 89 of file GenCrossSection.h.

References HepMC::GenCrossSection::write().

6.3.3.19 std::ostream & HepMC::operator<< (std::ostream & *ostr*, const Polarization & *polar*)

write theta and phi to the output stream

print polarization information

Definition at line 129 of file HepMC-2.06.09/src/Polarization.cc.

References HepMC::Polarization::phi(), and HepMC::Polarization::theta().

6.3.3.20 std::ostream & HepMC::operator<< (std::ostream & *os*, PdfInfo const * *pdf*)

Definition at line 21 of file HepMC-2.06.09/src/PdfInfo.cc.

References HepMC::PdfInfo::id1(), HepMC::PdfInfo::id2(), HepMC::detail::output(), HepMC::PdfInfo::pdf1(), HepMC::PdfInfo::pdf2(), HepMC::PdfInfo::pdf_id1(), HepMC::PdfInfo::pdf_id2(), HepMC::PdfInfo::scalePDF(), HepMC::PdfInfo::x1(), and HepMC::PdfInfo::x2().

6.3.3.21 std::ostream & HepMC::operator<< (std::ostream & *os*, HeavyIon const * *ion*)

Write the contents of **HeavyIon** (p. 146) to an output stream.

Write the contents of **HeavyIon** (p. 146) to an output stream. **GenEvent** (p. 72) stores a pointer to a **HeavyIon** (p. 146).

Definition at line 23 of file HepMC-2.06.09/src/HeavyIon.cc.

References HepMC::HeavyIon::eccentricity(), HepMC::HeavyIon::event_plane_angle(), HepMC::HeavyIon::impact_parameter(), HepMC::HeavyIon::N_Nwounded_collisions(), HepMC::HeavyIon::Ncoll(), HepMC::HeavyIon::Ncoll_hard(), HepMC::HeavyIon::Npart_proj(), HepMC::HeavyIon::Npart_targ(), HepMC::HeavyIon::Nwounded_N_collisions(), HepMC::HeavyIon::Nwounded_Nwounded_collisions(), HepMC::detail::output(), HepMC::HeavyIon::sigma_inel_NN(), HepMC::HeavyIon::spectator_neutrons(), and HepMC::HeavyIon::spectator_protons().

6.3.3.22 std::ostream & HepMC::operator<< (std::ostream & *ostr,* const GenParticle & *part*)

Dump this particle's full info to ostr.

print particle

Definition at line 189 of file HepMC-2.06.09/src/GenParticle.cc.

References HepMC::GenVertex::barcode(), HepMC::GenParticle::barcode(), HepMC::FourVector::e(), HepMC::GenParticle::end_vertex(), HepMC::GenParticle::momentum(), HepMC::GenParticle::pdg_id(), HepMC::FourVector::px(), HepMC::FourVector::py(), HepMC::FourVector::pz(), and HepMC::GenParticle::status().

6.3.3.23 std::ostream & HepMC::operator<< (std::ostream & *ostr,* const Flow & *f*)

send **Flow** (p. 44) informatin to ostr for printing

for printing

Definition at line 190 of file HepMC-2.06.09/src/Flow.cc.

6.3.3.24 std::ostream & HepMC::operator<< (std::ostream & *ostr,* const GenVertex & *vtx*)

send vertex information to ostr for printing

print vertex information

Definition at line 440 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::position(), and HepMC::FourVector::x().

6.3.3.25 std::ostream & HepMC::operator<< (std::ostream & *os,* GenEvent & *evt*)

standard streaming IO output operator

Writes evt to an output stream.

Writes evt to an output stream.

Writes evt to an output stream.

Writes evt to an output stream.

Definition at line 355 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::GenEvent::write().

6.3.3.26 std::istream & HepMC::operator>> (std::istream & *is,* GenCrossSection & *xs*) [inline]

Definition at line 92 of file GenCrossSection.h.

References HepMC::GenCrossSection::read().

6.3.3.27 std::istream & HepMC::operator>> (std::istream & *is,* PdfInfo * *pdf*)

Definition at line 59 of file HepMC-2.06.09/src/PdfInfo.cc.

References HepMC::PdfInfo::set_id1(), HepMC::PdfInfo::set_id2(), HepMC::PdfInfo::set_pdf1(), HepMC::PdfInfo::set_pdf2(), HepMC::PdfInfo::set_pdf_id1(), HepMC::PdfInfo::set_pdf_id2(), HepMC::PdfInfo::set_scalePDF(), HepMC::PdfInfo::set_x1(), HepMC::PdfInfo::set_x2(), and x1.

6.3.3.28 std::istream & HepMC::operator>> (std::istream & *is,* HeavyIon * *ion*)

Read the contents of **HeavyIon** (p. 146) from an input stream.

Read the contents of **HeavyIon** (p. 146) from an input stream. **GenEvent** (p. 72) stores a pointer to a **HeavyIon** (p. 146).

Definition at line 72 of file HepMC-2.06.09/src/HeavyIon.cc.

References HepMC::HeavyIon::set_centrality(), HepMC::HeavyIon::set_eccentricity(), HepMC::HeavyIon::set_event_plane_angle(), HepMC::HeavyIon::set_impact_parameter(), HepMC::HeavyIon::set_N_Nwounded_collisions(), HepMC::HeavyIon::set_Ncoll(), HepMC::HeavyIon::set_Ncoll_hard(), HepMC::HeavyIon::set_Npart_proj(), HepMC::HeavyIon::set_Npart_targ(), HepMC::HeavyIon::set_Nwounded_N_collisions(), HepMC::HeavyIon::set_Nwounded_Nwounded_collisions(), HepMC::HeavyIon::set_sigma_inel_NN(), HepMC::HeavyIon::set_spectator_neutrons(), and HepMC::HeavyIon::set_spectator_protons().

6.3.3.29 std::istream & HepMC::operator>> (std::istream & *is,* GenEvent & *evt*)

standard streaming IO input operator

Definition at line 362 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::GenEvent::read().

6.3.3.30 std::istream & HepMC::set_input_units (std::istream & *is,* Units::MomentumUnit *mom,* Units::LengthUnit *len*)

set the units for this input stream

Examples:

testStreamIO.cc.in.

Definition at line 370 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References get_stream_info(), and HepMC::StreamInfo::use_input_units().

Referenced by HepMC::IO_GenEvent::use_input_units().

6.3.3.31 void HepMC::version (std::ostream & *os* = `std::cout`) [inline]

print **HepMC** (p. 15) version

Examples:

testMass.cc.in.

Definition at line 27 of file Version.h.

References versionName().

Referenced by main().

6.3.3.32 std::string HepMC::versionName ( ) [inline]

return **HepMC** (p. 15) version

Definition at line 22 of file Version.h.

References HEPMC_VERSION.

Referenced by version(), HepMC::IO_AsciiParticles::write_event(), write_HepMC_IO_block_begin(), and writeVersion().

6.3.3.33 std::ostream & HepMC::write_HepMC_IO_block_begin (std::ostream & *os*)

Explicitly write the begin block lines that **IO_GenEvent** (p. 186) uses.

Examples:

fio/example_PythiaStreamIO.cc, and **testStreamIO.cc.in**.

Definition at line 382 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), get_stream_info(), HepMC::StreamInfo::IO_GenEvent_Key(), and versionName().

Referenced by read_from_stream4(), readPythiaStreamIO(), HepMC::IO_GenEvent::write_event(), write_to_stream(), write_to_stream3(), and writePythiaStreamIO().

6.3.3.34 std::ostream & HepMC::write_HepMC_IO_block_end (std::ostream & *os*)

Explicitly write the end block line that **IO_GenEvent** (p. 186) uses.

Examples:

fio/example_PythiaStreamIO.cc, and **testStreamIO.cc.in**.

Definition at line 395 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), get_stream_info(), and HepMC::StreamInfo::IO_GenEvent_End().

Referenced by read_from_stream4(), readPythiaStreamIO(), HepMC::IO_GenEvent::write_comment(), write_to_stream(), write_to_stream3(), writePythiaStreamIO(), and HepMC::IO_GenEvent::~IO_GenEvent().

6.3.3.35 void HepMC::writeVersion (std::ostream & *os*) [inline]

write **HepMC** (p. 15) version to os

Definition at line 33 of file Version.h.

References versionName().

Referenced by HepMC::GenEvent::print_version().

6.3.4 Variable Documentation

6.3.4.1 const double HepMC::HepMC_pi = 3.14159265358979323846 [static]

Definition at line 19 of file Polarization.h.

6.3.4.2 const double HepMC::HepMC_pi = 3.14159265358979323846 [static]

Definition at line 19 of file 2.06.09/HepMC/Polarization.h.

6.4 HepMC::detail Namespace Reference

Classes

- struct **enable_if**

internal - used to decide if a class is arithmetic

- struct **enable_if**< **true, T** >
  internal - use if class T is arithmetic
- struct **disable_if**
  internal - used by SimpleVector to decide if a class is arithmetic
- struct **disable_if**< **false, T** >
  internal - used by SimpleVector to decide if a class is arithmetic
- struct **is_arithmetic**
  undefined and therefore non-arithmetic
- struct **is_arithmetic**< **char** >
  character is arithmetic
- struct **is_arithmetic**< **unsigned char** >
  unsigned character is arithmetic
- struct **is_arithmetic**< **signed char** >
  signed character is arithmetic
- struct **is_arithmetic**< **short** >
  short is arithmetic
- struct **is_arithmetic**< **unsigned short** >
  unsigned short is arithmetic
- struct **is_arithmetic**< **int** >
  int is arithmetic
- struct **is_arithmetic**< **unsigned int** >
  unsigned int is arithmetic
- struct **is_arithmetic**< **long** >
  long is arithmetic
- struct **is_arithmetic**< **unsigned long** >
  unsigned long is arithmetic
- struct **is_arithmetic**< **float** >
  float is arithmetic
- struct **is_arithmetic**< **double** >
  double is arithmetic
- struct **is_arithmetic**< **long double** >
  long double is arithmetic

Functions

- std::ostream & **establish_output_stream_info** (std::ostream &)
  *used by **IO_GenEvent** (p. 186) constructor*
- std::istream & **establish_input_stream_info** (std::istream &)
  *used by **IO_GenEvent** (p. 186) constructor*
- std::istream & **read_vertex** (std::istream &, **TempParticleMap** &, **GenVertex** ∗)
- std::istream & **read_particle** (std::istream &, **TempParticleMap** &, **GenParticle** ∗)
- std::ostream & **output** (std::ostream &os, const double &d)
  write a double - for internal use by streaming IO
- std::ostream & **output** (std::ostream &os, const float &d)
  write a float - for internal use by streaming IO
- std::ostream & **output** (std::ostream &os, const int &i)
  write an int - for internal use by streaming IO
- std::ostream & **output** (std::ostream &os, const long &i)
  write a long - for internal use by streaming IO
- std::ostream & **output** (std::ostream &os, const char &c)
  write a single char - for internal use by streaming IO
- std::istream & **find_event_end** (std::istream &)
  used to read to the end of a bad event

6.4.1 Function Documentation

6.4.1.1 std::istream & HepMC::detail::establish_input_stream_info (std::istream & *is*)

used by **IO_GenEvent** (p. 186) constructor

Definition at line 783 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), and HepMC::get_stream_info().

Referenced by HepMC::IO_GenEvent::IO_GenEvent().

6.4.1.2 std::ostream & HepMC::detail::establish_output_stream_info (std::ostream & *os*)

used by **IO_GenEvent** (p. 186) constructor

Definition at line 769 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::StreamInfo::finished_first_event(), and HepMC::get_stream_info().

Referenced by HepMC::IO_GenEvent::IO_GenEvent().

6.4.1.3 std::istream & HepMC::detail::find_event_end (std::istream & *is*)

used to read to the end of a bad event

Definition at line 98 of file HepMC-2.06.09/src/StreamHelpers.cc.

Referenced by HepMC::GenEvent::read().

6.4.1.4 std::ostream & HepMC::detail::output (std::ostream & *os,* const double & *d*) [inline]

write a double - for internal use by streaming IO

Definition at line 35 of file StreamHelpers.h.

Referenced by HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::operator<<(), HepMC::GenEvent::write(), and HepMC::IO_AsciiParticles::write_event().

6.4.1.5 std::ostream & HepMC::detail::output (std::ostream & *os,* const float & *d*) [inline]

write a float - for internal use by streaming IO

Definition at line 47 of file StreamHelpers.h.

6.4.1.6 std::ostream & HepMC::detail::output (std::ostream & *os,* const int & *i*) [inline]

write an int - for internal use by streaming IO

Definition at line 59 of file StreamHelpers.h.

6.4.1.7 std::ostream & HepMC::detail::output (std::ostream & *os,* const long & *i*) [inline]

write a long - for internal use by streaming IO

Definition at line 71 of file StreamHelpers.h.

6.4.1.8 std::ostream & HepMC::detail::output (std::ostream & *os,* const char & *c*) [inline]

write a single char - for internal use by streaming IO

Definition at line 83 of file StreamHelpers.h.

6.4.1.9 std::istream & HepMC::detail::read_particle (std::istream & *is,* TempParticleMap & *particle_to_end_vertex,* GenParticle * *p*)

get a **GenParticle** (p. 106) from ASCII input **TempParticleMap** (p. 267) is used to track the associations of particles with vertices

Definition at line 688 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::TempParticleMap::addEndParticle(), HepMC::ascii, HepMC::get_stream_info(), HepMC::StreamInfo::io_type(), HepMC::FourVector::m(), HepMC::GenParticle::momentum(), p, HepMC::GenParticle::set_flow(), HepMC::GenParticle::set_generated_mass(), HepMC::Flow::set_icode(), HepMC::GenParticle::set_momentum(), HepMC::GenParticle::set_pdg_id(), HepMC::GenParticle::set_polarization(), HepMC::GenParticle::set_status(), and HepMC::GenParticle::suggest_barcode().

Referenced by read_vertex().

6.4.1.10 std::istream & HepMC::detail::read_vertex (std::istream & *is,* TempParticleMap & *particle_to_end_vertex,* GenVertex * *v*)

get a **GenVertex** (p. 125) from ASCII input **TempParticleMap** (p. 267) is used to track the associations of particles with vertices

Definition at line 23 of file HepMC-2.06.09/src/StreamHelpers.cc.

References HepMC::GenVertex::add_particle_out(), read_particle(), HepMC::GenVertex::set_id(), HepMC::GenVertex::set_position(), HepMC::GenVertex::suggest_barcode(), and HepMC::GenVertex::weights().

Referenced by HepMC::GenEvent::read().

6.5 HepMC::Units Namespace Reference

Enumerations

- enum **MomentumUnit** { **MEV**, **GEV**, **MEV**, **GEV** }
- enum **LengthUnit** { **MM**, **CM**, **MM**, **CM** }
- enum **MomentumUnit** { **MEV**, **GEV**, **MEV**, **GEV** }
- enum **LengthUnit** { **MM**, **CM**, **MM**, **CM** }

Functions

- **LengthUnit default_length_unit** ()

  default unit is defined by configure
- **MomentumUnit default_momentum_unit** ()

  default unit is defined by configure
- std::string **name** (**MomentumUnit**)

  convert enum to string
- std::string **name** (**LengthUnit**)

  convert enum to string
- double **conversion_factor** (**MomentumUnit** from, **MomentumUnit** to)

  scaling factor relative to MeV
- double **conversion_factor** (**LengthUnit** from, **LengthUnit** to)

6.5.1 Enumeration Type Documentation

6.5.1.1 enum HepMC::Units::LengthUnit

Enumerator

MM

CM

MM

CM

Definition at line 26 of file Units.h.

6.5.1.2 enum HepMC::Units::LengthUnit

Enumerator

MM

CM

MM

CM

Definition at line 26 of file 2.06.09/HepMC/Units.h.

6.5.1.3 enum HepMC::Units::MomentumUnit

Enumerator

MEV

GEV

MEV

GEV

Definition at line 25 of file 2.06.09/HepMC/Units.h.

6.5.1.4 enum HepMC::Units::MomentumUnit

Enumerator

MEV

GEV

MEV

GEV

Definition at line 25 of file Units.h.

6.5.2 Function Documentation

6.5.2.1 double HepMC::Units::conversion_factor (MomentumUnit *from,* MomentumUnit *to*)

scaling factor relative to MeV

Examples:

testUnits.cc.

Definition at line 32 of file HepMC-2.06.09/src/Units.cc.

References GEV, and MEV.

Referenced by main(), and repairUnits().

6.5.2.2 double HepMC::Units::conversion_factor (LengthUnit *from,* LengthUnit *to*)

Definition at line 42 of file HepMC-2.06.09/src/Units.cc.

References CM, and MM.

6.5.2.3 LengthUnit HepMC::Units::default_length_unit ( )

default unit is defined by configure

Examples:

testUnits.cc.

Definition at line 53 of file HepMC-2.06.09/src/Units.cc.

References MM.

Referenced by HepMC::GenEvent::clear(), and main().

6.5.2.4 MomentumUnit HepMC::Units::default_momentum_unit ( )

default unit is defined by configure

Examples:

testUnits.cc.

Definition at line 58 of file HepMC-2.06.09/src/Units.cc.

References GEV.

Referenced by HepMC::GenEvent::clear(), and main().

6.5.2.5 std::string HepMC::Units::name (MomentumUnit *m*)

convert enum to string

Examples:

testHepMC.cc.in, **testStreamIO.cc.in**, and **testUnits.cc**.

Definition at line 16 of file HepMC-2.06.09/src/Units.cc.

References GEV, and MEV.

Referenced by main(), HepMC::WeightContainer::push_back(), read_variousFormats(), HepMC::WeightContainer::write(), HepMC::GenEvent::write(), and HepMC::GenEvent::write_units().

6.5.2.6 std::string HepMC::Units::name (LengthUnit /)

convert enum to string

Definition at line 24 of file HepMC-2.06.09/src/Units.cc.

References CM, and MM.

6.6 Units Namespace Reference

6.6.1 Detailed Description

Allow units to be specified within **HepMC** (p. 15). The default units are set at compile time.

Chapter 7

Class Documentation

7.1 HepMC::ConstGenEventParticleRange Class Reference

ConstGenEventParticleRange (p. 33) acts like a collection of particles.

`#include <GenRanges.h>`

Public Member Functions

- **ConstGenEventParticleRange** (**GenEvent** const &e)
  *the constructor requires a const **GenEvent** (p. 72)*
- **GenEvent::particle_const_iterator begin** () const
- **GenEvent::particle_const_iterator end** () const
- **ConstGenEventParticleRange** (**GenEvent** const &e)
  *the constructor requires a const **GenEvent** (p. 72)*
- **GenEvent::particle_const_iterator begin** () const
- **GenEvent::particle_const_iterator end** () const

7.1.1 Detailed Description

ConstGenEventParticleRange (p. 33) acts like a collection of particles.

HepMC::ConstGenEventParticleRange (p. 33) is used to mimic a collection of particles for ease of use - especially with utilities such as the Boost foreach funtion This is the const partner of **GenEventParticleRange** (p. 103)

Definition at line 112 of file GenRanges.h.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 HepMC::ConstGenEventParticleRange::ConstGenEventParticleRange (GenEvent const & *e*) [inline]

the constructor requires a const **GenEvent** (p. 72)

Definition at line 117 of file GenRanges.h.

7.1.2.2 HepMC::ConstGenEventParticleRange::ConstGenEventParticleRange (GenEvent const & *e*) [inline]

the constructor requires a const **GenEvent** (p. 72)

Definition at line 117 of file 2.06.09/HepMC/GenRanges.h.

7.1.3 Member Function Documentation

7.1.3.1 GenEvent::particle_const_iterator HepMC::ConstGenEventParticleRange::begin ( ) const [inline]

Definition at line 119 of file GenRanges.h.

References HepMC::GenEvent::particles_begin().

7.1.3.2 GenEvent::particle_const_iterator HepMC::ConstGenEventParticleRange::begin ( ) const [inline]

Definition at line 119 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::particles_begin().

7.1.3.3 GenEvent::particle_const_iterator HepMC::ConstGenEventParticleRange::end ( ) const [inline]

Definition at line 120 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::particles_end().

7.1.3.4 GenEvent::particle_const_iterator HepMC::ConstGenEventParticleRange::end ( ) const [inline]

Definition at line 120 of file GenRanges.h.

References HepMC::GenEvent::particles_end().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.2 HepMC::ConstGenEventVertexRange Class Reference

ConstGenEventVertexRange (p. 34) acts like a collection of vertices.

`#include <GenRanges.h>`

Public Member Functions

- **ConstGenEventVertexRange** (**GenEvent** const &e)
  *the constructor requires a const **GenEvent*** (p. *72*)
- **GenEvent::vertex_const_iterator begin** () const
- **GenEvent::vertex_const_iterator end** () const
- **ConstGenEventVertexRange** (**GenEvent** const &e)
  *the constructor requires a const **GenEvent*** (p. *72*)
- **GenEvent::vertex_const_iterator begin** () const
- **GenEvent::vertex_const_iterator end** () const

7.2.1 Detailed Description

ConstGenEventVertexRange (p. 34) acts like a collection of vertices.

HepMC::ConstGenEventVertexRange (p. 34) is used to mimic a collection of vertices for ease of use - especially with utilities such as the Boost foreach funtion This is the const partner of **GenEventVertexRange** (p. 104)

Definition at line 55 of file GenRanges.h.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 HepMC::ConstGenEventVertexRange::ConstGenEventVertexRange (GenEvent const & *e*) [inline]

the constructor requires a const **GenEvent** (p. 72)

Definition at line 60 of file GenRanges.h.

7.2.2.2 HepMC::ConstGenEventVertexRange::ConstGenEventVertexRange (GenEvent const & *e*) [inline]

the constructor requires a const **GenEvent** (p. 72)

Definition at line 60 of file 2.06.09/HepMC/GenRanges.h.

7.2.3 Member Function Documentation

7.2.3.1 GenEvent::vertex_const_iterator HepMC::ConstGenEventVertexRange::begin ( ) const [inline]

Definition at line 62 of file GenRanges.h.

References HepMC::GenEvent::vertices_begin().

7.2.3.2 GenEvent::vertex_const_iterator HepMC::ConstGenEventVertexRange::begin ( ) const [inline]

Definition at line 62 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::vertices_begin().

7.2.3.3 GenEvent::vertex_const_iterator HepMC::ConstGenEventVertexRange::end ( ) const [inline]

Definition at line 63 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::vertices_end().

7.2.3.4 GenEvent::vertex_const_iterator HepMC::ConstGenEventVertexRange::end ( ) const [inline]

Definition at line 63 of file GenRanges.h.

References HepMC::GenEvent::vertices_end().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.3 HepMC::ConstGenParticleEndRange Class Reference

`#include <GenRanges.h>`

Public Member Functions

- **ConstGenParticleEndRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)

  *the constructor requires a **GenParticle** (p. 106)*
- **GenVertex::particle_iterator begin** ()

begin iterator throws an error if the particle end_vertex is undefined

- **GenVertex::particle_iterator end** ()

  end iterator throws an error if the particle end_vertex is undefined

- **ConstGenParticleEndRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)

  *the constructor requires a **GenParticle*** (p. *106*)

- **GenVertex::particle_iterator begin** ()

  begin iterator throws an error if the particle end_vertex is undefined

- **GenVertex::particle_iterator end** ()

  end iterator throws an error if the particle end_vertex is undefined

7.3.1 Detailed Description

Definition at line 247 of file GenRanges.h.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 HepMC::ConstGenParticleEndRange::ConstGenParticleEndRange (GenParticle const & *p,* IteratorRange *range =* relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 252 of file GenRanges.h.

7.3.2.2 HepMC::ConstGenParticleEndRange::ConstGenParticleEndRange (GenParticle const & *p,* IteratorRange *range =* relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 252 of file 2.06.09/HepMC/GenRanges.h.

7.3.3 Member Function Documentation

7.3.3.1 GenVertex::particle_iterator HepMC::ConstGenParticleEndRange::begin ( ) [inline]

begin iterator throws an error if the particle end_vertex is undefined

Definition at line 313 of file GenRanges.h.

References HepMC::GenParticle::end_vertex(), and HepMC::GenVertex::particles_begin().

7.3.3.2 GenVertex::particle_iterator HepMC::ConstGenParticleEndRange::begin ( )

begin iterator throws an error if the particle end_vertex is undefined

7.3.3.3 GenVertex::particle_iterator HepMC::ConstGenParticleEndRange::end ( )

end iterator throws an error if the particle end_vertex is undefined

7.3.3.4 GenVertex::particle_iterator HepMC::ConstGenParticleEndRange::end ( ) [inline]

end iterator throws an error if the particle end_vertex is undefined

Definition at line 319 of file GenRanges.h.

References HepMC::GenParticle::end_vertex(), and HepMC::GenVertex::particles_end().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.4 HepMC::ConstGenParticleProductionRange Class Reference

`#include <GenRanges.h>`

Public Member Functions

- **ConstGenParticleProductionRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)
  *the constructor requires a **GenParticle*** (p. *106*)
- **GenVertex::particle_iterator begin** ()
  begin iterator throws an error if the particle production_vertex is undefined
- **GenVertex::particle_iterator end** ()
  end iterator throws an error if the particle production_vertex is undefined
- **ConstGenParticleProductionRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)
  *the constructor requires a **GenParticle*** (p. *106*)
- **GenVertex::particle_iterator begin** ()
  begin iterator throws an error if the particle production_vertex is undefined
- **GenVertex::particle_iterator end** ()
  end iterator throws an error if the particle production_vertex is undefined

7.4.1 Detailed Description

Definition at line 193 of file GenRanges.h.

7.4.2 Constructor & Destructor Documentation

7.4.2.1 HepMC::ConstGenParticleProductionRange::ConstGenParticleProductionRange (GenParticle const & *p*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 198 of file GenRanges.h.

7.4.2.2 HepMC::ConstGenParticleProductionRange::ConstGenParticleProductionRange (GenParticle const & *p*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 198 of file 2.06.09/HepMC/GenRanges.h.

7.4.3 Member Function Documentation

7.4.3.1 GenVertex::particle_iterator HepMC::ConstGenParticleProductionRange::begin ( ) [inline]

begin iterator throws an error if the particle production_vertex is undefined

Definition at line 286 of file GenRanges.h.

References HepMC::GenVertex::particles_begin(), and HepMC::GenParticle::production_vertex().

7.4.3.2 GenVertex::particle_iterator HepMC::ConstGenParticleProductionRange::begin ( )

begin iterator throws an error if the particle production_vertex is undefined

7.4.3.3 GenVertex::particle_iterator HepMC::ConstGenParticleProductionRange::end ( )

end iterator throws an error if the particle production_vertex is undefined

7.4.3.4 GenVertex::particle_iterator HepMC::ConstGenParticleProductionRange::end ( ) [inline]

end iterator throws an error if the particle production_vertex is undefined

Definition at line 293 of file GenRanges.h.

References HepMC::GenVertex::particles_end(), and HepMC::GenParticle::production_vertex().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.5 HepMC::detail::disable_if< bool, class > Struct Template Reference

internal - used by SimpleVector to decide if a class is arithmetic

`#include <enable_if.h>`

7.5.1 Detailed Description

template<bool, class>struct HepMC::detail::disable_if< bool, class >

internal - used by SimpleVector to decide if a class is arithmetic

Definition at line 33 of file enable_if.h.

The documentation for this struct was generated from the following file:

- **enable_if.h**

7.6 HepMC::detail::disable_if< false, T > Struct Template Reference

internal - used by SimpleVector to decide if a class is arithmetic

`#include <enable_if.h>`

Public Types

- typedef T **type**
  check type of class T
- typedef T **type**
  check type of class T

7.6.1 Detailed Description

template<class T>struct HepMC::detail::disable_if< false, T >

internal - used by SimpleVector to decide if a class is arithmetic

Definition at line 38 of file enable_if.h.

7.6.2 Member Typedef Documentation

7.6.2.1 template<class T > typedef T HepMC::detail::disable_if< false, T >::type

check type of class T

Definition at line 40 of file enable_if.h.

7.6.2.2 template<class T > typedef T HepMC::detail::disable_if< false, T >::type

check type of class T

Definition at line 40 of file 2.06.09/HepMC/enable_if.h.

The documentation for this struct was generated from the following files:

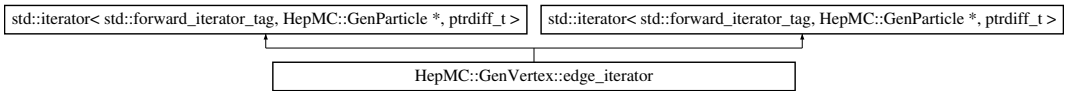
- **enable_if.h**
- **2.06.09/HepMC/enable_if.h**

7.7 HepMC::GenVertex::edge_iterator Class Reference

edge iterator

`#include <GenVertex.h>`

Inheritance diagram for HepMC::GenVertex::edge_iterator:



Public Member Functions

- **edge_iterator** ()
- **edge_iterator** (const **GenVertex** &vtx, **IteratorRange** range=**family**)
  used to set limits on the iteration
- **edge_iterator** (const **edge_iterator** &**p**)
  copy
- virtual ∼**edge_iterator** ()
- **edge_iterator** & **operator=** (const **edge_iterator** &**p**)
  make a copy
- **GenParticle** ∗ **operator**∗ (void) const
  return a pointer to a particle
- **edge_iterator** & **operator++** (void)
  Pre-fix increment.
- **edge_iterator operator++** (int)

Post-fix increment.

- bool **operator==** (const **edge_iterator** &a) const

  equality
- bool **operator!=** (const **edge_iterator** &a) const

  inequality
- bool **is_parent** () const

  true if parent of root vtx
- bool **is_child** () const

  true if child of root vtx
- const **GenVertex** ∗ **vertex_root** () const

  root vertex of this iteration
- **edge_iterator** ()
- **edge_iterator** (const **GenVertex** &vtx, **IteratorRange** range=**family**)

  used to set limits on the iteration
- **edge_iterator** (const **edge_iterator** &**p**)

  copy
- virtual ∼**edge_iterator** ()
- **edge_iterator** & **operator=** (const **edge_iterator** &**p**)

  make a copy
- **GenParticle** ∗ **operator**∗ (void) const

  return a pointer to a particle
- **edge_iterator** & **operator++** (void)

  Pre-fix increment.
- **edge_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **edge_iterator** &a) const

  equality
- bool **operator!=** (const **edge_iterator** &a) const

  inequality
- bool **is_parent** () const

  true if parent of root vtx
- bool **is_child** () const

  true if child of root vtx
- const **GenVertex** ∗ **vertex_root** () const

  root vertex of this iteration

7.7.1 Detailed Description

edge iterator

iterate over the family of edges connected to m_vertex begins with parents (incoming particles) then children (outgoing) This is not a recursive iterator ... it is a building block for the public iterators and is intended for internal use only. The acceptable Iterator Ranges are: family, parents, children

Definition at line 194 of file GenVertex.h.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 HepMC::GenVertex::edge_iterator::edge_iterator ( )

Definition at line 462 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.2.2 HepMC::GenVertex::edge_iterator::edge_iterator (const GenVertex & *vtx,* IteratorRange *range =* family)

used to set limits on the iteration

Definition at line 466 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::ancestors, HepMC::children, HepMC::descendants, HepMC::family, and HepMC::parents.

7.7.2.3 HepMC::GenVertex::edge_iterator::edge_iterator (const edge_iterator & *p*)

copy

Definition at line 517 of file HepMC-2.06.09/src/GenVertex.cc.

References p.

7.7.2.4 HepMC::GenVertex::edge_iterator::~edge_iterator ( ) [virtual]

Definition at line 521 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.2.5 HepMC::GenVertex::edge_iterator::edge_iterator ( )

7.7.2.6 HepMC::GenVertex::edge_iterator::edge_iterator (const GenVertex & *vtx,* IteratorRange *range =* family)

used to set limits on the iteration

7.7.2.7 HepMC::GenVertex::edge_iterator::edge_iterator (const edge_iterator & *p*)

copy

7.7.2.8 virtual HepMC::GenVertex::edge_iterator::~edge_iterator ( ) [virtual]

7.7.3 Member Function Documentation

7.7.3.1 bool HepMC::GenVertex::edge_iterator::is_child ( ) const

true if child of root vtx

Definition at line 590 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.3.2 bool HepMC::GenVertex::edge_iterator::is_child ( ) const

true if child of root vtx

7.7.3.3 bool HepMC::GenVertex::edge_iterator::is_parent ( ) const

true if parent of root vtx

Definition at line 585 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.3.4 bool HepMC::GenVertex::edge_iterator::is_parent ( ) const

true if parent of root vtx

7.7.3.5 bool HepMC::GenVertex::edge_iterator::operator!= (const edge_iterator & *a*) const [inline]

inequality

Definition at line 467 of file GenVertex.h.

7.7.3.6 bool HepMC::GenVertex::edge_iterator::operator!= (const edge_iterator & *a*) const

inequality

7.7.3.7 GenParticle* HepMC::GenVertex::edge_iterator::operator* (void) const

return a pointer to a particle

7.7.3.8 GenParticle * HepMC::GenVertex::edge_iterator::operator* (void) const

return a pointer to a particle

Definition at line 533 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.3.9 edge_iterator& HepMC::GenVertex::edge_iterator::operator++ (void)

Pre-fix increment.

7.7.3.10 GenVertex::edge_iterator & HepMC::GenVertex::edge_iterator::operator++ (void)

Pre-fix increment.

Definition at line 538 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::family, and HepMC::parents.

7.7.3.11 edge_iterator HepMC::GenVertex::edge_iterator::operator++ (int)

Post-fix increment.

7.7.3.12 GenVertex::edge_iterator HepMC::GenVertex::edge_iterator::operator++ (int)

Post-fix increment.

Definition at line 578 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.3.13 GenVertex::edge_iterator & HepMC::GenVertex::edge_iterator::operator= (const edge_iterator & *p*)

make a copy

Definition at line 523 of file HepMC-2.06.09/src/GenVertex.cc.

7.7.3.14 edge_iterator& HepMC::GenVertex::edge_iterator::operator= (const edge_iterator & *p*)

make a copy

7.7.3.15 bool HepMC::GenVertex::edge_iterator::operator== (const edge_iterator & *a*) const

equality

7.7.3.16 bool HepMC::GenVertex::edge_iterator::operator== (const edge_iterator & *a*) const [inline]

equality

Definition at line 462 of file GenVertex.h.

7.7.3.17 const GenVertex * HepMC::GenVertex::edge_iterator::vertex_root ( ) const [inline]

root vertex of this iteration

Definition at line 472 of file GenVertex.h.

7.7.3.18 const GenVertex* HepMC::GenVertex::edge_iterator::vertex_root ( ) const

root vertex of this iteration

The documentation for this class was generated from the following files:

- **GenVertex.h**
- **2.06.09/HepMC/GenVertex.h**
- **HepMC-2.06.09/src/GenVertex.cc**
- **src/GenVertex.cc**

7.8 HepMC::detail::enable_if< bool, class > Struct Template Reference

internal - used to decide if a class is arithmetic

`#include <enable_if.h>`

7.8.1 Detailed Description

template<bool, class>struct HepMC::detail::enable_if< bool, class >

internal - used to decide if a class is arithmetic

Definition at line 17 of file enable_if.h.

The documentation for this struct was generated from the following file:

- **enable_if.h**

7.9 HepMC::detail::enable_if< true, T > Struct Template Reference

internal - use if class T is arithmetic

`#include <enable_if.h>`

Public Types

- typedef T **type**
  check type of class T
- typedef T **type**
  check type of class T

7.9.1 Detailed Description

template<class T>struct HepMC::detail::enable_if< true, T >

internal - use if class T is arithmetic

Definition at line 22 of file enable_if.h.

7.9.2 Member Typedef Documentation

7.9.2.1 template<class T > typedef T HepMC::detail::enable_if< true, T >::type

check type of class T

Definition at line 24 of file enable_if.h.

7.9.2.2 template<class T > typedef T HepMC::detail::enable_if< true, T >::type

check type of class T

Definition at line 24 of file 2.06.09/HepMC/enable_if.h.

The documentation for this struct was generated from the following files:

- **enable_if.h**
- **2.06.09/HepMC/enable_if.h**

7.10 HepMC::Flow Class Reference

The flow object.

`#include <Flow.h>`

Public Types

- typedef std::map< int, int >
  ::**iterator iterator**
  iterator for flow pattern container
- typedef std::map< int, int >
  ::**const_iterator const_iterator**
  const iterator for flow pattern container
- typedef std::map< int, int >
  ::**iterator iterator**
  iterator for flow pattern container
- typedef std::map< int, int >
  ::**const_iterator const_iterator**
  const iterator for flow pattern container

Public Member Functions

- **Flow** (**GenParticle** ∗**particle_owner**=0)
  default constructor
- **Flow** (const **Flow** &)
  copy
- virtual ∼**Flow** ()
- void **swap** (**Flow** &other)
  swap
- **Flow** & **operator=** (const **Flow** &)
  make a copy
- bool **operator==** (const **Flow** &a) const
  equality
- bool **operator!=** (const **Flow** &a) const
  inequality
- void **print** (std::ostream &ostr=std::cout) const
  *print **Flow** (p. 44) information to ostr*
- std::vector< **HepMC::GenParticle** ∗ > **connected_partners** (int code, int code_index=1, int num_indices=2) const
- std::vector< **HepMC::GenParticle** ∗ > **dangling_connected_partners** (int code, int code_index=1, int num_indices=2) const
- const **GenParticle** ∗ **particle_owner** () const
  *find particle owning this **Flow** (p. 44)*
- int **icode** (int code_index=1) const
  flow code
- **Flow set_icode** (int code_index, int code)
  set flow code
- **Flow set_unique_icode** (int code_index=1)
  set unique flow code
- bool **empty** () const
  return true if there is no flow container
- int **size** () const
  size of flow pattern container
- void **clear** ()
  clear flow patterns
- bool **erase** (int code_index)
  empty flow pattern container
- **iterator begin** ()
  beginning of flow pattern container
- **iterator end** ()
  end of flow pattern container
- **const_iterator begin** () const
  beginning of flow pattern container
- **const_iterator end** () const
  end of flow pattern container
- **Flow** (**GenParticle** ∗**particle_owner**=0)
  default constructor
- **Flow** (const **Flow** &)
  copy
- virtual ∼**Flow** ()
- void **swap** (**Flow** &other)

swap

- **Flow** & **operator=** (const **Flow** &)

  make a copy
- bool **operator==** (const **Flow** &a) const

  equality
- bool **operator!=** (const **Flow** &a) const

  inequality
- void **print** (std::ostream &ostr=std::cout) const

  *print **Flow*** (p. *44) information to ostr*
- std::vector< **HepMC::GenParticle** * > **connected_partners** (int code, int code_index=1, int num_indices=2) const
- std::vector< **HepMC::GenParticle** * > **dangling_connected_partners** (int code, int code_index=1, int num_indices=2) const
- const **GenParticle** * **particle_owner** () const

  *find particle owning this **Flow*** (p. *44)*
- int **icode** (int code_index=1) const

  flow code
- **Flow set_icode** (int code_index, int code)

  set flow code
- **Flow set_unique_icode** (int code_index=1)

  set unique flow code
- bool **empty** () const

  return true if there is no flow container
- int **size** () const

  size of flow pattern container
- void **clear** ()

  clear flow patterns
- bool **erase** (int code_index)

  empty flow pattern container
- **iterator begin** ()

  beginning of flow pattern container
- **iterator end** ()

  end of flow pattern container
- **const_iterator begin** () const

  beginning of flow pattern container
- **const_iterator end** () const

  end of flow pattern container

Protected Member Functions

- void **connected_partners** (std::vector< **HepMC::GenParticle** * > *output, int code, int code_index, int num_indices) const

  for internal use only
- void **dangling_connected_partners** (std::vector< **HepMC::GenParticle** * > *output, std::vector< **HepMC::GenParticle** * > *visited_particles, int code, int code_index, int num_indices) const

  for internal use only
- void **connected_partners** (std::vector< **HepMC::GenParticle** * > *output, int code, int code_index, int num_indices) const

  for internal use only
- void **dangling_connected_partners** (std::vector< **HepMC::GenParticle** * > *output, std::vector< **HepMC::GenParticle** * > *visited_particles, int code, int code_index, int num_indices) const

  for internal use only

Friends

- std::ostream & **operator**<< (std::ostream &ostr, const **Flow** &f)

  for printing
- std::ostream & **operator**<< (std::ostream &ostr, const **Flow** &f)

  for printing

7.10.1 Detailed Description

The flow object.

The particle's flow object keeps track of an arbitrary number of flow patterns within a graph (i.e. color flow, charge flow, lepton number flow, ...) **Flow** (p. 44) patterns are coded with an integer, in the same manner as in Herwig.

Examples:

testFlow.cc.

Definition at line 66 of file Flow.h.

7.10.2 Member Typedef Documentation

7.10.2.1 typedef std::map<int,int>::const_iterator HepMC::Flow::const_iterator

const iterator for flow pattern container

Definition at line 128 of file Flow.h.

7.10.2.2 typedef std::map<int,int>::const_iterator HepMC::Flow::const_iterator

const iterator for flow pattern container

Definition at line 128 of file 2.06.09/HepMC/Flow.h.

7.10.2.3 typedef std::map<int,int>::iterator HepMC::Flow::iterator

iterator for flow pattern container

Definition at line 126 of file 2.06.09/HepMC/Flow.h.

7.10.2.4 typedef std::map<int,int>::iterator HepMC::Flow::iterator

iterator for flow pattern container

Definition at line 126 of file Flow.h.

7.10.3 Constructor & Destructor Documentation

7.10.3.1 HepMC::Flow::Flow (GenParticle * *particle_owner* = 0)

default constructor

Definition at line 13 of file HepMC-2.06.09/src/Flow.cc.

7.10.3.2 HepMC::Flow::Flow (const Flow & *inflow*)

copy

copies both the m_icode AND the m_particle_owner

copies both the m_icode AND the m_particle_owner

Definition at line 17 of file HepMC-2.06.09/src/Flow.cc.

7.10.3.3 HepMC::Flow::~Flow ( ) [virtual]

Definition at line 24 of file HepMC-2.06.09/src/Flow.cc.

7.10.3.4 HepMC::Flow::Flow (GenParticle * *particle_owner* = 0)

default constructor

7.10.3.5 HepMC::Flow::Flow (const Flow &)

copy

7.10.3.6 virtual HepMC::Flow::~Flow ( ) [virtual]

7.10.4 Member Function Documentation

7.10.4.1 Flow::iterator HepMC::Flow::begin ( ) [inline]

beginning of flow pattern container

Definition at line 184 of file Flow.h.

7.10.4.2 iterator HepMC::Flow::begin ( )

beginning of flow pattern container

7.10.4.3 Flow::const_iterator HepMC::Flow::begin ( ) const [inline]

beginning of flow pattern container

Definition at line 186 of file Flow.h.

7.10.4.4 const_iterator HepMC::Flow::begin ( ) const

beginning of flow pattern container

7.10.4.5 void HepMC::Flow::clear ( )

clear flow patterns

7.10.4.6 void HepMC::Flow::clear ( ) [inline]

clear flow patterns

Definition at line 179 of file Flow.h.

7.10.4.7 std::vector< GenParticle * > HepMC::Flow::connected_partners (int *code*, int *code_index* = 1, int *num_indices* = 2) const

returns all connected particles which have "code" in any of the num_indices beginning with index code_index. Returns all flow partners which have "code" in any of the num_indices beginning with index code_index. m_particle-_owner is included in the result. Return is by value since the set should never be very big. EXAMPLE: if you want to find all flow partners that have the same code in indices 2,3,4 as particle p has in index 2, you would use: set<GenParticle*> result = p->flow().connected_partners(p->flow().icode(2),2,3);

Returns all flow partners which have "code" in any of the num_indices beginning with index code_index. m_particle-_owner is included in the result. Return is by value since the set should never be very big. EXAMPLE: if you want to find all flow partners that have the same code in indices 2,3,4 as particle p has in index 2, you would use: set<GenParticle*> result = p->flow().connected_partners(p->flow().icode(2),2,3);

Examples:

testFlow.cc.

Definition at line 38 of file HepMC-2.06.09/src/Flow.cc.

References icode(), and HepMC::detail::output().

Referenced by main().

7.10.4.8 std::vector<HepMC::GenParticle*> HepMC::Flow::connected_partners (int *code*, int *code_index* = 1, int *num_indices* = 2) const

returns all connected particles which have "code" in any of the num_indices beginning with index code_index.

7.10.4.9 void HepMC::Flow::connected_partners (std::vector< HepMC::GenParticle * > * *output*, int *code*, int *code_index*, int *num_indices*) const [protected]

for internal use only

protected: for recursive use by **Flow::connected_partners()** (p. 49)

protected: for recursive use by **Flow::connected_partners()** (p. 49)

Definition at line 60 of file HepMC-2.06.09/src/Flow.cc.

References HepMC::GenParticle::end_vertex(), HepMC::family, HepMC::not_in_vector(), p, HepMC::GenVertex-::particles_begin(), HepMC::GenVertex::particles_end(), and HepMC::GenParticle::production_vertex().

7.10.4.10 void HepMC::Flow::connected_partners (std::vector< HepMC::GenParticle * > * *output*, int *code*, int *code_index*, int *num_indices*) const [protected]

for internal use only

7.10.4.11 std::vector< GenParticle * > HepMC::Flow::dangling_connected_partners (int *code*, int *code_index* = 1, int *num_indices* = 2) const

same as connected_partners, but returns only those particles which are connected to <=1 other particles (i.e. the flow line "dangles" at these particles)

Examples:

testFlow.cc.

Definition at line 108 of file HepMC-2.06.09/src/Flow.cc.

References icode(), and HepMC::detail::output().

Referenced by main().

7.10.4.12 std::vector<HepMC::GenParticle*> HepMC::Flow::dangling_connected_partners (int *code*, int *code_index* = 1, int *num_indices* = 2) const

same as connected_partners, but returns only those particles which are connected to <=1 other particles (i.e. the flow line "dangles" at these particles)

7.10.4.13 void HepMC::Flow::dangling_connected_partners (std::vector< HepMC::GenParticle * > * *output*, std::vector< HepMC::GenParticle * > * *visited_particles*, int *code*, int *code_index*, int *num_indices*) const [protected]

for internal use only

protected: for recursive use by **Flow::dangling_connected_partners** (p. 49)

protected: for recursive use by **Flow::dangling_connected_partners** (p. 49)

Definition at line 123 of file HepMC-2.06.09/src/Flow.cc.

References HepMC::GenParticle::end_vertex(), HepMC::family, HepMC::not_in_vector(), p, HepMC::GenVertex::particles_begin(), HepMC::GenVertex::particles_end(), and HepMC::GenParticle::production_vertex().

7.10.4.14 void HepMC::Flow::dangling_connected_partners (std::vector< HepMC::GenParticle * > * *output*, std::vector< HepMC::GenParticle * > * *visited_particles*, int *code*, int *code_index*, int *num_indices*) const [protected]

for internal use only

7.10.4.15 bool HepMC::Flow::empty ( ) const

return true if there is no flow container

7.10.4.16 bool HepMC::Flow::empty ( ) const [inline]

return true if there is no flow container

Definition at line 177 of file Flow.h.

7.10.4.17 Flow::iterator HepMC::Flow::end ( ) [inline]

end of flow pattern container

Definition at line 185 of file Flow.h.

7.10.4.18 iterator HepMC::Flow::end ( )

end of flow pattern container

7.10.4.19 const_iterator HepMC::Flow::end ( ) const

end of flow pattern container

7.10.4.20 Flow::const_iterator HepMC::Flow::end ( ) const [inline]

end of flow pattern container

Definition at line 187 of file Flow.h.

7.10.4.21 bool HepMC::Flow::erase (int *code_index*)

empty flow pattern container

7.10.4.22 bool HepMC::Flow::erase (int *code_index*) [inline]

empty flow pattern container

Examples:

testFlow.cc.

Definition at line 180 of file Flow.h.

Referenced by main().

7.10.4.23 int HepMC::Flow::icode (int *code_index* = 1) const [inline]

flow code

Examples:

testFlow.cc.

Definition at line 163 of file Flow.h.

Referenced by connected_partners(), dangling_connected_partners(), HepMC::GenParticle::flow(), and main().

7.10.4.24 int HepMC::Flow::icode (int *code_index* = 1) const

flow code

7.10.4.25 bool HepMC::Flow::operator!= (const Flow & *a*) const

inequality

7.10.4.26 bool HepMC::Flow::operator!= (const Flow & *a*) const [inline]

inequality

Definition at line 199 of file Flow.h.

7.10.4.27 Flow & HepMC::Flow::operator= (const Flow & *inflow*) [inline]

make a copy

copies only the m_icode ... not the particle_owner this is intuitive behaviour so you can do oneparticle->flow() = otherparticle->flow()

copies only the m_icode ... not the particle_owner this is intuitive behaviour so you can do oneparticle->flow() = otherparticle->flow()

Definition at line 202 of file Flow.h.

7.10.4.28 Flow& HepMC::Flow::operator= (const Flow &)

make a copy

7.10.4.29 bool HepMC::Flow::operator== (const Flow & *a*) const [inline]

equality

equivalent flows have the same flow codes for all flow_numbers (i.e. their m_icode maps are identical), but they need not have the same m_particle owner

equivalent flows have the same flow codes for all flow_numbers (i.e. their m_icode maps are identical), but they need not have the same m_particle owner

Definition at line 193 of file Flow.h.

7.10.4.30 bool HepMC::Flow::operator== (const Flow & *a*) const

equality

7.10.4.31 const GenParticle* HepMC::Flow::particle_owner ( ) const

find particle owning this **Flow** (p. 44)

7.10.4.32 const GenParticle * HepMC::Flow::particle_owner ( ) const [inline]

find particle owning this **Flow** (p. 44)

Definition at line 160 of file Flow.h.

7.10.4.33 void HepMC::Flow::print (std::ostream & *ostr* = std::cout) const

print **Flow** (p. 44) information to ostr

Definition at line 34 of file HepMC-2.06.09/src/Flow.cc.

7.10.4.34 void HepMC::Flow::print (std::ostream & *ostr* = std::cout) const

print **Flow** (p. 44) information to ostr

7.10.4.35 Flow HepMC::Flow::set_icode (int *code_index,* int *code*) [inline]

set flow code

Definition at line 167 of file Flow.h.

Referenced by HepMC::detail::read_particle(), and HepMC::GenParticle::set_flow().

7.10.4.36 Flow HepMC::Flow::set_icode (int *code_index,* int *code*)

set flow code

7.10.4.37 Flow HepMC::Flow::set_unique_icode (int *code_index* = 1)

set unique flow code

7.10.4.38 Flow HepMC::Flow::set_unique_icode (int *code_index* = 1) [inline]

set unique flow code
use this method if you want to assign a unique flow code, but do not want the burden of choosing it yourself
use this method if you want to assign a unique flow code, but do not want the burden of choosing it yourself
Definition at line 171 of file Flow.h.
Referenced by HepMC::GenParticle::set_flow().

7.10.4.39 int HepMC::Flow::size ( ) const

size of flow pattern container

7.10.4.40 int HepMC::Flow::size ( ) const [inline]

size of flow pattern container
Definition at line 178 of file Flow.h.

7.10.4.41 void HepMC::Flow::swap (Flow & *other*)

swap
Definition at line 28 of file HepMC-2.06.09/src/Flow.cc.
Referenced by HepMC::GenParticle::swap().

7.10.4.42 void HepMC::Flow::swap (Flow & *other*)

swap

7.10.5 Friends And Related Function Documentation

7.10.5.1 std::ostream& operator<< (std::ostream & *ostr,* const Flow & *f*) [friend]

for printing
Definition at line 190 of file HepMC-2.06.09/src/Flow.cc.

7.10.5.2 std::ostream& operator<< (std::ostream & *ostr,* const **Flow** & *f*) [friend]

for printing

for printing

Definition at line 190 of file HepMC-2.06.09/src/Flow.cc.

The documentation for this class was generated from the following files:

- **Flow.h**
- **2.06.09/HepMC/Flow.h**
- **HepMC-2.06.09/src/Flow.cc**
- **src/Flow.cc**

7.11 HepMC::FourVector Class Reference

FourVector (p. 54) is a simple representation of a physics 4 vector.

`#include <SimpleVector.h>`

Public Member Functions

- **FourVector** (double xin, double yin, double zin, double tin=0)
  constructor requiring at least x, y, and z
- **FourVector** (double tin)
  constructor requiring only t
- **FourVector** ()
- template<class T >
  FourVector (const T &**v**, typename **detail::disable_if**< **detail::is_arithmetic**< T >::value, void >::type *=0)
- **FourVector** (const **FourVector** &**v**)
  copy constructor
- void **swap** (**FourVector** &other)
  swap
- double **px** () const
  return px
- double **py** () const
  return py
- double **pz** () const
  return pz
- double **e** () const
  return E
- double **x** () const
  return x
- double **y** () const
  return y
- double **z** () const
  return z
- double **t** () const
  return t
- double **m2** () const
  Invariant mass squared.
- double **m** () const

*Invariant mass. If **m2()** (p. 59) is negative then -sqrt(-m2()) is returned.*

- double **perp2** () const

  Transverse component of the spatial vector squared.
- double **perp** () const

  Transverse component of the spatial vector (R in cylindrical system).
- double **theta** () const

  The polar angle.
- double **phi** () const

  The azimuth angle.
- double **rho** () const

  spatial vector component magnitude
- **FourVector** & **operator=** (const **FourVector** &)

  make a copy
- bool **operator==** (const **FourVector** &) const

  equality
- bool **operator!=** (const **FourVector** &) const

  inequality
- double **pseudoRapidity** () const

  Returns the pseudo-rapidity, i.e. -ln(tan(theta/2))
- double **eta** () const

  Pseudorapidity (of the space part)
- void **set** (double **x**, double **y**, double **z**, double **t**)

  set x, y, z, and t
- void **setX** (double xin)

  set x
- void **setY** (double yin)

  set y
- void **setZ** (double zin)

  set z
- void **setT** (double tin)

  set t
- void **setPx** (double xin)

  set px
- void **setPy** (double yin)

  set py
- void **setPz** (double zin)

  set pz
- void **setE** (double tin)

  set E
- **FourVector** (double xin, double yin, double zin, double tin=0)

  constructor requiring at least x, y, and z
- **FourVector** (double tin)

  constructor requiring only t
- **FourVector** ()
- template<class T >
  FourVector (const T &**v**, typename **detail::disable_if**< **detail::is_arithmetic**< T >::value, void >::type *=0)
- **FourVector** (const **FourVector** &**v**)

  copy constructor
- void **swap** (**FourVector** &other)

  swap
- double **px** () const

return px

- double **py** () const

  return py

- double **pz** () const

  return pz

- double **e** () const

  return E

- double **x** () const

  return x

- double **y** () const

  return y

- double **z** () const

  return z

- double **t** () const

  return t

- double **m2** () const

  Invariant mass squared.

- double **m** () const

  *Invariant mass. If **m2()** (p. 59) is negative then -sqrt(-m2()) is returned.*

- double **perp2** () const

  Transverse component of the spatial vector squared.

- double **perp** () const

  Transverse component of the spatial vector (R in cylindrical system).

- double **theta** () const

  The polar angle.

- double **phi** () const

  The azimuth angle.

- double **rho** () const

  spatial vector component magnitude

- **FourVector** & **operator=** (const **FourVector** &)

  make a copy

- bool **operator==** (const **FourVector** &) const

  equality

- bool **operator!=** (const **FourVector** &) const

  inequality

- double **pseudoRapidity** () const

  Returns the pseudo-rapidity, i.e. -ln(tan(theta/2))

- double **eta** () const

  Pseudorapidity (of the space part)

- void **set** (double **x**, double **y**, double **z**, double **t**)

  set x, y, z, and t

- void **setX** (double xin)

  set x

- void **setY** (double yin)

  set y

- void **setZ** (double zin)

  set z

- void **setT** (double tin)

  set t

- void **setPx** (double xin)

  set px

- void **setPy** (double yin)

  set py
- void **setPz** (double zin)

  set pz
- void **setE** (double tin)

  set E

7.11.1 Detailed Description

FourVector (p. 54) is a simple representation of a physics 4 vector.

For compatibility with existing code, the basic expected geometrical access methods are povided. Also, there is a templated constructor that will take another vector (HepLorentzVector, GenVector, ...) which must have the following methods: **x()** (p. 66), **y()** (p. 66), **z()** (p. 66), **t()** (p. 65).

Examples:

example_BuildEventFromScratch.cc, **testFlow.cc**, **testPrintBug.cc**, **testSimpleVector.cc**, and **VectorConversion.h**.

Definition at line 42 of file SimpleVector.h.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 HepMC::FourVector::FourVector (double *xin,* double *yin,* double *zin,* double *tin* = 0) [inline]

constructor requiring at least x, y, and z

Definition at line 47 of file SimpleVector.h.

7.11.2.2 HepMC::FourVector::FourVector (double *tin*) [inline]

constructor requiring only t

Definition at line 51 of file SimpleVector.h.

7.11.2.3 HepMC::FourVector::FourVector ( ) [inline]

Definition at line 54 of file SimpleVector.h.

7.11.2.4 template<class T > HepMC::FourVector::FourVector (const T & *v,* typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type * = 0) [inline]

templated constructor this is used ONLY if T is not arithmetic

Definition at line 60 of file SimpleVector.h.

7.11.2.5 HepMC::FourVector::FourVector (const FourVector & *v*) [inline]

copy constructor

Definition at line 65 of file SimpleVector.h.

7.11.2.6 HepMC::FourVector::FourVector (double *xin,* double *yin,* double *zin,* double *tin* = 0) [inline]

constructor requiring at least x, y, and z

Definition at line 47 of file 2.06.09/HepMC/SimpleVector.h.

7.11.2.7 HepMC::FourVector::FourVector (double *tin*) [inline]

constructor requiring only t

Definition at line 51 of file 2.06.09/HepMC/SimpleVector.h.

7.11.2.8 HepMC::FourVector::FourVector ( ) [inline]

Definition at line 54 of file 2.06.09/HepMC/SimpleVector.h.

7.11.2.9 template<class T > HepMC::FourVector::FourVector (const T & *v,* typename **detail::disable_if**< **detail::is_arithmetic**< T >::value, void >::type * = 0) [inline]

templated constructor this is used ONLY if T is not arithmetic

Definition at line 60 of file 2.06.09/HepMC/SimpleVector.h.

7.11.2.10 HepMC::FourVector::FourVector (const **FourVector** & *v*) [inline]

copy constructor

Definition at line 65 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3 Member Function Documentation

7.11.3.1 double HepMC::FourVector::e ( ) const [inline]

return E

Examples:

testSimpleVector.cc.

Definition at line 73 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), repairUnits(), and HepMC::IO_HEPEVT::write_event().

7.11.3.2 double HepMC::FourVector::e ( ) const [inline]

return E

Definition at line 73 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.3 double HepMC::FourVector::eta ( ) const

Pseudorapidity (of the space part)

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.4 double HepMC::FourVector::eta ( ) const

Pseudorapidity (of the space part)

7.11.3.5 double HepMC::FourVector::m ( ) const

Invariant mass. If **m2()** (p. 59) is negative then -sqrt(-m2()) is returned.

Examples:

testSimpleVector.cc.

Referenced by main(), and HepMC::detail::read_particle().

7.11.3.6 double HepMC::FourVector::m ( ) const

Invariant mass. If **m2()** (p. 59) is negative then -sqrt(-m2()) is returned.

7.11.3.7 double HepMC::FourVector::m2 ( ) const

Invariant mass squared.

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.8 double HepMC::FourVector::m2 ( ) const

Invariant mass squared.

7.11.3.9 bool HepMC::FourVector::operator!= (const FourVector &) const

inequality

7.11.3.10 bool HepMC::FourVector::operator!= (const FourVector &) const

inequality

7.11.3.11 FourVector& HepMC::FourVector::operator= (const FourVector &)

make a copy

7.11.3.12 FourVector& HepMC::FourVector::operator= (const FourVector &)

make a copy

7.11.3.13 bool HepMC::FourVector::operator== (const FourVector &) const

equality

7.11.3.14 bool HepMC::FourVector::operator== (const FourVector &) const

equality

7.11.3.15 double HepMC::FourVector::perp ( ) const

Transverse component of the spatial vector (R in cylindrical system).

7.11.3.16 double HepMC::FourVector::perp ( ) const

Transverse component of the spatial vector (R in cylindrical system).

Examples:

example_UsingIterators.cc, and **testSimpleVector.cc**.

Referenced by IsPhoton(), main(), and IsPhoton::operator()().

7.11.3.17 double HepMC::FourVector::perp2 ( ) const

Transverse component of the spatial vector squared.

7.11.3.18 double HepMC::FourVector::perp2 ( ) const

Transverse component of the spatial vector squared.

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.19 double HepMC::FourVector::phi ( ) const

The azimuth angle.

7.11.3.20 double HepMC::FourVector::phi ( ) const

The azimuth angle.

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.21 double HepMC::FourVector::pseudoRapidity ( ) const

Returns the pseudo-rapidity, i.e. -ln(tan(theta/2))

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.22 double HepMC::FourVector::pseudoRapidity ( ) const

Returns the pseudo-rapidity, i.e. -ln(tan(theta/2))

7.11.3.23 double HepMC::FourVector::px ( ) const [inline]

return px

Definition at line 70 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.24 double HepMC::FourVector::px ( ) const [inline]

return px

Examples:

testSimpleVector.cc.

Definition at line 70 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), repairUnits(), and HepMC::IO_HEPEVT::write_event().

7.11.3.25 double HepMC::FourVector::py ( ) const [inline]

return py

Definition at line 71 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.26 double HepMC::FourVector::py ( ) const [inline]

return py

Examples:

testSimpleVector.cc.

Definition at line 71 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), repairUnits(), and HepMC::IO_HEPEVT::write_event().

7.11.3.27 double HepMC::FourVector::pz ( ) const [inline]

return pz

Definition at line 72 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.28 double HepMC::FourVector::pz ( ) const [inline]

return pz

Examples:

testSimpleVector.cc.

Definition at line 72 of file SimpleVector.h.

Referenced by HepMC::GenParticle::convert_momentum(), main(), HepMC::operator<<(), HepMC::GenParticle::print(), repairUnits(), and HepMC::IO_HEPEVT::write_event().

7.11.3.29 double HepMC::FourVector::rho ( ) const

spatial vector component magnitude

Examples:

testMass.cc.in, and **testSimpleVector.cc**.

Referenced by main().

7.11.3.30 double HepMC::FourVector::rho ( ) const

spatial vector component magnitude

7.11.3.31 void HepMC::FourVector::set (double *x*, double *y*, double *z*, double *t*)

set x, y, z, and t

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.32 void HepMC::FourVector::set (double *x*, double *y*, double *z*, double *t*)

set x, y, z, and t

7.11.3.33 void HepMC::FourVector::setE (double *tin*) [inline]

set E

Examples:

testSimpleVector.cc.

Definition at line 110 of file SimpleVector.h.

Referenced by main().

7.11.3.34 void HepMC::FourVector::setE (double *tin*) [inline]

set E

Definition at line 110 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.35 void HepMC::FourVector::setPx (double *xin*) [inline]

set px

Examples:

testSimpleVector.cc.

Definition at line 107 of file SimpleVector.h.

Referenced by main().

7.11.3.36 void HepMC::FourVector::setPx (double *xin*) [inline]

set px

Definition at line 107 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.37 void HepMC::FourVector::setPy (double *yin*) [inline]

set py

Examples:

testSimpleVector.cc.

Definition at line 108 of file SimpleVector.h.

Referenced by main().

7.11.3.38 void HepMC::FourVector::setPy (double *yin*) [inline]

set py

Definition at line 108 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.39 void HepMC::FourVector::setPz (double *zin*) [inline]

set pz

Definition at line 109 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.40 void HepMC::FourVector::setPz (double *zin*) [inline]

set pz

Examples:

testSimpleVector.cc.

Definition at line 109 of file SimpleVector.h.

Referenced by main().

7.11.3.41 void HepMC::FourVector::setT (double *tin*) [inline]

set t

Examples:

testSimpleVector.cc.

Definition at line 105 of file SimpleVector.h.

Referenced by main().

7.11.3.42 void HepMC::FourVector::setT (double *tin*) [inline]

set t

Definition at line 105 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.43 void HepMC::FourVector::setX (double *xin*) [inline]

set x

Definition at line 102 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.44 void HepMC::FourVector::setX (double *xin*) [inline]

set x

Examples:

testSimpleVector.cc.

Definition at line 102 of file SimpleVector.h.

Referenced by main().

7.11.3.45 void HepMC::FourVector::setY (double *yin*) [inline]

set y

Definition at line 103 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.46 void HepMC::FourVector::setY (double *yin*) [inline]

set y

Examples:

testSimpleVector.cc.

Definition at line 103 of file SimpleVector.h.

Referenced by main().

7.11.3.47 void HepMC::FourVector::setZ (double *zin*) [inline]

set z

Examples:

testSimpleVector.cc.

Definition at line 104 of file SimpleVector.h.

Referenced by main().

7.11.3.48 void HepMC::FourVector::setZ (double *zin*) [inline]

set z

Definition at line 104 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.49 void HepMC::FourVector::swap (FourVector & *other*)

swap

Referenced by HepMC::GenParticle::swap(), and HepMC::GenVertex::swap().

7.11.3.50 void HepMC::FourVector::swap (FourVector & *other*)

swap

7.11.3.51 double HepMC::FourVector::t ( ) const [inline]

return t

Definition at line 78 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.52 double HepMC::FourVector::t ( ) const [inline]

return t

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 78 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), convertTo(), main(), HepMC::GenVertex::print(), and HepMC::IO_HEPEVT::write_event().

7.11.3.53 double HepMC::FourVector::theta ( ) const

The polar angle.

Examples:

testSimpleVector.cc.

Referenced by main().

7.11.3.54 double HepMC::FourVector::theta ( ) const

The polar angle.

7.11.3.55 double HepMC::FourVector::x ( ) const [inline]

return x

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 75 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), convertTo(), main(), HepMC::operator<<(), HepMC::GenVertex::point3d(), HepMC::GenVertex::print(), and HepMC::IO_HEPEVT::write_event().

7.11.3.56 double HepMC::FourVector::x ( ) const [inline]

return x

Definition at line 75 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.57 double HepMC::FourVector::y ( ) const [inline]

return y

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 76 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), convertTo(), main(), HepMC::GenVertex::point3d(), HepMC::GenVertex::print(), and HepMC::IO_HEPEVT::write_event().

7.11.3.58 double HepMC::FourVector::y ( ) const [inline]

return y

Definition at line 76 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.59 double HepMC::FourVector::z ( ) const [inline]

return z

Definition at line 77 of file 2.06.09/HepMC/SimpleVector.h.

7.11.3.60 double HepMC::FourVector::z ( ) const [inline]

return z

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 77 of file SimpleVector.h.

Referenced by HepMC::GenVertex::convert_position(), convertTo(), main(), HepMC::GenVertex::point3d(), HepMC::GenVertex::print(), and HepMC::IO_HEPEVT::write_event().

The documentation for this class was generated from the following files:

- **SimpleVector.h**
- **2.06.09/HepMC/SimpleVector.h**

7.12 HepMC::GenCrossSection Class Reference

The **GenCrossSection** (p. 67) class stores the generated cross section.

`#include <GenCrossSection.h>`

Public Member Functions

- **GenCrossSection** ()
- ~**GenCrossSection** ()
- **GenCrossSection** (**GenCrossSection** const &orig)

  copy
- void **swap** (**GenCrossSection** &other)

  swap
- **GenCrossSection** & **operator=** (**GenCrossSection** const &rhs)
- bool **operator==** (const **GenCrossSection** &) const

  check for equality
- bool **operator!=** (const **GenCrossSection** &) const

  check for inequality
- double **cross_section** () const

  cross section in pb
- double **cross_section_error** () const

  error associated with this cross section in pb
- bool **is_set** () const

  True if the cross section has been set. False by default.
- void **set_cross_section** (double xs, double xs_err)

  Set cross section and error in pb.
- void **set_cross_section** (double)

  set cross section in pb
- void **set_cross_section_error** (double)

  set error associated with this cross section in pb
- void **clear** ()
- std::ostream & **write** (std::ostream &) const

  write to an output stream
- std::istream & **read** (std::istream &)

  read from an input stream
- **GenCrossSection** ()
- ~**GenCrossSection** ()
- **GenCrossSection** (**GenCrossSection** const &orig)

  copy
- void **swap** (**GenCrossSection** &other)

  swap

- **GenCrossSection** & **operator=** (**GenCrossSection** const &rhs)
- bool **operator==** (const **GenCrossSection** &) const

  check for equality
- bool **operator!=** (const **GenCrossSection** &) const

  check for inequality
- double **cross_section** () const

  cross section in pb
- double **cross_section_error** () const

  error associated with this cross section in pb
- bool **is_set** () const

  True if the cross section has been set. False by default.
- void **set_cross_section** (double xs, double xs_err)

  Set cross section and error in pb.
- void **set_cross_section** (double)

  set cross section in pb
- void **set_cross_section_error** (double)

  set error associated with this cross section in pb
- void **clear** ()
- std::ostream & **write** (std::ostream &) const

  write to an output stream
- std::istream & **read** (std::istream &)

  read from an input stream

7.12.1 Detailed Description

The **GenCrossSection** (p. 67) class stores the generated cross section.

HepMC::GenCrossSection (p. 67) is used to store the generated cross section. This class is meant to be used to pass, on an event by event basis, the current best guess of the total cross section. It is expected that the final cross section will be stored elsewhere.

- double cross_section; // cross section in pb
- double cross_section_error; // error associated with this cross section

The units of cross_section and cross_section_error are expected to be pb.

GenCrossSection (p. 67) information will be written if **GenEvent** (p. 72) contains a pointer to a valid **GenCrossSection** (p. 67) object.

Examples:

testHepMC.cc.in.

Definition at line 32 of file GenCrossSection.h.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 HepMC::GenCrossSection::GenCrossSection ( ) [inline]

Definition at line 35 of file GenCrossSection.h.

7.12.2.2 HepMC::GenCrossSection::~GenCrossSection ( ) [inline]

Definition at line 40 of file GenCrossSection.h.

7.12.2.3 HepMC::GenCrossSection::GenCrossSection (GenCrossSection const & *orig*)

copy

Definition at line 19 of file HepMC-2.06.09/src/GenCrossSection.cc.

7.12.2.4 HepMC::GenCrossSection::GenCrossSection ( ) [inline]

Definition at line 35 of file 2.06.09/HepMC/GenCrossSection.h.

7.12.2.5 HepMC::GenCrossSection::~GenCrossSection ( ) [inline]

Definition at line 40 of file 2.06.09/HepMC/GenCrossSection.h.

7.12.2.6 HepMC::GenCrossSection::GenCrossSection (GenCrossSection const & *orig*)

copy

7.12.3 Member Function Documentation

7.12.3.1 void HepMC::GenCrossSection::clear ( )

Clear all **GenCrossSection** (p. 67) info (disables output of **GenCrossSection** (p. 67) until the cross section is set again)

Definition at line 52 of file HepMC-2.06.09/src/GenCrossSection.cc.

7.12.3.2 void HepMC::GenCrossSection::clear ( )

Clear all **GenCrossSection** (p. 67) info (disables output of **GenCrossSection** (p. 67) until the cross section is set again)

7.12.3.3 double HepMC::GenCrossSection::cross_section ( ) const [inline]

cross section in pb

Examples:

fio/example_PythiaStreamIO.cc, and **testHepMC.cc.in**.

Definition at line 55 of file GenCrossSection.h.

Referenced by operator==(), readPythiaStreamIO(), readWithCrossSection(), and HepMC::GenEvent::write_cross_section().

7.12.3.4 double HepMC::GenCrossSection::cross_section ( ) const [inline]

cross section in pb

Definition at line 55 of file 2.06.09/HepMC/GenCrossSection.h.

7.12.3.5 double HepMC::GenCrossSection::cross_section_error ( ) const [inline]

error associated with this cross section in pb

Definition at line 57 of file 2.06.09/HepMC/GenCrossSection.h.

7.12.3.6 double HepMC::GenCrossSection::cross_section_error ( ) const [inline]

error associated with this cross section in pb

Definition at line 57 of file GenCrossSection.h.

Referenced by operator==(), and HepMC::GenEvent::write_cross_section().

7.12.3.7 bool HepMC::GenCrossSection::is_set ( ) const [inline]

True if the cross section has been set. False by default.

Definition at line 60 of file 2.06.09/HepMC/GenCrossSection.h.

7.12.3.8 bool HepMC::GenCrossSection::is_set ( ) const [inline]

True if the cross section has been set. False by default.

Definition at line 60 of file GenCrossSection.h.

Referenced by HepMC::GenEvent::read(), and write().

7.12.3.9 bool HepMC::GenCrossSection::operator!= (const GenCrossSection &) const

check for inequality

7.12.3.10 bool HepMC::GenCrossSection::operator!= (const GenCrossSection & *rhs*) const

check for inequality

Definition at line 46 of file HepMC-2.06.09/src/GenCrossSection.cc.

7.12.3.11 GenCrossSection & HepMC::GenCrossSection::operator= (GenCrossSection const & *rhs*)

shallow

Definition at line 32 of file HepMC-2.06.09/src/GenCrossSection.cc.

References swap().

7.12.3.12 GenCrossSection& HepMC::GenCrossSection::operator= (GenCrossSection const & *rhs*)

shallow

7.12.3.13 bool HepMC::GenCrossSection::operator== (const GenCrossSection &) const

check for equality

7.12.3.14 bool HepMC::GenCrossSection::operator== (const GenCrossSection & *rhs*) const

check for equality

Definition at line 39 of file HepMC-2.06.09/src/GenCrossSection.cc.

References cross_section(), and cross_section_error().

7.12.3.15 std::istream& HepMC::GenCrossSection::read (std::istream &)

read from an input stream

7.12.3.16 std::istream & HepMC::GenCrossSection::read (std::istream & *is*)

read from an input stream

Definition at line 76 of file HepMC-2.06.09/src/GenCrossSection.cc.

References set_cross_section().

Referenced by HepMC::operator>>(), and HepMC::GenEvent::read().

7.12.3.17 void HepMC::GenCrossSection::set_cross_section (double *xs,* double *xs_err*) [inline]

Set cross section and error in pb.

Examples:

testHepMC.cc.in.

Definition at line 98 of file GenCrossSection.h.

References set_cross_section_error().

Referenced by HepMC::getHerwigCrossSection(), HepMC::getPythiaCrossSection(), read(), and writeWithCrossSection().

7.12.3.18 void HepMC::GenCrossSection::set_cross_section (double *xs,* double *xs_err*)

Set cross section and error in pb.

7.12.3.19 void HepMC::GenCrossSection::set_cross_section (double *xs*) [inline]

set cross section in pb

Definition at line 103 of file GenCrossSection.h.

7.12.3.20 void HepMC::GenCrossSection::set_cross_section (double)

set cross section in pb

7.12.3.21 void HepMC::GenCrossSection::set_cross_section_error (double *xserr*) [inline]

set error associated with this cross section in pb

Definition at line 109 of file GenCrossSection.h.

Referenced by set_cross_section().

7.12.3.22 void HepMC::GenCrossSection::set_cross_section_error (double)

set error associated with this cross section in pb

7.12.3.23 void HepMC::GenCrossSection::swap (GenCrossSection & *other*)

swap

Definition at line 25 of file HepMC-2.06.09/src/GenCrossSection.cc.

Referenced by operator=().

7.12.3.24 void HepMC::GenCrossSection::swap (GenCrossSection & *other*)

swap

7.12.3.25 std::ostream & HepMC::GenCrossSection::write (std::ostream & *os*) const

write to an output stream

Definition at line 59 of file HepMC-2.06.09/src/GenCrossSection.cc.

References is_set().

Referenced by HepMC::operator<<(), and HepMC::GenEvent::write().

7.12.3.26 std::ostream& HepMC::GenCrossSection::write (std::ostream &) const

write to an output stream

The documentation for this class was generated from the following files:

- **GenCrossSection.h**
- **2.06.09/HepMC/GenCrossSection.h**
- **HepMC-2.06.09/src/GenCrossSection.cc**
- **src/GenCrossSection.cc**

7.13 HepMC::GenEvent Class Reference

The **GenEvent** (p. 72) class is the core of **HepMC** (p. 15).

`#include <GenEvent.h>`

Classes

- class **particle_const_iterator**
  const particle iterator
- class **particle_iterator**
  non-const particle iterator
- class **vertex_const_iterator**
  const vertex iterator
- class **vertex_iterator**
  non-const vertex iterator

Public Member Functions

- **GenEvent** (int **signal_process_id**=0, int **event_number**=0, **GenVertex** ∗signal_vertex=0, const **WeightContainer** &**weights**=std::vector< double >(), const std::vector< long > &randomstates=std::vector< long >(), **Units::MomentumUnit**=**Units::default_momentum_unit**(), **Units::LengthUnit**=**Units::default_length_unit**())

  *default constructor creates null pointers to **HeavyIon** (p. 146), **PdfInfo** (p. 233), and **GenCrossSection** (p. 67)*
- **GenEvent** (int **signal_process_id**, int **event_number**, **GenVertex** ∗signal_vertex, const **WeightContainer** &**weights**, const std::vector< long > &randomstates, const **HeavyIon** &ion, const **PdfInfo** &pdf, **Units::MomentumUnit**=**Units::default_momentum_unit**(), **Units::LengthUnit**=**Units::default_length_unit**())

  *explicit constructor that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)*
- **GenEvent** (**Units::MomentumUnit**, **Units::LengthUnit**, int **signal_process_id**=0, int **event_number**=0, **GenVertex** ∗signal_vertex=0, const **WeightContainer** &**weights**=std::vector< double >(), const std::vector< long > &randomstates=std::vector< long >())

  constructor requiring units - all else is default
- **GenEvent** (**Units::MomentumUnit**, **Units::LengthUnit**, int **signal_process_id**, int **event_number**, **GenVertex** ∗signal_vertex, const **WeightContainer** &**weights**, const std::vector< long > &randomstates, const **HeavyIon** &ion, const **PdfInfo** &pdf)

  *explicit constructor with units first that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)*
- **GenEvent** (const **GenEvent** &inevent)

  deep copy
- **GenEvent** & **operator=** (const **GenEvent** &inevent)

  make a deep copy
- virtual ∼**GenEvent** ()

  deletes all vertices/particles in this evt
- void **swap** (**GenEvent** &other)

  swap
- void **print** (std::ostream &ostr=std::cout) const

  dumps to ostr
- void **print_version** (std::ostream &ostr=std::cout) const

  dumps release version to ostr
- **GenParticle** ∗ **barcode_to_particle** (int barCode) const

  assign a barcode to a particle
- **GenVertex** ∗ **barcode_to_vertex** (int barCode) const

  assign a barcode to a vertex
- int **signal_process_id** () const

  unique signal process id
- int **event_number** () const

  event number
- int **mpi** () const

  number of multi parton interactions
- double **event_scale** () const

  energy scale, see hep-ph/0109068
- double **alphaQCD** () const

  QCD coupling, see hep-ph/0109068.
- double **alphaQED** () const
- **GenVertex** ∗ **signal_process_vertex** () const

  pointer to the vertex containing the signal process
- bool **valid_beam_particles** () const

  test to see if we have two valid beam particles
- std::pair< **HepMC::GenParticle** ∗, **HepMC::GenParticle** ∗ > **beam_particles** () const

pair of pointers to the two incoming beam particles

- bool **is_valid** () const
- **WeightContainer** & **weights** ()

  *direct access to **WeightContainer** (p. 292)*
- const **WeightContainer** & **weights** () const

  *direct access to **WeightContainer** (p. 292)*
- **GenCrossSection** const * **cross_section** () const

  *access the **GenCrossSection** (p. 67) container if it exists*
- **GenCrossSection** * **cross_section** ()
- **HeavyIon** const * **heavy_ion** () const

  *access the **HeavyIon** (p. 146) container if it exists*
- **HeavyIon** * **heavy_ion** ()
- **PdfInfo** const * **pdf_info** () const

  *access the **PdfInfo** (p. 233) container if it exists*
- **PdfInfo** * **pdf_info** ()
- const std::vector< long > & **random_states** () const

  vector of integers containing information about the random state
- int **particles_size** () const

  how many particle barcodes exist?
- bool **particles_empty** () const

  return true if there are no particle barcodes
- int **vertices_size** () const

  how many vertex barcodes exist?
- bool **vertices_empty** () const

  return true if there are no vertex barcodes
- void **write_units** (std::ostream &os=std::cout) const
- void **write_cross_section** (std::ostream &ostr=std::cout) const
- **Units::MomentumUnit momentum_unit** () const

  ***Units** (p. 29) used by the **GenParticle** (p. 106) momentum **FourVector** (p. 54).*
- **Units::LengthUnit length_unit** () const

  ***Units** (p. 29) used by the **GenVertex** (p. 125) position **FourVector** (p. 54).*
- std::ostream & **write** (std::ostream &)
- std::istream & **read** (std::istream &)
- bool **add_vertex** (**GenVertex** *vtx)

  adds to evt and adopts
- bool **remove_vertex** (**GenVertex** *vtx)

  erases vtx from evt
- void **clear** ()

  empties the entire event
- void **set_signal_process_id** (int id)

  set unique signal process id
- void **set_event_number** (int eventno)

  set event number
- void **set_mpi** (int)

  set number of multi parton interactions
- void **set_event_scale** (double scale)

  set energy scale
- void **set_alphaQCD** (double a)

  set QCD coupling
- void **set_alphaQED** (double a)

  set QED coupling

- void **set_signal_process_vertex** (**GenVertex** ∗)

  set pointer to the vertex containing the signal process
- bool **set_beam_particles** (**GenParticle** ∗, **GenParticle** ∗)

  set incoming beam particles
- bool **set_beam_particles** (std::pair< **HepMC::GenParticle** ∗, **HepMC::GenParticle** ∗ > const &)

  use a pair of GenParticle∗'s to set incoming beam particles
- void **set_random_states** (const std::vector< long > &randomstates)

  provide random state information
- void **set_cross_section** (const **GenCrossSection** &)

  *provide a pointer to the **GenCrossSection*** (p. *67*) *container*
- void **set_heavy_ion** (const **HeavyIon** &ion)

  *provide a pointer to the **HeavyIon*** (p. *146*) *container*
- void **set_pdf_info** (const **PdfInfo** &**p**)

  *provide a pointer to the **PdfInfo*** (p. *233*) *container*
- void **use_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- void **use_units** (std::string &, std::string &)
- void **define_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- void **define_units** (std::string &, std::string &)
- **GenEventVertexRange vertex_range** ()

  vertex range
- **ConstGenEventVertexRange vertex_range** () const

  vertex range
- **GenEventParticleRange particle_range** ()

  particle range
- **ConstGenEventParticleRange particle_range** () const

  particle range
- **vertex_const_iterator vertices_begin** () const

  begin vertex iteration
- **vertex_const_iterator vertices_end** () const

  end vertex iteration
- **vertex_iterator vertices_begin** ()

  begin vertex iteration
- **vertex_iterator vertices_end** ()

  end vertex iteration
- **particle_const_iterator particles_begin** () const

  begin particle iteration
- **particle_const_iterator particles_end** () const

  end particle iteration
- **particle_iterator particles_begin** ()

  begin particle iteration
- **particle_iterator particles_end** ()

  end particle iteration
- **GenEvent** (int **signal_process_id**=0, int **event_number**=0, **GenVertex** ∗signal_vertex=0, const **WeightContainer** &**weights**=std::vector< double >(), const std::vector< long > &randomstates=std::vector< long >(), **Units::MomentumUnit**=**Units::default_momentum_unit**(), **Units::LengthUnit**=**Units::default_length_unit**())

  *default constructor creates null pointers to **HeavyIon*** (p. *146*)*, **PdfInfo*** (p. *233*)*, and **GenCrossSection*** (p. *67*)
- **GenEvent** (int **signal_process_id**, int **event_number**, **GenVertex** ∗signal_vertex, const **WeightContainer** &**weights**, const std::vector< long > &randomstates, const **HeavyIon** &ion, const **PdfInfo** &pdf, **Units::MomentumUnit**=**Units::default_momentum_unit**(), **Units::LengthUnit**=**Units::default_length_unit**())

  *explicit constructor that takes **HeavyIon*** (p. *146*) *and **PdfInfo*** (p. *233*)

- **GenEvent** (**Units::MomentumUnit**, **Units::LengthUnit**, int **signal_process_id**=0, int **event_number**=0, **GenVertex** ∗signal_vertex=0, const **WeightContainer** &**weights**=std::vector< double >(), const std::vector< long > &randomstates=std::vector< long >())

  constructor requiring units - all else is default
- **GenEvent** (**Units::MomentumUnit**, **Units::LengthUnit**, int **signal_process_id**, int **event_number**, **GenVertex** ∗signal_vertex, const **WeightContainer** &**weights**, const std::vector< long > &randomstates, const **HeavyIon** &ion, const **PdfInfo** &pdf)

  *explicit constructor with units first that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)*
- **GenEvent** (const **GenEvent** &inevent)

  deep copy
- **GenEvent** & **operator=** (const **GenEvent** &inevent)

  make a deep copy
- virtual ∼**GenEvent** ()

  deletes all vertices/particles in this evt
- void **swap** (**GenEvent** &other)

  swap
- void **print** (std::ostream &ostr=std::cout) const

  dumps to ostr
- void **print_version** (std::ostream &ostr=std::cout) const

  dumps release version to ostr
- **GenParticle** ∗ **barcode_to_particle** (int barCode) const

  assign a barcode to a particle
- **GenVertex** ∗ **barcode_to_vertex** (int barCode) const

  assign a barcode to a vertex
- int **signal_process_id** () const

  unique signal process id
- int **event_number** () const

  event number
- int **mpi** () const

  number of multi parton interactions
- double **event_scale** () const

  energy scale, see hep-ph/0109068
- double **alphaQCD** () const

  QCD coupling, see hep-ph/0109068.
- double **alphaQED** () const
- **GenVertex** ∗ **signal_process_vertex** () const

  pointer to the vertex containing the signal process
- bool **valid_beam_particles** () const

  test to see if we have two valid beam particles
- std::pair< **HepMC::GenParticle** ∗, **HepMC::GenParticle** ∗ > **beam_particles** () const

  pair of pointers to the two incoming beam particles
- bool **is_valid** () const
- **WeightContainer** & **weights** ()

  *direct access to **WeightContainer** (p. 292)*
- const **WeightContainer** & **weights** () const

  *direct access to **WeightContainer** (p. 292)*
- **GenCrossSection** const ∗ **cross_section** () const

  *access the **GenCrossSection** (p. 67) container if it exists*
- **GenCrossSection** ∗ **cross_section** ()
- **HeavyIon** const ∗ **heavy_ion** () const

*access the **HeavyIon** (p. 146) container if it exists*

- **HeavyIon** ∗ **heavy_ion** ()
- **PdfInfo** const ∗ **pdf_info** () const

  *access the **PdfInfo** (p. 233) container if it exists*
- **PdfInfo** ∗ **pdf_info** ()
- const std::vector< long > & **random_states** () const

  vector of integers containing information about the random state
- int **particles_size** () const

  how many particle barcodes exist?
- bool **particles_empty** () const

  return true if there are no particle barcodes
- int **vertices_size** () const

  how many vertex barcodes exist?
- bool **vertices_empty** () const

  return true if there are no vertex barcodes
- void **write_units** (std::ostream &os=std::cout) const
- void **write_cross_section** (std::ostream &ostr=std::cout) const
- **Units::MomentumUnit momentum_unit** () const

  ***Units** (p. 29) used by the **GenParticle** (p. 106) momentum **FourVector** (p. 54).*
- **Units::LengthUnit length_unit** () const

  ***Units** (p. 29) used by the **GenVertex** (p. 125) position **FourVector** (p. 54).*
- std::ostream & **write** (std::ostream &)
- std::istream & **read** (std::istream &)
- bool **add_vertex** (**GenVertex** ∗vtx)

  adds to evt and adopts
- bool **remove_vertex** (**GenVertex** ∗vtx)

  erases vtx from evt
- void **clear** ()

  empties the entire event
- void **set_signal_process_id** (int id)

  set unique signal process id
- void **set_event_number** (int eventno)

  set event number
- void **set_mpi** (int)

  set number of multi parton interactions
- void **set_event_scale** (double scale)

  set energy scale
- void **set_alphaQCD** (double a)

  set QCD coupling
- void **set_alphaQED** (double a)

  set QED coupling
- void **set_signal_process_vertex** (**GenVertex** ∗)

  set pointer to the vertex containing the signal process
- bool **set_beam_particles** (**GenParticle** ∗, **GenParticle** ∗)

  set incoming beam particles
- bool **set_beam_particles** (std::pair< **HepMC::GenParticle** ∗, **HepMC::GenParticle** ∗ > const &)

  use a pair of GenParticle∗'s to set incoming beam particles
- void **set_random_states** (const std::vector< long > &randomstates)

  provide random state information
- void **set_cross_section** (const **GenCrossSection** &)

  *provide a pointer to the **GenCrossSection** (p. 67) container*

- void **set_heavy_ion** (const **HeavyIon** &ion)

  *provide a pointer to the **HeavyIon** (p. 146) container*
- void **set_pdf_info** (const **PdfInfo** &**p**)

  *provide a pointer to the **PdfInfo** (p. 233) container*
- void **use_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- void **use_units** (std::string &, std::string &)
- void **define_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- void **define_units** (std::string &, std::string &)
- **GenEventVertexRange vertex_range** ()

  vertex range
- **ConstGenEventVertexRange vertex_range** () const

  vertex range
- **GenEventParticleRange particle_range** ()

  particle range
- **ConstGenEventParticleRange particle_range** () const

  particle range
- **vertex_const_iterator vertices_begin** () const

  begin vertex iteration
- **vertex_const_iterator vertices_end** () const

  end vertex iteration
- **vertex_iterator vertices_begin** ()

  begin vertex iteration
- **vertex_iterator vertices_end** ()

  end vertex iteration
- **particle_const_iterator particles_begin** () const

  begin particle iteration
- **particle_const_iterator particles_end** () const

  end particle iteration
- **particle_iterator particles_begin** ()

  begin particle iteration
- **particle_iterator particles_end** ()

  end particle iteration

Protected Member Functions

- bool **set_barcode** (**GenParticle** ∗**p**, int suggested_barcode=false)

  *set the barcode - intended for use by **GenParticle** (p. 106)*
- bool **set_barcode** (**GenVertex** ∗**v**, int suggested_barcode=false)

  *set the barcode - intended for use by **GenVertex** (p. 125)*
- void **remove_barcode** (**GenParticle** ∗**p**)

  *intended for use by **GenParticle** (p. 106)*
- void **remove_barcode** (**GenVertex** ∗**v**)

  *intended for use by **GenVertex** (p. 125)*
- void **delete_all_vertices** ()

  delete all vertices owned by this event
- bool **set_barcode** (**GenParticle** ∗**p**, int suggested_barcode=false)

  *set the barcode - intended for use by **GenParticle** (p. 106)*
- bool **set_barcode** (**GenVertex** ∗**v**, int suggested_barcode=false)

  *set the barcode - intended for use by **GenVertex** (p. 125)*
- void **remove_barcode** (**GenParticle** ∗**p**)

  *intended for use by **GenParticle** (p. 106)*

- void **remove_barcode** (**GenVertex** ∗**v**)

  intended for use by ***GenVertex*** *(*p. *125)*
- void **delete_all_vertices** ()

  delete all vertices owned by this event

Friends

- class **GenParticle**
- class **GenVertex**
- class **vertex_const_iterator**
- class **vertex_iterator**
- class **particle_const_iterator**
- class **particle_iterator**

7.13.1 Detailed Description

The **GenEvent** (p. 72) class is the core of **HepMC** (p. 15).

HepMC::GenEvent (p. 72) contains information about generated particles. **GenEvent** (p. 72) is structured as a set of vertices which contain the particles.

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, **example_MyPythiaOnlyToHepMC.cc**, **example_UsingIterators.cc**, **example_VectorConversion.cc**, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.cc**, **testFlow.cc**, **testHepMC.cc.in**, **testHepMCIteration.cc.in**, **testMass.cc.in**, **testMultipleCopies.cc.in**, **testPrintBug.cc**, and **testStreamIO.cc.in**.

Definition at line 155 of file GenEvent.h.

7.13.2 Constructor & Destructor Documentation

7.13.2.1 HepMC::GenEvent::GenEvent (int *signal_process_id* = 0, int *event_number* = 0, GenVertex ∗ *signal_vertex* = 0, const WeightContainer & *weights* = std::vector<double>(), const std::vector< long > & *randomstates* = std::vector<long>(), Units::MomentumUnit *mom* = Units::default_momentum_unit(), Units::LengthUnit *len* = Units::default_length_unit())

default constructor creates null pointers to **HeavyIon** (p. 146), **PdfInfo** (p. 233), and **GenCrossSection** (p. 67)

This constructor only allows null pointers to **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

This constructor only allows null pointers to **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 22 of file HepMC-2.06.09/src/GenEvent.cc.

7.13.2.2 HepMC::GenEvent::GenEvent (int *signal_process_id*, int *event_number*, GenVertex ∗ *signal_vertex*, const WeightContainer & *weights*, const std::vector< long > & *randomstates*, const HeavyIon & *ion*, const PdfInfo & *pdf*, Units::MomentumUnit *mom* = Units::default_momentum_unit(), Units::LengthUnit *len* = Units::default_length_unit())

explicit constructor that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

GenEvent (p. 72) makes its own copy of **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

GenEvent (p. 72) makes its own copy of **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 55 of file HepMC-2.06.09/src/GenEvent.cc.

7.13.2.3 HepMC::GenEvent::GenEvent (Units::MomentumUnit *mom,* Units::LengthUnit *len,* int *signal_process_id* = 0, int *event_number* = 0, GenVertex * *signal_vertex* = 0, const WeightContainer & *weights* = std::vector<double>(), const std::vector< long > & *randomstates* = std::vector<long>())

constructor requiring units - all else is default

constructor requiring units - all else is default This constructor only allows null pointers to **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

constructor requiring units - all else is default This constructor only allows null pointers to **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 88 of file HepMC-2.06.09/src/GenEvent.cc.

7.13.2.4 HepMC::GenEvent::GenEvent (Units::MomentumUnit *mom,* Units::LengthUnit *len,* int *signal_process_id,* int *event_number,* GenVertex * *signal_vertex,* const WeightContainer & *weights,* const std::vector< long > & *randomstates,* const HeavyIon & *ion,* const PdfInfo & *pdf*)

explicit constructor with units first that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

explicit constructor with units first that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233) **GenEvent** (p. 72) makes its own copy of **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

explicit constructor with units first that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233) **GenEvent** (p. 72) makes its own copy of **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

note: default values for m_event_scale, m_alphaQCD, m_alphaQED are as suggested in hep-ph/0109068, "Generic Interface..."

Definition at line 122 of file HepMC-2.06.09/src/GenEvent.cc.

7.13.2.5 HepMC::GenEvent::GenEvent (const GenEvent & *inevent*)

deep copy

deep copy - makes a copy of all vertices!

deep copy - makes a copy of all vertices!

Definition at line 156 of file HepMC-2.06.09/src/GenEvent.cc.

References add_vertex(), beam_particles(), HepMC::GenParticle::end_vertex(), GenParticle, GenVertex, p, particles_begin(), particles_end(), HepMC::GenParticle::production_vertex(), random_states(), set_beam_particles(), set_random_states(), set_signal_process_vertex(), signal_process_vertex(), HepMC::GenVertex-

::suggest_barcode(), v, vertices_begin(), vertices_end(), and weights().

7.13.2.6 HepMC::GenEvent::~GenEvent ( ) [virtual]

deletes all vertices/particles in this evt

Deep destructor. deletes all vertices/particles in this **GenEvent** (p. 72) deletes the associated **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

Deep destructor. deletes all vertices/particles in this **GenEvent** (p. 72) deletes the associated **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

Definition at line 258 of file HepMC-2.06.09/src/GenEvent.cc.

References delete_all_vertices().

7.13.2.7 HepMC::GenEvent::GenEvent (int *signal_process_id* = 0, int *event_number* = 0, GenVertex * *signal_vertex* = 0, const WeightContainer & *weights* = std::vector< double >(), const std::vector< long > & *randomstates* = std::vector< long >(), Units::MomentumUnit = Units::default_momentum_unit(), Units::LengthUnit = Units::default_length_unit())

default constructor creates null pointers to **HeavyIon** (p. 146), **PdfInfo** (p. 233), and **GenCrossSection** (p. 67)

7.13.2.8 HepMC::GenEvent::GenEvent (int *signal_process_id,* int *event_number,* GenVertex * *signal_vertex,* const WeightContainer & *weights,* const std::vector< long > & *randomstates,* const HeavyIon & *ion,* const PdfInfo & *pdf,* Units::MomentumUnit = Units::default_momentum_unit(), Units::LengthUnit = Units::default_length_unit())

explicit constructor that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

7.13.2.9 HepMC::GenEvent::GenEvent (Units::MomentumUnit , Units::LengthUnit , int *signal_process_id* = 0, int *event_number* = 0, GenVertex * *signal_vertex* = 0, const WeightContainer & *weights* = std::vector< double >(), const std::vector< long > & *randomstates* = std::vector< long >())

constructor requiring units - all else is default

7.13.2.10 HepMC::GenEvent::GenEvent (Units::MomentumUnit , Units::LengthUnit , int *signal_process_id,* int *event_number,* GenVertex * *signal_vertex,* const WeightContainer & *weights,* const std::vector< long > & *randomstates,* const HeavyIon & *ion,* const PdfInfo & *pdf*)

explicit constructor with units first that takes **HeavyIon** (p. 146) and **PdfInfo** (p. 233)

7.13.2.11 HepMC::GenEvent::GenEvent (const GenEvent & *inevent*)

deep copy

7.13.2.12 virtual HepMC::GenEvent::~GenEvent ( ) [virtual]

deletes all vertices/particles in this evt

7.13.3 Member Function Documentation

7.13.3.1 bool HepMC::GenEvent::add_vertex (GenVertex * *vtx*)

adds to evt and adopts

returns true if successful - generally will only return false if the inserted vertex is already included in the event.

returns true if successful - generally will only return false if the inserted vertex is already included in the event.

Examples:

example_BuildEventFromScratch.cc, **example_VectorConversion.cc**, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 334 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::parent_event(), remove_vertex(), and HepMC::GenVertex::set_parent_event_().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_HERWIG::fill_next_event(), GenEvent(), main(), read(), and set_signal_process_vertex().

7.13.3.2 bool HepMC::GenEvent::add_vertex (GenVertex * *vtx*)

adds to evt and adopts

7.13.3.3 double HepMC::GenEvent::alphaQCD ( ) const [inline]

QCD coupling, see hep-ph/0109068.

Definition at line 690 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.4 double HepMC::GenEvent::alphaQCD ( ) const

QCD coupling, see hep-ph/0109068.

7.13.3.5 double HepMC::GenEvent::alphaQED ( ) const

QED coupling, see hep-ph/0109068

7.13.3.6 double HepMC::GenEvent::alphaQED ( ) const [inline]

QED coupling, see hep-ph/0109068

Definition at line 692 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.7 GenParticle * HepMC::GenEvent::barcode_to_particle (int *barCode*) const [inline]

assign a barcode to a particle

Each vertex or particle has a barcode, which is just an integer which uniquely identifies it inside the event (i.e. there is a one to one mapping between particle memory addresses and particle barcodes... and the same applied for vertices).

The value of a barcode has NO MEANING and NO ORDER! For the user's convenience, when an event is read in via an IO_method from an indexed list (like the HEPEVT common block), then the index will become the barcode for that particle.

Particle barcodes are always positive integers. The barcodes are chosen and set automatically when a vertex or particle comes under the ownership of an event (i.e. it is contained in an event).

Please note that the barcodes are intended for internal use within **HepMC** (p. 15) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Definition at line 798 of file GenEvent.h.

7.13.3.8 GenParticle* HepMC::GenEvent::barcode_to_particle (int *barCode*) const

assign a barcode to a particle

7.13.3.9 GenVertex* HepMC::GenEvent::barcode_to_vertex (int *barCode*) const

assign a barcode to a vertex

7.13.3.10 GenVertex * HepMC::GenEvent::barcode_to_vertex (int *barCode*) const [inline]

assign a barcode to a vertex

Each vertex or particle has a barcode, which is just an integer which uniquely identifies it inside the event (i.e. there is a one to one mapping between particle memory addresses and particle barcodes... and the same applied for vertices).

The value of a barcode has NO MEANING and NO ORDER! For the user's convenience, when an event is read in via an IO_method from an indexed list (like the HEPEVT common block), then the index will become the barcode for that particle.

Vertex barcodes are always negative integers. The barcodes are chosen and set automatically when a vertex or particle comes under the ownership of an event (i.e. it is contained in an event).

Please note that the barcodes are intended for internal use within **HepMC** (p. 15) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Definition at line 823 of file GenEvent.h.

Referenced by HepMC::compareVertices(), and read().

7.13.3.11 std::pair<HepMC::GenParticle*,HepMC::GenParticle*> HepMC::GenEvent::beam_particles ( ) const

pair of pointers to the two incoming beam particles

7.13.3.12 std::pair< HepMC::GenParticle *, HepMC::GenParticle * > HepMC::GenEvent::beam_particles ( ) const [inline]

pair of pointers to the two incoming beam particles

Examples:

testMass.cc.in.

Definition at line 844 of file GenEvent.h.

Referenced by HepMC::compareBeamParticles(), filterEvent(), GenEvent(), main(), print(), and write().

7.13.3.13 void HepMC::GenEvent::clear ( )

empties the entire event

remove all information from the event deletes all vertices/particles in this evt

remove all information from the event deletes all vertices/particles in this evt

Examples:

testHepMCIteration.cc.in, and **testStreamIO.cc.in**.

Definition at line 365 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::Units::default_length_unit(), HepMC::Units::default_momentum_unit(), and delete_all_vertices().

Referenced by HepMC::IO_GenEvent::fill_next_event(), main(), and read().

7.13.3.14 void HepMC::GenEvent::clear ( )

empties the entire event

7.13.3.15 GenCrossSection const * HepMC::GenEvent::cross_section ( ) const [inline]

access the **GenCrossSection** (p. 67) container if it exists

Examples:

fio/example_PythiaStreamIO.cc, and **testHepMC.cc.in**.

Definition at line 704 of file GenEvent.h.

Referenced by readPythiaStreamIO(), readWithCrossSection(), readWithWeight(), and write_cross_section().

7.13.3.16 GenCrossSection const* HepMC::GenEvent::cross_section ( ) const

access the **GenCrossSection** (p. 67) container if it exists

7.13.3.17 GenCrossSection * HepMC::GenEvent::cross_section ( ) [inline]

Definition at line 707 of file GenEvent.h.

7.13.3.18 GenCrossSection* HepMC::GenEvent::cross_section ( )

7.13.3.19 void HepMC::GenEvent::define_units (Units::MomentumUnit *new_m*, Units::LengthUnit *new_l*) [inline]

set the units using enums This method will NOT convert momentum and position data

Examples:

testHepMC.cc.in.

Definition at line 866 of file GenEvent.h.

Referenced by read_testUnits().

7.13.3.20 void HepMC::GenEvent::define_units (Units::MomentumUnit , Units::LengthUnit)

set the units using enums This method will NOT convert momentum and position data

7.13.3.21 void HepMC::GenEvent::define_units (std::string & *new_m,* std::string & *new_l*)

set the units using strings the string must match the enum exactly This method will NOT convert momentum and position data

Definition at line 665 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::Units::CM, HepMC::Units::GEV, HepMC::Units::MEV, and HepMC::Units::MM.

7.13.3.22 void HepMC::GenEvent::define_units (std::string & , std::string &)

set the units using strings the string must match the enum exactly This method will NOT convert momentum and position data

7.13.3.23 void HepMC::GenEvent::delete_all_vertices ( ) [protected]

delete all vertices owned by this event

deletes all vertices in the vertex container (i.e. all vertices owned by this event) The vertices are the "owners" of the particles, so as we delete the vertices, the vertex desctructors are automatically deleting their particles.

deletes all vertices in the vertex container (i.e. all vertices owned by this event) The vertices are the "owners" of the particles, so as we delete the vertices, the vertex desctructors are automatically deleting their particles.

Definition at line 403 of file HepMC-2.06.09/src/GenEvent.cc.

References particles_empty(), and vertices_empty().

Referenced by clear(), and ~GenEvent().

7.13.3.24 void HepMC::GenEvent::delete_all_vertices ( ) [protected]

delete all vertices owned by this event

7.13.3.25 int HepMC::GenEvent::event_number ( ) const [inline]

event number

Examples:

example_EventSelection.cc, **fio/example_MyPythia.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.-cc**, **testHepMC.cc.in**, **testHepMCIteration.cc.in**, **testMass.cc.in**, **testMultipleCopies.cc.in**, and **test-StreamIO.cc.in**.

Definition at line 682 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), findW(), main(), massInfo(), particleTypes(), print(), pythia_in(), pythia-_in_out(), read_from_stream4(), read_nan(), read_testIOGenEvent(), read_testUnits(), read_variousFormats(), readWithCrossSection(), readWithWeight(), simpleIter(), simpleIter2(), simpleIter3(), simpleIter4(), write(), Hep-MC::IO_HEPEVT::write_event(), HepMC::IO_AsciiParticles::write_event(), write_to_stream(), write_to_stream3(), writeWithCrossSection(), and writeWithWeight().

7.13.3.26 int HepMC::GenEvent::event_number ( ) const

event number

7.13.3.27 double HepMC::GenEvent::event_scale ( ) const

energy scale, see hep-ph/0109068

7.13.3.28 double HepMC::GenEvent::event_scale ( ) const [inline]

energy scale, see hep-ph/0109068

Definition at line 688 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.29 HeavyIon const * HepMC::GenEvent::heavy_ion ( ) const [inline]

access the **HeavyIon** (p. 146) container if it exists

Definition at line 710 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), and write().

7.13.3.30 HeavyIon const* HepMC::GenEvent::heavy_ion ( ) const

access the **HeavyIon** (p. 146) container if it exists

7.13.3.31 HeavyIon * HepMC::GenEvent::heavy_ion ( ) [inline]

Definition at line 713 of file GenEvent.h.

7.13.3.32 HeavyIon* HepMC::GenEvent::heavy_ion ( )

7.13.3.33 bool HepMC::GenEvent::is_valid ( ) const

check **GenEvent** (p. 72) for validity A **GenEvent** (p. 72) is presumed valid if it has particles and/or vertices. A **GenEvent** (p. 72) is presumed valid if it has both associated particles and vertices. No other information is checked.

A **GenEvent** (p. 72) is presumed valid if it has both associated particles and vertices. No other information is checked.

Examples:

fio/example_PythiaStreamIO.cc, and **testStreamIO.cc.in**.

Definition at line 677 of file HepMC-2.06.09/src/GenEvent.cc.

References particles_empty(), and vertices_empty().

Referenced by HepMC::IO_GenEvent::fill_next_event(), read_from_stream4(), and readPythiaStreamIO().

7.13.3.34 bool HepMC::GenEvent::is_valid ( ) const

check **GenEvent** (p. 72) for validity A **GenEvent** (p. 72) is presumed valid if it has particles and/or vertices.

7.13.3.35 Units::LengthUnit HepMC::GenEvent::length_unit ( ) const [inline]

Units (p. 29) used by the **GenVertex** (p. 125) position **FourVector** (p. 54).

Definition at line 852 of file GenEvent.h.

Referenced by write(), and write_units().

7.13.3.36 Units::LengthUnit HepMC::GenEvent::length_unit ( ) const

Units (p. 29) used by the **GenVertex** (p. 125) position **FourVector** (p. 54).

7.13.3.37 Units::MomentumUnit HepMC::GenEvent::momentum_unit ( ) const [inline]

Units (p. 29) used by the **GenParticle** (p. 106) momentum **FourVector** (p. 54).

Definition at line 849 of file GenEvent.h.

Referenced by write(), and write_units().

7.13.3.38 Units::MomentumUnit HepMC::GenEvent::momentum_unit ( ) const

Units (p. 29) used by the **GenParticle** (p. 106) momentum **FourVector** (p. 54).

7.13.3.39 int HepMC::GenEvent::mpi ( ) const [inline]

number of multi parton interactions

Returns the number of multi parton interactions in the event. This number is -1 if it is not set.

Definition at line 686 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), and write().

7.13.3.40 int HepMC::GenEvent::mpi ( ) const

number of multi parton interactions

7.13.3.41 GenEvent& HepMC::GenEvent::operator= (const GenEvent & *inevent*)

make a deep copy

7.13.3.42 GenEvent & HepMC::GenEvent::operator= (const GenEvent & *inevent*)

make a deep copy

best practices implementation

best practices implementation

Definition at line 269 of file HepMC-2.06.09/src/GenEvent.cc.

References swap().

7.13.3.43 GenEventParticleRange HepMC::GenEvent::particle_range ( )

particle range

7.13.3.44 GenEventParticleRange HepMC::GenEvent::particle_range ( )

particle range

Examples:

testHepMCIteration.cc.in.

Definition at line 26 of file HepMC-2.06.09/src/GenRanges.cc.

Referenced by simpleIter4().

7.13.3.45 ConstGenEventParticleRange HepMC::GenEvent::particle_range ( ) const

particle range

7.13.3.46 ConstGenEventParticleRange HepMC::GenEvent::particle_range ( ) const

particle range

Definition at line 31 of file HepMC-2.06.09/src/GenRanges.cc.

7.13.3.47 particle_const_iterator HepMC::GenEvent::particles_begin ( ) const [inline]

begin particle iteration

Definition at line 507 of file 2.06.09/HepMC/GenEvent.h.

References particle_const_iterator.

7.13.3.48 particle_const_iterator HepMC::GenEvent::particles_begin ( ) const [inline]

begin particle iteration

Examples:

example_EventSelection.cc, **example_UsingIterators.cc**, **example_VectorConversion.cc**, **fio/example_MyPythia.cc**, **testHepMCIteration.cc.in**, **testMass.cc.in**, and **testMultipleCopies.cc.in**.

Definition at line 507 of file GenEvent.h.

References particle_const_iterator.

Referenced by HepMC::GenEventParticleRange::begin(), HepMC::ConstGenEventParticleRange::begin(), HepMC::compareParticles(), filterEvent(), findPiZero(), findW(), GenEvent(), main(), massInfo(), IsGoodEvent::operator()(), IsEventGood::operator()(), IsGoodEventMyPythia::operator()(), particleTypes(), repairUnits(), simpleIter(), simpleIter2(), valid_beam_particles(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.49 particle_iterator HepMC::GenEvent::particles_begin ( ) [inline]

begin particle iteration

Definition at line 565 of file GenEvent.h.

References particle_iterator.

7.13.3.50 particle_iterator HepMC::GenEvent::particles_begin ( ) [inline]

begin particle iteration

Definition at line 565 of file 2.06.09/HepMC/GenEvent.h.

References particle_iterator.

7.13.3.51 bool HepMC::GenEvent::particles_empty ( ) const

return true if there are no particle barcodes

7.13.3.52 bool HepMC::GenEvent::particles_empty ( ) const [inline]

return true if there are no particle barcodes

Definition at line 833 of file GenEvent.h.

Referenced by delete_all_vertices(), and is_valid().

7.13.3.53 particle_const_iterator HepMC::GenEvent::particles_end ( ) const [inline]

end particle iteration

Definition at line 511 of file 2.06.09/HepMC/GenEvent.h.

References particle_const_iterator.

7.13.3.54 particle_const_iterator HepMC::GenEvent::particles_end ( ) const [inline]

end particle iteration

Examples:

example_EventSelection.cc, **example_UsingIterators.cc**, **example_VectorConversion.cc**, **fio/example_MyPythia.cc**, **testHepMCIteration.cc.in**, **testMass.cc.in**, and **testMultipleCopies.cc.in**.

Definition at line 511 of file GenEvent.h.

References particle_const_iterator.

Referenced by HepMC::compareParticles(), HepMC::GenEventParticleRange::end(), HepMC::ConstGenEventParticleRange::end(), filterEvent(), findPiZero(), findW(), GenEvent(), main(), massInfo(), IsGoodEvent::operator()(), IsEventGood::operator()(), IsGoodEventMyPythia::operator()(), particleTypes(), repairUnits(), simpleIter(), simpleIter2(), valid_beam_particles(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.55 particle_iterator HepMC::GenEvent::particles_end ( ) [inline]

end particle iteration

Definition at line 569 of file GenEvent.h.

References particle_iterator.

7.13.3.56 particle_iterator HepMC::GenEvent::particles_end ( ) [inline]

end particle iteration

Definition at line 569 of file 2.06.09/HepMC/GenEvent.h.

References particle_iterator.

7.13.3.57 int HepMC::GenEvent::particles_size ( ) const

how many particle barcodes exist?

7.13.3.58 int HepMC::GenEvent::particles_size ( ) const [inline]

how many particle barcodes exist?

Examples:

testMultipleCopies.cc.in.

Definition at line 830 of file GenEvent.h.

Referenced by HepMC::compareParticles(), main(), particleTypes(), print(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.59 PdfInfo const * HepMC::GenEvent::pdf_info ( ) const [inline]

access the **PdfInfo** (p. 233) container if it exists

Definition at line 716 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), and write().

7.13.3.60 PdfInfo const* HepMC::GenEvent::pdf_info ( ) const

access the **PdfInfo** (p. 233) container if it exists

7.13.3.61 PdfInfo * HepMC::GenEvent::pdf_info ( ) [inline]

Definition at line 719 of file GenEvent.h.

7.13.3.62 PdfInfo* HepMC::GenEvent::pdf_info ( )

7.13.3.63 void HepMC::GenEvent::print (std::ostream & *ostr* = std::cout) const

dumps to ostr

7.13.3.64 void HepMC::GenEvent::print (std::ostream & *ostr* = std::cout) const

dumps to ostr

dumps the content of this event to ostr to dump to cout use: event.print(); if you want to write this event to file outfile.txt you could use: std::ofstream outfile("outfile.txt"); event.print(outfile);

dumps the content of this event to ostr to dump to cout use: event.print(); if you want to write this event to file outfile.txt you could use: std::ofstream outfile("outfile.txt"); event.print(outfile);

Examples:

example_BuildEventFromScratch.cc, **example_VectorConversion.cc**, **fio/example_MyHerwig.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.cc**, **testFlow.cc**, **testHepMC.cc.in**, **testMultipleCopies.cc.in**, and **testPrintBug.cc**.

Definition at line 277 of file HepMC-2.06.09/src/GenEvent.cc.

References alphaQCD(), alphaQED(), HepMC::GenVertex::barcode(), beam_particles(), event_number(), event_scale(), particles_size(), HepMC::WeightContainer::print(), signal_process_id(), signal_process_vertex(), HepMC::WeightContainer::size(), vertices_begin(), vertices_end(), vertices_size(), weights(), write_cross_section(), and write_units().

Referenced by main(), writeWithCrossSection(), and writeWithWeight().

7.13.3.65 void HepMC::GenEvent::print_version (std::ostream & *ostr* = std::cout) const

dumps release version to ostr

7.13.3.66 void HepMC::GenEvent::print_version (std::ostream & *ostr* = `std::cout`) const

dumps release version to ostr

Definition at line 328 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::writeVersion().

7.13.3.67 const std::vector<long>& HepMC::GenEvent::random_states ( ) const

vector of integers containing information about the random state

7.13.3.68 const std::vector< long > & HepMC::GenEvent::random_states ( ) const [inline]

vector of integers containing information about the random state

Vector of integers which specify the random number generator's state for this event. It is left to the generator to make use of this. We envision a vector of RndmStatesTags to be included with a run class which would specify the meaning of the random_states.

Definition at line 727 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), GenEvent(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.69 std::istream & HepMC::GenEvent::read (std::istream & *is*)

read a **GenEvent** (p. 72) from streaming input

read a **GenEvent** (p. 72) from streaming input

Examples:

fio/example_PythiaStreamIO.cc, and **testStreamIO.cc.in**.

Definition at line 155 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References HepMC::GenVertex::add_particle_in(), add_vertex(), HepMC::GenParticle::barcode(), barcode_to_vertex(), clear(), HepMC::TempParticleMap::end_vertex(), HepMC::GenParticle::end_vertex(), HepMC::extascii, HepMC::detail::find_event_end(), HepMC::StreamInfo::finished_first_event(), HepMC::gen, GenVertex, HepMC::get_stream_info(), HepMC::StreamInfo::has_key(), HepMC::StreamInfo::io_momentum_unit(), HepMC::StreamInfo::io_position_unit(), HepMC::StreamInfo::io_type(), HepMC::GenCrossSection::is_set(), HepMC::PdfInfo::is_valid(), HepMC::HeavyIon::is_valid(), HepMC::TempParticleMap::order_begin(), HepMC::TempParticleMap::order_end(), p, HepMC::GenParticle::production_vertex(), HepMC::GenCrossSection::read(), HepMC::detail::read_vertex(), HepMC::StreamInfo::reading_event_header(), set_beam_particles(), set_cross_section(), HepMC::StreamInfo::set_finished_first_event(), set_heavy_ion(), set_pdf_info(), HepMC::StreamInfo::set_reading_event_header(), set_signal_process_vertex(), signal_process_vertex(), use_units(), and v.

Referenced by HepMC::operator>>(), read_from_stream4(), and readPythiaStreamIO().

7.13.3.70 std::istream& HepMC::GenEvent::read (std::istream &)

7.13.3.71 void HepMC::GenEvent::remove_barcode (GenParticle * *p*) [inline],[protected]

intended for use by **GenParticle** (p. 106)

Definition at line 774 of file GenEvent.h.

References HepMC::GenParticle::barcode().

Referenced by HepMC::GenParticle::set_end_vertex_(), HepMC::GenVertex::set_parent_event_(), HepMC::GenParticle::set_production_vertex_(), HepMC::GenParticle::~GenParticle(), and HepMC::GenVertex::~GenVertex().

7.13.3.72 void HepMC::GenEvent::remove_barcode (GenParticle * p) [protected]

intended for use by **GenParticle** (p. 106)

7.13.3.73 void HepMC::GenEvent::remove_barcode (GenVertex * v) [inline], [protected]

intended for use by **GenVertex** (p. 125)

Definition at line 777 of file GenEvent.h.

References HepMC::GenVertex::barcode().

7.13.3.74 void HepMC::GenEvent::remove_barcode (GenVertex * v) [protected]

intended for use by **GenVertex** (p. 125)

7.13.3.75 bool HepMC::GenEvent::remove_vertex (GenVertex * vtx)

erases vtx from evt

this removes vtx from the event but does NOT delete it. returns True if an entry vtx existed in the table and was erased

this removes vtx from the event but does NOT delete it. returns True if an entry vtx existed in the table and was erased

Definition at line 357 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::parent_event(), and HepMC::GenVertex::set_parent_event_().

Referenced by add_vertex().

7.13.3.76 bool HepMC::GenEvent::remove_vertex (GenVertex * vtx)

erases vtx from evt

7.13.3.77 void HepMC::GenEvent::set_alphaQCD (double a) [inline]

set QCD coupling

Definition at line 743 of file GenEvent.h.

7.13.3.78 void HepMC::GenEvent::set_alphaQCD (double a)

set QCD coupling

7.13.3.79 void HepMC::GenEvent::set_alphaQED (double a) [inline]

set QED coupling

Definition at line 745 of file GenEvent.h.

7.13.3.80 void HepMC::GenEvent::set_alphaQED (double a)

set QED coupling

7.13.3.81 bool HepMC::GenEvent::set_barcode (GenParticle * p, int suggested_barcode = false) [protected]

set the barcode - intended for use by **GenParticle** (p. 106)

7.13.3.82 bool HepMC::GenEvent::set_barcode (GenParticle * p, int suggested_barcode = false) [protected]

set the barcode - intended for use by **GenParticle** (p. 106)

Definition at line 430 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::GenParticle::barcode(), p, HepMC::GenParticle::parent_event(), and HepMC::GenParticle::set_barcode_().

Referenced by HepMC::GenParticle::set_end_vertex_(), HepMC::GenVertex::set_parent_event_(), HepMC::GenParticle::set_production_vertex_(), HepMC::GenVertex::suggest_barcode(), and HepMC::GenParticle::suggest_barcode().

7.13.3.83 bool HepMC::GenEvent::set_barcode (GenVertex * v, int suggested_barcode = false) [protected]

set the barcode - intended for use by **GenVertex** (p. 125)

Definition at line 501 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::GenVertex::barcode(), HepMC::GenVertex::parent_event(), HepMC::GenVertex::set_barcode_(), and v.

7.13.3.84 bool HepMC::GenEvent::set_barcode (GenVertex * v, int suggested_barcode = false) [protected]

set the barcode - intended for use by **GenVertex** (p. 125)

7.13.3.85 bool HepMC::GenEvent::set_beam_particles (GenParticle * bp1, GenParticle * bp2)

set incoming beam particles

construct the beam particle information using pointers to **GenParticle** (p. 106) returns false if either GenParticle* is null

Definition at line 586 of file HepMC-2.06.09/src/GenEvent.cc.

Referenced by HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_HERWIG::fill_next_event(), GenEvent(), read(), and set_beam_particles().

7.13.3.86 bool HepMC::GenEvent::set_beam_particles (GenParticle * , GenParticle *)

set incoming beam particles

7.13.3.87 bool HepMC::GenEvent::set_beam_particles (std::pair< HepMC::GenParticle *, HepMC::GenParticle * > const &)

use a pair of GenParticle*'s to set incoming beam particles

7.13.3.88 bool HepMC::GenEvent::set_beam_particles (std::pair< HepMC::GenParticle *, HepMC::GenParticle * > const & bp)

use a pair of GenParticle*'s to set incoming beam particles

construct the beam particle information using a std::pair of pointers to **GenParticle** (p. 106) returns false if either GenParticle* is null

Definition at line 595 of file HepMC-2.06.09/src/GenEvent.cc.

References set_beam_particles().

7.13.3.89 void HepMC::GenEvent::set_cross_section (const GenCrossSection & *xs*) [inline]

provide a pointer to the **GenCrossSection** (p. 67) container

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.cc**, and **testHepMC.cc.in**.

Definition at line 752 of file GenEvent.h.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), read(), writePythiaStreamIO(), and writeWithCrossSection().

7.13.3.90 void HepMC::GenEvent::set_cross_section (const GenCrossSection &)

provide a pointer to the **GenCrossSection** (p. 67) container

7.13.3.91 void HepMC::GenEvent::set_event_number (int *eventno*) [inline]

set event number

Examples:

fio/example_MyHerwig.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testHerwigCopies.cc**.

Definition at line 733 of file GenEvent.h.

Referenced by HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_HERWIG::fill_next_event(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

7.13.3.92 void HepMC::GenEvent::set_event_number (int *eventno*)

set event number

7.13.3.93 void HepMC::GenEvent::set_event_scale (double *scale*) [inline]

set energy scale

Definition at line 741 of file GenEvent.h.

7.13.3.94 void HepMC::GenEvent::set_event_scale (double *scale*)

set energy scale

7.13.3.95 void HepMC::GenEvent::set_heavy_ion (const HeavyIon & *ion*) [inline]

provide a pointer to the **HeavyIon** (p. 146) container

Examples:

testMass.cc.in.

Definition at line 758 of file GenEvent.h.

Referenced by main(), and read().

7.13.3.96 void HepMC::GenEvent::set_heavy_ion (const HeavyIon & *ion*)

provide a pointer to the **HeavyIon** (p. 146) container

7.13.3.97 void HepMC::GenEvent::set_mpi (int *nmpi*) [inline]

set number of multi parton interactions

Use this to set the number of multi parton interactions in each event.

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 737 of file GenEvent.h.

Referenced by event_selection(), main(), pythia_out(), and writePythiaStreamIO().

7.13.3.98 void HepMC::GenEvent::set_mpi (int)

set number of multi parton interactions

7.13.3.99 void HepMC::GenEvent::set_pdf_info (const PdfInfo & *p*) [inline]

provide a pointer to the **PdfInfo** (p. 233) container

Examples:

testMass.cc.in.

Definition at line 764 of file GenEvent.h.

Referenced by main(), and read().

7.13.3.100 void HepMC::GenEvent::set_pdf_info (const PdfInfo & *p*)

provide a pointer to the **PdfInfo** (p. 233) container

7.13.3.101 void HepMC::GenEvent::set_random_states (const std::vector< long > & *randomstates*) [inline]

provide random state information

Definition at line 770 of file GenEvent.h.

Referenced by GenEvent().

7.13.3.102 void HepMC::GenEvent::set_random_states (const std::vector< long > & *randomstates*)

provide random state information

7.13.3.103 void HepMC::GenEvent::set_signal_process_id (int *id*) [inline]

set unique signal process id

Examples:

fio/example_MyHerwig.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testHerwigCopies.cc**.

Definition at line 730 of file GenEvent.h.

Referenced by main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

7.13.3.104 void HepMC::GenEvent::set_signal_process_id (int *id*)

set unique signal process id

7.13.3.105 void HepMC::GenEvent::set_signal_process_vertex (GenVertex * *vtx*) [inline]

set pointer to the vertex containing the signal process

Examples:

example_BuildEventFromScratch.cc, **example_VectorConversion.cc**, and **testFlow.cc**.

Definition at line 747 of file GenEvent.h.

References add_vertex().

Referenced by HepMC::IO_HERWIG::fill_next_event(), GenEvent(), main(), and read().

7.13.3.106 void HepMC::GenEvent::set_signal_process_vertex (GenVertex *)

set pointer to the vertex containing the signal process

7.13.3.107 int HepMC::GenEvent::signal_process_id ( ) const

unique signal process id

7.13.3.108 int HepMC::GenEvent::signal_process_id ( ) const [inline]

unique signal process id

The integer ID that uniquely specifies this signal process, i.e. MSUB in Pythia. It is necessary to package this with each event rather than with the run because many processes may be generated within one run.

Definition at line 679 of file GenEvent.h.

Referenced by HepMC::compareGenEvent(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.109 GenVertex* HepMC::GenEvent::signal_process_vertex ( ) const

pointer to the vertex containing the signal process

7.13.3.110 GenVertex * HepMC::GenEvent::signal_process_vertex ( ) const [inline]

pointer to the vertex containing the signal process

returns a (mutable) pointer to the signal process vertex

returns a (mutable) pointer to the signal process vertex

Definition at line 694 of file GenEvent.h.

Referenced by HepMC::compareSignalProcessVertex(), GenEvent(), print(), read(), write(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.111 void HepMC::GenEvent::swap (GenEvent & *other*)

swap

Definition at line 226 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::WeightContainer::swap(), vertices_begin(), and vertices_end().

Referenced by operator=().

7.13.3.112 void HepMC::GenEvent::swap (GenEvent & *other*)

swap

7.13.3.113 void HepMC::GenEvent::use_units (Units::MomentumUnit *new_m*, Units::LengthUnit *new_l*) [inline]

set the units using enums This method will convert momentum and position data if necessary

Examples:

example_BuildEventFromScratch.cc, **example_MyPythiaOnlyToHepMC.cc**, **example_VectorConversion.cc**, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.cc**, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 856 of file GenEvent.h.

Referenced by HepMC::convert_units(), event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), read(), and writePythiaStreamIO().

7.13.3.114 void HepMC::GenEvent::use_units (Units::MomentumUnit , Units::LengthUnit)

set the units using enums This method will convert momentum and position data if necessary

7.13.3.115 void HepMC::GenEvent::use_units (std::string & *new_m*, std::string & *new_l*) [inline]

set the units using strings the string must match the enum exactly This method will convert momentum and position data if necessary

Definition at line 861 of file GenEvent.h.

7.13.3.116 void HepMC::GenEvent::use_units (std::string & , std::string &)

set the units using strings the string must match the enum exactly This method will convert momentum and position data if necessary

7.13.3.117 bool HepMC::GenEvent::valid_beam_particles ( ) const

test to see if we have two valid beam particles

7.13.3.118 bool HepMC::GenEvent::valid_beam_particles ( ) const

test to see if we have two valid beam particles

Examples:

testMass.cc.in.

Definition at line 568 of file HepMC-2.06.09/src/GenEvent.cc.

References p, particles_begin(), and particles_end().

Referenced by main().

7.13.3.119 GenEventVertexRange HepMC::GenEvent::vertex_range ( )

vertex range

Examples:

testHepMCIteration.cc.in.

Definition at line 16 of file HepMC-2.06.09/src/GenRanges.cc.

Referenced by simpleIter4().

7.13.3.120 GenEventVertexRange HepMC::GenEvent::vertex_range ( )

vertex range

7.13.3.121 ConstGenEventVertexRange HepMC::GenEvent::vertex_range ( ) const

vertex range

Definition at line 21 of file HepMC-2.06.09/src/GenRanges.cc.

7.13.3.122 ConstGenEventVertexRange HepMC::GenEvent::vertex_range ( ) const

vertex range

7.13.3.123 vertex_const_iterator HepMC::GenEvent::vertices_begin ( ) const [inline]

begin vertex iteration

Definition at line 377 of file 2.06.09/HepMC/GenEvent.h.

References vertex_const_iterator.

7.13.3.124 vertex_const_iterator HepMC::GenEvent::vertices_begin ( ) const [inline]

begin vertex iteration

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 377 of file GenEvent.h.

References vertex_const_iterator.

Referenced by HepMC::GenEventVertexRange::begin(), HepMC::ConstGenEventVertexRange::begin(), HepMC::compareVertices(), filterEvent(), GenEvent(), main(), print(), simpleIter(), simpleIter2(), swap(), write(), and HepMC::IO_HEPEVT::write_event().

7.13.3.125 vertex_iterator HepMC::GenEvent::vertices_begin ( ) [inline]

begin vertex iteration

Definition at line 440 of file GenEvent.h.

References vertex_iterator.

7.13.3.126 vertex_iterator HepMC::GenEvent::vertices_begin ( ) [inline]

begin vertex iteration

Definition at line 440 of file 2.06.09/HepMC/GenEvent.h.

References vertex_iterator.

7.13.3.127 bool HepMC::GenEvent::vertices_empty ( ) const [inline]

return true if there are no vertex barcodes

Definition at line 839 of file GenEvent.h.

Referenced by delete_all_vertices(), and is_valid().

7.13.3.128 bool HepMC::GenEvent::vertices_empty ( ) const

return true if there are no vertex barcodes

7.13.3.129 vertex_const_iterator HepMC::GenEvent::vertices_end ( ) const [inline]

end vertex iteration

Definition at line 381 of file 2.06.09/HepMC/GenEvent.h.

References vertex_const_iterator.

7.13.3.130 vertex_const_iterator HepMC::GenEvent::vertices_end ( ) const [inline]

end vertex iteration

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 381 of file GenEvent.h.

References vertex_const_iterator.

Referenced by HepMC::compareVertices(), HepMC::GenEventVertexRange::end(), HepMC::ConstGenEventVertexRange::end(), filterEvent(), GenEvent(), main(), print(), simpleIter(), simpleIter2(), swap(), write(), and HepMC::IO_HEPEVT::write_event().

7.13.3.131 vertex_iterator HepMC::GenEvent::vertices_end ( ) [inline]

end vertex iteration

Definition at line 444 of file GenEvent.h.

References vertex_iterator.

7.13.3.132 vertex_iterator HepMC::GenEvent::vertices_end ( ) [inline]

end vertex iteration

Definition at line 444 of file 2.06.09/HepMC/GenEvent.h.

References vertex_iterator.

7.13.3.133 int HepMC::GenEvent::vertices_size ( ) const [inline]

how many vertex barcodes exist?

Examples:

testMultipleCopies.cc.in.

Definition at line 836 of file GenEvent.h.

Referenced by HepMC::compareVertices(), main(), print(), write(), and HepMC::IO_AsciiParticles::write_event().

7.13.3.134 int HepMC::GenEvent::vertices_size ( ) const

how many vertex barcodes exist?

7.13.3.135 WeightContainer & HepMC::GenEvent::weights ( ) [inline]

direct access to **WeightContainer** (p. 292)

direct access to the weights container is allowed. Thus you can use myevt.weights()[2]; to access element 2 of the weights. or use myevt.weights().push_back(mywgt); to add an element. and you can set the weights with myevt.weights() = myvector;

Examples:

fio/testPythiaCopies.cc, **testHepMC.cc.in**, and **testMass.cc.in**.

Definition at line 699 of file GenEvent.h.

Referenced by HepMC::compareWeights(), GenEvent(), main(), print(), write(), HepMC::IO_AsciiParticles::write_event(), writeWithCrossSection(), and writeWithWeight().

7.13.3.136 WeightContainer& HepMC::GenEvent::weights ( )

direct access to **WeightContainer** (p. 292)

direct access to the weights container is allowed. Thus you can use myevt.weights()[2]; to access element 2 of the weights. or use myevt.weights().push_back(mywgt); to add an element. and you can set the weights with myevt.weights() = myvector;

7.13.3.137 const WeightContainer& HepMC::GenEvent::weights ( ) const

direct access to **WeightContainer** (p. 292)

7.13.3.138 const WeightContainer & HepMC::GenEvent::weights ( ) const [inline]

direct access to **WeightContainer** (p. 292)

Definition at line 701 of file GenEvent.h.

7.13.3.139 std::ostream & HepMC::GenEvent::write (std::ostream & *os*)

Writes evt to an output stream.

Writes evt to an output stream.

Examples:

fio/example_PythiaStreamIO.cc, **testFlow.cc**, and **testStreamIO.cc.in**.

Definition at line 72 of file HepMC-2.06.09/src/GenEventStreamIO.cc.

References alphaQCD(), alphaQED(), beam_particles(), event_number(), event_scale(), HepMC::StreamInfo-::finished_first_event(), HepMC::get_stream_info(), heavy_ion(), length_unit(), momentum_unit(), mpi(), HepMC::-Units::name(), HepMC::detail::output(), pdf_info(), HepMC::StreamInfo::set_finished_first_event(), signal_process_-id(), signal_process_vertex(), HepMC::WeightContainer::size(), v, vertices_begin(), vertices_end(), vertices_size(), weights(), and HepMC::GenCrossSection::write().

Referenced by main(), HepMC::operator<<(), read_from_stream4(), readPythiaStreamIO(), and write_to_-stream3().

7.13.3.140 std::ostream& HepMC::GenEvent::write (std::ostream &)

7.13.3.141 void HepMC::GenEvent::write_cross_section (std::ostream & *ostr* = `std::cout`) const

If the cross section is defined, write the cross section information to an output stream. If the output stream is not defined, use std::cout.

7.13.3.142 void HepMC::GenEvent::write_cross_section (std::ostream & *ostr* = `std::cout`) const

If the cross section is defined, write the cross section information to an output stream. If the output stream is not defined, use std::cout.

Examples:

testHepMC.cc.in.

Definition at line 605 of file HepMC-2.06.09/src/GenEvent.cc.

References HepMC::GenCrossSection::cross_section(), cross_section(), and HepMC::GenCrossSection::cross_-section_error().

Referenced by print(), and writeWithCrossSection().

7.13.3.143 void HepMC::GenEvent::write_units (std::ostream & *os* = `std::cout`) const

Write the unit information to an output stream. If the output stream is not defined, use std::cout.

Examples:

testHepMC.cc.in, and **testStreamIO.cc.in**.

Definition at line 599 of file HepMC-2.06.09/src/GenEvent.cc.

References length_unit(), momentum_unit(), and HepMC::Units::name().

Referenced by print(), read_testIOGenEvent(), read_testUnits(), and read_variousFormats().

7.13.3.144 void HepMC::GenEvent::write_units (std::ostream & *os* = `std::cout`) const

Write the unit information to an output stream. If the output stream is not defined, use std::cout.

7.13.4 Friends And Related Function Documentation

7.13.4.1 GenParticle [friend]

Definition at line 156 of file GenEvent.h.

Referenced by GenEvent().

7.13.4.2 GenVertex [friend]

Definition at line 157 of file GenEvent.h.

Referenced by GenEvent(), and read().

7.13.4.3 particle_const_iterator [friend]

Definition at line 505 of file GenEvent.h.

Referenced by HepMC::GenEvent::particle_iterator::operator particle_const_iterator(), particles_begin(), and particles_end().

7.13.4.4 particle_iterator [friend]

Definition at line 563 of file GenEvent.h.

Referenced by particles_begin(), and particles_end().

7.13.4.5 vertex_const_iterator [friend]

Definition at line 375 of file GenEvent.h.

Referenced by HepMC::GenEvent::vertex_iterator::operator vertex_const_iterator(), vertices_begin(), and vertices_end().

7.13.4.6 vertex_iterator [friend]

Definition at line 438 of file GenEvent.h.

Referenced by vertices_begin(), and vertices_end().

The documentation for this class was generated from the following files:

- **GenEvent.h**
- **2.06.09/HepMC/GenEvent.h**

- **HepMC-2.06.09/src/GenEvent.cc**
- **HepMC-2.06.09/src/GenEventStreamIO.cc**
- **HepMC-2.06.09/src/GenRanges.cc**
- **src/GenEvent.cc**
- **src/GenEventStreamIO.cc**
- **src/GenRanges.cc**

7.14 HepMC::GenEventParticleRange Class Reference

GenEventParticleRange (p. 103) acts like a collection of particles.

`#include <GenRanges.h>`

Public Member Functions

- **GenEventParticleRange** (**GenEvent** &e)

  *the constructor requires a **GenEvent*** (p. *72*)
- **GenEvent::particle_iterator begin** ()
- **GenEvent::particle_iterator end** ()
- **GenEventParticleRange** (**GenEvent** &e)

  *the constructor requires a **GenEvent*** (p. *72*)
- **GenEvent::particle_iterator begin** ()
- **GenEvent::particle_iterator end** ()

7.14.1 Detailed Description

GenEventParticleRange (p. 103) acts like a collection of particles.

HepMC::GenEventParticleRange (p. 103) is used to mimic a collection of particles for ease of use - especially with utilities such as the Boost foreach funtion

Examples:

testHepMCIteration.cc.in.

Definition at line 83 of file GenRanges.h.

7.14.2 Constructor & Destructor Documentation

7.14.2.1 HepMC::GenEventParticleRange::GenEventParticleRange (GenEvent & *e*) [inline]

the constructor requires a **GenEvent** (p. 72)

Definition at line 88 of file GenRanges.h.

7.14.2.2 HepMC::GenEventParticleRange::GenEventParticleRange (GenEvent & *e*) [inline]

the constructor requires a **GenEvent** (p. 72)

Definition at line 88 of file 2.06.09/HepMC/GenRanges.h.

7.14.3 Member Function Documentation

7.14.3.1 GenEvent::particle_iterator HepMC::GenEventParticleRange::begin ( ) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 90 of file GenRanges.h.

References HepMC::GenEvent::particles_begin().

Referenced by simpleIter3(), and simpleIter4().

7.14.3.2 GenEvent::particle_iterator HepMC::GenEventParticleRange::begin ( ) [inline]

Definition at line 90 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::particles_begin().

7.14.3.3 GenEvent::particle_iterator HepMC::GenEventParticleRange::end ( ) [inline]

Definition at line 91 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::particles_end().

7.14.3.4 GenEvent::particle_iterator HepMC::GenEventParticleRange::end ( ) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 91 of file GenRanges.h.

References HepMC::GenEvent::particles_end().

Referenced by simpleIter3(), and simpleIter4().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.15 HepMC::GenEventVertexRange Class Reference

GenEventVertexRange (p. 104) acts like a collection of vertices.

`#include <GenRanges.h>`

Public Member Functions

- **GenEventVertexRange** (**GenEvent** &e)
  *the constructor requires a **GenEvent*** (p. *72*)
- **GenEvent::vertex_iterator begin** ()
- **GenEvent::vertex_iterator end** ()
- **GenEventVertexRange** (**GenEvent** &e)
  *the constructor requires a **GenEvent*** (p. *72*)
- **GenEvent::vertex_iterator begin** ()
- **GenEvent::vertex_iterator end** ()

7.15.1 Detailed Description

GenEventVertexRange (p. 104) acts like a collection of vertices.

HepMC::GenEventVertexRange (p. 104) is used to mimic a collection of vertices for ease of use - especially with utilities such as the Boost foreach funtion

Examples:

testHepMCIteration.cc.in.

Definition at line 26 of file GenRanges.h.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 HepMC::GenEventVertexRange::GenEventVertexRange (GenEvent & *e*) [inline]

the constructor requires a **GenEvent** (p. 72)

Definition at line 31 of file GenRanges.h.

7.15.2.2 HepMC::GenEventVertexRange::GenEventVertexRange (GenEvent & *e*) [inline]

the constructor requires a **GenEvent** (p. 72)

Definition at line 31 of file 2.06.09/HepMC/GenRanges.h.

7.15.3 Member Function Documentation

7.15.3.1 GenEvent::vertex_iterator HepMC::GenEventVertexRange::begin ( ) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 33 of file GenRanges.h.

References HepMC::GenEvent::vertices_begin().

Referenced by simpleIter3(), and simpleIter4().

7.15.3.2 GenEvent::vertex_iterator HepMC::GenEventVertexRange::begin ( ) [inline]

Definition at line 33 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::vertices_begin().

7.15.3.3 GenEvent::vertex_iterator HepMC::GenEventVertexRange::end ( ) [inline]

Definition at line 34 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenEvent::vertices_end().

7.15.3.4 GenEvent::vertex_iterator HepMC::GenEventVertexRange::end ( ) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 34 of file GenRanges.h.

References HepMC::GenEvent::vertices_end().

Referenced by simpleIter3(), and simpleIter4().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.16 HepMC::GenParticle Class Reference

The **GenParticle** (p. 106) class contains information about generated particles.

`#include <GenParticle.h>`

Public Member Functions

- **GenParticle** (void)
  default constructor
- **GenParticle** (const **FourVector** &**momentum**, int **pdg_id**, int **status**=0, const **Flow** &itsflow=**Flow**(), const **Polarization** &polar=**Polarization**(0, 0))
  constructor requires momentum and particle ID
- **GenParticle** (const **GenParticle** &inparticle)
  shallow copy.
- virtual ∼**GenParticle** ()
- void **swap** (**GenParticle** &other)
  swap
- **GenParticle** & **operator=** (const **GenParticle** &inparticle)
- bool **operator==** (const **GenParticle** &) const
  check for equality
- bool **operator!=** (const **GenParticle** &) const
  check for inequality
- void **print** (std::ostream &ostr=std::cout) const
  dump this particle's full info to ostr
- **operator HepMC::FourVector** () const
  conversion operator
- const **FourVector** & **momentum** () const
  standard 4 momentum
- int **pdg_id** () const
  particle ID
- int **status** () const
  HEPEVT decay status.
- const **Flow** & **flow** () const
  particle flow
- int **flow** (int code_index) const
  particle flow index
- const **Polarization** & **polarization** () const
  polarization information
- **GenVertex** ∗ **production_vertex** () const
  pointer to the production vertex
- **GenVertex** ∗ **end_vertex** () const

pointer to the decay vertex

- **GenEvent** ∗ **parent_event** () const

  pointer to the event that owns this particle

- double **generated_mass** () const

  mass as generated

- double **generatedMass** () const

  generatedMass() *(*p. *113) is included for backwards compatibility with* ***CLHEP*** *(*p. *15)* ***HepMC*** *(*p. *15)*

- int **barcode** () const

  particle barcode

- bool **is_undecayed** () const

  Convenience method. Returns true if status==1.

- bool **has_decayed** () const

  Convenience method. Returns true if status==2.

- bool **is_beam** () const
- **GenParticleProductionRange particles_in** (**IteratorRange** range=**relatives**)

  incoming particle range

- **ConstGenParticleProductionRange particles_in** (**IteratorRange** range=**relatives**) const

  incoming particle range

- **GenParticleEndRange particles_out** (**IteratorRange** range=**relatives**)

  outgoing particle range

- **ConstGenParticleEndRange particles_out** (**IteratorRange** range=**relatives**) const

  outgoing particle range

- bool **suggest_barcode** (int the_bar_code)

  In general there is no reason to "suggest_barcode".

- void **set_momentum** (const **FourVector** &vec4)

  set standard 4 momentum

- void **set_pdg_id** (int id)

  set particle ID

- void **set_status** (int **status**=0)

  set decay status

- void **set_flow** (const **Flow** &f)

  set particle flow

- void **set_flow** (int code_index, int code=0)
- void **set_polarization** (const **Polarization** &pol=**Polarization**(0, 0))

  set polarization

- void **set_generated_mass** (const double &m)

  define the actual generated mass

- void **setGeneratedMass** (const double &m)

  setGeneratedMass() *(*p. *120) is included for backwards compatibility with* ***CLHEP*** *(*p. *15)* ***HepMC*** *(*p. *15)*

- **GenParticle** (void)

  default constructor

- **GenParticle** (const **FourVector** &**momentum**, int **pdg_id**, int **status**=0, const **Flow** &itsflow=**Flow**(), const **Polarization** &polar=**Polarization**(0, 0))

  constructor requires momentum and particle ID

- **GenParticle** (const **GenParticle** &inparticle)

  shallow copy.

- virtual ∼**GenParticle** ()
- void **swap** (**GenParticle** &other)

  swap

- **GenParticle** & **operator=** (const **GenParticle** &inparticle)
- bool **operator==** (const **GenParticle** &) const

check for equality

- bool **operator!=** (const **GenParticle** &) const

  check for inequality

- void **print** (std::ostream &ostr=std::cout) const

  dump this particle's full info to ostr

- **operator HepMC::FourVector** () const

  conversion operator

- const **FourVector** & **momentum** () const

  standard 4 momentum

- int **pdg_id** () const

  particle ID

- int **status** () const

  HEPEVT decay status.

- const **Flow** & **flow** () const

  particle flow

- int **flow** (int code_index) const

  particle flow index

- const **Polarization** & **polarization** () const

  polarization information

- **GenVertex** ∗ **production_vertex** () const

  pointer to the production vertex

- **GenVertex** ∗ **end_vertex** () const

  pointer to the decay vertex

- **GenEvent** ∗ **parent_event** () const

  pointer to the event that owns this particle

- double **generated_mass** () const

  mass as generated

- double **generatedMass** () const

  generatedMass() *(*p. *113) is included for backwards compatibility with* ***CLHEP*** *(*p. *15)* ***HepMC*** *(*p. *15)*

- int **barcode** () const

  particle barcode

- bool **is_undecayed** () const

  Convenience method. Returns true if status==1.

- bool **has_decayed** () const

  Convenience method. Returns true if status==2.

- bool **is_beam** () const
- **GenParticleProductionRange particles_in** (**IteratorRange** range=**relatives**)

  incoming particle range

- **ConstGenParticleProductionRange particles_in** (**IteratorRange** range=**relatives**) const

  incoming particle range

- **GenParticleEndRange particles_out** (**IteratorRange** range=**relatives**)

  outgoing particle range

- **ConstGenParticleEndRange particles_out** (**IteratorRange** range=**relatives**) const

  outgoing particle range

- bool **suggest_barcode** (int the_bar_code)

  In general there is no reason to "suggest_barcode".

- void **set_momentum** (const **FourVector** &vec4)

  set standard 4 momentum

- void **set_pdg_id** (int id)

  set particle ID

- void **set_status** (int **status**=0)

set decay status

- void **set_flow** (const **Flow** &f)

  set particle flow
- void **set_flow** (int code_index, int code=0)
- void **set_polarization** (const **Polarization** &pol=**Polarization**(0, 0))

  set polarization
- void **set_generated_mass** (const double &m)

  define the actual generated mass
- void **setGeneratedMass** (const double &m)

  setGeneratedMass() *(p. 120) is included for backwards compatibility with* ***CLHEP*** *(p. 15)* ***HepMC*** *(p. 15)*

Protected Member Functions

- void **set_production_vertex_** (**GenVertex** ∗productionvertex=0)

  set production vertex - for internal use only
- void **set_end_vertex_** (**GenVertex** ∗decayvertex=0)

  set decay vertex - for internal use only
- void **set_barcode_** (int the_bar_code)

  for use by ***GenEvent*** *(p. 72) only*
- void **convert_momentum** (const double &)
- void **set_production_vertex_** (**GenVertex** ∗productionvertex=0)

  set production vertex - for internal use only
- void **set_end_vertex_** (**GenVertex** ∗decayvertex=0)

  set decay vertex - for internal use only
- void **set_barcode_** (int the_bar_code)

  for use by ***GenEvent*** *(p. 72) only*
- void **convert_momentum** (const double &)

Friends

- class **GenVertex**
- class **GenEvent**
- std::ostream & **operator**<< (std::ostream &, const **GenParticle** &)

  print particle
- std::ostream & **operator**<< (std::ostream &, const **GenParticle** &)

  print particle

7.16.1 Detailed Description

The **GenParticle** (p. 106) class contains information about generated particles.

HepMC::GenParticle (p. 106) contains momentum, generated mass, particle ID, decay status, flow, polarization, pointers to production and decay vertices and a unique barcode identfier.

Examples:

example_BuildEventFromScratch.cc, **example_UsingIterators.cc**, **example_VectorConversion.cc**, **testFlow.cc**, **testHepMCIteration.cc.in**, **testMass.cc.in**, and **testPrintBug.cc**.

Definition at line 60 of file GenParticle.h.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 HepMC::GenParticle::GenParticle (void)

default constructor

Definition at line 14 of file HepMC-2.06.09/src/GenParticle.cc.

7.16.2.2 HepMC::GenParticle::GenParticle (const FourVector & *momentum,* int *pdg_id,* int *status* = 0, const Flow & *itsflow* = Flow (), const Polarization & *polar* = Polarization (0,0))

constructor requires momentum and particle ID

Definition at line 23 of file HepMC-2.06.09/src/GenParticle.cc.

References set_flow().

7.16.2.3 HepMC::GenParticle::GenParticle (const GenParticle & *inparticle*)

shallow copy.

Shallow copy: does not copy the vertex pointers (note - impossible to copy vertex pointers which having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Shallow copy: does not copy the vertex pointers (note - impossible to copy vertex pointers which having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 37 of file HepMC-2.06.09/src/GenParticle.cc.

References barcode(), set_end_vertex_(), set_production_vertex_(), and suggest_barcode().

7.16.2.4 HepMC::GenParticle::~GenParticle ( ) [virtual]

Definition at line 58 of file HepMC-2.06.09/src/GenParticle.cc.

References parent_event(), and HepMC::GenEvent::remove_barcode().

7.16.2.5 HepMC::GenParticle::GenParticle (void)

default constructor

7.16.2.6 HepMC::GenParticle::GenParticle (const FourVector & *momentum,* int *pdg_id,* int *status* = 0, const Flow & *itsflow* = Flow (), const Polarization & *polar* = Polarization (0, 0))

constructor requires momentum and particle ID

7.16.2.7 HepMC::GenParticle::GenParticle (const GenParticle & *inparticle*)

shallow copy.

7.16.2.8 virtual HepMC::GenParticle::~GenParticle ( ) [virtual]

7.16.3 Member Function Documentation

7.16.3.1 int HepMC::GenParticle::barcode ( ) const [inline]

particle barcode

The barcode is the particle's reference number, every vertex in the event has a unique barcode. Particle barcodes are positive numbers, vertex barcodes are negative numbers.

Please note that the barcodes are intended for internal use within **HepMC** (p. 15) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Examples:

testFlow.cc.

Definition at line 252 of file GenParticle.h.

Referenced by HepMC::TempParticleMap::addEndParticle(), GenParticle(), main(), PrintChildren::operator()(), HepMC::operator<<(), print(), HepMC::GenEvent::read(), HepMC::GenEvent::remove_barcode(), HepMC::GenEvent::set_barcode(), set_end_vertex_(), and set_production_vertex_().

7.16.3.2 int HepMC::GenParticle::barcode ( ) const

particle barcode

The barcode is the particle's reference number, every vertex in the event has a unique barcode. Particle barcodes are positive numbers, vertex barcodes are negative numbers.

Please note that the barcodes are intended for internal use within **HepMC** (p. 15) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

7.16.3.3 void HepMC::GenParticle::convert_momentum (const double & *f*) [protected]

scale the momentum vector and generated mass this method is only for use by **GenEvent** (p. 72)

Definition at line 246 of file HepMC-2.06.09/src/GenParticle.cc.

References HepMC::FourVector::e(), HepMC::FourVector::px(), HepMC::FourVector::py(), and HepMC::FourVector::pz().

7.16.3.4 void HepMC::GenParticle::convert_momentum (const double &) [protected]

scale the momentum vector and generated mass this method is only for use by **GenEvent** (p. 72)

7.16.3.5 GenVertex * HepMC::GenParticle::end_vertex ( ) const [inline]

pointer to the decay vertex

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 221 of file GenParticle.h.

Referenced by HepMC::GenVertex::add_particle_in(), HepMC::GenParticleEndRange::begin(), HepMC::ConstGenParticleEndRange::begin(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::GenParticleEndRange::end(), HepMC::ConstGenParticleEndRange::end(), filterEvent(), HepMC::GenEvent::GenEvent(), IsFinalState::operator()(), PrintW::operator()(), IsStateFinal::operator()(), PrintConstW::operator()(), HepMC::operator<<(), parent_event(), print(), HepMC::GenEvent::read(), and HepMC::GenVertex::remove_particle().

7.16.3.6 GenVertex* HepMC::GenParticle::end_vertex ( ) const

pointer to the decay vertex

7.16.3.7 const Flow & HepMC::GenParticle::flow ( ) const [inline]

particle flow

Examples:

testFlow.cc.

Definition at line 223 of file GenParticle.h.

Referenced by main().

7.16.3.8 const Flow& HepMC::GenParticle::flow ( ) const

particle flow

7.16.3.9 int HepMC::GenParticle::flow (int *code_index*) const [inline]

particle flow index

Definition at line 225 of file GenParticle.h.

References HepMC::Flow::icode().

7.16.3.10 int HepMC::GenParticle::flow (int *code_index*) const

particle flow index

7.16.3.11 double HepMC::GenParticle::generated_mass ( ) const

mass as generated

Because of precision issues, the generated mass is not always the same as the mass calculated from the momentum 4 vector. If the generated mass has been set, then **generated_mass()** (p. 112) returns that value. If the generated mass has not been set, then **generated_mass()** (p. 112) returns the mass calculated from the momentum 4 vector.

Definition at line 236 of file HepMC-2.06.09/src/GenParticle.cc.

Referenced by generatedMass(), and operator==().

7.16.3.12 double HepMC::GenParticle::generated_mass ( ) const

mass as generated

Because of precision issues, the generated mass is not always the same as the mass calculated from the momentum 4 vector. If the generated mass has been set, then **generated_mass()** (p. 112) returns that value. If the generated mass has not been set, then **generated_mass()** (p. 112) returns the mass calculated from the momentum 4 vector.

7.16.3.13 double HepMC::GenParticle::generatedMass ( ) const [inline]

generatedMass() (p. 113) is included for backwards compatibility with **CLHEP** (p. 15) **HepMC** (p. 15)

Definition at line 121 of file GenParticle.h.

References generated_mass().

7.16.3.14 double HepMC::GenParticle::generatedMass ( ) const [inline]

generatedMass() (p. 113) is included for backwards compatibility with **CLHEP** (p. 15) **HepMC** (p. 15)

Definition at line 121 of file 2.06.09/HepMC/GenParticle.h.

References generated_mass().

7.16.3.15 bool HepMC::GenParticle::has_decayed ( ) const [inline]

Convenience method. Returns true if status==2.

Definition at line 259 of file GenParticle.h.

7.16.3.16 bool HepMC::GenParticle::has_decayed ( ) const

Convenience method. Returns true if status==2.

7.16.3.17 bool HepMC::GenParticle::is_beam ( ) const [inline]

Convenience method. Returns true if status==4 Note that using status 4 for beam particles is a new convention which may not have been implemented by the code originating this **GenEvent** (p. 72).

Definition at line 262 of file GenParticle.h.

7.16.3.18 bool HepMC::GenParticle::is_beam ( ) const

Convenience method. Returns true if status==4 Note that using status 4 for beam particles is a new convention which may not have been implemented by the code originating this **GenEvent** (p. 72).

7.16.3.19 bool HepMC::GenParticle::is_undecayed ( ) const [inline]

Convenience method. Returns true if status==1.

Definition at line 256 of file GenParticle.h.

7.16.3.20 bool HepMC::GenParticle::is_undecayed ( ) const

Convenience method. Returns true if status==1.

7.16.3.21 const FourVector & HepMC::GenParticle::momentum ( ) const [inline]

standard 4 momentum

Examples:

example_UsingIterators.cc, and **testMass.cc.in**.

Definition at line 211 of file GenParticle.h.

Referenced by IsPhoton(), main(), IsPhoton::operator()(), HepMC::operator<<(), operator==(), print(), and HepMC::detail::read_particle().

7.16.3.22 const FourVector& HepMC::GenParticle::momentum ( ) const

standard 4 momentum

7.16.3.23 HepMC::GenParticle::operator HepMC::FourVector ( ) const

conversion operator

7.16.3.24 HepMC::GenParticle::operator HepMC::FourVector ( ) const [inline]

conversion operator

Definition at line 208 of file GenParticle.h.

7.16.3.25 bool HepMC::GenParticle::operator!= (const GenParticle &) const

check for inequality

7.16.3.26 bool HepMC::GenParticle::operator!= (const GenParticle & *a*) const

check for inequality

Definition at line 102 of file HepMC-2.06.09/src/GenParticle.cc.

7.16.3.27 GenParticle& HepMC::GenParticle::operator= (const GenParticle & *inparticle*)

shallow.

7.16.3.28 GenParticle & HepMC::GenParticle::operator= (const GenParticle & *inparticle*)

shallow. Shallow: does not copy the vertex pointers (note - impossible to copy vertex pointers which having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Shallow: does not copy the vertex pointers (note - impossible to copy vertex pointers which having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 77 of file HepMC-2.06.09/src/GenParticle.cc.

References swap().

7.16.3.29 bool HepMC::GenParticle::operator== (const GenParticle &) const

check for equality

7.16.3.30 bool HepMC::GenParticle::operator== (const GenParticle & *a*) const

check for equality

consistent with the definition of the copy constructor as a shallow constructor,.. this operator does not test the vertex pointers. Does not compare barcodes.

consistent with the definition of the copy constructor as a shallow constructor,.. this operator does not test the vertex pointers. Does not compare barcodes.

Definition at line 89 of file HepMC-2.06.09/src/GenParticle.cc.

References generated_mass(), momentum(), pdg_id(), polarization(), and status().

7.16.3.31 GenEvent* HepMC::GenParticle::parent_event ( ) const

pointer to the event that owns this particle

7.16.3.32 GenEvent * HepMC::GenParticle::parent_event ( ) const

pointer to the event that owns this particle

Definition at line 123 of file HepMC-2.06.09/src/GenParticle.cc.

References end_vertex(), HepMC::GenVertex::parent_event(), and production_vertex().

Referenced by HepMC::GenEvent::set_barcode(), set_end_vertex_(), set_production_vertex_(), suggest_barcode(), and ~GenParticle().

7.16.3.33 GenParticleProductionRange HepMC::GenParticle::particles_in (IteratorRange *range* = relatives)

incoming particle range

Examples:

testHepMCIteration.cc.in.

Definition at line 61 of file HepMC-2.06.09/src/GenRanges.cc.

Referenced by PrintW::operator()(), and PrintConstW::operator()().

7.16.3.34 GenParticleProductionRange HepMC::GenParticle::particles_in (IteratorRange *range* = relatives)

incoming particle range

7.16.3.35 ConstGenParticleProductionRange HepMC::GenParticle::particles_in (IteratorRange *range* = relatives) const

incoming particle range

7.16.3.36 ConstGenParticleProductionRange HepMC::GenParticle::particles_in (IteratorRange *range* = relatives) const

incoming particle range

Definition at line 67 of file HepMC-2.06.09/src/GenRanges.cc.

7.16.3.37 GenParticleEndRange HepMC::GenParticle::particles_out (IteratorRange *range* = relatives)

outgoing particle range

Examples:

testHepMCIteration.cc.in.

Definition at line 73 of file HepMC-2.06.09/src/GenRanges.cc.

Referenced by PrintW::operator()(), and PrintConstW::operator()().

7.16.3.38 GenParticleEndRange HepMC::GenParticle::particles_out (IteratorRange *range* = relatives)

outgoing particle range

7.16.3.39 ConstGenParticleEndRange HepMC::GenParticle::particles_out (IteratorRange *range* = relatives) const

outgoing particle range

7.16.3.40 ConstGenParticleEndRange HepMC::GenParticle::particles_out (IteratorRange *range* = relatives) const

outgoing particle range

Definition at line 79 of file HepMC-2.06.09/src/GenRanges.cc.

7.16.3.41 int HepMC::GenParticle::pdg_id ( ) const [inline]

particle ID

Examples:

example_UsingIterators.cc.

Definition at line 214 of file GenParticle.h.

Referenced by IsPhoton(), IsWBoson(), IsPhoton::operator()(), IsW_Boson::operator()(), PrintChildren::operator()(), HepMC::operator<<(), operator==(), and print().

7.16.3.42 int HepMC::GenParticle::pdg_id ( ) const

particle ID

7.16.3.43 const Polarization& HepMC::GenParticle::polarization ( ) const

polarization information

7.16.3.44 const Polarization & HepMC::GenParticle::polarization ( ) const [inline]

polarization information

Definition at line 228 of file GenParticle.h.

Referenced by operator==(), and print().

7.16.3.45 void HepMC::GenParticle::print (std::ostream & *ostr* = std::cout) const

dump this particle's full info to ostr

Dump this particle's full info to ostr, where by default particle.print(); will dump to cout.

Dump this particle's full info to ostr, where by default particle.print(); will dump to cout.

Examples:

testHepMCIteration.cc.in.

Definition at line 106 of file HepMC-2.06.09/src/GenParticle.cc.

References HepMC::GenVertex::barcode(), barcode(), HepMC::FourVector::e(), end_vertex(), momentum(), pdg_id(), polarization(), production_vertex(), HepMC::FourVector::px(), HepMC::FourVector::py(), HepMC::FourVector::pz(), and status().

Referenced by PrintW::operator()(), PrintPhoton::operator()(), PrintParticle::operator()(), PrintConstW::operator()(), and PrintDescendants::operator()().

7.16.3.46 void HepMC::GenParticle::print (std::ostream & *ostr* = `std::cout`) const

dump this particle's full info to ostr

7.16.3.47 GenVertex* HepMC::GenParticle::production_vertex ( ) const

pointer to the production vertex

7.16.3.48 GenVertex * HepMC::GenParticle::production_vertex ( ) const [inline]

pointer to the production vertex

Examples:

testHepMCIteration.cc.in.

Definition at line 218 of file GenParticle.h.

Referenced by HepMC::GenVertex::add_particle_out(), HepMC::GenParticleProductionRange::begin(), HepMC::ConstGenParticleProductionRange::begin(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::GenParticleProductionRange::end(), HepMC::ConstGenParticleProductionRange::end(), filterEvent(), HepMC::GenEvent::GenEvent(), PrintW::operator()(), PrintConstW::operator()(), parent_event(), print(), HepMC::GenEvent::read(), and HepMC::GenVertex::remove_particle().

7.16.3.49 void HepMC::GenParticle::set_barcode_ (int *the_bar_code*) [inline], [protected]

for use by **GenEvent** (p. 72) only

Definition at line 254 of file GenParticle.h.

Referenced by HepMC::GenEvent::set_barcode(), and suggest_barcode().

7.16.3.50 void HepMC::GenParticle::set_barcode_ (int *the_bar_code*) [protected]

for use by **GenEvent** (p. 72) only

7.16.3.51 void HepMC::GenParticle::set_end_vertex_ (GenVertex * *decayvertex* = 0) [protected]

set decay vertex - for internal use only

7.16.3.52 void HepMC::GenParticle::set_end_vertex_ (GenVertex * *decayvertex* = 0) [protected]

set decay vertex - for internal use only

Definition at line 142 of file HepMC-2.06.09/src/GenParticle.cc.

References barcode(), parent_event(), HepMC::GenEvent::remove_barcode(), and HepMC::GenEvent::set_barcode().

Referenced by HepMC::GenVertex::add_particle_in(), GenParticle(), and HepMC::GenVertex::remove_particle().

7.16.3.53 void HepMC::GenParticle::set_flow (const Flow & *f*)

set particle flow

7.16.3.54 void HepMC::GenParticle::set_flow (const Flow & *f*) [inline]

set particle flow

Examples:

testFlow.cc.

Definition at line 238 of file GenParticle.h.

Referenced by GenParticle(), main(), and HepMC::detail::read_particle().

7.16.3.55 void HepMC::GenParticle::set_flow (int *code_index,* int *code* = 0)

set particle flow index

7.16.3.56 void HepMC::GenParticle::set_flow (int *code_index,* int *code* = 0) [inline]

set particle flow index

Definition at line 240 of file GenParticle.h.

References HepMC::Flow::set_icode(), and HepMC::Flow::set_unique_icode().

7.16.3.57 void HepMC::GenParticle::set_generated_mass (const double & *m*)

define the actual generated mass

If you do not call **set_generated_mass()** (p. 118), then **generated_mass()** (p. 112) will simply return the mass calculated from **momentum()** (p. 113)

Definition at line 240 of file HepMC-2.06.09/src/GenParticle.cc.

Referenced by HepMC::detail::read_particle(), and setGeneratedMass().

7.16.3.58 void HepMC::GenParticle::set_generated_mass (const double & *m*)

define the actual generated mass

If you do not call **set_generated_mass()** (p. 118), then **generated_mass()** (p. 112) will simply return the mass calculated from **momentum()** (p. 113)

7.16.3.59 void HepMC::GenParticle::set_momentum (const FourVector & *vec4*)

set standard 4 momentum

7.16.3.60 void HepMC::GenParticle::set_momentum (const FourVector & *vec4*) [inline]

set standard 4 momentum

Definition at line 231 of file GenParticle.h.

Referenced by HepMC::detail::read_particle().

7.16.3.61 void HepMC::GenParticle::set_pdg_id (int *id*) [inline]

set particle ID

Definition at line 234 of file GenParticle.h.

Referenced by HepMC::detail::read_particle().

7.16.3.62 void HepMC::GenParticle::set_pdg_id (int *id*)

set particle ID

7.16.3.63 void HepMC::GenParticle::set_polarization (const Polarization & *pol* = Polarization (0,0)) [inline]

set polarization

Definition at line 249 of file GenParticle.h.

Referenced by main(), and HepMC::detail::read_particle().

7.16.3.64 void HepMC::GenParticle::set_polarization (const Polarization & *pol* = Polarization (0, 0))

set polarization

7.16.3.65 void HepMC::GenParticle::set_production_vertex_ (GenVertex * *productionvertex* = 0) [protected]

set production vertex - for internal use only

7.16.3.66 void HepMC::GenParticle::set_production_vertex_ (GenVertex * *productionvertex* = 0) [protected]

set production vertex - for internal use only

Definition at line 129 of file HepMC-2.06.09/src/GenParticle.cc.

References barcode(), parent_event(), HepMC::GenEvent::remove_barcode(), and HepMC::GenEvent::set_barcode().

Referenced by HepMC::GenVertex::add_particle_out(), GenParticle(), and HepMC::GenVertex::remove_particle().

7.16.3.67 void HepMC::GenParticle::set_status (int *status* = 0)

set decay status

7.16.3.68 void HepMC::GenParticle::set_status (int *status* = 0) [inline]

set decay status

Definition at line 236 of file GenParticle.h.

Referenced by HepMC::detail::read_particle().

7.16.3.69 void HepMC::GenParticle::setGeneratedMass (const double & *m*) [inline]

setGeneratedMass() (p. 120) is included for backwards compatibility with **CLHEP** (p. 15) **HepMC** (p. 15)

Definition at line 173 of file GenParticle.h.

References set_generated_mass().

Referenced by HepMC::IO_HEPEVT::build_particle(), and HepMC::IO_HERWIG::build_particle().

7.16.3.70 void HepMC::GenParticle::setGeneratedMass (const double & *m*) [inline]

setGeneratedMass() (p. 120) is included for backwards compatibility with **CLHEP** (p. 15) **HepMC** (p. 15)

Definition at line 173 of file 2.06.09/HepMC/GenParticle.h.

References set_generated_mass().

7.16.3.71 int HepMC::GenParticle::status ( ) const [inline]

HEPEVT decay status.

Examples:

example_UsingIterators.cc.

Definition at line 216 of file GenParticle.h.

Referenced by IsFinalState::operator()(), IsStateFinal::operator()(), PrintChildren::operator()(), HepMC::operator<<(), operator==(), and print().

7.16.3.72 int HepMC::GenParticle::status ( ) const

HEPEVT decay status.

7.16.3.73 bool HepMC::GenParticle::suggest_barcode (int *the_bar_code*)

In general there is no reason to "suggest_barcode".

allows a barcode to be suggested for this particle. In general it is better to let the event pick the barcode for you, which is automatic. Returns TRUE if the suggested barcode has been accepted (i.e. the suggested barcode has not already been used in the event, and so it was used). Returns FALSE if the suggested barcode was rejected, or if the particle is not yet part of an event, such that it is not yet possible to know if the suggested barcode will be accepted).

allows a barcode to be suggested for this particle. In general it is better to let the event pick the barcode for you, which is automatic. Returns TRUE if the suggested barcode has been accepted (i.e. the suggested barcode has not already been used in the event, and so it was used). Returns FALSE if the suggested barcode was rejected, or if the particle is not yet part of an event, such that it is not yet possible to know if the suggested barcode will be accepted).

Definition at line 153 of file HepMC-2.06.09/src/GenParticle.cc.

References parent_event(), HepMC::GenEvent::set_barcode(), and set_barcode_().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), GenParticle(), and HepMC::detail::read_particle().

7.16.3.74 bool HepMC::GenParticle::suggest_barcode (int *the_bar_code*)

In general there is no reason to "suggest_barcode".

7.16.3.75 void HepMC::GenParticle::swap (GenParticle & *other*)

swap

7.16.3.76 void HepMC::GenParticle::swap (GenParticle & *other*)

swap

Definition at line 63 of file HepMC-2.06.09/src/GenParticle.cc.

References HepMC::Polarization::swap(), HepMC::FourVector::swap(), and HepMC::Flow::swap().

Referenced by operator=().

7.16.4 Friends And Related Function Documentation

7.16.4.1 GenEvent [friend]

Definition at line 63 of file GenParticle.h.

7.16.4.2 GenVertex [friend]

Definition at line 62 of file GenParticle.h.

7.16.4.3 std::ostream& operator<< (std::ostream & *ostr,* const GenParticle & *part*) [friend]

print particle

Definition at line 189 of file HepMC-2.06.09/src/GenParticle.cc.

7.16.4.4 std::ostream& operator<< (std::ostream & *ostr,* const GenParticle & *part*) [friend]

print particle

print particle

Definition at line 189 of file HepMC-2.06.09/src/GenParticle.cc.

The documentation for this class was generated from the following files:

- **GenParticle.h**
- **2.06.09/HepMC/GenParticle.h**
- **HepMC-2.06.09/src/GenParticle.cc**
- **HepMC-2.06.09/src/GenRanges.cc**
- **src/GenParticle.cc**
- **src/GenRanges.cc**

7.17 HepMC::GenParticleEndRange Class Reference

GenParticleEndRange (p. 122) acts like a collection of particles.

`#include <GenRanges.h>`

Public Member Functions

- **GenParticleEndRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)
  *the constructor requires a **GenParticle** (p. 106)*
- **GenVertex::particle_iterator begin** ()
  begin iterator throws an error if the particle end_vertex is undefined
- **GenVertex::particle_iterator end** ()
  end iterator throws an error if the particle end_vertex is undefined
- **GenParticleEndRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)
  *the constructor requires a **GenParticle** (p. 106)*
- **GenVertex::particle_iterator begin** ()
  begin iterator throws an error if the particle end_vertex is undefined
- **GenVertex::particle_iterator end** ()
  end iterator throws an error if the particle end_vertex is undefined

7.17.1 Detailed Description

GenParticleEndRange (p. 122) acts like a collection of particles.

HepMC::GenParticleEndRange (p. 122) is used to mimic a collection of particles associated with the particle's end vertex for ease of use Utilities such as the Boost foreach funtion will want to use this class.

Definition at line 224 of file GenRanges.h.

7.17.2 Constructor & Destructor Documentation

7.17.2.1 HepMC::GenParticleEndRange::GenParticleEndRange (GenParticle const & *p*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 229 of file GenRanges.h.

7.17.2.2 HepMC::GenParticleEndRange::GenParticleEndRange (GenParticle const & *p*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 229 of file 2.06.09/HepMC/GenRanges.h.

7.17.3 Member Function Documentation

7.17.3.1 GenVertex::particle_iterator HepMC::GenParticleEndRange::begin ( ) [inline]

begin iterator throws an error if the particle end_vertex is undefined

Examples:

testHepMCIteration.cc.in.

Definition at line 300 of file GenRanges.h.

References HepMC::GenParticle::end_vertex(), and HepMC::GenVertex::particles_begin().

Referenced by PrintW::operator()(), and PrintConstW::operator()().

7.17.3.2 GenVertex::particle_iterator HepMC::GenParticleEndRange::begin ( )

begin iterator throws an error if the particle end_vertex is undefined

7.17.3.3 GenVertex::particle_iterator HepMC::GenParticleEndRange::end ( )

end iterator throws an error if the particle end_vertex is undefined

7.17.3.4 GenVertex::particle_iterator HepMC::GenParticleEndRange::end ( ) [inline]

end iterator throws an error if the particle end_vertex is undefined

Examples:

testHepMCIteration.cc.in.

Definition at line 306 of file GenRanges.h.

References HepMC::GenParticle::end_vertex(), and HepMC::GenVertex::particles_end().

Referenced by PrintW::operator()(), and PrintConstW::operator()().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.18 HepMC::GenParticleProductionRange Class Reference

GenParticleProductionRange (p. 123) acts like a collection of particles.

`#include <GenRanges.h>`

Public Member Functions

- **GenParticleProductionRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)
  *the constructor requires a **GenParticle** (p. 106)*
- **GenVertex::particle_iterator begin** ()
  begin iterator throws an error if the particle production_vertex is undefined
- **GenVertex::particle_iterator end** ()
  end iterator throws an error if the particle production_vertex is undefined
- **GenParticleProductionRange** (**GenParticle** const &**p**, **IteratorRange** range=**relatives**)
  *the constructor requires a **GenParticle** (p. 106)*
- **GenVertex::particle_iterator begin** ()
  begin iterator throws an error if the particle production_vertex is undefined
- **GenVertex::particle_iterator end** ()
  end iterator throws an error if the particle production_vertex is undefined

7.18.1 Detailed Description

GenParticleProductionRange (p. 123) acts like a collection of particles.

HepMC::GenParticleProductionRange (p. 123) is used to mimic a collection of particles associated with the particle's production vertex for ease of use Utilities such as the Boost foreach funtion will want to use this class.

Definition at line 170 of file GenRanges.h.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 HepMC::GenParticleProductionRange::GenParticleProductionRange (GenParticle const & *p*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 175 of file GenRanges.h.

7.18.2.2 HepMC::GenParticleProductionRange::GenParticleProductionRange (GenParticle const & *p*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenParticle** (p. 106)

Definition at line 175 of file 2.06.09/HepMC/GenRanges.h.

7.18.3 Member Function Documentation

7.18.3.1 GenVertex::particle_iterator HepMC::GenParticleProductionRange::begin ( ) [inline]

begin iterator throws an error if the particle production_vertex is undefined

Examples:

testHepMCIteration.cc.in.

Definition at line 271 of file GenRanges.h.

References HepMC::GenVertex::particles_begin(), and HepMC::GenParticle::production_vertex().

Referenced by PrintW::operator()(), and PrintConstW::operator()().

7.18.3.2 GenVertex::particle_iterator HepMC::GenParticleProductionRange::begin ( )

begin iterator throws an error if the particle production_vertex is undefined

7.18.3.3 GenVertex::particle_iterator HepMC::GenParticleProductionRange::end ( )

end iterator throws an error if the particle production_vertex is undefined

7.18.3.4 GenVertex::particle_iterator HepMC::GenParticleProductionRange::end ( ) [inline]

end iterator throws an error if the particle production_vertex is undefined

Examples:

testHepMCIteration.cc.in.

Definition at line 278 of file GenRanges.h.

References HepMC::GenVertex::particles_end(), and HepMC::GenParticle::production_vertex().

Referenced by PrintW::operator()(), and PrintConstW::operator()().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.19 HepMC::GenVertex Class Reference

GenVertex (p. 125) contains information about decay vertices.

`#include <GenVertex.h>`

Classes

- class **edge_iterator**
  edge iterator
- class **particle_iterator**
  particle iterator
- class **vertex_iterator**
  vertex iterator

Public Types

- typedef std::vector
  < **HepMC::GenParticle** * >
  ::const_iterator **particles_in_const_iterator**
  const iterator for incoming particles
- typedef std::vector
  < **HepMC::GenParticle** * >
  ::const_iterator **particles_out_const_iterator**
  const iterator for outgoing particles
- typedef std::vector
  < **HepMC::GenParticle** * >
  ::const_iterator **particles_in_const_iterator**
  const iterator for incoming particles
- typedef std::vector
  < **HepMC::GenParticle** * >
  ::const_iterator **particles_out_const_iterator**
  const iterator for outgoing particles

Public Member Functions

- **GenVertex** (const **FourVector** &**position**=**FourVector**(0, 0, 0, 0), int **id**=0, const **WeightContainer** &**weights**=std::vector< double >())
  default constructor
- **GenVertex** (const **GenVertex** &invertex)
  shallow copy
- virtual ~**GenVertex** ()

- void **swap** (**GenVertex** &other)

  swap
- **GenVertex** & **operator=** (const **GenVertex** &invertex)

  shallow
- bool **operator==** (const **GenVertex** &a) const

  equality
- bool **operator!=** (const **GenVertex** &a) const

  inequality
- void **print** (std::ostream &ostr=std::cout) const

  print vertex information
- double **check_momentum_conservation** () const

  |*Sum (three_mom_in-three_mom_out)*|
- void **add_particle_in** (**GenParticle** ∗inparticle)

  add incoming particle
- void **add_particle_out** (**GenParticle** ∗outparticle)

  add outgoing particle
- **GenParticle** ∗ **remove_particle** (**GenParticle** ∗particle)

  remove a particle
- **operator HepMC::FourVector** () const

  conversion operator
- **operator HepMC::ThreeVector** () const

  conversion operator
- **GenEvent** ∗ **parent_event** () const

  pointer to the event that owns this vertex
- **ThreeVector point3d** () const

  vertex position
- const **FourVector** & **position** () const

  vertex position and time
- void **set_position** (const **FourVector** &**position**=**FourVector**(0, 0, 0, 0))

  set vertex position and time
- int **id** () const

  vertex ID
- void **set_id** (int **id**)

  set vertex ID
- int **barcode** () const

  unique identifier
- bool **suggest_barcode** (int the_bar_code)

  In general there is no reason to "suggest_barcode".
- **WeightContainer** & **weights** ()

  direct access to the weights container is allowed.
- const **WeightContainer** & **weights** () const

  const direct access to the weights container
- **GenVertexParticleRange particles** (**IteratorRange** range=**relatives**)

  particle range
- **GenParticleProductionRange particles_in** (**GenParticle** &, **IteratorRange** range=**relatives**)

  incoming particle range
- **ConstGenParticleProductionRange particles_in** (**GenParticle** const &, **IteratorRange** range=**relatives**) const

  incoming particle range
- **GenParticleEndRange particles_out** (**GenParticle** &, **IteratorRange** range=**relatives**)

  outgoing particle range

- **ConstGenParticleEndRange particles_out** (**GenParticle** const &, **IteratorRange** range=**relatives**) const
  outgoing particle range
- **particles_in_const_iterator particles_in_const_begin** () const
  begin iteration of incoming particles
- **particles_in_const_iterator particles_in_const_end** () const
  end iteration of incoming particles
- **particles_out_const_iterator particles_out_const_begin** () const
  begin iteration of outgoing particles
- **particles_out_const_iterator particles_out_const_end** () const
  end iteration of outgoing particles
- int **particles_in_size** () const
  number of incoming particles
- int **particles_out_size** () const
  number of outgoing particles
- **vertex_iterator vertices_begin** (**IteratorRange** range=**relatives**)
  begin vertex range
- **vertex_iterator vertices_end** (**IteratorRange**)
  end vertex range
- **particle_iterator particles_begin** (**IteratorRange** range=**relatives**)
  begin particle range
- **particle_iterator particles_end** (**IteratorRange**)
  end particle range
- **GenVertex** (const **FourVector** &**position**=**FourVector**(0, 0, 0, 0), int **id**=0, const **WeightContainer** &**weights**=std::vector< double >())
  default constructor
- **GenVertex** (const **GenVertex** &invertex)
  shallow copy
- virtual ~**GenVertex** ()
- void **swap** (**GenVertex** &other)
  swap
- **GenVertex** & **operator=** (const **GenVertex** &invertex)
  shallow
- bool **operator==** (const **GenVertex** &a) const
  equality
- bool **operator!=** (const **GenVertex** &a) const
  inequality
- void **print** (std::ostream &ostr=std::cout) const
  print vertex information
- double **check_momentum_conservation** () const
  |*Sum (three_mom_in-three_mom_out)*|
- void **add_particle_in** (**GenParticle** *inparticle)
  add incoming particle
- void **add_particle_out** (**GenParticle** *outparticle)
  add outgoing particle
- **GenParticle** * **remove_particle** (**GenParticle** *particle)
  remove a particle
- **operator HepMC::FourVector** () const
  conversion operator
- **operator HepMC::ThreeVector** () const
  conversion operator
- **GenEvent** * **parent_event** () const

pointer to the event that owns this vertex

- **ThreeVector point3d** () const

  vertex position

- const **FourVector** & **position** () const

  vertex position and time

- void **set_position** (const **FourVector** &**position**=**FourVector**(0, 0, 0, 0))

  set vertex position and time

- int **id** () const

  vertex ID

- void **set_id** (int **id**)

  set vertex ID

- int **barcode** () const

  unique identifier

- bool **suggest_barcode** (int the_bar_code)

  In general there is no reason to "suggest_barcode".

- **WeightContainer** & **weights** ()

  direct access to the weights container is allowed.

- const **WeightContainer** & **weights** () const

  const direct access to the weights container

- **GenVertexParticleRange particles** (**IteratorRange** range=**relatives**)

  particle range

- **GenParticleProductionRange particles_in** (**GenParticle** &, **IteratorRange** range=**relatives**)

  incoming particle range

- **ConstGenParticleProductionRange particles_in** (**GenParticle** const &, **IteratorRange** range=**relatives**) const

  incoming particle range

- **GenParticleEndRange particles_out** (**GenParticle** &, **IteratorRange** range=**relatives**)

  outgoing particle range

- **ConstGenParticleEndRange particles_out** (**GenParticle** const &, **IteratorRange** range=**relatives**) const

  outgoing particle range

- **particles_in_const_iterator particles_in_const_begin** () const

  begin iteration of incoming particles

- **particles_in_const_iterator particles_in_const_end** () const

  end iteration of incoming particles

- **particles_out_const_iterator particles_out_const_begin** () const

  begin iteration of outgoing particles

- **particles_out_const_iterator particles_out_const_end** () const

  end iteration of outgoing particles

- int **particles_in_size** () const

  number of incoming particles

- int **particles_out_size** () const

  number of outgoing particles

- **vertex_iterator vertices_begin** (**IteratorRange** range=**relatives**)

  begin vertex range

- **vertex_iterator vertices_end** (**IteratorRange**)

  end vertex range

- **particle_iterator particles_begin** (**IteratorRange** range=**relatives**)

  begin particle range

- **particle_iterator particles_end** (**IteratorRange**)

  end particle range

Protected Member Functions

- void **set_parent_event_** (**GenEvent** ∗evt)
  set parent event
- void **set_barcode_** (int the_bar_code)
  set identifier
- void **change_parent_event_** (**GenEvent** ∗evt)
  for use with swap
- int **edges_size** (**IteratorRange** range=**family**) const
  size
- **edge_iterator edges_begin** (**IteratorRange** range=**family**) const
  begin range
- **edge_iterator edges_end** (**IteratorRange**) const
  end range
- void **delete_adopted_particles** ()
  for internal use only
- void **remove_particle_in** (**GenParticle** ∗)
  for internal use only - remove particle from incoming list
- void **remove_particle_out** (**GenParticle** ∗)
  for internal use only - remove particle from outgoing list
- void **convert_position** (const double &)
- void **set_parent_event_** (**GenEvent** ∗evt)
  set parent event
- void **set_barcode_** (int the_bar_code)
  set identifier
- void **change_parent_event_** (**GenEvent** ∗evt)
  for use with swap
- int **edges_size** (**IteratorRange** range=**family**) const
  size
- **edge_iterator edges_begin** (**IteratorRange** range=**family**) const
  begin range
- **edge_iterator edges_end** (**IteratorRange**) const
  end range
- void **delete_adopted_particles** ()
  for internal use only
- void **remove_particle_in** (**GenParticle** ∗)
  for internal use only - remove particle from incoming list
- void **remove_particle_out** (**GenParticle** ∗)
  for internal use only - remove particle from outgoing list
- void **convert_position** (const double &)

Friends

- class **GenEvent**
- class **edge_iterator**
- class **vertex_iterator**
- class **particle_iterator**
- std::ostream & **operator**<< (std::ostream &, const **GenVertex** &)
  print vertex information
- std::ostream & **operator**<< (std::ostream &, const **GenVertex** &)
  print vertex information

7.19.1 Detailed Description

GenVertex (p. 125) contains information about decay vertices.

HepMC::GenVertex (p. 125) contains the position in space and time of a decay. It also contains lists of incoming and outgoing particles.

Examples:

example_BuildEventFromScratch.cc, **example_VectorConversion.cc**, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 52 of file GenVertex.h.

7.19.2 Member Typedef Documentation

7.19.2.1 typedef std::vector<HepMC::GenParticle*>::const_iterator HepMC::GenVertex::particles_in_const_iterator

const iterator for incoming particles

Definition at line 152 of file GenVertex.h.

7.19.2.2 typedef std::vector<HepMC::GenParticle*>::const_iterator HepMC::GenVertex::particles_in_const_iterator

const iterator for incoming particles

Definition at line 152 of file 2.06.09/HepMC/GenVertex.h.

7.19.2.3 typedef std::vector<HepMC::GenParticle*>::const_iterator HepMC::GenVertex::particles_out_const_iterator

const iterator for outgoing particles

Definition at line 155 of file 2.06.09/HepMC/GenVertex.h.

7.19.2.4 typedef std::vector<HepMC::GenParticle*>::const_iterator HepMC::GenVertex::particles_out_const_iterator

const iterator for outgoing particles

Definition at line 155 of file GenVertex.h.

7.19.3 Constructor & Destructor Documentation

7.19.3.1 HepMC::GenVertex::GenVertex (const FourVector & *position* = FourVector(0,0,0,0), int *id* = 0, const WeightContainer & *weights* = std::vector<double>())

default constructor

Definition at line 14 of file HepMC-2.06.09/src/GenVertex.cc.

7.19.3.2 HepMC::GenVertex::GenVertex (const GenVertex & *invertex*)

shallow copy

Shallow copy: does not copy the FULL list of particle pointers. Creates a copy of - invertex

- outgoing particles of invertex, but sets the decay vertex of these particles to NULL
- all incoming particles which do not have a creation vertex. (i.e. it creates copies of all particles which it owns) (note - impossible to copy the FULL list of particle pointers while having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Shallow copy: does not copy the FULL list of particle pointers. Creates a copy of - invertex

- outgoing particles of invertex, but sets the decay vertex of these particles to NULL
- all incoming particles which do not have a creation vertex. (i.e. it creates copies of all particles which it owns) (note - impossible to copy the FULL list of particle pointers while having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 23 of file HepMC-2.06.09/src/GenVertex.cc.

References add_particle_in(), add_particle_out(), barcode(), particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), particles_out_const_end(), and suggest_barcode().

7.19.3.3 HepMC::GenVertex::~GenVertex ( ) [virtual]

Definition at line 63 of file HepMC-2.06.09/src/GenVertex.cc.

References delete_adopted_particles(), parent_event(), and HepMC::GenEvent::remove_barcode().

7.19.3.4 HepMC::GenVertex::GenVertex (const FourVector & *position* = FourVector(0, 0, 0, 0), int *id* = 0, const WeightContainer & *weights* = std::vector< double >())

default constructor

7.19.3.5 HepMC::GenVertex::GenVertex (const GenVertex & *invertex*)

shallow copy

7.19.3.6 virtual HepMC::GenVertex::~GenVertex ( ) [virtual]

7.19.4 Member Function Documentation

7.19.4.1 void HepMC::GenVertex::add_particle_in (GenParticle * *inparticle*)

add incoming particle

Examples:

example_BuildEventFromScratch.cc, **example_VectorConversion.cc**, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 273 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenParticle::end_vertex(), remove_particle_in(), and HepMC::GenParticle::set_end_vertex_-().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_-HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HERWIG::fill_-next_event(), GenVertex(), main(), and HepMC::GenEvent::read().

7.19.4.2 void HepMC::GenVertex::add_particle_in (GenParticle * *inparticle*)

add incoming particle

7.19.4.3 void HepMC::GenVertex::add_particle_out (GenParticle * *outparticle*)

add outgoing particle

Examples:

example_BuildEventFromScratch.cc, **example_VectorConversion.cc**, **testFlow.cc**, and **testPrintBug.cc**.

Definition at line 284 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenParticle::production_vertex(), remove_particle_out(), and HepMC::GenParticle::set_production_vertex_().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_HERWIG::fill_next_event(), filterEvent(), GenVertex(), main(), and HepMC::detail::read_vertex().

7.19.4.4 void HepMC::GenVertex::add_particle_out (GenParticle * *outparticle*)

add outgoing particle

7.19.4.5 int HepMC::GenVertex::barcode ( ) const [inline]

unique identifier

The barcode is the vertex's reference number, every vertex in the event has a unique barcode. Vertex barcodes are negative numbers, particle barcodes are positive numbers.

Please note that the barcodes are intended for internal use within **HepMC** (p. 15) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

Definition at line 416 of file GenVertex.h.

Referenced by HepMC::GenEvent::add_vertex(), HepMC::compareVertex(), GenVertex(), HepMC::operator<<(), print(), HepMC::GenParticle::print(), HepMC::GenEvent::print(), HepMC::GenEvent::remove_barcode(), HepMC::GenEvent::remove_vertex(), HepMC::GenEvent::set_barcode(), set_parent_event_(), and HepMC::IO_AsciiParticles::write_event().

7.19.4.6 int HepMC::GenVertex::barcode ( ) const

unique identifier

The barcode is the vertex's reference number, every vertex in the event has a unique barcode. Vertex barcodes are negative numbers, particle barcodes are positive numbers.

Please note that the barcodes are intended for internal use within **HepMC** (p. 15) as a unique identifier for the particles and vertices. Using the barcode to encode extra information is an abuse of the barcode data member and causes confusion among users.

7.19.4.7 void HepMC::GenVertex::change_parent_event_ (GenEvent * *evt*) [protected]

for use with swap

Definition at line 419 of file HepMC-2.06.09/src/GenVertex.cc.

7.19.4.8 void HepMC::GenVertex::change_parent_event_ (GenEvent * *evt*) [protected]

for use with swap

7.19.4.9 double HepMC::GenVertex::check_momentum_conservation ( ) const

|Sum (three_mom_in-three_mom_out)|

finds the difference between the total momentum out and the total momentum in vectors, and returns the magnitude of this vector i.e. returns | vec{p_in} - vec{p_out} |

finds the difference between the total momentum out and the total momentum in vectors, and returns the magnitude of this vector i.e. returns | vec{p_in} - vec{p_out} |

Definition at line 253 of file HepMC-2.06.09/src/GenVertex.cc.

References particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), and particles_out_-const_end().

7.19.4.10 double HepMC::GenVertex::check_momentum_conservation ( ) const

|Sum (three_mom_in-three_mom_out)|

7.19.4.11 void HepMC::GenVertex::convert_position (const double & *f*) [protected]

scale the position vector this method is only for use by **GenEvent** (p. 72)

scale the position vector this method is only for use by **GenEvent** (p. 72) convert_position assumes that 4th component of the position vector is ctau rather than time and has units of length-time

Definition at line 918 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::FourVector::t(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector-::z().

7.19.4.12 void HepMC::GenVertex::convert_position (const double &) [protected]

scale the position vector this method is only for use by **GenEvent** (p. 72)

7.19.4.13 void HepMC::GenVertex::delete_adopted_particles ( ) [protected]

for internal use only

7.19.4.14 void HepMC::GenVertex::delete_adopted_particles ( ) [protected]

for internal use only

deletes all particles which this vertex owns to be used by the vertex destructor and operator=

deletes all particles which this vertex owns to be used by the vertex destructor and operator=

Definition at line 329 of file HepMC-2.06.09/src/GenVertex.cc.

Referenced by ~GenVertex().

7.19.4.15 GenVertex::edge_iterator HepMC::GenVertex::edges_begin (IteratorRange *range* = family) const [inline], [protected]

begin range

Definition at line 476 of file GenVertex.h.

References edge_iterator.

Referenced by HepMC::GenVertex::vertex_iterator::vertex_iterator().

7.19.4.16 edge_iterator HepMC::GenVertex::edges_begin (IteratorRange *range* = family) const [protected]

begin range

7.19.4.17 GenVertex::edge_iterator HepMC::GenVertex::edges_end (IteratorRange) const [inline], [protected]

end range

Definition at line 481 of file GenVertex.h.

References edge_iterator.

Referenced by HepMC::GenVertex::vertex_iterator::vertex_iterator().

7.19.4.18 edge_iterator HepMC::GenVertex::edges_end (IteratorRange) const [protected]

end range

7.19.4.19 int HepMC::GenVertex::edges_size (IteratorRange *range* = family) const [protected]

size

Definition at line 595 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::children, HepMC::family, and HepMC::parents.

7.19.4.20 int HepMC::GenVertex::edges_size (IteratorRange *range* = family) const [protected]

size

7.19.4.21 int HepMC::GenVertex::id ( ) const [inline]

vertex ID

we don't define what you use the id for – but we imagine, for example it might code the meaning of the **weights()** (p. 143)

Definition at line 414 of file GenVertex.h.

Referenced by print().

7.19.4.22 int HepMC::GenVertex::id ( ) const

vertex ID

we don't define what you use the id for – but we imagine, for example it might code the meaning of the **weights()** (p. 143)

7.19.4.23 HepMC::GenVertex::operator HepMC::FourVector ( ) const [inline]

conversion operator

Definition at line 402 of file GenVertex.h.

7.19.4.24 HepMC::GenVertex::operator HepMC::FourVector ( ) const

conversion operator

7.19.4.25 HepMC::GenVertex::operator HepMC::ThreeVector ( ) const [inline]

conversion operator

Definition at line 404 of file GenVertex.h.

7.19.4.26 HepMC::GenVertex::operator HepMC::ThreeVector ( ) const

conversion operator

7.19.4.27 bool HepMC::GenVertex::operator!= (const GenVertex & *a*) const

inequality

7.19.4.28 bool HepMC::GenVertex::operator!= (const GenVertex & *a*) const

inequality

Definition at line 140 of file HepMC-2.06.09/src/GenVertex.cc.

7.19.4.29 GenVertex & HepMC::GenVertex::operator= (const GenVertex & *invertex*)

shallow

Shallow: does not copy the FULL list of particle pointers. Creates a copy of - invertex

- outgoing particles of invertex, but sets the decay vertex of these particles to NULL
- all incoming particles which do not have a creation vertex.
- it does not alter ∗this's m_event (!) (i.e. it creates copies of all particles which it owns) (note - impossible to copy the FULL list of particle pointers while having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Shallow: does not copy the FULL list of particle pointers. Creates a copy of - invertex

- outgoing particles of invertex, but sets the decay vertex of these particles to NULL
- all incoming particles which do not have a creation vertex.
- it does not alter ∗this's m_event (!) (i.e. it creates copies of all particles which it owns) (note - impossible to copy the FULL list of particle pointers while having the vertex and particles in/out point-back to one another – unless you copy the entire tree – which we don't want to do)

Definition at line 82 of file HepMC-2.06.09/src/GenVertex.cc.

References swap().

7.19.4.30 GenVertex& HepMC::GenVertex::operator= (const GenVertex & *invertex*)

shallow

7.19.4.31 bool HepMC::GenVertex::operator== (const GenVertex & *a*) const

equality

7.19.4.32 bool HepMC::GenVertex::operator== (const GenVertex & *a*) const

equality

Returns true if the positions and the particles in the lists of a and this are identical. Does not compare barcodes. Note that it is impossible for two vertices to point to the same particle's address, so we need to do more than just compare the particle pointers

Returns true if the positions and the particles in the lists of a and this are identical. Does not compare barcodes. Note that it is impossible for two vertices to point to the same particle's address, so we need to do more than just compare the particle pointers

Definition at line 103 of file HepMC-2.06.09/src/GenVertex.cc.

References particles_in_const_begin(), particles_in_const_end(), particles_in_size(), particles_out_const_begin(), particles_out_const_end(), particles_out_size(), and position().

7.19.4.33 GenEvent * HepMC::GenVertex::parent_event ( ) const [inline]

pointer to the event that owns this vertex

Definition at line 408 of file GenVertex.h.

Referenced by HepMC::GenEvent::add_vertex(), HepMC::GenParticle::parent_event(), HepMC::GenEvent::remove_vertex(), HepMC::GenEvent::set_barcode(), suggest_barcode(), and ~GenVertex().

7.19.4.34 GenEvent* HepMC::GenVertex::parent_event ( ) const

pointer to the event that owns this vertex

7.19.4.35 GenVertexParticleRange HepMC::GenVertex::particles (IteratorRange *range* = relatives)

particle range

7.19.4.36 GenVertexParticleRange HepMC::GenVertex::particles (IteratorRange *range* = relatives)

particle range

Definition at line 36 of file HepMC-2.06.09/src/GenRanges.cc.

7.19.4.37 GenVertex::particle_iterator HepMC::GenVertex::particles_begin (IteratorRange *range* = relatives) [inline]

begin particle range

Definition at line 525 of file GenVertex.h.

References particle_iterator.

Referenced by HepMC::GenVertexParticleRange::begin(), HepMC::GenParticleProductionRange::begin(), HepMC::ConstGenParticleProductionRange::begin(), HepMC::GenParticleEndRange::begin(), HepMC::ConstGenParticleEndRange::begin(), HepMC::Flow::connected_partners(), and HepMC::Flow::dangling_connected_partners().

7.19.4.38 particle_iterator HepMC::GenVertex::particles_begin (IteratorRange *range* = relatives)

begin particle range

7.19.4.39 particle_iterator HepMC::GenVertex::particles_end (IteratorRange)

end particle range

7.19.4.40 GenVertex::particle_iterator HepMC::GenVertex::particles_end (IteratorRange) [inline]

end particle range

Definition at line 530 of file GenVertex.h.

References particle_iterator.

Referenced by HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::GenVertexParticleRange::end(), HepMC::GenParticleProductionRange::end(), HepMC::ConstGenParticleProductionRange::end(), HepMC::GenParticleEndRange::end(), and HepMC::ConstGenParticleEndRange::end().

7.19.4.41 GenParticleProductionRange HepMC::GenVertex::particles_in (GenParticle & , IteratorRange *range* = relatives)

incoming particle range

7.19.4.42 GenParticleProductionRange HepMC::GenVertex::particles_in (GenParticle & *p*, IteratorRange *range* = relatives)

incoming particle range

Definition at line 41 of file HepMC-2.06.09/src/GenRanges.cc.

7.19.4.43 ConstGenParticleProductionRange HepMC::GenVertex::particles_in (GenParticle const & , IteratorRange *range* = relatives) const

incoming particle range

7.19.4.44 ConstGenParticleProductionRange HepMC::GenVertex::particles_in (GenParticle const & *p*, IteratorRange *range* = relatives) const

incoming particle range

Definition at line 46 of file HepMC-2.06.09/src/GenRanges.cc.

7.19.4.45 particles_in_const_iterator HepMC::GenVertex::particles_in_const_begin ( ) const

begin iteration of incoming particles

7.19.4.46 GenVertex::particles_in_const_iterator HepMC::GenVertex::particles_in_const_begin ( ) const [inline]

begin iteration of incoming particles

Definition at line 435 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), GenVertex(), operator==(), print(), and set_parent_event_().

7.19.4.47 particles_in_const_iterator HepMC::GenVertex::particles_in_const_end ( ) const

end iteration of incoming particles

7.19.4.48 GenVertex::particles_in_const_iterator HepMC::GenVertex::particles_in_const_end ( ) const [inline]

end iteration of incoming particles

Definition at line 440 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), GenVertex(), operator==(), print(), and set_parent_event_().

7.19.4.49 int HepMC::GenVertex::particles_in_size ( ) const

number of incoming particles

7.19.4.50 int HepMC::GenVertex::particles_in_size ( ) const [inline]

number of incoming particles

Definition at line 454 of file GenVertex.h.

Referenced by HepMC::compareVertex(), and operator==().

7.19.4.51 GenParticleEndRange HepMC::GenVertex::particles_out (GenParticle & , IteratorRange *range* = relatives)

outgoing particle range

7.19.4.52 GenParticleEndRange HepMC::GenVertex::particles_out (GenParticle & *p*, IteratorRange *range* = relatives)

outgoing particle range

Definition at line 51 of file HepMC-2.06.09/src/GenRanges.cc.

7.19.4.53 ConstGenParticleEndRange HepMC::GenVertex::particles_out (GenParticle const & , IteratorRange *range* = relatives) const

outgoing particle range

7.19.4.54 ConstGenParticleEndRange HepMC::GenVertex::particles_out (GenParticle const & *p*, IteratorRange *range* = relatives) const

outgoing particle range

Definition at line 56 of file HepMC-2.06.09/src/GenRanges.cc.

7.19.4.55 particles_out_const_iterator HepMC::GenVertex::particles_out_const_begin ( ) const

begin iteration of outgoing particles

7.19.4.56 GenVertex::particles_out_const_iterator HepMC::GenVertex::particles_out_const_begin ( ) const [inline]

begin iteration of outgoing particles

Definition at line 445 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), filterEvent(), GenVertex(), operator==(), print(), and set_parent_event_().

7.19.4.57 particles_out_const_iterator HepMC::GenVertex::particles_out_const_end ( ) const

end iteration of outgoing particles

7.19.4.58 GenVertex::particles_out_const_iterator HepMC::GenVertex::particles_out_const_end ( ) const [inline]

end iteration of outgoing particles

Definition at line 450 of file GenVertex.h.

Referenced by check_momentum_conservation(), HepMC::compareVertex(), filterEvent(), GenVertex(), operator==(), print(), and set_parent_event_().

7.19.4.59 int HepMC::GenVertex::particles_out_size ( ) const [inline]

number of outgoing particles

Definition at line 458 of file GenVertex.h.

Referenced by HepMC::compareVertex(), filterEvent(), and operator==().

7.19.4.60 int HepMC::GenVertex::particles_out_size ( ) const

number of outgoing particles

7.19.4.61 ThreeVector HepMC::GenVertex::point3d ( ) const [inline]

vertex position

Definition at line 410 of file GenVertex.h.

References HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

7.19.4.62 ThreeVector HepMC::GenVertex::point3d ( ) const

vertex position

7.19.4.63 const FourVector & HepMC::GenVertex::position ( ) const [inline]

vertex position and time

Definition at line 406 of file GenVertex.h.

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::compareVertex(), HepMC::operator<<(), operator==(), and print().

7.19.4.64 const FourVector& HepMC::GenVertex::position ( ) const

vertex position and time

7.19.4.65 void HepMC::GenVertex::print (std::ostream & *ostr* = `std::cout`) const

print vertex information

Definition at line 145 of file HepMC-2.06.09/src/GenVertex.cc.

References barcode(), HepMC::WeightContainer::end(), id(), particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), particles_out_const_end(), position(), HepMC::WeightContainer::size(), HepMC::FourVector::t(), weights(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

Referenced by HepMC::IO_HERWIG::build_production_vertex().

7.19.4.66 void HepMC::GenVertex::print (std::ostream & *ostr* = `std::cout`) const

print vertex information

7.19.4.67 GenParticle* HepMC::GenVertex::remove_particle (GenParticle * *particle*)

remove a particle

remove_particle finds *particle in the in and/or out list and removes it from these lists ... it DOES NOT DELETE THE PARTICLE or its relations. You could delete the particle too as follows: delete vtx->remove_particle(particle);

7.19.4.68 GenParticle * HepMC::GenVertex::remove_particle (GenParticle * *particle*)

remove a particle

remove_particle finds *particle in the in and/or out list and removes it from these lists ... it DOES NOT DELETE THE PARTICLE or its relations. You could delete the particle too as follows: delete vtx->remove_particle(particle); this finds *particle in the in and/or out list and removes it from these lists ... it DOES NOT DELETE THE PARTICLE or its relations. you could delete the particle too as follows: delete vtx->remove_particle(particle); or if the particle has an end vertex, you could: delete vtx->remove_particle(particle)->end_vertex(); which would delete the particle's end vertex, and thus would also delete the particle, since the particle would be owned by the end vertex.

this finds *particle in the in and/or out list and removes it from these lists ... it DOES NOT DELETE THE PARTICLE or its relations. you could delete the particle too as follows: delete vtx->remove_particle(particle); or if the particle has an end vertex, you could: delete vtx->remove_particle(particle)->end_vertex(); which would delete the particle's end vertex, and thus would also delete the particle, since the particle would be owned by the end vertex.

Definition at line 295 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenParticle::end_vertex(), HepMC::GenParticle::production_vertex(), remove_particle_in(), remove_particle_out(), HepMC::GenParticle::set_end_vertex_(), and HepMC::GenParticle::set_production_vertex_().

Referenced by filterEvent().

7.19.4.69 void HepMC::GenVertex::remove_particle_in (GenParticle * *particle*) [protected]

for internal use only - remove particle from incoming list

this finds *particle in m_particles_in and removes it from that list

this finds *particle in m_particles_in and removes it from that list

Definition at line 317 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::already_in_vector().

Referenced by add_particle_in(), and remove_particle().

7.19.4.70 void HepMC::GenVertex::remove_particle_in (GenParticle *) [protected]

for internal use only - remove particle from incoming list

7.19.4.71 void HepMC::GenVertex::remove_particle_out (GenParticle *) [protected]

for internal use only - remove particle from outgoing list

7.19.4.72 void HepMC::GenVertex::remove_particle_out (GenParticle * *particle*) [protected]

for internal use only - remove particle from outgoing list

this finds *particle in m_particles_out and removes it from that list

this finds *particle in m_particles_out and removes it from that list

Definition at line 323 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::already_in_vector().

Referenced by add_particle_out(), and remove_particle().

7.19.4.73 void HepMC::GenVertex::set_barcode_ (int *the_bar_code*) [inline], [protected]

set identifier

Definition at line 417 of file GenVertex.h.

Referenced by HepMC::GenEvent::set_barcode(), and suggest_barcode().

7.19.4.74 void HepMC::GenVertex::set_barcode_ (int *the_bar_code*) [protected]

set identifier

7.19.4.75 void HepMC::GenVertex::set_id (int *id*) [inline]

set vertex ID

Definition at line 428 of file GenVertex.h.

Referenced by HepMC::detail::read_vertex().

7.19.4.76 void HepMC::GenVertex::set_id (int *id*)

set vertex ID

7.19.4.77 void HepMC::GenVertex::set_parent_event_ (GenEvent * *evt*) [protected]

set parent event

only the **GenEvent** (p. 72) (friend) is allowed to set the parent_event, and barcode. It is done automatically anytime you add a vertex to an event

Definition at line 388 of file HepMC-2.06.09/src/GenVertex.cc.

References barcode(), particles_in_const_begin(), particles_in_const_end(), particles_out_const_begin(), particles_out_const_end(), HepMC::GenEvent::remove_barcode(), and HepMC::GenEvent::set_barcode().

Referenced by HepMC::GenEvent::add_vertex(), and HepMC::GenEvent::remove_vertex().

7.19.4.78 void HepMC::GenVertex::set_parent_event_ (GenEvent * *evt*) [protected]

set parent event

only the **GenEvent** (p. 72) (friend) is allowed to set the parent_event, and barcode. It is done automatically anytime you add a vertex to an event

7.19.4.79 void HepMC::GenVertex::set_position (const FourVector & *position* = FourVector(0, 0, 0, 0))

set vertex position and time

7.19.4.80 void HepMC::GenVertex::set_position (const FourVector & *position* = FourVector(0,0,0,0)) [inline]

set vertex position and time

Definition at line 424 of file GenVertex.h.

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), and HepMC::detail::read_vertex().

7.19.4.81 bool HepMC::GenVertex::suggest_barcode (int *the_bar_code*)

In general there is no reason to "suggest_barcode".

7.19.4.82 bool HepMC::GenVertex::suggest_barcode (int *the_bar_code*)

In general there is no reason to "suggest_barcode".

allows a barcode to be suggested for this vertex. In general it is better to let the event pick the barcode for you, which is automatic. Returns TRUE if the suggested barcode has been accepted (i.e. the suggested barcode has not already been used in the event, and so it was used). Returns FALSE if the suggested barcode was rejected, or if the vertex is not yet part of an event, such that it is not yet possible to know if the suggested barcode will be accepted).

allows a barcode to be suggested for this vertex. In general it is better to let the event pick the barcode for you, which is automatic. Returns TRUE if the suggested barcode has been accepted (i.e. the suggested barcode has not already been used in the event, and so it was used). Returns FALSE if the suggested barcode was rejected, or if the vertex is not yet part of an event, such that it is not yet possible to know if the suggested barcode will be accepted).

Definition at line 363 of file HepMC-2.06.09/src/GenVertex.cc.

References parent_event(), HepMC::GenEvent::set_barcode(), and set_barcode_().

Referenced by HepMC::GenEvent::GenEvent(), GenVertex(), and HepMC::detail::read_vertex().

7.19.4.83 void HepMC::GenVertex::swap (GenVertex & *other*)

swap

7.19.4.84 void HepMC::GenVertex::swap (GenVertex & *other*)

swap

Definition at line 71 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::WeightContainer::swap(), and HepMC::FourVector::swap().

Referenced by operator=().

7.19.4.85 vertex_iterator HepMC::GenVertex::vertices_begin (IteratorRange *range* = relatives)

begin vertex range

7.19.4.86 GenVertex::vertex_iterator HepMC::GenVertex::vertices_begin (IteratorRange *range* = relatives) [inline]

begin vertex range

Definition at line 504 of file GenVertex.h.

References vertex_iterator.

7.19.4.87 vertex_iterator HepMC::GenVertex::vertices_end (IteratorRange)

end vertex range

7.19.4.88 GenVertex::vertex_iterator HepMC::GenVertex::vertices_end (IteratorRange) [inline]

end vertex range

Definition at line 510 of file GenVertex.h.

References vertex_iterator.

7.19.4.89 WeightContainer & HepMC::GenVertex::weights ( ) [inline]

direct access to the weights container is allowed.

Definition at line 419 of file GenVertex.h.

Referenced by print(), and HepMC::detail::read_vertex().

7.19.4.90 WeightContainer& HepMC::GenVertex::weights ( )

direct access to the weights container is allowed.

7.19.4.91 const WeightContainer& HepMC::GenVertex::weights ( ) const

const direct access to the weights container

7.19.4.92 const WeightContainer & HepMC::GenVertex::weights ( ) const [inline]

const direct access to the weights container

Definition at line 421 of file GenVertex.h.

7.19.5 Friends And Related Function Documentation

7.19.5.1 edge_iterator [friend]

Definition at line 233 of file GenVertex.h.

Referenced by edges_begin(), edges_end(), HepMC::GenVertex::particle_iterator::operator++(), and HepMC::GenVertex::particle_iterator::particle_iterator().

7.19.5.2 GenEvent [friend]

Definition at line 56 of file GenVertex.h.

7.19.5.3 std::ostream& operator<< (std::ostream & *ostr,* const GenVertex & *vtx*) [friend]

print vertex information

Definition at line 440 of file HepMC-2.06.09/src/GenVertex.cc.

7.19.5.4 std::ostream& operator<< (std::ostream & *ostr,* const GenVertex & *vtx*) [friend]

print vertex information

print vertex information

Definition at line 440 of file HepMC-2.06.09/src/GenVertex.cc.

7.19.5.5 particle_iterator [friend]

Definition at line 366 of file GenVertex.h.

Referenced by particles_begin(), and particles_end().

7.19.5.6 vertex_iterator [friend]

Definition at line 318 of file GenVertex.h.

Referenced by HepMC::GenVertex::vertex_iterator::copy_recursive_iterator_(), HepMC::GenVertex::vertex_iterator::follow_edge_(), HepMC::GenVertex::particle_iterator::particle_iterator(), vertices_begin(), and vertices_end().

The documentation for this class was generated from the following files:

- **GenVertex.h**
- **2.06.09/HepMC/GenVertex.h**
- **HepMC-2.06.09/src/GenRanges.cc**
- **HepMC-2.06.09/src/GenVertex.cc**
- **src/GenRanges.cc**
- **src/GenVertex.cc**

7.20 HepMC::GenVertexParticleRange Class Reference

GenVertexParticleRange (p. 144) acts like a collection of particles.

`#include <GenRanges.h>`

Public Member Functions

- **GenVertexParticleRange** (**GenVertex** &**v**, **IteratorRange** range=**relatives**)

  *the constructor requires a **GenVertex** (p. 125)*
- **GenVertex::particle_iterator begin** ()
- **GenVertex::particle_iterator end** ()
- **GenVertexParticleRange** (**GenVertex** &**v**, **IteratorRange** range=**relatives**)

  *the constructor requires a **GenVertex** (p. 125)*
- **GenVertex::particle_iterator begin** ()
- **GenVertex::particle_iterator end** ()

7.20.1 Detailed Description

GenVertexParticleRange (p. 144) acts like a collection of particles.

HepMC::GenVertexParticleRange (p. 144) is used to mimic a collection of particles for ease of use - especially with utilities such as the Boost foreach funtion

Definition at line 140 of file GenRanges.h.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 HepMC::GenVertexParticleRange::GenVertexParticleRange (GenVertex & *v*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenVertex** (p. 125)

Definition at line 145 of file GenRanges.h.

7.20.2.2 HepMC::GenVertexParticleRange::GenVertexParticleRange (GenVertex & *v*, IteratorRange *range* = relatives) [inline]

the constructor requires a **GenVertex** (p. 125)

Definition at line 145 of file 2.06.09/HepMC/GenRanges.h.

7.20.3 Member Function Documentation

7.20.3.1 GenVertex::particle_iterator HepMC::GenVertexParticleRange::begin ( ) [inline]

Definition at line 148 of file GenRanges.h.

References HepMC::GenVertex::particles_begin().

7.20.3.2 GenVertex::particle_iterator HepMC::GenVertexParticleRange::begin ( ) [inline]

Definition at line 148 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenVertex::particles_begin().

7.20.3.3 GenVertex::particle_iterator HepMC::GenVertexParticleRange::end ( ) [inline]

Definition at line 149 of file 2.06.09/HepMC/GenRanges.h.

References HepMC::GenVertex::particles_end().

7.20.3.4 GenVertex::particle_iterator HepMC::GenVertexParticleRange::end ( ) [inline]

Definition at line 149 of file GenRanges.h.

References HepMC::GenVertex::particles_end().

The documentation for this class was generated from the following files:

- **GenRanges.h**
- **2.06.09/HepMC/GenRanges.h**

7.21 HepMC::Heavyion Class Reference

The **Heavyion** (p. 146) class stores information about heavy ions.

`#include <HeavyIon.h>`

Public Member Functions

- **Heavyion** ()
  default constructor
- **Heavyion** (int nh, int np, int nt, int nc, int ns, int nsp, int nnw=0, int nwn=0, int nwnw=0, float im=0., float pl=0., float ec=0., float s=0., float fc=0.)
  The first 6 values must be provided.
- ~**Heavyion** ()
- **Heavyion** (**Heavyion** const &orig)
  copy constructor
- **Heavyion** & **operator=** (**Heavyion** const &rhs)
  make a copy
- void **swap** (**Heavyion** &other)
  *swap two **Heavyion** (p. 146) objects*
- bool **operator==** (const **Heavyion** &) const
  check for equality
- bool **operator!=** (const **Heavyion** &) const
  check for inequality
- int **Ncoll_hard** () const
  Number of hard scatterings.
- int **Npart_proj** () const
  Number of projectile participants.
- int **Npart_targ** () const
  Number of target participants.
- int **Ncoll** () const
  Number of NN (nucleon-nucleon) collisions.
- int **spectator_neutrons** () const
  Number of spectator neutrons.
- int **spectator_protons** () const
  Number of spectator protons.
- int **N_Nwounded_collisions** () const
  Number of N-Nwounded collisions.
- int **Nwounded_N_collisions** () const
  Number of Nwounded-N collisons.
- int **Nwounded_Nwounded_collisions** () const

Number of Nwounded-Nwounded collisions.

- float **impact_parameter** () const

  Impact Parameter(in fm) of collision.
- float **event_plane_angle** () const

  Azimuthal angle of event plane.
- float **eccentricity** () const
- float **sigma_inel_NN** () const

  nucleon-nucleon inelastic (including diffractive) cross-section
- float **centrality** () const

  centrality (percentile of geometric cross section. Negaitve if not set.)
- bool **is_valid** () const

  verify that the instance contains non-zero information
- void **set_Ncoll_hard** (const int &i)

  set number of hard scatterings
- void **set_Npart_proj** (const int &i)

  set number of projectile participants
- void **set_Npart_targ** (const int &i)

  set number of target participants
- void **set_Ncoll** (const int &i)

  set number of NN (nucleon-nucleon) collisions
- void **set_spectator_neutrons** (const int &i)

  set number of spectator neutrons
- void **set_spectator_protons** (const int &i)

  set number of spectator protons
- void **set_N_Nwounded_collisions** (const int &i)

  set number of N-Nwounded collisions
- void **set_Nwounded_N_collisions** (const int &i)

  set number of Nwounded-N collisons
- void **set_Nwounded_Nwounded_collisions** (const int &i)

  set number of Nwounded-Nwounded collisions
- void **set_impact_parameter** (const float &f)

  set Impact Parameter in fm
- void **set_event_plane_angle** (const float &f)

  set azimuthal angle of event plane
- void **set_eccentricity** (const float &f)

  set eccentricity of participating nucleons in the transverse plane
- void **set_sigma_inel_NN** (const float &f)

  set nucleon-nucleon inelastic cross-section
- void **set_centrality** (const float &f)

  set centrality percentile [0:100]
- **Heavylon** ()

  default constructor
- **Heavylon** (int nh, int np, int nt, int nc, int ns, int nsp, int nnw=0, int nwn=0, int nwnw=0, float im=0., float pl=0., float ec=0., float s=0., float fc=0.)

  The first 6 values must be provided.
- ~**Heavylon** ()
- **Heavylon** (**Heavylon** const &orig)

  copy constructor
- **Heavylon** & **operator=** (**Heavylon** const &rhs)

  make a copy
- void **swap** (**Heavylon** &other)

*swap two **Heavylon** (p. 146) objects*

- bool **operator==** (const **Heavylon** &) const

  check for equality

- bool **operator!=** (const **Heavylon** &) const

  check for inequality

- int **Ncoll_hard** () const

  Number of hard scatterings.

- int **Npart_proj** () const

  Number of projectile participants.

- int **Npart_targ** () const

  Number of target participants.

- int **Ncoll** () const

  Number of NN (nucleon-nucleon) collisions.

- int **spectator_neutrons** () const

  Number of spectator neutrons.

- int **spectator_protons** () const

  Number of spectator protons.

- int **N_Nwounded_collisions** () const

  Number of N-Nwounded collisions.

- int **Nwounded_N_collisions** () const

  Number of Nwounded-N collisons.

- int **Nwounded_Nwounded_collisions** () const

  Number of Nwounded-Nwounded collisions.

- float **impact_parameter** () const

  Impact Parameter(in fm) of collision.

- float **event_plane_angle** () const

  Azimuthal angle of event plane.

- float **eccentricity** () const
- float **sigma_inel_NN** () const

  nucleon-nucleon inelastic (including diffractive) cross-section

- float **centrality** () const

  centrality (percentile of geometric cross section. Negaitve if not set.)

- bool **is_valid** () const

  verify that the instance contains non-zero information

- void **set_Ncoll_hard** (const int &i)

  set number of hard scatterings

- void **set_Npart_proj** (const int &i)

  set number of projectile participants

- void **set_Npart_targ** (const int &i)

  set number of target participants

- void **set_Ncoll** (const int &i)

  set number of NN (nucleon-nucleon) collisions

- void **set_spectator_neutrons** (const int &i)

  set number of spectator neutrons

- void **set_spectator_protons** (const int &i)

  set number of spectator protons

- void **set_N_Nwounded_collisions** (const int &i)

  set number of N-Nwounded collisions

- void **set_Nwounded_N_collisions** (const int &i)

  set number of Nwounded-N collisons

- void **set_Nwounded_Nwounded_collisions** (const int &i)

set number of Nwounded-Nwounded collisions

- void **set_impact_parameter** (const float &f)

  set Impact Parameter in fm

- void **set_event_plane_angle** (const float &f)

  set azimuthal angle of event plane

- void **set_eccentricity** (const float &f)

  set eccentricity of participating nucleons in the transverse plane

- void **set_sigma_inel_NN** (const float &f)

  set nucleon-nucleon inelastic cross-section

- void **set_centrality** (const float &f)

  set centrality percentile [0:100]

7.21.1 Detailed Description

The **HeavyIon** (p. 146) class stores information about heavy ions.

HepMC::HeavyIon (p. 146) provides additional information storage for Heavy Ion generators in **GenEvent** (p. 72). Creation and use of this information is optional.

Examples:

testMass.cc.in.

Definition at line 51 of file HeavyIon.h.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 HepMC::HeavyIon::HeavyIon ( ) [inline]

default constructor

Definition at line 56 of file HeavyIon.h.

7.21.2.2 HepMC::HeavyIon::HeavyIon (int *nh*, int *np*, int *nt*, int *nc*, int *ns*, int *nsp*, int *nnw* = 0, int *nwn* = 0, int *nwnw* = 0, float *im* = 0., float *pl* = 0., float *ec* = 0., float *s* = 0., float *c* = 0.) [inline]

The first 6 values must be provided.

Required members are the number of hard scatterings, the number of projectile participants. the number of target participants. the number of nucleon-nucleon collisions, the number of spectator neutrons, and the number of spectator protons.

Definition at line 190 of file HeavyIon.h.

7.21.2.3 HepMC::HeavyIon::~HeavyIon ( ) [inline]

Definition at line 79 of file HeavyIon.h.

7.21.2.4 HepMC::HeavyIon::HeavyIon (HeavyIon const & *orig*) [inline]

copy constructor

Definition at line 209 of file HeavyIon.h.

7.21.2.5 HepMC::HeavyIon::HeavyIon ( ) [inline]

default constructor

Definition at line 56 of file 2.06.09/HepMC/HeavyIon.h.

7.21.2.6 HepMC::HeavyIon::HeavyIon (int *nh*, int *np*, int *nt*, int *nc*, int *ns*, int *nsp*, int *nnw* = 0, int *nwn* = 0, int *nwnw* = 0, float *im* = 0., float *pl* = 0., float *ec* = 0., float *s* = 0., float *fc* = 0.)

The first 6 values must be provided.

7.21.2.7 HepMC::HeavyIon::~HeavyIon ( ) [inline]

Definition at line 79 of file 2.06.09/HepMC/HeavyIon.h.

7.21.2.8 HepMC::HeavyIon::HeavyIon (HeavyIon const & *orig*)

copy constructor

7.21.3 Member Function Documentation

7.21.3.1 float HepMC::HeavyIon::centrality ( ) const [inline]

centrality (percentile of geometric cross section. Negaitve if not set.)

Definition at line 121 of file HeavyIon.h.

Referenced by operator==().

7.21.3.2 float HepMC::HeavyIon::centrality ( ) const [inline]

centrality (percentile of geometric cross section. Negaitve if not set.)

Definition at line 121 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.3 float HepMC::HeavyIon::eccentricity ( ) const [inline]

eccentricity of participating nucleons in the transverse plane (as in phobos nucl-ex/0510031)

Definition at line 117 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.4 float HepMC::HeavyIon::eccentricity ( ) const [inline]

eccentricity of participating nucleons in the transverse plane (as in phobos nucl-ex/0510031)

Definition at line 117 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.5 float HepMC::HeavyIon::event_plane_angle ( ) const [inline]

Azimuthal angle of event plane.

Definition at line 114 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.6 float HepMC::HeavyIon::event_plane_angle ( ) const [inline]

Azimuthal angle of event plane.

Definition at line 114 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.7 float HepMC::HeavyIon::impact_parameter ( ) const [inline]

Impact Parameter(in fm) of collision.

Definition at line 112 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.8 float HepMC::HeavyIon::impact_parameter ( ) const [inline]

Impact Parameter(in fm) of collision.

Definition at line 112 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.9 bool HepMC::HeavyIon::is_valid ( ) const [inline]

verify that the instance contains non-zero information

Definition at line 275 of file HeavyIon.h.

Referenced by HepMC::GenEvent::read().

7.21.3.10 bool HepMC::HeavyIon::is_valid ( ) const

verify that the instance contains non-zero information

7.21.3.11 int HepMC::HeavyIon::N_Nwounded_collisions ( ) const [inline]

Number of N-Nwounded collisions.

Definition at line 106 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.12 int HepMC::HeavyIon::N_Nwounded_collisions ( ) const [inline]

Number of N-Nwounded collisions.

Definition at line 106 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.13 int HepMC::HeavyIon::Ncoll ( ) const [inline]

Number of NN (nucleon-nucleon) collisions.

Definition at line 100 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.14 int HepMC::HeavyIon::Ncoll ( ) const [inline]

Number of NN (nucleon-nucleon) collisions.

Definition at line 100 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.15 int HepMC::HeavyIon::Ncoll_hard ( ) const [inline]

Number of hard scatterings.

Definition at line 94 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.16 int HepMC::HeavyIon::Ncoll_hard ( ) const [inline]

Number of hard scatterings.

Definition at line 94 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.17 int HepMC::HeavyIon::Npart_proj ( ) const [inline]

Number of projectile participants.

Definition at line 96 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.18 int HepMC::HeavyIon::Npart_proj ( ) const [inline]

Number of projectile participants.

Definition at line 96 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.19 int HepMC::HeavyIon::Npart_targ ( ) const [inline]

Number of target participants.

Definition at line 98 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.20 int HepMC::HeavyIon::Npart_targ ( ) const [inline]

Number of target participants.

Definition at line 98 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.21 int HepMC::HeavyIon::Nwounded_N_collisions ( ) const [inline]

Number of Nwounded-N collisons.

Definition at line 108 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.22 int HepMC::HeavyIon::Nwounded_N_collisions ( ) const [inline]

Number of Nwounded-N collisons.

Definition at line 108 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.23 int HepMC::HeavyIon::Nwounded_Nwounded_collisions ( ) const [inline]

Number of Nwounded-Nwounded collisions.

Definition at line 110 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.24 int HepMC::HeavyIon::Nwounded_Nwounded_collisions ( ) const [inline]

Number of Nwounded-Nwounded collisions.

Definition at line 110 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.25 bool HepMC::HeavyIon::operator!= (const HeavyIon & *a*) const [inline]

check for inequality

any nonmatching member generates inequality

any nonmatching member generates inequality

Definition at line 269 of file HeavyIon.h.

7.21.3.26 bool HepMC::HeavyIon::operator!= (const HeavyIon &) const

check for inequality

7.21.3.27 HeavyIon& HepMC::HeavyIon::operator= (HeavyIon const & *rhs*)

make a copy

7.21.3.28 HeavyIon & HepMC::HeavyIon::operator= (HeavyIon const & *rhs*) [inline]

make a copy

Definition at line 226 of file HeavyIon.h.

References swap().

7.21.3.29 bool HepMC::HeavyIon::operator== (const HeavyIon &) const

check for equality

7.21.3.30 bool HepMC::HeavyIon::operator== (const HeavyIon & *a*) const [inline]

check for equality

equality requires that each member match

equality requires that each member match

Definition at line 251 of file HeavyIon.h.

References centrality(), eccentricity(), event_plane_angle(), impact_parameter(), N_Nwounded_collisions(), Ncoll(), Ncoll_hard(), Npart_proj(), Npart_targ(), Nwounded_N_collisions(), Nwounded_Nwounded_collisions(), sigma_inel_NN(), spectator_neutrons(), and spectator_protons().

7.21.3.31 void HepMC::HeavyIon::set_centrality (const float & *f*) [inline]

set centrality percentile [0:100]

Definition at line 155 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.32 void HepMC::HeavyIon::set_centrality (const float & *f*) [inline]

set centrality percentile [0:100]

Definition at line 155 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.33 void HepMC::HeavyIon::set_eccentricity (const float & *f*) [inline]

set eccentricity of participating nucleons in the transverse plane

Definition at line 151 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.34 void HepMC::HeavyIon::set_eccentricity (const float & *f*) [inline]

set eccentricity of participating nucleons in the transverse plane

Definition at line 151 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.35 void HepMC::HeavyIon::set_event_plane_angle (const float & *f*) [inline]

set azimuthal angle of event plane

Definition at line 149 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.36 void HepMC::HeavyIon::set_event_plane_angle (const float & *f*) [inline]

set azimuthal angle of event plane

Definition at line 149 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.37 void HepMC::HeavyIon::set_impact_parameter (const float & *f*) [inline]

set Impact Parameter in fm

Definition at line 147 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.38 void HepMC::HeavyIon::set_impact_parameter (const float & *f*) [inline]

set Impact Parameter in fm

Definition at line 147 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.39 void HepMC::HeavyIon::set_N_Nwounded_collisions (const int & *i*) [inline]

set number of N-Nwounded collisions

Definition at line 140 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.40 void HepMC::HeavyIon::set_N_Nwounded_collisions (const int & *i*) [inline]

set number of N-Nwounded collisions

Definition at line 140 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.41 void HepMC::HeavyIon::set_Ncoll (const int & *i*) [inline]

set number of NN (nucleon-nucleon) collisions

Definition at line 134 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.42 void HepMC::HeavyIon::set_Ncoll (const int & *i*) [inline]

set number of NN (nucleon-nucleon) collisions

Definition at line 134 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.43 void HepMC::HeavyIon::set_Ncoll_hard (const int & *i*) [inline]

set number of hard scatterings

Definition at line 128 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.44 void HepMC::HeavyIon::set_Ncoll_hard (const int & *i*) [inline]

set number of hard scatterings

Definition at line 128 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.45 void HepMC::HeavyIon::set_Npart_proj (const int & *i*) [inline]

set number of projectile participants

Definition at line 130 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.46 void HepMC::HeavyIon::set_Npart_proj (const int & *i*) [inline]

set number of projectile participants

Definition at line 130 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.47 void HepMC::HeavyIon::set_Npart_targ (const int & *i*) [inline]

set number of target participants

Definition at line 132 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.48 void HepMC::HeavyIon::set_Npart_targ (const int & *i*) [inline]

set number of target participants

Definition at line 132 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.49 void HepMC::HeavyIon::set_Nwounded_N_collisions (const int & *i*) [inline]

set number of Nwounded-N collisons

Definition at line 142 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.50 void HepMC::HeavyIon::set_Nwounded_N_collisions (const int & *i*) [inline]

set number of Nwounded-N collisons

Definition at line 142 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.51 void HepMC::HeavyIon::set_Nwounded_Nwounded_collisions (const int & *i*) [inline]

set number of Nwounded-Nwounded collisions

Definition at line 144 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.52 void HepMC::HeavyIon::set_Nwounded_Nwounded_collisions (const int & *i*) [inline]

set number of Nwounded-Nwounded collisions

Definition at line 144 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.53 void HepMC::HeavyIon::set_sigma_inel_NN (const float & *f*) [inline]

set nucleon-nucleon inelastic cross-section

Definition at line 153 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.54 void HepMC::HeavyIon::set_sigma_inel_NN (const float & *f*) [inline]

set nucleon-nucleon inelastic cross-section

Definition at line 153 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.55 void HepMC::HeavyIon::set_spectator_neutrons (const int & *i*) [inline]

set number of spectator neutrons

Definition at line 136 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.56 void HepMC::HeavyIon::set_spectator_neutrons (const int & *i*) [inline]

set number of spectator neutrons

Definition at line 136 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.57 void HepMC::HeavyIon::set_spectator_protons (const int & *i*) [inline]

set number of spectator protons

Definition at line 138 of file HeavyIon.h.

Referenced by HepMC::operator>>().

7.21.3.58 void HepMC::HeavyIon::set_spectator_protons (const int & *i*) [inline]

set number of spectator protons

Definition at line 138 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.59 float HepMC::HeavyIon::sigma_inel_NN ( ) const [inline]

nucleon-nucleon inelastic (including diffractive) cross-section

Definition at line 119 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.60 float HepMC::HeavyIon::sigma_inel_NN ( ) const [inline]

nucleon-nucleon inelastic (including diffractive) cross-section

Definition at line 119 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.61 int HepMC::HeavyIon::spectator_neutrons ( ) const [inline]

Number of spectator neutrons.

Definition at line 102 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.62 int HepMC::HeavyIon::spectator_neutrons ( ) const [inline]

Number of spectator neutrons.

Definition at line 102 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.63 int HepMC::HeavyIon::spectator_protons ( ) const [inline]

Number of spectator protons.

Definition at line 104 of file HeavyIon.h.

Referenced by HepMC::operator<<(), and operator==().

7.21.3.64 int HepMC::HeavyIon::spectator_protons ( ) const [inline]

Number of spectator protons.

Definition at line 104 of file 2.06.09/HepMC/HeavyIon.h.

7.21.3.65 void HepMC::HeavyIon::swap (HeavyIon & *other*) [inline]

swap two **HeavyIon** (p. 146) objects

Definition at line 233 of file HeavyIon.h.

Referenced by operator=().

7.21.3.66 void HepMC::HeavyIon::swap (HeavyIon & *other*)

swap two **HeavyIon** (p. 146) objects

The documentation for this class was generated from the following files:

- **HeavyIon.h**
- **2.06.09/HepMC/HeavyIon.h**

7.22 HepMC::HEPEVT_Wrapper Class Reference

Generic Wrapper for the fortran HEPEVT common block.

`#include <HEPEVT_Wrapper.h>`

Static Public Member Functions

- static void **print_hepevt** (std::ostream &ostr=std::cout)
  write information from HEPEVT common block
- static void **print_hepevt_particle** (int index, std::ostream &ostr=std::cout)
  write particle information to ostr
- static bool **is_double_precision** ()
  True if common block uses double.
- static bool **check_hepevt_consistency** (std::ostream &ostr=std::cout)
  check for problems with HEPEVT common block
- static void **zero_everything** ()
  set all entries in HEPEVT to zero
- static int **event_number** ()
  event number
- static int **number_entries** ()
  num entries in current evt
- static int **status** (int index)

status code

- static int **id** (int index)

  PDG particle id.
- static int **first_parent** (int index)

  index of 1st mother
- static int **last_parent** (int index)

  index of last mother
- static int **number_parents** (int index)

  number of parents
- static int **first_child** (int index)

  index of 1st daughter
- static int **last_child** (int index)

  index of last daughter
- static int **number_children** (int index)

  number of children
- static double **px** (int index)

  X momentum.
- static double **py** (int index)

  Y momentum.
- static double **pz** (int index)

  Z momentum.
- static double **e** (int index)

  Energy.
- static double **m** (int index)

  generated mass
- static double **x** (int index)

  X Production vertex.
- static double **y** (int index)

  Y Production vertex.
- static double **z** (int index)

  Z Production vertex.
- static double **t** (int index)

  production time
- static void **set_event_number** (int evtno)

  set event number
- static void **set_number_entries** (int noentries)

  set number of entries in HEPEVT
- static void **set_status** (int index, int **status**)

  set particle status
- static void **set_id** (int index, int **id**)

  set particle ID
- static void **set_parents** (int index, int firstparent, int lastparent)

  define parents of a particle
- static void **set_children** (int index, int firstchild, int lastchild)

  define children of a particle
- static void **set_momentum** (int index, double **px**, double **py**, double **pz**, double **e**)

  set particle momentum
- static void **set_mass** (int index, double mass)

  set particle mass
- static void **set_position** (int index, double **x**, double **y**, double **z**, double **t**)

  set particle production vertex

- static unsigned int **sizeof_int** ()

  size of integer in bytes
- static unsigned int **sizeof_real** ()

  size of real in bytes
- static int **max_number_entries** ()

  size of common block
- static void **set_sizeof_int** (unsigned int)

  define size of integer
- static void **set_sizeof_real** (unsigned int)

  define size of real
- static void **set_max_number_entries** (unsigned int)

  define size of common block
- static void **print_hepevt** (std::ostream &ostr=std::cout)

  write information from HEPEVT common block
- static void **print_hepevt_particle** (int index, std::ostream &ostr=std::cout)

  write particle information to ostr
- static bool **is_double_precision** ()

  True if common block uses double.
- static bool **check_hepevt_consistency** (std::ostream &ostr=std::cout)

  check for problems with HEPEVT common block
- static void **zero_everything** ()

  set all entries in HEPEVT to zero
- static int **event_number** ()

  event number
- static int **number_entries** ()

  num entries in current evt
- static int **status** (int index)

  status code
- static int **id** (int index)

  PDG particle id.
- static int **first_parent** (int index)

  index of 1st mother
- static int **last_parent** (int index)

  index of last mother
- static int **number_parents** (int index)

  number of parents
- static int **first_child** (int index)

  index of 1st daughter
- static int **last_child** (int index)

  index of last daughter
- static int **number_children** (int index)

  number of children
- static double **px** (int index)

  X momentum.
- static double **py** (int index)

  Y momentum.
- static double **pz** (int index)

  Z momentum.
- static double **e** (int index)

  Energy.
- static double **m** (int index)

generated mass

- static double **x** (int index)

  X Production vertex.
- static double **y** (int index)

  Y Production vertex.
- static double **z** (int index)

  Z Production vertex.
- static double **t** (int index)

  production time
- static void **set_event_number** (int evtno)

  set event number
- static void **set_number_entries** (int noentries)

  set number of entries in HEPEVT
- static void **set_status** (int index, int **status**)

  set particle status
- static void **set_id** (int index, int **id**)

  set particle ID
- static void **set_parents** (int index, int firstparent, int lastparent)

  define parents of a particle
- static void **set_children** (int index, int firstchild, int lastchild)

  define children of a particle
- static void **set_momentum** (int index, double **px**, double **py**, double **pz**, double **e**)

  set particle momentum
- static void **set_mass** (int index, double mass)

  set particle mass
- static void **set_position** (int index, double **x**, double **y**, double **z**, double **t**)

  set particle production vertex
- static unsigned int **sizeof_int** ()

  size of integer in bytes
- static unsigned int **sizeof_real** ()

  size of real in bytes
- static int **max_number_entries** ()

  size of common block
- static void **set_sizeof_int** (unsigned int)

  define size of integer
- static void **set_sizeof_real** (unsigned int)

  define size of real
- static void **set_max_number_entries** (unsigned int)

  define size of common block

Static Protected Member Functions

- static double **byte_num_to_double** (unsigned int)

  navigate a byte array
- static int **byte_num_to_int** (unsigned int)

  navigate a byte array
- static void **write_byte_num** (double, unsigned int)

  pretend common block is an array of bytes
- static void **write_byte_num** (int, unsigned int)

  pretend common block is an array of bytes

- static void **print_legend** (std::ostream &ostr=std::cout)

  print output legend
- static double **byte_num_to_double** (unsigned int)

  navigate a byte array
- static int **byte_num_to_int** (unsigned int)

  navigate a byte array
- static void **write_byte_num** (double, unsigned int)

  pretend common block is an array of bytes
- static void **write_byte_num** (int, unsigned int)

  pretend common block is an array of bytes
- static void **print_legend** (std::ostream &ostr=std::cout)

  print output legend

7.22.1 Detailed Description

Generic Wrapper for the fortran HEPEVT common block.

This class is intended for static use only - it makes no sense to instantiate it.

Definition at line 130 of file HEPEVT_Wrapper.h.

7.22.2 Member Function Documentation

7.22.2.1 double HepMC::HEPEVT_Wrapper::byte_num_to_double (unsigned int *b*) [inline], [static], [protected]

navigate a byte array

Definition at line 255 of file HEPEVT_Wrapper.h.

References hepevt, and hepevt_bytes_allocation.

Referenced by e(), m(), px(), py(), pz(), t(), x(), y(), and z().

7.22.2.2 static double HepMC::HEPEVT_Wrapper::byte_num_to_double (unsigned *int*) [static], [protected]

navigate a byte array

7.22.2.3 int HepMC::HEPEVT_Wrapper::byte_num_to_int (unsigned int *b*) [inline], [static], [protected]

navigate a byte array

Definition at line 273 of file HEPEVT_Wrapper.h.

References hepevt, and hepevt_bytes_allocation.

Referenced by event_number(), first_child(), first_parent(), id(), last_child(), last_parent(), number_entries(), and status().

7.22.2.4 static int HepMC::HEPEVT_Wrapper::byte_num_to_int (unsigned *int*) [static], [protected]

navigate a byte array

7.22.2.5 bool HepMC::HEPEVT_Wrapper::check_hepevt_consistency (std::ostream & *ostr* = std::cout) [static]

check for problems with HEPEVT common block

This method inspects the HEPEVT common block and looks for inconsistencies in the mother/daughter pointers

This method inspects the HEPEVT common block and looks for inconsistencies in the mother/daughter pointers

Definition at line 88 of file fio/HEPEVT_Wrapper.cc.

References event_number(), first_child(), first_parent(), last_child(), last_parent(), m(), number_entries(), print_hepevt_particle(), and print_legend().

7.22.2.6 static bool HepMC::HEPEVT_Wrapper::check_hepevt_consistency (std::ostream & *ostr* = std::cout) [static]

check for problems with HEPEVT common block

7.22.2.7 double HepMC::HEPEVT_Wrapper::e (int *index*) [inline], [static]

Energy.

Definition at line 446 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.8 static double HepMC::HEPEVT_Wrapper::e (int *index*) [static]

Energy.

7.22.2.9 int HepMC::HEPEVT_Wrapper::event_number ( ) [inline], [static]

event number

Definition at line 343 of file HEPEVT_Wrapper.h.

References byte_num_to_int().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), check_hepevt_consistency(), HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_HERWIG::fill_next_event(), and print_hepevt().

7.22.2.10 static int HepMC::HEPEVT_Wrapper::event_number ( ) [static]

event number

7.22.2.11 int HepMC::HEPEVT_Wrapper::first_child (int *index*) [inline], [static]

index of 1st daughter

Definition at line 394 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), check_hepevt_consistency(), last_child(), number_children(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.12 static int HepMC::HEPEVT_Wrapper::first_child (int *index*) [static]

index of 1st daughter

7.22.2.13 int HepMC::HEPEVT_Wrapper::first_parent (int *index*) [inline], [static]

index of 1st mother

Definition at line 362 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), check_hepevt_consistency(), HepMC::IO_HERWIG::fill_next_event(), last_parent(), number_parents(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.14 static int HepMC::HEPEVT_Wrapper::first_parent (int *index*) [static]

index of 1st mother

7.22.2.15 int HepMC::HEPEVT_Wrapper::id (int *index*) [inline], [static]

PDG particle id.

Definition at line 356 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), and sizeof_int().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.16 static int HepMC::HEPEVT_Wrapper::id (int *index*) [static]

PDG particle id.

7.22.2.17 bool HepMC::HEPEVT_Wrapper::is_double_precision ( ) [inline], [static]

True if common block uses double.

Definition at line 337 of file HEPEVT_Wrapper.h.

References sizeof_real().

Referenced by print_hepevt().

7.22.2.18 static bool HepMC::HEPEVT_Wrapper::is_double_precision ( ) [static]

True if common block uses double.

7.22.2.19 int HepMC::HEPEVT_Wrapper::last_child (int *index*) [inline], [static]

index of last daughter

Definition at line 402 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), first_child(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), check_hepevt_consistency(), number_children(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.20 static int HepMC::HEPEVT_Wrapper::last_child (int *index*) [static]

index of last daughter

7.22.2.21 int HepMC::HEPEVT_Wrapper::last_parent (int *index*) [inline], [static]

index of last mother

Definition at line 370 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), first_parent(), max_number_entries(), number_entries(), and sizeof_int().

Referenced by HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), check_hepevt_consistency(), number_parents(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.22 static int HepMC::HEPEVT_Wrapper::last_parent (int *index*) [static]

index of last mother

7.22.2.23 static double HepMC::HEPEVT_Wrapper::m (int *index*) [static]

generated mass

7.22.2.24 double HepMC::HEPEVT_Wrapper::m (int *index*) [inline], [static]

generated mass

Definition at line 452 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), check_hepevt_consistency(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.25 int HepMC::HEPEVT_Wrapper::max_number_entries ( ) [inline], [static]

size of common block

Definition at line 229 of file HEPEVT_Wrapper.h.

Referenced by e(), first_child(), first_parent(), id(), last_child(), last_parent(), m(), number_entries(), print_hepevt(), px(), py(), pz(), set_children(), set_id(), set_mass(), set_momentum(), set_parents(), set_position(), set_status(), t(), HepMC::IO_HEPEVT::write_event(), x(), y(), z(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.26 static int HepMC::HEPEVT_Wrapper::max_number_entries ( ) [static]

size of common block

7.22.2.27 static int HepMC::HEPEVT_Wrapper::number_children (int *index*) [static]

number of children

7.22.2.28 int HepMC::HEPEVT_Wrapper::number_children (int *index*) [inline], [static]

number of children

Definition at line 420 of file HEPEVT_Wrapper.h.

References first_child(), and last_child().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), and HepMC::IO_HERWIG::build_end_vertex().

7.22.2.29 static int HepMC::HEPEVT_Wrapper::number_entries ( ) [static]

num entries in current evt

7.22.2.30 int HepMC::HEPEVT_Wrapper::number_entries ( ) [inline], [static]

num entries in current evt

Definition at line 346 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), max_number_entries(), and sizeof_int().

Referenced by check_hepevt_consistency(), HepMC::IO_HEPEVT::fill_next_event(), HepMC::IO_HERWIG::fill_next_event(), first_child(), first_parent(), last_child(), last_parent(), print_hepevt(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.31 int HepMC::HEPEVT_Wrapper::number_parents (int *index*) [inline], [static]

number of parents

Definition at line 388 of file HEPEVT_Wrapper.h.

References first_parent(), and last_parent().

Referenced by HepMC::IO_HEPEVT::build_production_vertex(), and HepMC::IO_HERWIG::build_production_vertex().

7.22.2.32 static int HepMC::HEPEVT_Wrapper::number_parents (int *index*) [static]

number of parents

7.22.2.33 static void HepMC::HEPEVT_Wrapper::print_hepevt (std::ostream & *ostr* = std::cout) [static]

write information from HEPEVT common block

7.22.2.34 void HepMC::HEPEVT_Wrapper::print_hepevt (std::ostream & *ostr* = std::cout) [static]

write information from HEPEVT common block

dumps the content of this HEPEVT event to ostr (Width is 80)

dumps the content of this HEPEVT event to ostr (Width is 80)

Examples:

fio/example_MyHerwig.cc.

Definition at line 27 of file fio/HEPEVT_Wrapper.cc.

References event_number(), is_double_precision(), max_number_entries(), number_entries(), print_hepevt_particle(), print_legend(), sizeof_int(), and sizeof_real().

Referenced by main().

7.22.2.35 void HepMC::HEPEVT_Wrapper::print_hepevt_particle (int *index,* std::ostream & *ostr* = std::cout) [static]

write particle information to ostr

dumps the content HEPEVT particle entry i (Width is 120) here i is the C array index (i.e. it starts at 0 ... whereas the fortran array index starts at 1) So if there's 100 particles, the last valid index is 100-1=99

dumps the content HEPEVT particle entry i (Width is 120) here i is the C array index (i.e. it starts at 0 ... whereas the fortran array index starts at 1) So if there's 100 particles, the last valid index is 100-1=99

Definition at line 68 of file fio/HEPEVT_Wrapper.cc.

References e(), first_child(), first_parent(), last_child(), last_parent(), m(), px(), py(), pz(), status(), t(), x(), y(), and z().

Referenced by check_hepevt_consistency(), and print_hepevt().

7.22.2.36 static void HepMC::HEPEVT_Wrapper::print_hepevt_particle (int *index,* std::ostream & *ostr* = std::cout) [static]

write particle information to ostr

7.22.2.37 static void HepMC::HEPEVT_Wrapper::print_legend (std::ostream & *ostr* = std::cout) [static], [protected]

print output legend

7.22.2.38 void HepMC::HEPEVT_Wrapper::print_legend (std::ostream & *ostr* = std::cout) [static], [protected]

print output legend

Definition at line 55 of file fio/HEPEVT_Wrapper.cc.

Referenced by check_hepevt_consistency(), and print_hepevt().

7.22.2.39 double HepMC::HEPEVT_Wrapper::px (int *index*) [inline], [static]

X momentum.

Definition at line 427 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.40 static double HepMC::HEPEVT_Wrapper::px (int *index*) [static]

X momentum.

7.22.2.41 double HepMC::HEPEVT_Wrapper::py (int *index*) [inline], [static]

Y momentum.

Definition at line 433 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.42 static double HepMC::HEPEVT_Wrapper::py (int *index*) [static]

Y momentum.

7.22.2.43 double HepMC::HEPEVT_Wrapper::pz (int *index*) [inline], [static]

Z momentum.

Definition at line 440 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.44 static double HepMC::HEPEVT_Wrapper::pz (int *index*) [static]

Z momentum.

7.22.2.45 static void HepMC::HEPEVT_Wrapper::set_children (int *index,* int *firstchild,* int *lastchild*) [static]

define children of a particle

7.22.2.46 void HepMC::HEPEVT_Wrapper::set_children (int *index,* int *firstchild,* int *lastchild*) [inline], [static]

define children of a particle

Definition at line 514 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HERWIG::repair_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.47 static void HepMC::HEPEVT_Wrapper::set_event_number (int *evtno*) [static]

set event number

7.22.2.48 void HepMC::HEPEVT_Wrapper::set_event_number (int *evtno*) [inline], [static]

set event number

Definition at line 486 of file HEPEVT_Wrapper.h.

References write_byte_num().

Referenced by HepMC::IO_HEPEVT::write_event(), and zero_everything().

7.22.2.49 static void HepMC::HEPEVT_Wrapper::set_id (int *index,* int *id*) [static]

set particle ID

7.22.2.50 void HepMC::HEPEVT_Wrapper::set_id (int *index,* int *id*) [inline], [static]

set particle ID

Definition at line 498 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HERWIG::repair_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.51 void HepMC::HEPEVT_Wrapper::set_mass (int *index,* double *mass*) [inline], [static]

set particle mass

Definition at line 538 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), sizeof_real(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.52 static void HepMC::HEPEVT_Wrapper::set_mass (int *index,* double *mass*) [static]

set particle mass

7.22.2.53 void HepMC::HEPEVT_Wrapper::set_max_number_entries (unsigned int *size*) [inline], [static]

define size of common block

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 251 of file HEPEVT_Wrapper.h.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

7.22.2.54 static void HepMC::HEPEVT_Wrapper::set_max_number_entries (unsigned *int*) [static]

define size of common block

7.22.2.55 static void HepMC::HEPEVT_Wrapper::set_momentum (int *index,* double *px,* double *py,* double *pz,* double *e*) [static]

set particle momentum

7.22.2.56 void HepMC::HEPEVT_Wrapper::set_momentum (int *index,* double *px,* double *py,* double *pz,* double *e*) [inline], [static]

set particle momentum

Definition at line 524 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), sizeof_real(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.57 static void HepMC::HEPEVT_Wrapper::set_number_entries (int *noentries*) [static]

set number of entries in HEPEVT

7.22.2.58 void HepMC::HEPEVT_Wrapper::set_number_entries (int *noentries*) [inline], [static]

set number of entries in HEPEVT

Definition at line 489 of file HEPEVT_Wrapper.h.

References sizeof_int(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), and zero_everything().

7.22.2.59 static void HepMC::HEPEVT_Wrapper::set_parents (int *index,* int *firstparent,* int *lastparent*) [static]

define parents of a particle

7.22.2.60 void HepMC::HEPEVT_Wrapper::set_parents (int *index,* int *firstparent,* int *lastparent*) [inline], [static]

define parents of a particle

Definition at line 504 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HERWIG::repair_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.61 void HepMC::HEPEVT_Wrapper::set_position (int *index,* double *x,* double *y,* double *z,* double *t*) [inline], [static]

set particle production vertex

Definition at line 545 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), sizeof_real(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.62 static void HepMC::HEPEVT_Wrapper::set_position (int *index,* double *x,* double *y,* double *z,* double *t*) [static]

set particle production vertex

7.22.2.63 static void HepMC::HEPEVT_Wrapper::set_sizeof_int (unsigned *int*) [static]

define size of integer

7.22.2.64 void HepMC::HEPEVT_Wrapper::set_sizeof_int (unsigned int *size*) [inline], [static]

define size of integer

Definition at line 232 of file HEPEVT_Wrapper.h.

7.22.2.65 static void HepMC::HEPEVT_Wrapper::set_sizeof_real (unsigned *int*) [static]

define size of real

7.22.2.66 void HepMC::HEPEVT_Wrapper::set_sizeof_real (unsigned int *size*) [inline], [static]

define size of real

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 242 of file HEPEVT_Wrapper.h.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

7.22.2.67 void HepMC::HEPEVT_Wrapper::set_status (int *index,* int *status*) [inline], [static]

set particle status

Definition at line 492 of file HEPEVT_Wrapper.h.

References max_number_entries(), sizeof_int(), and write_byte_num().

Referenced by HepMC::IO_HERWIG::remove_gaps_in_hepevt(), HepMC::IO_HEPEVT::write_event(), zero_everything(), and HepMC::IO_HERWIG::zero_hepevt_entry().

7.22.2.68 static void HepMC::HEPEVT_Wrapper::set_status (int *index,* int *status*) [static]

set particle status

7.22.2.69 unsigned int HepMC::HEPEVT_Wrapper::sizeof_int ( ) [inline], [static]

size of integer in bytes

Definition at line 225 of file HEPEVT_Wrapper.h.

Referenced by e(), first_child(), first_parent(), id(), last_child(), last_parent(), m(), number_entries(), print_hepevt(), px(), py(), pz(), set_children(), set_id(), set_mass(), set_momentum(), set_number_entries(), set_parents(), set_position(), set_status(), status(), t(), x(), y(), and z().

7.22.2.70 static unsigned int HepMC::HEPEVT_Wrapper::sizeof_int ( ) [static]

size of integer in bytes

7.22.2.71 unsigned int HepMC::HEPEVT_Wrapper::sizeof_real ( ) [inline], [static]

size of real in bytes

Definition at line 227 of file HEPEVT_Wrapper.h.

Referenced by e(), is_double_precision(), m(), print_hepevt(), px(), py(), pz(), set_mass(), set_momentum(), set_position(), t(), x(), y(), and z().

7.22.2.72 static unsigned int HepMC::HEPEVT_Wrapper::sizeof_real ( ) [static]

size of real in bytes

7.22.2.73 static int HepMC::HEPEVT_Wrapper::status (int *index*) [static]

status code

7.22.2.74 int HepMC::HEPEVT_Wrapper::status (int *index*) [inline], [static]

status code

Definition at line 353 of file HEPEVT_Wrapper.h.

References byte_num_to_int(), and sizeof_int().

Referenced by HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), HepMC::IO_HERWIG::fill_next_event(), print_hepevt_particle(), HepMC::IO_HERWIG::remove_gaps_in_hepevt(), and HepMC::IO_HERWIG::repair_hepevt().

7.22.2.75 static double HepMC::HEPEVT_Wrapper::t (int *index*) [static]

production time

7.22.2.76 double HepMC::HEPEVT_Wrapper::t (int *index*) [inline], [static]

production time

Definition at line 479 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.77 static void HepMC::HEPEVT_Wrapper::write_byte_num (double , unsigned *int*) [static], [protected]

pretend common block is an array of bytes

7.22.2.78 void HepMC::HEPEVT_Wrapper::write_byte_num (double *in,* unsigned int *b*) [inline], [static], [protected]

pretend common block is an array of bytes

Definition at line 295 of file HEPEVT_Wrapper.h.

References hepevt, and hepevt_bytes_allocation.

Referenced by set_children(), set_event_number(), set_id(), set_mass(), set_momentum(), set_number_entries(), set_parents(), set_position(), and set_status().

7.22.2.79 static void HepMC::HEPEVT_Wrapper::write_byte_num (int , unsigned *int*) [static], [protected]

pretend common block is an array of bytes

7.22.2.80 void HepMC::HEPEVT_Wrapper::write_byte_num (int *in,* unsigned int *b*) [inline], [static], [protected]

pretend common block is an array of bytes

Definition at line 312 of file HEPEVT_Wrapper.h.

References hepevt, and hepevt_bytes_allocation.

7.22.2.81 static double HepMC::HEPEVT_Wrapper::x (int *index*) [static]

X Production vertex.

7.22.2.82 double HepMC::HEPEVT_Wrapper::x (int *index*) [inline], [static]

X Production vertex.

Definition at line 458 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.83 static double HepMC::HEPEVT_Wrapper::y (int *index*) [static]

Y Production vertex.

7.22.2.84 double HepMC::HEPEVT_Wrapper::y (int *index*) [inline], [static]

Y Production vertex.

Definition at line 465 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.85 static double HepMC::HEPEVT_Wrapper::z (int *index*) [static]

Z Production vertex.

7.22.2.86 double HepMC::HEPEVT_Wrapper::z (int *index*) [inline], [static]

Z Production vertex.

Definition at line 472 of file HEPEVT_Wrapper.h.

References byte_num_to_double(), max_number_entries(), sizeof_int(), and sizeof_real().

Referenced by HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), print_hepevt_particle(), and HepMC::IO_HERWIG::remove_gaps_in_hepevt().

7.22.2.87 void HepMC::HEPEVT_Wrapper::zero_everything ( ) [static]

set all entries in HEPEVT to zero

Definition at line 212 of file fio/HEPEVT_Wrapper.cc.

References max_number_entries(), set_children(), set_event_number(), set_id(), set_mass(), set_momentum(), set_number_entries(), set_parents(), set_position(), and set_status().

7.22.2.88 static void HepMC::HEPEVT_Wrapper::zero_everything ( ) [static]

set all entries in HEPEVT to zero

The documentation for this class was generated from the following files:

- **HEPEVT_Wrapper.h**
- **2.06.09/HepMC/HEPEVT_Wrapper.h**
- **fio/HEPEVT_Wrapper.cc**
- **HepMC-2.06.09/fio/HEPEVT_Wrapper.cc**

7.23 hwgev Struct Reference

`#include <HerwigWrapper.h>`

Public Attributes

- double **AVWGT**
- double **EVWGT**
- double **GAMWT**
- double **TLOUT**
- double **WBIGST**
- double **WGTMAX**
- double **WGTSUM**
- double **WSQSUM**
- int **IDHW** [**herwig_hepevt_size**]
- int **IERROR**
- int **ISTAT**
- int **LWEVT**
- int **MAXER**
- int **MAXPR**
- int **NOWGT**
- int **NRN** [2]
- int **NUMER**
- int **NUMERU**
- int **NWGTS**
- int **GENSOF**

7.23.1 Detailed Description

Definition at line 56 of file HerwigWrapper.h.

7.23.2 Member Data Documentation

7.23.2.1 double hwgev::AVWGT

Definition at line 57 of file HerwigWrapper.h.

7.23.2.2 double hwgev::EVWGT

Definition at line 57 of file HerwigWrapper.h.

7.23.2.3 double hwgev::GAMWT

Definition at line 57 of file HerwigWrapper.h.

7.23.2.4 int hwgev::GENSOF

Definition at line 60 of file HerwigWrapper.h.

7.23.2.5 int hwgev::IDHW

Definition at line 58 of file HerwigWrapper.h.

7.23.2.6 int hwgev::IERROR

Definition at line 58 of file HerwigWrapper.h.

7.23.2.7 int hwgev::ISTAT

Definition at line 58 of file HerwigWrapper.h.

7.23.2.8 int hwgev::LWEVT

Definition at line 58 of file HerwigWrapper.h.

7.23.2.9 int hwgev::MAXER

Definition at line 58 of file HerwigWrapper.h.

7.23.2.10 int hwgev::MAXPR

Definition at line 58 of file HerwigWrapper.h.

7.23.2.11 int hwgev::NOWGT

Definition at line 59 of file HerwigWrapper.h.

7.23.2.12 int hwgev::NRN

Definition at line 59 of file HerwigWrapper.h.

7.23.2.13 int hwgev::NUMER

Definition at line 59 of file HerwigWrapper.h.

7.23.2.14 int hwgev::NUMERU

Definition at line 59 of file HerwigWrapper.h.

7.23.2.15 int hwgev::NWGTS

Definition at line 59 of file HerwigWrapper.h.

7.23.2.16 double hwgev::TLOUT

Definition at line 57 of file HerwigWrapper.h.

7.23.2.17 double hwgev::WBIGST

Definition at line 57 of file HerwigWrapper.h.

7.23.2.18 double hwgev::WGTMAX

Definition at line 57 of file HerwigWrapper.h.

7.23.2.19 double hwgev::WGTSUM

Definition at line 57 of file HerwigWrapper.h.

7.23.2.20 double hwgev::WSQSUM

Definition at line 57 of file HerwigWrapper.h.

The documentation for this struct was generated from the following files:

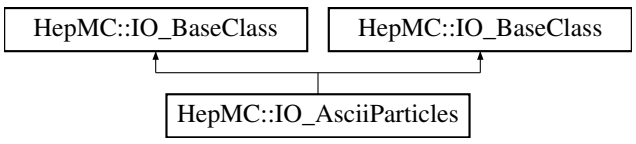
- **HerwigWrapper.h**
- **2.06.09/HepMC/HerwigWrapper.h**

7.24 HepMC::IO_AsciiParticles Class Reference

event input/output in ascii format for eye and machine reading

`#include <IO_AsciiParticles.h>`

Inheritance diagram for HepMC::IO_AsciiParticles:



Public Member Functions

- **IO_AsciiParticles** (const char *filename="IO_AsciiParticles.dat", std::ios::openmode mode=std::ios::out)
  constructor requiring a file name and std::ios mode
- virtual ~**IO_AsciiParticles** ()
- void **write_event** (const **GenEvent** *evt)
  write this event
- bool **fill_next_event** (**GenEvent** *evt)
  get the next event
- void **write_comment** (const std::string comment)
- void **setPrecision** (int iprec)
  set output precision
- int **rdstate** () const
  check the state of the IO stream
- void **clear** ()
  clear the IO stream
- void **print** (std::ostream &ostr=std::cout) const
  write to ostr
- **IO_AsciiParticles** (const char *filename="IO_AsciiParticles.dat", std::ios::openmode mode=std::ios::out)
  constructor requiring a file name and std::ios mode
- virtual ~**IO_AsciiParticles** ()
- void **write_event** (const **GenEvent** *evt)
  write this event
- bool **fill_next_event** (**GenEvent** *evt)
  get the next event
- void **write_comment** (const std::string comment)
- void **setPrecision** (int iprec)
  set output precision
- int **rdstate** () const
  check the state of the IO stream
- void **clear** ()
  clear the IO stream
- void **print** (std::ostream &ostr=std::cout) const
  write to ostr

Protected Member Functions

- bool **write_end_listing** ()
  write end tag
- bool **write_end_listing** ()
  write end tag

7.24.1 Detailed Description

event input/output in ascii format for eye and machine reading

Strategy for reading or writing events as machine readable ascii to a file. When instantiating, the mode of file to be created must be specified.

Examples:

fio/example_MyPythia.cc, **testHepMC.cc.in**, and **testStreamIO.cc.in**.

Definition at line 54 of file IO_AsciiParticles.h.

7.24.2 Constructor & Destructor Documentation

7.24.2.1 HepMC::IO_AsciiParticles::IO_AsciiParticles (const char * *filename* = "IO_AsciiParticles.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

Definition at line 17 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

7.24.2.2 HepMC::IO_AsciiParticles::~IO_AsciiParticles ( ) [virtual]

Definition at line 46 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

7.24.2.3 HepMC::IO_AsciiParticles::IO_AsciiParticles (const char * *filename* = "IO_AsciiParticles.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

7.24.2.4 virtual HepMC::IO_AsciiParticles::~IO_AsciiParticles ( ) [virtual]

7.24.3 Member Function Documentation

7.24.3.1 void HepMC::IO_AsciiParticles::clear ( ) [inline]

clear the IO stream

Definition at line 97 of file IO_AsciiParticles.h.

7.24.3.2 void HepMC::IO_AsciiParticles::clear ( )

clear the IO stream

7.24.3.3 bool HepMC::IO_AsciiParticles::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 182).

7.24.3.4 bool HepMC::IO_AsciiParticles::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 182).

Definition at line 181 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

7.24.3.5 void HepMC::IO_AsciiParticles::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

7.24.3.6 void HepMC::IO_AsciiParticles::print (std::ostream & *ostr* = `std::cout`) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

Definition at line 53 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

7.24.3.7 int HepMC::IO_AsciiParticles::rdstate ( ) const [inline]

check the state of the IO stream

Definition at line 96 of file IO_AsciiParticles.h.

7.24.3.8 int HepMC::IO_AsciiParticles::rdstate ( ) const

check the state of the IO stream

7.24.3.9 void HepMC::IO_AsciiParticles::setPrecision (int *iprec*)

set output precision

7.24.3.10 void HepMC::IO_AsciiParticles::setPrecision (int *iprec*) [inline]

set output precision

Definition at line 98 of file IO_AsciiParticles.h.

7.24.3.11 void HepMC::IO_AsciiParticles::write_comment (const std::string *comment*)

insert a comment directly into the output file — normally you only want to do this at the beginning or end of the file. All comments are preceded with "HepMC::IO_AsciiParticles-COMMENT\n"

7.24.3.12 void HepMC::IO_AsciiParticles::write_comment (const std::string *comment*)

insert a comment directly into the output file — normally you only want to do this at the beginning or end of the file. All comments are preceded with "HepMC::IO_AsciiParticles-COMMENT\n"

Definition at line 204 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

References write_end_listing().

7.24.3.13 bool HepMC::IO_AsciiParticles::write_end_listing ( ) [protected]

write end tag

7.24.3.14 bool HepMC::IO_AsciiParticles::write_end_listing ( ) [protected]

write end tag

Definition at line 219 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

Referenced by write_comment().

7.24.3.15 void HepMC::IO_AsciiParticles::write_event (const GenEvent * *evt*) [virtual]

write this event

Implements **HepMC::IO_BaseClass** (p. 183).

Definition at line 63 of file HepMC-2.06.09/src/IO_AsciiParticles.cc.

References HepMC::GenEvent::alphaQCD(), HepMC::GenEvent::alphaQED(), HepMC::GenVertex::barcode(), HepMC::WeightContainer::begin(), HepMC::WeightContainer::end(), HepMC::GenEvent::event_number(), HepMC::GenEvent::event_scale(), HepMC::detail::output(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::particles_size(), HepMC::GenEvent::random_states(), HepMC::GenEvent::signal_process_id(), HepMC::GenEvent::signal_process_vertex(), HepMC::WeightContainer::size(), HepMC::versionName(), HepMC::GenEvent::vertices_size(), and HepMC::GenEvent::weights().

7.24.3.16 void HepMC::IO_AsciiParticles::write_event (const GenEvent * *evt*) [virtual]

write this event

Implements **HepMC::IO_BaseClass** (p. 183).

The documentation for this class was generated from the following files:

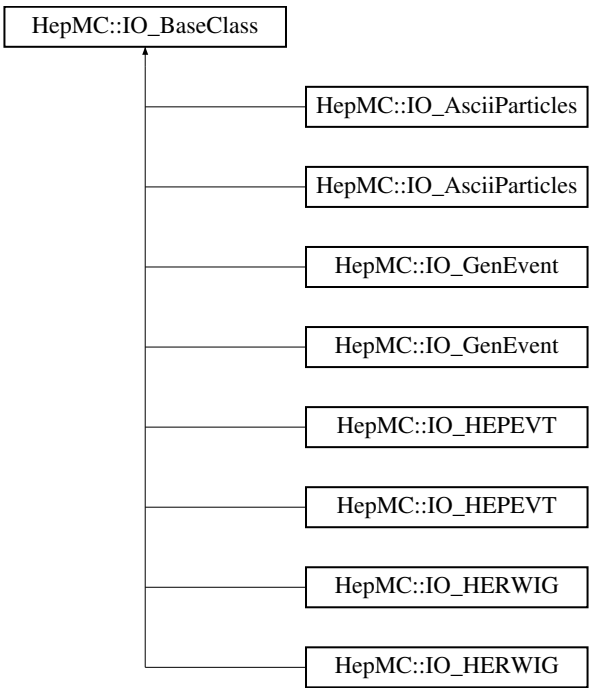
- **IO_AsciiParticles.h**
- **2.06.09/HepMC/IO_AsciiParticles.h**
- **HepMC-2.06.09/src/IO_AsciiParticles.cc**
- **src/IO_AsciiParticles.cc**

7.25 HepMC::IO_BaseClass Class Reference

all input/output classes inherit from **IO_BaseClass** (p. 180)

`#include <IO_BaseClass.h>`

Inheritance diagram for HepMC::IO_BaseClass:



Public Member Functions

- virtual ~**IO_BaseClass** ()
- virtual void **write_event** (const **GenEvent** *)=0

  *write this **GenEvent** (p. 72)*
- virtual bool **fill_next_event** (**GenEvent** *)=0

  *fill this **GenEvent** (p. 72)*
- virtual void **print** (std::ostream &ostr=std::cout) const

  write output to ostr
- **GenEvent** * **read_next_event** ()

  do not over-ride
- virtual **GenEvent** *& **operator**>> (**GenEvent** *&)

  the same as read_next_event
- virtual const **GenEvent** *& **operator**<< (const **GenEvent** *&)

  the same as write_event
- virtual **GenEvent** *& **operator**<< (**GenEvent** *&)

  the same as write_event
- virtual ~**IO_BaseClass** ()
- virtual void **write_event** (const **GenEvent** *)=0

  *write this **GenEvent** (p. 72)*
- virtual bool **fill_next_event** (**GenEvent** *)=0

  *fill this **GenEvent** (p. 72)*
- virtual void **print** (std::ostream &ostr=std::cout) const

  write output to ostr
- **GenEvent** * **read_next_event** ()

  do not over-ride
- virtual **GenEvent** *& **operator**>> (**GenEvent** *&)

  the same as read_next_event
- virtual const **GenEvent** *& **operator**<< (const **GenEvent** *&)

  the same as write_event
- virtual **GenEvent** *& **operator**<< (**GenEvent** *&)

  the same as write_event

7.25.1 Detailed Description

all input/output classes inherit from **IO_BaseClass** (p. 180)

If you want to write a new IO class, then inherit from this class and re-define read_event() and **write_event()** (p. 183)

Definition at line 34 of file IO_BaseClass.h.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 virtual HepMC::IO_BaseClass::~IO_BaseClass ( ) [inline], [virtual]

Definition at line 36 of file IO_BaseClass.h.

7.25.2.2 virtual HepMC::IO_BaseClass::~IO_BaseClass ( ) [inline], [virtual]

Definition at line 36 of file 2.06.09/HepMC/IO_BaseClass.h.

7.25.3 Member Function Documentation

7.25.3.1 virtual bool HepMC::IO_BaseClass::fill_next_event (GenEvent *) [pure virtual]

fill this **GenEvent** (p. 72)

Implemented in **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_AsciiParticles** (p. 178), **HepMC::IO_AsciiParticles** (p. 178), **HepMC::IO_HERWIG** (p. 202), **HepMC::IO_HERWIG** (p. 202), **HepMC::IO_HEPEVT** (p. 195), and **HepMC::IO_HEPEVT** (p. 195).

Referenced by read_next_event().

7.25.3.2 virtual bool HepMC::IO_BaseClass::fill_next_event (GenEvent *) [pure virtual]

fill this **GenEvent** (p. 72)

Implemented in **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_AsciiParticles** (p. 178), **HepMC::IO_AsciiParticles** (p. 178), **HepMC::IO_HERWIG** (p. 202), **HepMC::IO_HERWIG** (p. 202), **HepMC::IO_HEPEVT** (p. 195), and **HepMC::IO_HEPEVT** (p. 195).

7.25.3.3 const GenEvent *& HepMC::IO_BaseClass::operator<< (const GenEvent *& *evt*) [inline], [virtual]

the same as write_event

Definition at line 99 of file IO_BaseClass.h.

References write_event().

7.25.3.4 virtual const GenEvent*& HepMC::IO_BaseClass::operator<< (const GenEvent *&) [virtual]

the same as write_event

7.25.3.5 virtual GenEvent*& HepMC::IO_BaseClass::operator<< (GenEvent *&) [virtual]

the same as write_event

7.25.3.6 GenEvent *& HepMC::IO_BaseClass::operator<< (GenEvent *& *evt*) [inline], [virtual]

the same as write_event

Definition at line 105 of file IO_BaseClass.h.

References write_event().

7.25.3.7 GenEvent *& HepMC::IO_BaseClass::operator>> (GenEvent *& *evt*) [inline], [virtual]

the same as read_next_event

Definition at line 94 of file IO_BaseClass.h.

References read_next_event().

7.25.3.8 virtual GenEvent*& HepMC::IO_BaseClass::operator>> (GenEvent *&) [virtual]

the same as read_next_event

7.25.3.9 virtual void HepMC::IO_BaseClass::print (std::ostream & *ostr* = `std::cout`) const [virtual]

write output to ostr

Reimplemented in **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_AsciiParticles** (p. 179), **HepMC::IO_AsciiParticles** (p. 179), **HepMC::IO_HERWIG** (p. 203), **HepMC::IO_HERWIG** (p. 203), **HepMC::IO_HEPEVT** (p. 196), and **HepMC::IO_HEPEVT** (p. 196).

7.25.3.10 void HepMC::IO_BaseClass::print (std::ostream & *ostr* = `std::cout`) const [inline], [virtual]

write output to ostr

Reimplemented in **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_GenEvent** (p. 190), **HepMC::IO_AsciiParticles** (p. 179), **HepMC::IO_AsciiParticles** (p. 179), **HepMC::IO_HERWIG** (p. 203), **HepMC::IO_HERWIG** (p. 203), **HepMC::IO_HEPEVT** (p. 196), and **HepMC::IO_HEPEVT** (p. 196).

Definition at line 90 of file IO_BaseClass.h.

7.25.3.11 GenEvent* HepMC::IO_BaseClass::read_next_event ( )

do not over-ride

7.25.3.12 GenEvent * HepMC::IO_BaseClass::read_next_event ( ) [inline]

do not over-ride

creates a new event and fills it by calling the sister method read_next_event(GenEvent*)

creates a new event and fills it by calling the sister method read_next_event(GenEvent*)

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.cc**, and **testMultipleCopies.cc.in**.

Definition at line 74 of file IO_BaseClass.h.

References fill_next_event().

Referenced by event_selection(), main(), operator>>(), pythia_in(), pythia_in_out(), pythia_out(), pythia_particle_out(), read_nan(), read_testIOGenEvent(), read_testUnits(), read_variousFormats(), readWithCrossSection(), readWithWeight(), write_to_stream(), write_to_stream3(), writePythiaStreamIO(), writeWithCrossSection(), and writeWithWeight().

7.25.3.13 virtual void HepMC::IO_BaseClass::write_event (const GenEvent *) [pure virtual]

write this **GenEvent** (p. 72)

Implemented in **HepMC::IO_GenEvent** (p. 191), **HepMC::IO_GenEvent** (p. 191), **HepMC::IO_AsciiParticles** (p. 180), **HepMC::IO_AsciiParticles** (p. 180), **HepMC::IO_HEPEVT** (p. 198), and **HepMC::IO_HEPEVT** (p. 198).

7.25.3.14 virtual void HepMC::IO_BaseClass::write_event (const GenEvent *) [pure virtual]

write this **GenEvent** (p. 72)

Implemented in **HepMC::IO_GenEvent** (p. 191), **HepMC::IO_GenEvent** (p. 191), **HepMC::IO_AsciiParticles** (p. 180), **HepMC::IO_AsciiParticles** (p. 180), **HepMC::IO_HEPEVT** (p. 198), and **HepMC::IO_HEPEVT** (p. 198).

Referenced by operator<<().

The documentation for this class was generated from the following files:

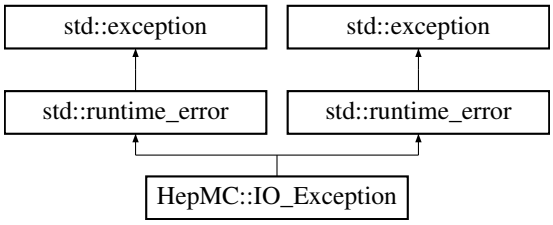
- **IO_BaseClass.h**
- **2.06.09/HepMC/IO_BaseClass.h**

7.26 HepMC::IO_Exception Class Reference

IO exception handling.

`#include <IO_Exception.h>`

Inheritance diagram for HepMC::IO_Exception:



Public Types

- enum **ErrorType** {
  OK, **NullEvent**, **WrongFileType**, **MissingStartKey**,
  EndOfStream, **EndKeyMismatch**, **MissingEndKey**, **InvalidData**,
  InputAndOutput, **BadOutputStream**, **BadInputStream**, **OK**,
  NullEvent, **WrongFileType**, **MissingStartKey**, **EndOfStream**,
  EndKeyMismatch, **MissingEndKey**, **InvalidData**, **InputAndOutput**,
  BadOutputStream, **BadInputStream** }

  IO error types.
- enum **ErrorType** {
  OK, **NullEvent**, **WrongFileType**, **MissingStartKey**,
  EndOfStream, **EndKeyMismatch**, **MissingEndKey**, **InvalidData**,
  InputAndOutput, **BadOutputStream**, **BadInputStream**, **OK**,
  NullEvent, **WrongFileType**, **MissingStartKey**, **EndOfStream**,
  EndKeyMismatch, **MissingEndKey**, **InvalidData**, **InputAndOutput**,
  BadOutputStream, **BadInputStream** }

  IO error types.

Public Member Functions

- **IO_Exception** (const std::string &msg)
- **IO_Exception** (const std::string &msg)

7.26.1 Detailed Description

IO exception handling.

IO_GenEvent (p. 186), etc. catch the throw and set data members with the error type and message Some of the messages are constructed with transient information (e.g., contents of a bad **GenParticle** (p. 106))

Examples:

testStreamIO.cc.in.

Definition at line 28 of file IO_Exception.h.

7.26.2 Member Enumeration Documentation

7.26.2.1 enum HepMC::IO_Exception::ErrorType

IO error types.

Enumerator

OK
NullEvent
WrongFileType
MissingStartKey
EndOfStream
EndKeyMismatch
MissingEndKey
InvalidData
InputAndOutput
BadOutputStream
BadInputStream
OK
NullEvent
WrongFileType
MissingStartKey
EndOfStream
EndKeyMismatch
MissingEndKey
InvalidData
InputAndOutput
BadOutputStream
BadInputStream

Definition at line 34 of file IO_Exception.h.

7.26.2.2 enum HepMC::IO_Exception::ErrorType

IO error types.

Enumerator

OK
NullEvent
WrongFileType
MissingStartKey
EndOfStream
EndKeyMismatch
MissingEndKey
InvalidData
InputAndOutput
BadOutputStream

BadInputStream

OK

NullEvent

WrongFileType

MissingStartKey

EndOfStream

EndKeyMismatch

MissingEndKey

InvalidData

InputAndOutput

BadOutputStream

BadInputStream

Definition at line 34 of file 2.06.09/HepMC/IO_Exception.h.

7.26.3 Constructor & Destructor Documentation

7.26.3.1 HepMC::IO_Exception::IO_Exception (const std::string & *msg*) [inline]

Definition at line 30 of file IO_Exception.h.

7.26.3.2 HepMC::IO_Exception::IO_Exception (const std::string & *msg*) [inline]

Definition at line 30 of file 2.06.09/HepMC/IO_Exception.h.

The documentation for this class was generated from the following files:

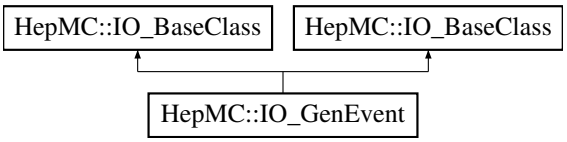
- **IO_Exception.h**
- **2.06.09/HepMC/IO_Exception.h**

7.27 HepMC::IO_GenEvent Class Reference

IO_GenEvent (p. 186) also deals with **HeavyIon** (p. 146) and **PdfInfo** (p. 233).

`#include <IO_GenEvent.h>`

Inheritance diagram for HepMC::IO_GenEvent:



Public Member Functions

- **IO_GenEvent** (const std::string &filename="IO_GenEvent.dat", std::ios::openmode mode=std::ios::out)
  constructor requiring a file name and std::ios mode
- **IO_GenEvent** (std::istream &)
  constructor requiring an input stream

- **IO_GenEvent** (std::ostream &)

  constructor requiring an output stream
- virtual ~**IO_GenEvent** ()
- void **write_event** (const **GenEvent** *evt)

  write this event
- bool **fill_next_event** (**GenEvent** *evt)

  get the next event
- void **write_comment** (const std::string comment)
- int **rdstate** () const

  check the state of the IO stream
- void **clear** ()

  clear the IO stream
- void **print** (std::ostream &ostr=std::cout) const

  write to ostr
- void **use_input_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- void **precision** (int)
- int **error_type** () const

  integer (enum) associated with read error
- const std::string & **error_message** () const

  the read error message string
- **IO_GenEvent** (const std::string &filename="IO_GenEvent.dat", std::ios::openmode mode=std::ios::out)

  constructor requiring a file name and std::ios mode
- **IO_GenEvent** (std::istream &)

  constructor requiring an input stream
- **IO_GenEvent** (std::ostream &)

  constructor requiring an output stream
- virtual ~**IO_GenEvent** ()
- void **write_event** (const **GenEvent** *evt)

  write this event
- bool **fill_next_event** (**GenEvent** *evt)

  get the next event
- void **write_comment** (const std::string comment)
- int **rdstate** () const

  check the state of the IO stream
- void **clear** ()

  clear the IO stream
- void **print** (std::ostream &ostr=std::cout) const

  write to ostr
- void **use_input_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- void **precision** (int)
- int **error_type** () const

  integer (enum) associated with read error
- const std::string & **error_message** () const

  the read error message string

7.27.1 Detailed Description

IO_GenEvent (p. 186) also deals with **HeavyIon** (p. 146) and **PdfInfo** (p. 233).

event input/output in ascii format for machine reading extended format contains **HeavyIon** (p. 146) and **PdfInfo** (p. 233) classes

Strategy for reading or writing events using iostreams When instantiating with a file name, the mode of file to be created must be specified. Options are: std::ios::in open file for input std::ios::out open file for output std::ios::trunc erase old file when opening (i.e. ios::out|ios::trunc removes oldfile, and creates a new one for output) std::ios::app append output to end of file for the purposes of this class, simultaneous input and output mode (std::ios::in | std::ios::out) is not allowed.

Event listings are preceded by the key: "HepMC::IO_GenEvent-START_EVENT_LISTING\n" and terminated by the key: "HepMC::IO_GenEvent-END_EVENT_LISTING\n" **GenParticle** (p. 106) Data tables are preceded by the key: "HepMC::IO_GenEvent-START_PARTICLE_DATA\n" and terminated by the key: "HepMC::IO_GenEvent-END_PARTICLE_DATA\n" Comments are allowed. They need not be preceded by anything, though if a comment is written using write_comment(const string) then it will be preceded by "HepMC::IO_GenEvent-COMMENT\n" Each event, vertex, particle, particle data, heavy ion, or pdf info line is preceded by "E ","V ","P ","D ","H ","F " respectively. Comments may appear anywhere in the file – so long as they do not contain any of the start/stop keys.

Examples:

example_EventSelection.cc, **example_UsingIterators.cc**, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **testFlow.cc**, **testHepMC.cc.in**, **testHepMCIteration.cc.in**, **testMass.cc.in**, **testMultipleCopies.cc.in**, and **testStreamIO.cc.in**.

Definition at line 63 of file IO_GenEvent.h.

7.27.2 Constructor & Destructor Documentation

7.27.2.1 HepMC::IO_GenEvent::IO_GenEvent (const std::string & *filename* = "IO_GenEvent.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

Definition at line 16 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::detail::establish_input_stream_info(), HepMC::detail::establish_output_stream_info(), and HepMC::IO_Exception::InputAndOutput.

7.27.2.2 HepMC::IO_GenEvent::IO_GenEvent (std::istream & *istr*)

constructor requiring an input stream

Definition at line 50 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::detail::establish_input_stream_info().

7.27.2.3 HepMC::IO_GenEvent::IO_GenEvent (std::ostream & *ostr*)

constructor requiring an output stream

Definition at line 61 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::detail::establish_output_stream_info().

7.27.2.4 HepMC::IO_GenEvent::~IO_GenEvent ( ) [virtual]

Definition at line 72 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::write_HepMC_IO_block_end().

7.27.2.5 HepMC::IO_GenEvent::IO_GenEvent (const std::string & *filename* = "IO_GenEvent.dat", std::ios::openmode *mode* = std::ios::out)

constructor requiring a file name and std::ios mode

7.27.2.6 HepMC::IO_GenEvent::IO_GenEvent (std::istream &)

constructor requiring an input stream

7.27.2.7 HepMC::IO_GenEvent::IO_GenEvent (std::ostream &)

constructor requiring an output stream

7.27.2.8 virtual HepMC::IO_GenEvent::~IO_GenEvent ( ) [virtual]

7.27.3 Member Function Documentation

7.27.3.1 void HepMC::IO_GenEvent::clear ( ) [inline]

clear the IO stream

Definition at line 133 of file IO_GenEvent.h.

7.27.3.2 void HepMC::IO_GenEvent::clear ( )

clear the IO stream

7.27.3.3 const std::string & HepMC::IO_GenEvent::error_message ( ) const [inline]

the read error message string

Definition at line 145 of file IO_GenEvent.h.

Referenced by read_nan().

7.27.3.4 const std::string& HepMC::IO_GenEvent::error_message ( ) const

the read error message string

7.27.3.5 int HepMC::IO_GenEvent::error_type ( ) const

integer (enum) associated with read error

7.27.3.6 int HepMC::IO_GenEvent::error_type ( ) const [inline]

integer (enum) associated with read error

Definition at line 141 of file IO_GenEvent.h.

Referenced by read_nan().

7.27.3.7 bool HepMC::IO_GenEvent::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 182).

Definition at line 109 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::GenEvent::clear(), HepMC::IO_Exception::InvalidData, HepMC::GenEvent::is_valid(), HepMC::IO_Exception::NullEvent, HepMC::IO_Exception::OK, and HepMC::IO_Exception::WrongFileType.

7.27.3.8 bool HepMC::IO_GenEvent::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 182).

7.27.3.9 void HepMC::IO_GenEvent::precision (int)

set output precision The default precision is 16.

7.27.3.10 void HepMC::IO_GenEvent::precision (int *size*)

set output precision The default precision is 16.

Definition at line 96 of file HepMC-2.06.09/src/IO_GenEvent.cc.

Referenced by read_testIOGenEvent().

7.27.3.11 void HepMC::IO_GenEvent::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

7.27.3.12 void HepMC::IO_GenEvent::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

Definition at line 86 of file HepMC-2.06.09/src/IO_GenEvent.cc.

7.27.3.13 int HepMC::IO_GenEvent::rdstate ( ) const

check the state of the IO stream

7.27.3.14 int HepMC::IO_GenEvent::rdstate ( ) const [inline]

check the state of the IO stream

Definition at line 123 of file IO_GenEvent.h.

Referenced by main().

7.27.3.15 void HepMC::IO_GenEvent::use_input_units (Units::MomentumUnit , Units::LengthUnit)

needed when reading a file without units if those units are different than the declared default units (e.g., the default units are MeV, but the file was written with GeV) This method is not necessary if the units are written in the file

7.27.3.16 void HepMC::IO_GenEvent::use_input_units (Units::MomentumUnit *mom,* Units::LengthUnit *len*)

needed when reading a file without units if those units are different than the declared default units (e.g., the default units are MeV, but the file was written with GeV) This method is not necessary if the units are written in the file

Definition at line 79 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::set_input_units().

Referenced by main(), read_nan(), read_testIOGenEvent(), read_variousFormats(), write_to_stream(), write_to_-stream3(), writeWithCrossSection(), and writeWithWeight().

7.27.3.17 void HepMC::IO_GenEvent::write_comment (const std::string *comment*)

insert a comment directly into the output file — normally you only want to do this at the beginning or end of the file. All comments are preceded with "HepMC::IO_GenEvent-COMMENT\n"

7.27.3.18 void HepMC::IO_GenEvent::write_comment (const std::string *comment*)

insert a comment directly into the output file — normally you only want to do this at the beginning or end of the file. All comments are preceded with "HepMC::IO_GenEvent-COMMENT\n"

Definition at line 162 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::write_HepMC_IO_block_end(), and HepMC::IO_Exception::WrongFileType.

7.27.3.19 void HepMC::IO_GenEvent::write_event (const GenEvent * *evt*) [virtual]

write this event

Implements **HepMC::IO_BaseClass** (p. 183).

7.27.3.20 void HepMC::IO_GenEvent::write_event (const GenEvent * *evt*) [virtual]

write this event

Writes evt to output stream. It does NOT delete the event after writing.

Writes evt to output stream. It does NOT delete the event after writing.

Implements **HepMC::IO_BaseClass** (p. 183).

Definition at line 143 of file HepMC-2.06.09/src/IO_GenEvent.cc.

References HepMC::write_HepMC_IO_block_begin(), and HepMC::IO_Exception::WrongFileType.

The documentation for this class was generated from the following files:

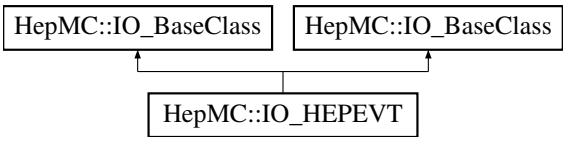
- **IO_GenEvent.h**
- **2.06.09/HepMC/IO_GenEvent.h**
- **HepMC-2.06.09/src/IO_GenEvent.cc**
- **src/IO_GenEvent.cc**

7.28 HepMC::IO_HEPEVT Class Reference

HEPEVT IO class.

`#include <IO_HEPEVT.h>`

Inheritance diagram for HepMC::IO_HEPEVT:



Public Member Functions

- **IO_HEPEVT** ()
- virtual ~**IO_HEPEVT** ()
- bool **fill_next_event** (**GenEvent** ∗)

  *fill this **GenEvent** (p. 72)*
- void **write_event** (const **GenEvent** ∗)

  *write this **GenEvent** (p. 72)*
- void **print** (std::ostream &ostr=std::cout) const

  write output to ostr
- bool **trust_both_mothers_and_daughters** () const

  default is false
- bool **trust_mothers_before_daughters** () const

  default is true
- bool **print_inconsistency_errors** () const

  default is true
- bool **trust_beam_particles** () const

  default is true
- void **set_trust_mothers_before_daughters** (bool b=true)

  define mother daughter trust rules
- void **set_trust_both_mothers_and_daughters** (bool b=false)

  define mother daughter trust rules
- void **set_print_inconsistency_errors** (bool b=true)
- void **set_trust_beam_particles** (bool b=true)

  declare whether or not beam particles exist
- **IO_HEPEVT** ()
- virtual ~**IO_HEPEVT** ()
- bool **fill_next_event** (**GenEvent** ∗)

  *fill this **GenEvent** (p. 72)*
- void **write_event** (const **GenEvent** ∗)

  *write this **GenEvent** (p. 72)*
- void **print** (std::ostream &ostr=std::cout) const

  write output to ostr
- bool **trust_both_mothers_and_daughters** () const

  default is false
- bool **trust_mothers_before_daughters** () const

  default is true
- bool **print_inconsistency_errors** () const

default is true

- bool **trust_beam_particles** () const

  default is true

- void **set_trust_mothers_before_daughters** (bool b=true)

  define mother daughter trust rules

- void **set_trust_both_mothers_and_daughters** (bool b=false)

  define mother daughter trust rules

- void **set_print_inconsistency_errors** (bool b=true)
- void **set_trust_beam_particles** (bool b=true)

  declare whether or not beam particles exist

Protected Member Functions

- **GenParticle** ∗ **build_particle** (int index)

  *create a **GenParticle** (p. 106)*

- void **build_production_vertex** (int i, std::vector< **HepMC::GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  create a production vertex

- void **build_end_vertex** (int i, std::vector< **HepMC::GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  create an end vertex

- int **find_in_map** (const std::map< **HepMC::GenParticle** ∗, int > &m, **GenParticle** ∗**p**) const

  find this particle in the particle map

- **GenParticle** ∗ **build_particle** (int index)

  *create a **GenParticle** (p. 106)*

- void **build_production_vertex** (int i, std::vector< **HepMC::GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  create a production vertex

- void **build_end_vertex** (int i, std::vector< **HepMC::GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  create an end vertex

- int **find_in_map** (const std::map< **HepMC::GenParticle** ∗, int > &m, **GenParticle** ∗**p**) const

  find this particle in the particle map

7.28.1 Detailed Description

HEPEVT IO class.

IO class for reading the standard HEPEVT common block.

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 39 of file IO_HEPEVT.h.

7.28.2 Constructor & Destructor Documentation

7.28.2.1 HepMC::IO_HEPEVT::IO_HEPEVT ( )

Definition at line 12 of file fio/IO_HEPEVT.cc.

7.28.2.2 HepMC::IO_HEPEVT::~IO_HEPEVT ( ) [virtual]

Definition at line 18 of file fio/IO_HEPEVT.cc.

7.28.2.3 HepMC::IO_HEPEVT::IO_HEPEVT ( )

7.28.2.4 virtual HepMC::IO_HEPEVT::~IO_HEPEVT ( ) [virtual]

7.28.3 Member Function Documentation

7.28.3.1 void HepMC::IO_HEPEVT::build_end_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create an end vertex

for particle in HEPEVT with index i, build an end vertex if appropriate, and add that vertex to the event

for particle in HEPEVT with index i, build an end vertex if appropriate, and add that vertex to the event

Definition at line 257 of file fio/IO_HEPEVT.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::GenParticle::end_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::number_children(), p, HepMC::GenVertex::position(), HepMC::GenParticle::production_vertex(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

7.28.3.2 void HepMC::IO_HEPEVT::build_end_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create an end vertex

7.28.3.3 GenParticle* HepMC::IO_HEPEVT::build_particle (int *index*) [protected]

create a **GenParticle** (p. 106)

7.28.3.4 GenParticle * HepMC::IO_HEPEVT::build_particle (int *index*) [protected]

create a **GenParticle** (p. 106)

Builds a particle object corresponding to index in HEPEVT

Builds a particle object corresponding to index in HEPEVT

Definition at line 325 of file fio/IO_HEPEVT.cc.

References HepMC::HEPEVT_Wrapper::e(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::m(), p, HepMC::HEPEVT_Wrapper::px(), HepMC::HEPEVT_Wrapper::py(), HepMC::HEPEVT_Wrapper::pz(), HepMC::GenParticle::setGeneratedMass(), HepMC::HEPEVT_Wrapper::status(), and HepMC::GenParticle::suggest_barcode().

Referenced by fill_next_event().

7.28.3.5 void HepMC::IO_HEPEVT::build_production_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create a production vertex

7.28.3.6 void HepMC::IO_HEPEVT::build_production_vertex (int *i*, std::vector< HepMC::GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

create a production vertex

for particle in HEPEVT with index i, build a production vertex if appropriate, and add that vertex to the event

for particle in HEPEVT with index i, build a production vertex if appropriate, and add that vertex to the event

Definition at line 191 of file fio/IO_HEPEVT.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::number_parents(), p, HepMC::GenVertex::position(), HepMC::GenParticle::production_vertex(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

7.28.3.7 bool HepMC::IO_HEPEVT::fill_next_event (GenEvent *) [virtual]

fill this **GenEvent** (p. 72)

Implements **HepMC::IO_BaseClass** (p. 182).

7.28.3.8 bool HepMC::IO_HEPEVT::fill_next_event (GenEvent *) [virtual]

fill this **GenEvent** (p. 72)

read one event from the HEPEVT common block and fill **GenEvent** (p. 72) return T/F =success/failure

For HEPEVT commons built with the luhepc routine of Pythia 5.7 the children pointers are not always correct (i.e. there is oftentimes an internal inconsistency between the parents and children pointers). The parent pointers always seem to be correct. Thus the switch trust_mothers_before_daughters=1 is appropriate for pythia. NOTE: you should also set the switch MSTP(128) = 2 in pythia (not the default!), so that pythia doesn't store two copies of resonances in the event record. The situation is opposite for the HEPEVT which comes from Isajet via stdhep, so then use the switch trust_mothers_before_daughters=0

sufficient to do one or the other.

read one event from the HEPEVT common block and fill **GenEvent** (p. 72) return T/F =success/failure

For HEPEVT commons built with the luhepc routine of Pythia 5.7 the children pointers are not always correct (i.e. there is oftentimes an internal inconsistency between the parents and children pointers). The parent pointers always seem to be correct. Thus the switch trust_mothers_before_daughters=1 is appropriate for pythia. NOTE: you should also set the switch MSTP(128) = 2 in pythia (not the default!), so that pythia doesn't store two copies of resonances in the event record. The situation is opposite for the HEPEVT which comes from Isajet via stdhep, so then use the switch trust_mothers_before_daughters=0

sufficient to do one or the other.

Implements **HepMC::IO_BaseClass** (p. 182).

Definition at line 31 of file fio/IO_HEPEVT.cc.

References HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), build_end_vertex(), build_particle(), build_production_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper-

::number_entries(), HepMC::GenEvent::set_beam_particles(), HepMC::GenEvent::set_event_number(), and trust-_beam_particles().

7.28.3.9 int HepMC::IO_HEPEVT::find_in_map (const std::map< HepMC::GenParticle *, int > & *m,* GenParticle * *p*) const [protected]

find this particle in the particle map

7.28.3.10 int HepMC::IO_HEPEVT::find_in_map (const std::map< HepMC::GenParticle *, int > & *m,* GenParticle * *p*) const [protected]

find this particle in the particle map

Definition at line 340 of file fio/IO_HEPEVT.cc.

Referenced by write_event().

7.28.3.11 void HepMC::IO_HEPEVT::print (std::ostream & *ostr* = `std::cout`) const [virtual]

write output to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

7.28.3.12 void HepMC::IO_HEPEVT::print (std::ostream & *ostr* = `std::cout`) const [virtual]

write output to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

Definition at line 20 of file fio/IO_HEPEVT.cc.

7.28.3.13 bool HepMC::IO_HEPEVT::print_inconsistency_errors ( ) const

default is true

7.28.3.14 bool HepMC::IO_HEPEVT::print_inconsistency_errors ( ) const [inline]

default is true

Definition at line 120 of file IO_HEPEVT.h.

7.28.3.15 void HepMC::IO_HEPEVT::set_print_inconsistency_errors (bool *b* = `true`)

Since HEPEVT has bi-directional pointers, it is possible that the mother/daughter pointers are inconsistent (though physically speaking this should never happen). In practise it happens often. When a conflict occurs (i.e. when mother/daughter pointers are in disagreement, where an empty (0) pointer is not considered a disagreement) an error is printed. These errors can be turned off with: myio_hepevt.set_print_inconsistency_errors(0); but it is STRONGLY recommended that you print the HEPEVT common and understand the inconsistency BEFORE you turn off the errors. The messages are there for a reason [remember, there is no message printed when the information is missing, ... only when is it inconsistent. User beware.] You can inspect the HEPEVT common block for inconsistencies with **HEPEVT_Wrapper::check_hepevt_consistency()** (p. 163)

There is a switch controlling whether the mother pointers or the daughters are to be trusted. For example, in Pythia the mother information is always correctly included, but the daughter information is often left unfilled: in this case we want to trust the mother pointers and not necessarily the daughters. [THIS IS THE DEFAULT]. Unfortunately the reverse happens for the stdhep(2001) translation of Isajet, so we need an option to toggle the choices.

7.28.3.16 void HepMC::IO_HEPEVT::set_print_inconsistency_errors (bool *b* = `true`) [inline]

Since HEPEVT has bi-directional pointers, it is possible that the mother/daughter pointers are inconsistent (though physically speaking this should never happen). In practise it happens often. When a conflict occurs (i.e. when mother/daughter pointers are in disagreement, where an empty (0) pointer is not considered a disagreement) an error is printed. These errors can be turned off with: myio_hepevt.set_print_inconsistency_errors(0); but it is STRONGLY recommended that you print the HEPEVT common and understand the inconsistency BEFORE you turn off the errors. The messages are there for a reason [remember, there is no message printed when the information is missing, ... only when is it inconsistent. User beware.] You can inspect the HEPEVT common block for inconsistencies with **HEPEVT_Wrapper::check_hepevt_consistency()** (p. 163)

There is a switch controlling whether the mother pointers or the daughters are to be trusted. For example, in Pythia the mother information is always correctly included, but the daughter information is often left unfilled: in this case we want to trust the mother pointers and not necessarily the daughters. [THIS IS THE DEFAULT]. Unfortunately the reverse happens for the stdhep(2001) translation of Isajet, so we need an option to toggle the choices.

Definition at line 129 of file IO_HEPEVT.h.

7.28.3.17 void HepMC::IO_HEPEVT::set_trust_beam_particles (bool *b* = `true`) [inline]

declare whether or not beam particles exist

Definition at line 135 of file IO_HEPEVT.h.

7.28.3.18 void HepMC::IO_HEPEVT::set_trust_beam_particles (bool *b* = `true`)

declare whether or not beam particles exist

7.28.3.19 void HepMC::IO_HEPEVT::set_trust_both_mothers_and_daughters (bool *b* = `false`) [inline]

define mother daughter trust rules

Definition at line 123 of file IO_HEPEVT.h.

7.28.3.20 void HepMC::IO_HEPEVT::set_trust_both_mothers_and_daughters (bool *b* = `false`)

define mother daughter trust rules

7.28.3.21 void HepMC::IO_HEPEVT::set_trust_mothers_before_daughters (bool *b* = `true`) [inline]

define mother daughter trust rules

Definition at line 126 of file IO_HEPEVT.h.

7.28.3.22 void HepMC::IO_HEPEVT::set_trust_mothers_before_daughters (bool *b* = `true`)

define mother daughter trust rules

7.28.3.23 bool HepMC::IO_HEPEVT::trust_beam_particles ( ) const

default is true

7.28.3.24 bool HepMC::IO_HEPEVT::trust_beam_particles ( ) const [inline]

default is true

Definition at line 132 of file IO_HEPEVT.h.

Referenced by fill_next_event().

7.28.3.25 bool HepMC::IO_HEPEVT::trust_both_mothers_and_daughters ( ) const

default is false

7.28.3.26 bool HepMC::IO_HEPEVT::trust_both_mothers_and_daughters ( ) const [inline]

default is false

Definition at line 114 of file IO_HEPEVT.h.

7.28.3.27 bool HepMC::IO_HEPEVT::trust_mothers_before_daughters ( ) const [inline]

default is true

Definition at line 117 of file IO_HEPEVT.h.

7.28.3.28 bool HepMC::IO_HEPEVT::trust_mothers_before_daughters ( ) const

default is true

7.28.3.29 void HepMC::IO_HEPEVT::write_event (const GenEvent *) [virtual]

write this **GenEvent** (p. 72)

This writes an event out to the HEPEVT common block. The daughters field is NOT filled, because it is possible to contruct graphs for which the mothers and daughters cannot both be make sequential. This is consistent with how pythia fills HEPEVT (daughters are not necessarily filled properly) and how **IO_HEPEVT** (p. 192) reads HEPEVT.

This writes an event out to the HEPEVT common block. The daughters field is NOT filled, because it is possible to contruct graphs for which the mothers and daughters cannot both be make sequential. This is consistent with how pythia fills HEPEVT (daughters are not necessarily filled properly) and how **IO_HEPEVT** (p. 192) reads HEPEVT.

Implements **HepMC::IO_BaseClass** (p. 183).

Definition at line 110 of file fio/IO_HEPEVT.cc.

References HepMC::FourVector::e(), HepMC::GenEvent::event_number(), find_in_map(), HepMC::HEPEVT_-Wrapper::max_number_entries(), p, HepMC::FourVector::px(), HepMC::FourVector::py(), HepMC::FourVector::pz(), HepMC::HEPEVT_Wrapper::set_children(), HepMC::HEPEVT_Wrapper::set_event_number(), HepMC::HEPE-VT_Wrapper::set_id(), HepMC::HEPEVT_Wrapper::set_mass(), HepMC::HEPEVT_Wrapper::set_momentum(), HepMC::HEPEVT_Wrapper::set_number_entries(), HepMC::HEPEVT_Wrapper::set_parents(), HepMC::HEPE-VT_Wrapper::set_position(), HepMC::HEPEVT_Wrapper::set_status(), HepMC::FourVector::t(), v, HepMC::Gen-Event::vertices_begin(), HepMC::GenEvent::vertices_end(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

7.28.3.30 void HepMC::IO_HEPEVT::write_event (const GenEvent *) [virtual]

write this **GenEvent** (p. 72)

Implements **HepMC::IO_BaseClass** (p. 183).

The documentation for this class was generated from the following files:

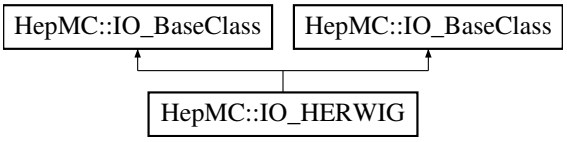
- **IO_HEPEVT.h**
- **2.06.09/HepMC/IO_HEPEVT.h**
- **fio/IO_HEPEVT.cc**
- **HepMC-2.06.09/fio/IO_HEPEVT.cc**

7.29 HepMC::IO_HERWIG Class Reference

IO_HERWIG (p. 199) is used to get Herwig information.

`#include <IO_HERWIG.h>`

Inheritance diagram for HepMC::IO_HERWIG:



Public Member Functions

- **IO_HERWIG** ()
- virtual ∼**IO_HERWIG** ()
- bool **fill_next_event** (**GenEvent** ∗)
  get the next event
- void **print** (std::ostream &ostr=std::cout) const
  write to ostr
- double **interfaces_to_version_number** () const
  this information is dubious
- bool **print_inconsistency_errors** () const
  default is true
- void **set_print_inconsistency_errors** (bool b=true)
  decide whether or not to print inconsistency errors
- bool **no_gaps_in_barcodes** () const
  ask how to deal with extra non-physical pseudo particles
- void **set_no_gaps_in_barcodes** (bool a)
- **IO_HERWIG** ()
- virtual ∼**IO_HERWIG** ()
- bool **fill_next_event** (**GenEvent** ∗)
  get the next event
- void **print** (std::ostream &ostr=std::cout) const
  write to ostr
- double **interfaces_to_version_number** () const
  this information is dubious
- bool **print_inconsistency_errors** () const
  default is true
- void **set_print_inconsistency_errors** (bool b=true)
  decide whether or not to print inconsistency errors
- bool **no_gaps_in_barcodes** () const
  ask how to deal with extra non-physical pseudo particles
- void **set_no_gaps_in_barcodes** (bool a)

Protected Member Functions

- bool **trust_both_mothers_and_daughters** () const

  default is true
- bool **trust_mothers_before_daughters** () const

  default is false
- void **set_trust_mothers_before_daughters** (bool b=true)

  define mother daughter trust rules
- void **set_trust_both_mothers_and_daughters** (bool b=false)

  define mother daughter trust rules
- **GenParticle** ∗ **build_particle** (int index)

  make a particle
- void **build_production_vertex** (int i, std::vector< **GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  make a production vertex
- void **build_end_vertex** (int i, std::vector< **GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  make a decay vertex
- int **find_in_map** (const std::map< **GenParticle** ∗, int > &m, **GenParticle** ∗**p**) const

  find this particle in the map
- void **repair_hepevt** () const

  make the HERWIG HEPEVT common block look like the standard
- void **remove_gaps_in_hepevt** () const

  deal with artifacts of repairing HEPEVT
- void **zero_hepevt_entry** (int i) const

  zero out a HEPEVT pseudo particle
- int **translate_herwig_to_pdg_id** (int i) const

  translate particle ID
- bool **trust_both_mothers_and_daughters** () const

  default is true
- bool **trust_mothers_before_daughters** () const

  default is false
- void **set_trust_mothers_before_daughters** (bool b=true)

  define mother daughter trust rules
- void **set_trust_both_mothers_and_daughters** (bool b=false)

  define mother daughter trust rules
- **GenParticle** ∗ **build_particle** (int index)

  make a particle
- void **build_production_vertex** (int i, std::vector< **GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  make a production vertex
- void **build_end_vertex** (int i, std::vector< **GenParticle** ∗ > &hepevt_particle, **GenEvent** ∗evt)

  make a decay vertex
- int **find_in_map** (const std::map< **GenParticle** ∗, int > &m, **GenParticle** ∗**p**) const

  find this particle in the map
- void **repair_hepevt** () const

  make the HERWIG HEPEVT common block look like the standard
- void **remove_gaps_in_hepevt** () const

  deal with artifacts of repairing HEPEVT
- void **zero_hepevt_entry** (int i) const

  zero out a HEPEVT pseudo particle
- int **translate_herwig_to_pdg_id** (int i) const

  translate particle ID

7.29.1 Detailed Description

IO_HERWIG (p. 199) is used to get Herwig information.

IO class for reading the HEPEVT common block from the Herwig monte carlo program.

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 56 of file IO_HERWIG.h.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 HepMC::IO_HERWIG::IO_HERWIG ( )

Definition at line 12 of file fio/IO_HERWIG.cc.

7.29.2.2 HepMC::IO_HERWIG::~IO_HERWIG ( ) [virtual]

Definition at line 83 of file fio/IO_HERWIG.cc.

7.29.2.3 HepMC::IO_HERWIG::IO_HERWIG ( )

7.29.2.4 virtual HepMC::IO_HERWIG::~IO_HERWIG ( ) [virtual]

7.29.3 Member Function Documentation

7.29.3.1 void HepMC::IO_HERWIG::build_end_vertex (int *i,* std::vector< GenParticle * > & *hepevt_particle,* GenEvent * *evt*) [protected]

make a decay vertex

for particle in HEPEVT with index i, build an end vertex if appropriate, and add that vertex to the event

for particle in HEPEVT with index i, build an end vertex if appropriate, and add that vertex to the event

Definition at line 304 of file fio/IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::GenParticle::end_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::number_children(), p, HepMC::GenVertex::position(), HepMC::GenParticle::production_vertex(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

7.29.3.2 void HepMC::IO_HERWIG::build_end_vertex (int *i,* std::vector< GenParticle * > & *hepevt_particle,* GenEvent * *evt*) [protected]

make a decay vertex

7.29.3.3 GenParticle * HepMC::IO_HERWIG::build_particle (int *index*) [protected]

make a particle

Builds a particle object corresponding to index in HEPEVT

Builds a particle object corresponding to index in HEPEVT

Definition at line 372 of file fio/IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::e(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::m(), p, HepMC::HEPEVT_Wrapper::px(), HepMC::HEPEVT_Wrapper::py(), HepMC::HEPEVT_Wrapper::pz(), HepMC::GenParticle::setGeneratedMass(), HepMC::HEPEVT_Wrapper::status(), and HepMC::GenParticle::suggest_barcode().

Referenced by fill_next_event().

7.29.3.4 GenParticle* HepMC::IO_HERWIG::build_particle (int *index*) [protected]

make a particle

7.29.3.5 void HepMC::IO_HERWIG::build_production_vertex (int *i*, std::vector< GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

make a production vertex

7.29.3.6 void HepMC::IO_HERWIG::build_production_vertex (int *i*, std::vector< GenParticle * > & *hepevt_particle*, GenEvent * *evt*) [protected]

make a production vertex

for particle in HEPEVT with index i, build a production vertex if appropriate, and add that vertex to the event

for particle in HEPEVT with index i, build a production vertex if appropriate, and add that vertex to the event

Definition at line 231 of file fio/IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::number_parents(), p, HepMC::GenVertex::position(), HepMC::GenVertex::print(), HepMC::GenParticle::production_vertex(), HepMC::GenVertex::set_position(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by fill_next_event().

7.29.3.7 bool HepMC::IO_HERWIG::fill_next_event (GenEvent *) [virtual]

get the next event

Implements **HepMC::IO_BaseClass** (p. 182).

7.29.3.8 bool HepMC::IO_HERWIG::fill_next_event (GenEvent * *evt*) [virtual]

get the next event

read one event from the Herwig HEPEVT common block and fill **GenEvent** (p. 72) return T/F =success/failure

sufficient to do one or the other.

read one event from the Herwig HEPEVT common block and fill **GenEvent** (p. 72) return T/F =success/failure

sufficient to do one or the other.

Implements **HepMC::IO_BaseClass** (p. 182).

Definition at line 96 of file fio/IO_HERWIG.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), build_end_vertex(), build_particle(), build_production_vertex(), HepMC::HEPEVT_Wrapper::event_number(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::number_entries(), repair_hepevt(), HepMC::GenEvent::set_beam_particles(), HepMC::GenEvent::set_event_number(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::HEPEVT_Wrapper::status().

7.29.3.9 int HepMC::IO_HERWIG::find_in_map (const std::map< GenParticle *, int > & *m*, GenParticle * *p*) const [protected]

find this particle in the map

7.29.3.10 int HepMC::IO_HERWIG::find_in_map (const std::map< GenParticle *, int > & *m*, GenParticle * *p*) const [protected]

find this particle in the map

Definition at line 387 of file fio/IO_HERWIG.cc.

7.29.3.11 double HepMC::IO_HERWIG::interfaces_to_version_number ( ) const [inline]

this information is dubious

Definition at line 65 of file 2.06.09/HepMC/IO_HERWIG.h.

7.29.3.12 double HepMC::IO_HERWIG::interfaces_to_version_number ( ) const [inline]

this information is dubious

Definition at line 65 of file IO_HERWIG.h.

7.29.3.13 bool HepMC::IO_HERWIG::no_gaps_in_barcodes ( ) const [inline]

ask how to deal with extra non-physical pseudo particles

Definition at line 74 of file IO_HERWIG.h.

7.29.3.14 bool HepMC::IO_HERWIG::no_gaps_in_barcodes ( ) const [inline]

ask how to deal with extra non-physical pseudo particles

Definition at line 74 of file 2.06.09/HepMC/IO_HERWIG.h.

7.29.3.15 void HepMC::IO_HERWIG::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

7.29.3.16 void HepMC::IO_HERWIG::print (std::ostream & *ostr* = std::cout) const [virtual]

write to ostr

Reimplemented from **HepMC::IO_BaseClass** (p. 183).

Definition at line 85 of file fio/IO_HERWIG.cc.

7.29.3.17 bool HepMC::IO_HERWIG::print_inconsistency_errors ( ) const [inline]

default is true

Definition at line 145 of file IO_HERWIG.h.

7.29.3.18 bool HepMC::IO_HERWIG::print_inconsistency_errors ( ) const

default is true

7.29.3.19 void HepMC::IO_HERWIG::remove_gaps_in_hepevt ( ) const [protected]

deal with artifacts of repairing HEPEVT

in this scenario, we do not allow there to be zero-ed entries in the HEPEVT common block, and so be reshuffle the common block, removing the zeero-ed entries as we go and making sure we keep the mother/daughter relationships appropriate

in this scenario, we do not allow there to be zero-ed entries in the HEPEVT common block, and so be reshuffle the common block, removing the zeero-ed entries as we go and making sure we keep the mother/daughter relationships appropriate

Definition at line 682 of file fio/IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::e(), HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_-Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HE-PEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::m(), HepMC::HEPEVT_Wrapper::number_entries(), HepMC::HEPEVT_Wrapper::px(), HepMC::HEPEVT_Wrapper::py(), HepMC::HEPEVT_Wrapper::pz(), HepMC::-HEPEVT_Wrapper::set_children(), HepMC::HEPEVT_Wrapper::set_id(), HepMC::HEPEVT_Wrapper::set_mass(), HepMC::HEPEVT_Wrapper::set_momentum(), HepMC::HEPEVT_Wrapper::set_number_entries(), HepMC::H-EPEVT_Wrapper::set_parents(), HepMC::HEPEVT_Wrapper::set_position(), HepMC::HEPEVT_Wrapper::set_-status(), HepMC::HEPEVT_Wrapper::status(), HepMC::HEPEVT_Wrapper::t(), HepMC::HEPEVT_Wrapper::x(), HepMC::HEPEVT_Wrapper::y(), and HepMC::HEPEVT_Wrapper::z().

Referenced by repair_hepevt().

7.29.3.20 void HepMC::IO_HERWIG::remove_gaps_in_hepevt ( ) const [protected]

deal with artifacts of repairing HEPEVT

7.29.3.21 void HepMC::IO_HERWIG::repair_hepevt ( ) const [protected]

make the HERWIG HEPEVT common block look like the standard

7.29.3.22 void HepMC::IO_HERWIG::repair_hepevt ( ) const [protected]

make the HERWIG HEPEVT common block look like the standard

This routine takes the HEPEVT common block as used in HERWIG, and converts it into the HEPEVT common block in the standard format

This means it:

- removes the color structure, which herwig overloads into the mother/daughter fields
- zeros extra entries for hard subprocess, etc.

Special HERWIG status codes 101,102 colliding beam particles 103 beam-beam collision CMS vector 120 hard subprocess CMS vector 121,122 hard subprocess colliding partons 123-129 hard subprocess outgoing particles 141-149 (ID=94) mirror image of hard subrpocess particles 100 (ID=0 cone)

Special HERWIG particle id's 91 clusters 94 jets 0 others with no pdg code

This routine takes the HEPEVT common block as used in HERWIG, and converts it into the HEPEVT common block in the standard format

This means it:

- removes the color structure, which herwig overloads into the mother/daughter fields
- zeros extra entries for hard subprocess, etc.

Special HERWIG status codes 101,102 colliding beam particles 103 beam-beam collision CMS vector 120 hard subprocess CMS vector 121,122 hard subprocess colliding partons 123-129 hard subprocess outgoing particles 141-149 (ID=94) mirror image of hard subrpocess particles 100 (ID=0 cone)

Special HERWIG particle id's 91 clusters 94 jets 0 others with no pdg code

Definition at line 394 of file fio/IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::first_child(), HepMC::HEPEVT_Wrapper::first_parent(), HepMC::HEPEVT_Wrapper::id(), HepMC::HEPEVT_Wrapper::last_child(), HepMC::HEPEVT_Wrapper::last_parent(), HepMC::HEPEVT_Wrapper::number_entries(), remove_gaps_in_hepevt(), HepMC::HEPEVT_Wrapper::set_children(), HepMC::HEPEVT_Wrapper::set_id(), HepMC::HEPEVT_Wrapper::set_parents(), HepMC::HEPEVT_Wrapper::status(), translate_herwig_to_pdg_id(), and zero_hepevt_entry().

Referenced by fill_next_event().

7.29.3.23 void HepMC::IO_HERWIG::set_no_gaps_in_barcodes (bool *a*) [inline]

The HERWIG HEPEVT common block has some EXTRA non-physical ENTRIES (such as CMS frame, HARD subprocess, and CONE). These are removed by **IO_HERWIG** (p. 199). Thus the **HepMC** (p. 15) event will APPEAR to have fewer particles in it that herwig did. There is a switch m_no_gaps_in_barcodes. For true - then the extra particles are removed from HEPEVT, with the result that the **HepMC** (p. 15) barcodes will be sequential, with no gaps. false - the barcodes will correspond directly to the HEPEVT index, but there will be gaps ... ie some barcodes will be unassigned. this switch requested by I Hinchliffe, October 31, 2002

Definition at line 87 of file 2.06.09/HepMC/IO_HERWIG.h.

7.29.3.24 void HepMC::IO_HERWIG::set_no_gaps_in_barcodes (bool *a*) [inline]

The HERWIG HEPEVT common block has some EXTRA non-physical ENTRIES (such as CMS frame, HARD subprocess, and CONE). These are removed by **IO_HERWIG** (p. 199). Thus the **HepMC** (p. 15) event will APPEAR to have fewer particles in it that herwig did. There is a switch m_no_gaps_in_barcodes. For true - then the extra particles are removed from HEPEVT, with the result that the **HepMC** (p. 15) barcodes will be sequential, with no gaps. false - the barcodes will correspond directly to the HEPEVT index, but there will be gaps ... ie some barcodes will be unassigned. this switch requested by I Hinchliffe, October 31, 2002

Definition at line 87 of file IO_HERWIG.h.

7.29.3.25 void HepMC::IO_HERWIG::set_print_inconsistency_errors (bool *b* = true) [inline]

decide whether or not to print inconsistency errors

Definition at line 154 of file IO_HERWIG.h.

7.29.3.26 void HepMC::IO_HERWIG::set_print_inconsistency_errors (bool *b* = `true`)

decide whether or not to print inconsistency errors

7.29.3.27 void HepMC::IO_HERWIG::set_trust_both_mothers_and_daughters (bool *b* = `false`) [inline], [protected]

define mother daughter trust rules

Definition at line 148 of file IO_HERWIG.h.

7.29.3.28 void HepMC::IO_HERWIG::set_trust_both_mothers_and_daughters (bool *b* = `false`) [protected]

define mother daughter trust rules

7.29.3.29 void HepMC::IO_HERWIG::set_trust_mothers_before_daughters (bool *b* = `true`) [inline], [protected]

define mother daughter trust rules

Definition at line 151 of file IO_HERWIG.h.

7.29.3.30 void HepMC::IO_HERWIG::set_trust_mothers_before_daughters (bool *b* = `true`) [protected]

define mother daughter trust rules

7.29.3.31 int HepMC::IO_HERWIG::translate_herwig_to_pdg_id (int *i*) const [protected]

translate particle ID

7.29.3.32 int HepMC::IO_HERWIG::translate_herwig_to_pdg_id (int *i*) const [protected]

translate particle ID

This routine is copied from Lynn Garren's stdhep 5.01. see `http:///cepa.fnal.gov/psm/stdhep/`

This routine is copied from Lynn Garren's stdhep 5.01. see `http:///cepa.fnal.gov/psm/stdhep/`

Definition at line 753 of file fio/IO_HERWIG.cc.

Referenced by repair_hepevt().

7.29.3.33 bool HepMC::IO_HERWIG::trust_both_mothers_and_daughters ( ) const [inline], [protected]

default is true

Definition at line 139 of file IO_HERWIG.h.

7.29.3.34 bool HepMC::IO_HERWIG::trust_both_mothers_and_daughters ( ) const [protected]

default is true

7.29.3.35 bool HepMC::IO_HERWIG::trust_mothers_before_daughters ( ) const [protected]

default is false

7.29.3.36 bool HepMC::IO_HERWIG::trust_mothers_before_daughters ( ) const [inline], [protected]

default is false

Definition at line 142 of file IO_HERWIG.h.

7.29.3.37 void HepMC::IO_HERWIG::zero_hepevt_entry (int *i*) const [protected]

zero out a HEPEVT pseudo particle

7.29.3.38 void HepMC::IO_HERWIG::zero_hepevt_entry (int *i*) const [protected]

zero out a HEPEVT pseudo particle

Definition at line 742 of file fio/IO_HERWIG.cc.

References HepMC::HEPEVT_Wrapper::max_number_entries(), HepMC::HEPEVT_Wrapper::set_children(), HepMC::HEPEVT_Wrapper::set_id(), HepMC::HEPEVT_Wrapper::set_mass(), HepMC::HEPEVT_Wrapper::set_momentum(), HepMC::HEPEVT_Wrapper::set_parents(), HepMC::HEPEVT_Wrapper::set_position(), and HepMC::HEPEVT_Wrapper::set_status().

Referenced by repair_hepevt().

The documentation for this class was generated from the following files:

- **IO_HERWIG.h**
- **2.06.09/HepMC/IO_HERWIG.h**
- **fio/IO_HERWIG.cc**
- **HepMC-2.06.09/fio/IO_HERWIG.cc**

7.30 HepMC::detail::is_arithmetic< T > Struct Template Reference

undefined and therefore non-arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = false

7.30.1 Detailed Description

template<class T>struct HepMC::detail::is_arithmetic< T >

undefined and therefore non-arithmetic

Definition at line 22 of file is_arithmetic.h.

7.30.2 Member Data Documentation

7.30.2.1 template<class T > static bool const HepMC::detail::is_arithmetic< T >::value = false [static]

Definition at line 24 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.31 HepMC::detail::is_arithmetic< char > Struct Template Reference

character is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.31.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< char >

character is arithmetic

Definition at line 29 of file is_arithmetic.h.

7.31.2 Member Data Documentation

7.31.2.1 static bool const HepMC::detail::is_arithmetic< char >::value = true [static]

Definition at line 30 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.32 HepMC::detail::is_arithmetic< double > Struct Template Reference

double is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.32.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< double >

double is arithmetic

Definition at line 79 of file is_arithmetic.h.

7.32.2 Member Data Documentation

7.32.2.1 static bool const HepMC::detail::is_arithmetic< double >::value = true [static]

Definition at line 80 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.33 HepMC::detail::is_arithmetic< float > Struct Template Reference

float is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.33.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< float >

float is arithmetic

Definition at line 74 of file is_arithmetic.h.

7.33.2 Member Data Documentation

7.33.2.1 static bool const HepMC::detail::is_arithmetic< float >::value = true [static]

Definition at line 75 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.34 HepMC::detail::is_arithmetic< int > Struct Template Reference

int is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.34.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< int >

int is arithmetic

Definition at line 54 of file is_arithmetic.h.

7.34.2 Member Data Documentation

7.34.2.1 static bool const HepMC::detail::is_arithmetic< int >::value = true [static]

Definition at line 55 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.35 HepMC::detail::is_arithmetic< long > Struct Template Reference

long is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.35.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< long >

long is arithmetic

Definition at line 64 of file is_arithmetic.h.

7.35.2 Member Data Documentation

7.35.2.1 static bool const HepMC::detail::is_arithmetic< long >::value = true [static]

Definition at line 65 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.36 HepMC::detail::is_arithmetic< long double > Struct Template Reference

long double is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.36.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< long double >

long double is arithmetic

Definition at line 84 of file is_arithmetic.h.

7.36.2 Member Data Documentation

7.36.2.1 static bool const HepMC::detail::is_arithmetic< long double >::value = true [static]

Definition at line 85 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.37 HepMC::detail::is_arithmetic< short > Struct Template Reference

short is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.37.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< short >

short is arithmetic

Definition at line 44 of file is_arithmetic.h.

7.37.2 Member Data Documentation

7.37.2.1 static bool const HepMC::detail::is_arithmetic< short >::value = true [static]

Definition at line 45 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.38 HepMC::detail::is_arithmetic< signed char > Struct Template Reference

signed character is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.38.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< signed char >

signed character is arithmetic

Definition at line 39 of file is_arithmetic.h.

7.38.2 Member Data Documentation

7.38.2.1 static bool const HepMC::detail::is_arithmetic< signed char >::value = true [static]

Definition at line 40 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.39 HepMC::detail::is_arithmetic< unsigned char > Struct Template Reference

unsigned character is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.39.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< unsigned char >

unsigned character is arithmetic

Definition at line 34 of file is_arithmetic.h.

7.39.2 Member Data Documentation

7.38.2.1 static bool const HepMC::detail::is_arithmetic< unsigned char >::value = true [static]

Definition at line 35 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.40 HepMC::detail::is_arithmetic< unsigned int > Struct Template Reference

unsigned int is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.40.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< unsigned int >

unsigned int is arithmetic

Definition at line 59 of file is_arithmetic.h.

7.40.2 Member Data Documentation

7.40.2.1 static bool const HepMC::detail::is_arithmetic< unsigned int >::value = true [static]

Definition at line 60 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.41 HepMC::detail::is_arithmetic< unsigned long > Struct Template Reference

unsigned long is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.41.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< unsigned long >

unsigned long is arithmetic

Definition at line 69 of file is_arithmetic.h.

7.41.2 Member Data Documentation

7.41.2.1 static bool const HepMC::detail::is_arithmetic< unsigned long >::value = true [static]

Definition at line 70 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.42 HepMC::detail::is_arithmetic< unsigned short > Struct Template Reference

unsigned short is arithmetic

`#include <is_arithmetic.h>`

Static Public Attributes

- static bool const **value** = true

7.42.1 Detailed Description

template<>struct HepMC::detail::is_arithmetic< unsigned short >

unsigned short is arithmetic

Definition at line 49 of file is_arithmetic.h.

7.42.2 Member Data Documentation

7.42.2.1 static bool const HepMC::detail::is_arithmetic< unsigned short >::value = true [static]

Definition at line 50 of file is_arithmetic.h.

The documentation for this struct was generated from the following files:

- **is_arithmetic.h**
- **2.06.09/HepMC/is_arithmetic.h**

7.43 IsEventGood Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenEvent** *evt)

  check this event for goodness
- bool **operator()** (const **HepMC::GenEvent** *evt)

  check this event for goodness

7.43.1 Detailed Description

example class

event selection predicate. returns true if the event contains a photon with pT $>$ 50 GeV

Examples:

example_EventSelection.cc.

Definition at line 20 of file examples/example_EventSelection.cc.

7.43.2 Member Function Documentation

7.43.2.1 bool IsEventGood::operator() (const HepMC::GenEvent * *evt*) [inline]

check this event for goodness

Examples:

example_EventSelection.cc.

Definition at line 23 of file examples/example_EventSelection.cc.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

7.43.2.2 bool IsEventGood::operator() (const HepMC::GenEvent * *evt*) [inline]

check this event for goodness

Definition at line 23 of file HepMC-2.06.09/examples/example_EventSelection.cc.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

The documentation for this class was generated from the following files:

- **examples/example_EventSelection.cc**
- **HepMC-2.06.09/examples/example_EventSelection.cc**

7.44 IsFinalState Class Reference

`#include <testHepMCIteration.h>`

Public Member Functions

- bool **operator()** (const **HepMC::GenParticle** *p)
  returns true if the GenParticle does not decay

7.44.1 Detailed Description

this predicate returns true if the input has no decay vertex

Examples:

testHepMCIteration.cc.in.

Definition at line 24 of file testHepMCIteration.h.

7.44.2 Member Function Documentation

7.44.2.1 bool IsFinalState::operator() (const HepMC::GenParticle * p) [inline]

returns true if the GenParticle does not decay

Definition at line 27 of file testHepMCIteration.h.

References HepMC::GenParticle::end_vertex(), and HepMC::GenParticle::status().

The documentation for this class was generated from the following file:

- **testHepMCIteration.h**

7.45 IsGoodEvent Class Reference

used in the tests

`#include <IsGoodEvent.h>`

Public Member Functions

- bool **operator()** (const **HepMC::GenEvent** *evt)

7.45.1 Detailed Description

used in the tests

event selection predicate. returns true if the event contains a photon with pT $>$ 50 GeV

Examples:

testHepMC.cc.in, **testHepMCIteration.cc.in**, **testMass.cc.in**, **testMultipleCopies.cc.in**, and **testStreamIO.-cc.in**.

Definition at line 14 of file IsGoodEvent.h.

7.45.2 Member Function Documentation

7.45.2.1 bool IsGoodEvent::operator() (const HepMC::GenEvent * evt) [inline]

Definition at line 16 of file IsGoodEvent.h.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

The documentation for this class was generated from the following file:

- **IsGoodEvent.h**

7.46 IsGoodEventMyPythia Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenEvent** ∗evt)
  returns true if event is "good"
- bool **operator()** (const **HepMC::GenEvent** ∗evt)
  returns true if event is "good"

7.46.1 Detailed Description

example class

example of generating events with Pythia using HepMC/PythiaWrapper.h Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy

To Compile: go to the **HepMC** (p. 15) directory and type: gmake examples/example_MyPythia.exe

In this example the precision and number of entries for the HEPEVT fortran common block are explicitly defined to correspond to those used in the Pythia version of the HEPEVT common block.

If you get funny output from HEPEVT in your own code, probably you have set these values incorrectly!

pythia_out() (p. 313): Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy and then output to file in ascii format using the IO_GenEvent strategy.

pythia_particle_out() (p. 313): Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy and then output to file in ascii format using the IO_AsciiParticles strategy. This is identical to **pythia_out()** (p. 313) except for the choice of output format.

event_selection() (p. 312): Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy and then a very simple event selection is performed.

pythia_in() (p. 312): Read the file created by **pythia_out()** (p. 313).

pythia_in_out() (p. 312): generate events with Pythia, write a file, and read the resulting output Notice that we use scope to explicitly close the ouput files. The two output files should be identical. event selection predicate. returns true if the event contains a photon with pT $>$ 25 GeV

Examples:

fio/example_MyPythia.cc.

Definition at line 61 of file examples/fio/example_MyPythia.cc.

7.46.2 Member Function Documentation

7.46.2.1 bool IsGoodEventMyPythia::operator() (const HepMC::GenEvent ∗ *evt*) [inline]

returns true if event is "good"

Examples:

fio/example_MyPythia.cc.

Definition at line 64 of file examples/fio/example_MyPythia.cc.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

7.46.2.2 bool IsGoodEventMyPythia::operator() (const HepMC::GenEvent ∗ *evt*) [inline]

returns true if event is "good"

Definition at line 64 of file HepMC-2.06.09/examples/fio/example_MyPythia.cc.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

The documentation for this class was generated from the following files:

- **examples/fio/example_MyPythia.cc**
- **HepMC-2.06.09/examples/fio/example_MyPythia.cc**

7.47 IsPhoton Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle is a photon with more than 10 GeV transverse momentum
- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle is a photon with more than 10 GeV transverse momentum

7.47.1 Detailed Description

example class

this predicate returns true if the input particle is a photon in the central region (eta $<$ 2.5) with pT $>$ 10 GeV

Examples:

example_UsingIterators.cc.

Definition at line 20 of file examples/example_UsingIterators.cc.

7.47.2 Member Function Documentation

7.47.2.1 bool IsPhoton::operator() (const HepMC::GenParticle ∗ *p*) [inline]

returns true if the GenParticle is a photon with more than 10 GeV transverse momentum

Examples:

example_UsingIterators.cc.

Definition at line 23 of file examples/example_UsingIterators.cc.

References HepMC::GenParticle::momentum(), HepMC::GenParticle::pdg_id(), and HepMC::FourVector::perp().

7.47.2.2 bool IsPhoton::operator() (const HepMC::GenParticle ∗ *p*) [inline]

returns true if the GenParticle is a photon with more than 10 GeV transverse momentum

Definition at line 23 of file HepMC-2.06.09/examples/example_UsingIterators.cc.

References HepMC::GenParticle::momentum(), HepMC::GenParticle::pdg_id(), and HepMC::FourVector::perp().

The documentation for this class was generated from the following files:

- **examples/example_UsingIterators.cc**
- **HepMC-2.06.09/examples/example_UsingIterators.cc**

7.48 IsStateFinal Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle does not decay
- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle does not decay

7.48.1 Detailed Description

example class

this predicate returns true if the input has no decay vertex

Examples:

example_UsingIterators.cc.

Definition at line 47 of file examples/example_UsingIterators.cc.

7.48.2 Member Function Documentation

7.48.2.1 bool IsStateFinal::operator() (const HepMC::GenParticle ∗ *p*) [inline]

returns true if the GenParticle does not decay

Examples:

example_UsingIterators.cc.

Definition at line 50 of file examples/example_UsingIterators.cc.

References HepMC::GenParticle::end_vertex(), and HepMC::GenParticle::status().

7.48.2.2 bool IsStateFinal::operator() (const HepMC::GenParticle ∗ *p*) [inline]

returns true if the GenParticle does not decay

Definition at line 50 of file HepMC-2.06.09/examples/example_UsingIterators.cc.

References HepMC::GenParticle::end_vertex(), and HepMC::GenParticle::status().

The documentation for this class was generated from the following files:

- **examples/example_UsingIterators.cc**
- **HepMC-2.06.09/examples/example_UsingIterators.cc**

7.49 IsW_Boson Class Reference

example class

Public Member Functions

- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle is a W
- bool **operator()** (const **HepMC::GenParticle** ∗**p**)
  returns true if the GenParticle is a W

7.49.1 Detailed Description

example class

this predicate returns true if the input particle is a W+/W-

Examples:

example_UsingIterators.cc.

Definition at line 34 of file examples/example_UsingIterators.cc.

7.49.2 Member Function Documentation

7.49.2.1 bool IsW_Boson::operator() (const HepMC::GenParticle ∗ *p*) [inline]

returns true if the GenParticle is a W

Examples:

example_UsingIterators.cc.

Definition at line 37 of file examples/example_UsingIterators.cc.

References HepMC::GenParticle::pdg_id().

7.49.2.2 bool IsW_Boson::operator() (const HepMC::GenParticle ∗ *p*) [inline]

returns true if the GenParticle is a W

Definition at line 37 of file HepMC-2.06.09/examples/example_UsingIterators.cc.

References HepMC::GenParticle::pdg_id().

The documentation for this class was generated from the following files:

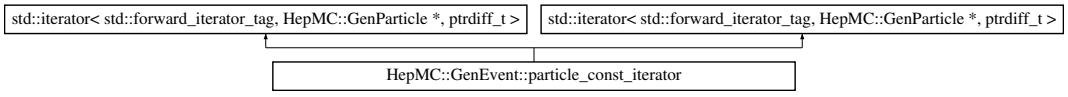
- **examples/example_UsingIterators.cc**
- **HepMC-2.06.09/examples/example_UsingIterators.cc**

7.50 HepMC::GenEvent::particle_const_iterator Class Reference

const particle iterator

`#include <GenEvent.h>`

Inheritance diagram for HepMC::GenEvent::particle_const_iterator:



Public Member Functions

- **particle_const_iterator** (const std::map< int, **HepMC::GenParticle** * >::const_iterator &i)

  iterate over particles
- **particle_const_iterator** ()
- **particle_const_iterator** (const **particle_const_iterator** &i)

  copy constructor
- virtual ~**particle_const_iterator** ()
- **particle_const_iterator** & **operator=** (const **particle_const_iterator** &i)

  make a copy
- **GenParticle** * **operator*** (void) const

  *return a pointer to **GenParticle*** (p. *106)*
- **particle_const_iterator** & **operator++** (void)

  Pre-fix increment.
- **particle_const_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **particle_const_iterator** &a) const

  equality
- bool **operator!=** (const **particle_const_iterator** &a) const

  inequality
- **particle_const_iterator** (const std::map< int, **HepMC::GenParticle** * >::const_iterator &i)

  iterate over particles
- **particle_const_iterator** ()
- **particle_const_iterator** (const **particle_const_iterator** &i)

  copy constructor
- virtual ~**particle_const_iterator** ()
- **particle_const_iterator** & **operator=** (const **particle_const_iterator** &i)

  make a copy
- **GenParticle** * **operator*** (void) const

  *return a pointer to **GenParticle*** (p. *106)*
- **particle_const_iterator** & **operator++** (void)

  Pre-fix increment.
- **particle_const_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **particle_const_iterator** &a) const

  equality
- bool **operator!=** (const **particle_const_iterator** &a) const

  inequality

Protected Attributes

- std::map< int,
  HepMC::GenParticle * >
  ::const_iterator **m_map_iterator**

  *const iterator to the **GenParticle*** (p. *106) map*

7.50.1 Detailed Description

const particle iterator

HepMC::GenEvent::particle_const_iterator (p. 220) is used to iterate over all particles in the event.

Examples:

example_EventSelection.cc, **example_VectorConversion.cc**, **fio/example_MyPythia.cc**, **testMass.cc.in**, and **testMultipleCopies.cc.in**.

Definition at line 464 of file GenEvent.h.

7.50.2 Constructor & Destructor Documentation

7.50.2.1 HepMC::GenEvent::particle_const_iterator::particle_const_iterator (const std::map< int, HepMC::GenParticle * >::const_iterator & *i*) [inline]

iterate over particles

Definition at line 469 of file GenEvent.h.

7.50.2.2 HepMC::GenEvent::particle_const_iterator::particle_const_iterator ( ) [inline]

Definition at line 472 of file GenEvent.h.

7.50.2.3 HepMC::GenEvent::particle_const_iterator::particle_const_iterator (const particle_const_iterator & *i*) [inline]

copy constructor

Definition at line 474 of file GenEvent.h.

7.50.2.4 virtual HepMC::GenEvent::particle_const_iterator::~particle_const_iterator ( ) [inline], [virtual]

Definition at line 476 of file GenEvent.h.

7.50.2.5 HepMC::GenEvent::particle_const_iterator::particle_const_iterator (const std::map< int, HepMC::GenParticle * >::const_iterator & *i*) [inline]

iterate over particles

Definition at line 469 of file 2.06.09/HepMC/GenEvent.h.

7.50.2.6 HepMC::GenEvent::particle_const_iterator::particle_const_iterator ( ) [inline]

Definition at line 472 of file 2.06.09/HepMC/GenEvent.h.

7.50.2.7 HepMC::GenEvent::particle_const_iterator::particle_const_iterator (const particle_const_iterator & *i*) [inline]

copy constructor

Definition at line 474 of file 2.06.09/HepMC/GenEvent.h.

7.50.2.8 virtual HepMC::GenEvent::particle_const_iterator::~particle_const_iterator ( ) [inline], [virtual]

Definition at line 476 of file 2.06.09/HepMC/GenEvent.h.

7.50.3 Member Function Documentation

7.50.3.1 bool HepMC::GenEvent::particle_const_iterator::operator!= (const particle_const_iterator & *a*) const [inline]

inequality

Definition at line 494 of file GenEvent.h.

References m_map_iterator.

7.50.3.2 bool HepMC::GenEvent::particle_const_iterator::operator!= (const particle_const_iterator & *a*) const [inline]

inequality

Definition at line 494 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.50.3.3 GenParticle* HepMC::GenEvent::particle_const_iterator::operator* (void) const [inline]

return a pointer to **GenParticle** (p. 106)

Definition at line 482 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.50.3.4 GenParticle* HepMC::GenEvent::particle_const_iterator::operator* (void) const [inline]

return a pointer to **GenParticle** (p. 106)

Definition at line 482 of file GenEvent.h.

References m_map_iterator.

7.50.3.5 particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 485 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.50.3.6 particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 485 of file GenEvent.h.

References m_map_iterator.

7.50.3.7 particle_const_iterator HepMC::GenEvent::particle_const_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 488 of file GenEvent.h.

7.50.3.8 particle_const_iterator HepMC::GenEvent::particle_const_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 488 of file 2.06.09/HepMC/GenEvent.h.

7.50.3.9 particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator= (const particle_const_iterator & *i*) [inline]

make a copy

Definition at line 478 of file GenEvent.h.

References m_map_iterator.

7.50.3.10 particle_const_iterator& HepMC::GenEvent::particle_const_iterator::operator= (const particle_const_iterator & *i*) [inline]

make a copy

Definition at line 478 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.50.3.11 bool HepMC::GenEvent::particle_const_iterator::operator== (const particle_const_iterator & *a*) const [inline]

equality

Definition at line 491 of file GenEvent.h.

References m_map_iterator.

7.50.3.12 bool HepMC::GenEvent::particle_const_iterator::operator== (const particle_const_iterator & *a*) const [inline]

equality

Definition at line 491 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.50.4 Member Data Documentation

7.50.4.1 std::map< int, HepMC::GenParticle * >::const_iterator HepMC::GenEvent::particle_const_iterator::m_map_iterator [protected]

const iterator to the **GenParticle** (p. 106) map

Definition at line 498 of file GenEvent.h.

Referenced by operator!=(), operator*(), operator++(), operator=(), and operator==().

The documentation for this class was generated from the following files:

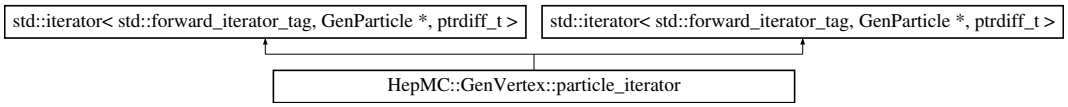
- **GenEvent.h**
- **2.06.09/HepMC/GenEvent.h**

7.51 HepMC::GenVertex::particle_iterator Class Reference

particle iterator

`#include <GenVertex.h>`

Inheritance diagram for HepMC::GenVertex::particle_iterator:



Public Member Functions

- **particle_iterator** ()
- **particle_iterator** (**GenVertex** &vertex_root, **IteratorRange** range)

  used to set limits on the iteration
- **particle_iterator** (const **particle_iterator** &)

  copy
- virtual ~**particle_iterator** ()
- **particle_iterator** & **operator=** (const **particle_iterator** &)

  make a copy
- **GenParticle** * **operator*** (void) const

  return a pointer to a particle
- **particle_iterator** & **operator++** (void)

  Pre-fix increment.
- **particle_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **particle_iterator** &) const

  equality
- bool **operator!=** (const **particle_iterator** &) const

  inequality
- **particle_iterator** ()
- **particle_iterator** (**GenVertex** &vertex_root, **IteratorRange** range)

  used to set limits on the iteration
- **particle_iterator** (const **particle_iterator** &)

  copy
- virtual ~**particle_iterator** ()
- **particle_iterator** & **operator=** (const **particle_iterator** &)

  make a copy
- **GenParticle** * **operator*** (void) const

  return a pointer to a particle
- **particle_iterator** & **operator++** (void)

  Pre-fix increment.
- **particle_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **particle_iterator** &) const

  equality
- bool **operator!=** (const **particle_iterator** &) const

  inequality

Protected Member Functions

- **GenParticle** ∗ **advance_to_first_** ()
  "first" particle
- **GenParticle** ∗ **advance_to_first_** ()
  "first" particle

7.51.1 Detailed Description

particle iterator

Iterates over all particles connected via a graph. by iterating through all vertices in the m_range. For each vertex it returns orphaned parent particles (i.e. parents without production vertices) then children ... in this way each particle is associated to exactly one vertex and so it is returned exactly once. Is made friend so that it can access protected edge iterator

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 339 of file GenVertex.h.

7.51.2 Constructor & Destructor Documentation

7.51.2.1 HepMC::GenVertex::particle_iterator::particle_iterator ( )

Definition at line 838 of file HepMC-2.06.09/src/GenVertex.cc.

7.51.2.2 HepMC::GenVertex::particle_iterator::particle_iterator (GenVertex & *vertex_root,* IteratorRange *range*)

used to set limits on the iteration

Definition at line 840 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenVertex::edge_iterator, HepMC::family, and HepMC::GenVertex::vertex_iterator.

7.51.2.3 HepMC::GenVertex::particle_iterator::particle_iterator (const particle_iterator & *p_iter*)

copy

Definition at line 854 of file HepMC-2.06.09/src/GenVertex.cc.

7.51.2.4 HepMC::GenVertex::particle_iterator::∼particle_iterator ( ) [virtual]

Definition at line 859 of file HepMC-2.06.09/src/GenVertex.cc.

7.51.2.5 HepMC::GenVertex::particle_iterator::particle_iterator ( )

7.51.2.6 HepMC::GenVertex::particle_iterator::particle_iterator (GenVertex & *vertex_root,* IteratorRange *range*)

used to set limits on the iteration

7.51.2.7 HepMC::GenVertex::particle_iterator::particle_iterator (const particle_iterator &)

copy

7.51.2.8 virtual HepMC::GenVertex::particle_iterator::~particle_iterator ( ) [virtual]

7.51.3 Member Function Documentation

7.51.3.1 GenParticle * HepMC::GenVertex::particle_iterator::advance_to_first_ ( ) [protected]

"first" particle

if the current edge is not a suitable return value (because it is a parent of the vertex root that itself belongs to a different vertex) it advances to the first suitable return value

if the current edge is not a suitable return value (because it is a parent of the vertex root that itself belongs to a different vertex) it advances to the first suitable return value

Definition at line 900 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::relatives.

7.51.3.2 GenParticle* HepMC::GenVertex::particle_iterator::advance_to_first_ ( ) [protected]

"first" particle

7.51.3.3 bool HepMC::GenVertex::particle_iterator::operator!= (const particle_iterator &) const

inequality

7.51.3.4 bool HepMC::GenVertex::particle_iterator::operator!= (const particle_iterator & *a*) const [inline]

inequality

Definition at line 520 of file GenVertex.h.

7.51.3.5 GenParticle* HepMC::GenVertex::particle_iterator::operator* (void) const

return a pointer to a particle

7.51.3.6 GenParticle * HepMC::GenVertex::particle_iterator::operator* (void) const

return a pointer to a particle

Definition at line 869 of file HepMC-2.06.09/src/GenVertex.cc.

7.51.3.7 particle_iterator& HepMC::GenVertex::particle_iterator::operator++ (void)

Pre-fix increment.

7.51.3.8 GenVertex::particle_iterator & HepMC::GenVertex::particle_iterator::operator++ (void)

Pre-fix increment.

Definition at line 874 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenVertex::edge_iterator.

7.51.3.9 GenVertex::particle_iterator HepMC::GenVertex::particle_iterator::operator++ (int)

Post-fix increment.

Definition at line 893 of file HepMC-2.06.09/src/GenVertex.cc.

7.51.3.10 particle_iterator HepMC::GenVertex::particle_iterator::operator++ (int)

Post-fix increment.

7.51.3.11 GenVertex::particle_iterator & HepMC::GenVertex::particle_iterator::operator= (const particle_iterator & *p_iter*)

make a copy

Definition at line 862 of file HepMC-2.06.09/src/GenVertex.cc.

7.51.3.12 particle_iterator& HepMC::GenVertex::particle_iterator::operator= (const particle_iterator &)

make a copy

7.51.3.13 bool HepMC::GenVertex::particle_iterator::operator== (const particle_iterator &) const

equality

7.51.3.14 bool HepMC::GenVertex::particle_iterator::operator== (const particle_iterator & *a*) const [inline]

equality

Definition at line 515 of file GenVertex.h.

The documentation for this class was generated from the following files:

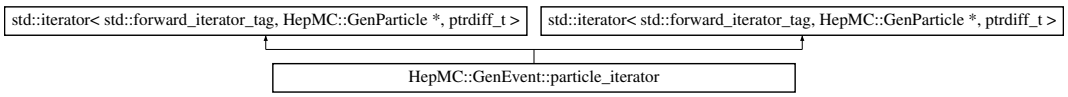
- **GenVertex.h**
- **2.06.09/HepMC/GenVertex.h**
- **HepMC-2.06.09/src/GenVertex.cc**
- **src/GenVertex.cc**

7.52 HepMC::GenEvent::particle_iterator Class Reference

non-const particle iterator

`#include <GenEvent.h>`

Inheritance diagram for HepMC::GenEvent::particle_iterator:



Public Member Functions

- **particle_iterator** (const std::map< int, **HepMC::GenParticle** * >::iterator &i)

iterate over particles

- **particle_iterator** ()
- **particle_iterator** (const **particle_iterator** &i)

  copy constructor
- virtual ∼**particle_iterator** ()
- **particle_iterator** & **operator=** (const **particle_iterator** &i)

  make a copy
- **operator particle_const_iterator** () const

  const particle iterator
- **GenParticle** ∗ **operator**∗ (void) const

  *return pointer to **GenParticle** (*p. *106)*
- **particle_iterator** & **operator++** (void)

  Pre-fix increment.
- **particle_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **particle_iterator** &a) const

  equality
- bool **operator!=** (const **particle_iterator** &a) const

  inequality
- **particle_iterator** (const std::map< int, **HepMC::GenParticle** ∗ >::iterator &i)

  iterate over particles
- **particle_iterator** ()
- **particle_iterator** (const **particle_iterator** &i)

  copy constructor
- virtual ∼**particle_iterator** ()
- **particle_iterator** & **operator=** (const **particle_iterator** &i)

  make a copy
- **operator particle_const_iterator** () const

  const particle iterator
- **GenParticle** ∗ **operator**∗ (void) const

  *return pointer to **GenParticle** (*p. *106)*
- **particle_iterator** & **operator++** (void)

  Pre-fix increment.
- **particle_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **particle_iterator** &a) const

  equality
- bool **operator!=** (const **particle_iterator** &a) const

  inequality

Protected Attributes

- std::map< int,
  HepMC::GenParticle ∗ >
  ::iterator **m_map_iterator**

  *iterator for **GenParticle** (*p. *106) map*

7.52.1 Detailed Description

non-const particle iterator

HepMC::GenEvent::particle_iterator (p. 228) is used to iterate over all particles in the event.

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 520 of file GenEvent.h.

7.52.2 Constructor & Destructor Documentation

7.52.2.1 HepMC::GenEvent::particle_iterator::particle_iterator (const std::map< int, HepMC::GenParticle * >::iterator & *i*) [inline]

iterate over particles

Definition at line 525 of file GenEvent.h.

7.52.2.2 HepMC::GenEvent::particle_iterator::particle_iterator ( ) [inline]

Definition at line 527 of file GenEvent.h.

7.52.2.3 HepMC::GenEvent::particle_iterator::particle_iterator (const particle_iterator & *i*) [inline]

copy constructor

Definition at line 529 of file GenEvent.h.

7.52.2.4 virtual HepMC::GenEvent::particle_iterator::~particle_iterator ( ) [inline], [virtual]

Definition at line 530 of file GenEvent.h.

7.52.2.5 HepMC::GenEvent::particle_iterator::particle_iterator (const std::map< int, HepMC::GenParticle * >::iterator & *i*) [inline]

iterate over particles

Definition at line 525 of file 2.06.09/HepMC/GenEvent.h.

7.52.2.6 HepMC::GenEvent::particle_iterator::particle_iterator ( ) [inline]

Definition at line 527 of file 2.06.09/HepMC/GenEvent.h.

7.52.2.7 HepMC::GenEvent::particle_iterator::particle_iterator (const particle_iterator & *i*) [inline]

copy constructor

Definition at line 529 of file 2.06.09/HepMC/GenEvent.h.

7.52.2.8 virtual HepMC::GenEvent::particle_iterator::~particle_iterator ( ) [inline], [virtual]

Definition at line 530 of file 2.06.09/HepMC/GenEvent.h.

7.52.3 Member Function Documentation

7.52.3.1 HepMC::GenEvent::particle_iterator::operator particle_const_iterator ( ) const [inline]

const particle iterator

Definition at line 537 of file GenEvent.h.

References m_map_iterator, and HepMC::GenEvent::particle_const_iterator.

7.52.3.2 HepMC::GenEvent::particle_iterator::operator particle_const_iterator ( ) const [inline]

const particle iterator

Definition at line 537 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator, and HepMC::GenEvent::particle_const_iterator.

7.52.3.3 bool HepMC::GenEvent::particle_iterator::operator!= (const particle_iterator & *a*) const [inline]

inequality

Definition at line 552 of file GenEvent.h.

References m_map_iterator.

7.52.3.4 bool HepMC::GenEvent::particle_iterator::operator!= (const particle_iterator & *a*) const [inline]

inequality

Definition at line 552 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.52.3.5 GenParticle* HepMC::GenEvent::particle_iterator::operator* (void) const [inline]

return pointer to **GenParticle** (p. 106)

Definition at line 540 of file GenEvent.h.

References m_map_iterator.

7.52.3.6 GenParticle* HepMC::GenEvent::particle_iterator::operator* (void) const [inline]

return pointer to **GenParticle** (p. 106)

Definition at line 540 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.52.3.7 particle_iterator& HepMC::GenEvent::particle_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 543 of file GenEvent.h.

References m_map_iterator.

7.52.3.8 particle_iterator& HepMC::GenEvent::particle_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 543 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.52.3.9 particle_iterator HepMC::GenEvent::particle_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 546 of file GenEvent.h.

7.52.3.10 particle_iterator HepMC::GenEvent::particle_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 546 of file 2.06.09/HepMC/GenEvent.h.

7.52.3.11 particle_iterator& HepMC::GenEvent::particle_iterator::operator= (const particle_iterator & *i*) [inline]

make a copy

Definition at line 532 of file GenEvent.h.

References m_map_iterator.

7.52.3.12 particle_iterator& HepMC::GenEvent::particle_iterator::operator= (const particle_iterator & *i*) [inline]

make a copy

Definition at line 532 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.52.3.13 bool HepMC::GenEvent::particle_iterator::operator== (const particle_iterator & *a*) const [inline]

equality

Definition at line 549 of file GenEvent.h.

References m_map_iterator.

7.52.3.14 bool HepMC::GenEvent::particle_iterator::operator== (const particle_iterator & *a*) const [inline]

equality

Definition at line 549 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.52.4 Member Data Documentation

7.52.4.1 std::map< int, HepMC::GenParticle * >::iterator HepMC::GenEvent::particle_iterator::m_map_iterator [protected]

iterator for **GenParticle** (p. 106) map

Definition at line 556 of file GenEvent.h.

Referenced by operator particle_const_iterator(), operator!=(), operator∗(), operator++(), operator=(), and operator==().

The documentation for this class was generated from the following files:

- **GenEvent.h**
- **2.06.09/HepMC/GenEvent.h**

7.53 HepMC::PdfInfo Class Reference

The **PdfInfo** (p. 233) class stores PDF information.

`#include <PdfInfo.h>`

Public Member Functions

- **PdfInfo** ()
  default constructor
- **PdfInfo** (int i1, int i2, double **x1**, double **x2**, double q, double p1, double p2, int **pdf_id1**=0, int **pdf_id2**=0)
  all values EXCEPT pdf_id1 and pdf_id2 must be provided
- ∼**PdfInfo** ()
- **PdfInfo** (**PdfInfo** const &orig)
  copy constructor
- **PdfInfo** & **operator=** (**PdfInfo** const &rhs)
  make a copy
- void **swap** (**PdfInfo** &other)
  *swap two **PdfInfo** (p. 233) objects*
- bool **operator==** (const **PdfInfo** &) const
  check for equality
- bool **operator!=** (const **PdfInfo** &) const
  check for inequality
- int **id1** () const
  flavour code of first parton
- int **id2** () const
  flavour code of second parton
- int **pdf_id1** () const
  LHAPDF set id of first parton.
- int **pdf_id2** () const
  LHAPDF set id of second parton.
- double **x1** () const
  fraction of beam momentum carried by first parton ("beam side")
- double **x2** () const
  fraction of beam momentum carried by second parton ("target side")
- double **scalePDF** () const
  Q-scale used in evaluation of PDF's (in GeV)
- double **pdf1** () const
  PDF (id1, x1, Q) - x∗f(x)
- double **pdf2** () const
  PDF (id2, x2, Q) - x∗f(x)
- bool **is_valid** () const

verify that the instance contains non-zero information

- void **set_id1** (const int &i)

  set flavour code of first parton
- void **set_id2** (const int &i)

  set flavour code of second parton
- void **set_pdf_id1** (const int &i)

  set LHAPDF set id of first parton
- void **set_pdf_id2** (const int &i)

  set LHAPDF set id of second parton
- void **set_x1** (const double &f)

  set fraction of beam momentum carried by first parton ("beam side")
- void **set_x2** (const double &f)

  set fraction of beam momentum carried by second parton ("target side")
- void **set_scalePDF** (const double &f)

  set Q-scale used in evaluation of PDF's (in GeV)
- void **set_pdf1** (const double &f)

  set x∗f(x) of first parton
- void **set_pdf2** (const double &f)

  set x∗f(x) of second parton
- **PdfInfo** ()

  default constructor
- **PdfInfo** (int i1, int i2, double **x1**, double **x2**, double q, double p1, double p2, int **pdf_id1**=0, int **pdf_id2**=0)

  all values EXCEPT pdf_id1 and pdf_id2 must be provided
- ∼**PdfInfo** ()
- **PdfInfo** (**PdfInfo** const &orig)

  copy constructor
- **PdfInfo** & **operator=** (**PdfInfo** const &rhs)

  make a copy
- void **swap** (**PdfInfo** &other)

  *swap two **PdfInfo** (p. 233) objects*
- bool **operator==** (const **PdfInfo** &) const

  check for equality
- bool **operator!=** (const **PdfInfo** &) const

  check for inequality
- int **id1** () const

  flavour code of first parton
- int **id2** () const

  flavour code of second parton
- int **pdf_id1** () const

  LHAPDF set id of first parton.
- int **pdf_id2** () const

  LHAPDF set id of second parton.
- double **x1** () const

  fraction of beam momentum carried by first parton ("beam side")
- double **x2** () const

  fraction of beam momentum carried by second parton ("target side")
- double **scalePDF** () const

  Q-scale used in evaluation of PDF's (in GeV)
- double **pdf1** () const

  PDF (id1, x1, Q) - x∗f(x)
- double **pdf2** () const

PDF (id2, x2, Q) - x∗f(x)

- bool **is_valid** () const

  verify that the instance contains non-zero information
- void **set_id1** (const int &i)

  set flavour code of first parton
- void **set_id2** (const int &i)

  set flavour code of second parton
- void **set_pdf_id1** (const int &i)

  set LHAPDF set id of first parton
- void **set_pdf_id2** (const int &i)

  set LHAPDF set id of second parton
- void **set_x1** (const double &f)

  set fraction of beam momentum carried by first parton ("beam side")
- void **set_x2** (const double &f)

  set fraction of beam momentum carried by second parton ("target side")
- void **set_scalePDF** (const double &f)

  set Q-scale used in evaluation of PDF's (in GeV)
- void **set_pdf1** (const double &f)

  set x∗f(x) of first parton
- void **set_pdf2** (const double &f)

  set x∗f(x) of second parton

7.53.1 Detailed Description

The **PdfInfo** (p. 233) class stores PDF information.

HepMC::PdfInfo (p. 233) stores additional PDF information for a **GenEvent** (p. 72). Creation and use of this information is optional.

- int id1; // flavour code of first parton
- int id2; // flavour code of second parton
- int pdf_id1; // LHAPDF set id of first parton (zero by default)
- int pdf_id2; // LHAPDF set id of second parton (zero by default)
- double x1; // fraction of beam momentum carried by first parton ("beam side")
- double x2; // fraction of beam momentum carried by second parton ("target side")
- double scalePDF; // Q-scale used in evaluation of PDF's (in GeV)
- double pdf1; // PDF (id1, x1, Q)
- double pdf2; // PDF (id2, x2, Q)

Input parton flavour codes id1 & id2 are expected to obey the PDG code conventions, especially g = 21.

The contents of pdf1 and pdf2 are expected to be x∗f(x). The LHAPDF set ids are the entries in the first column of `http:///projects.hepforge.org/lhapdf/PDFsets.index`

Examples:

testMass.cc.in.

Definition at line 37 of file PdfInfo.h.

7.53.2 Constructor & Destructor Documentation

7.53.2.1 HepMC::PdfInfo::PdfInfo ( ) [inline]

default constructor

Definition at line 43 of file PdfInfo.h.

7.53.2.2 HepMC::PdfInfo::PdfInfo (int *i1*, int *i2*, double *x1*, double *x2*, double *q*, double *p1*, double *p2*, int *pdf_id1* = 0, int *pdf_id2* = 0) [inline]

all values EXCEPT pdf_id1 and pdf_id2 must be provided

Definition at line 136 of file PdfInfo.h.

7.53.2.3 HepMC::PdfInfo::~PdfInfo ( ) [inline]

Definition at line 60 of file PdfInfo.h.

7.53.2.4 HepMC::PdfInfo::PdfInfo (PdfInfo const & *orig*) [inline]

copy constructor

Definition at line 150 of file PdfInfo.h.

7.53.2.5 HepMC::PdfInfo::PdfInfo ( ) [inline]

default constructor

Definition at line 43 of file 2.06.09/HepMC/PdfInfo.h.

7.53.2.6 HepMC::PdfInfo::PdfInfo (int *i1*, int *i2*, double *x1*, double *x2*, double *q*, double *p1*, double *p2*, int *pdf_id1* = 0, int *pdf_id2* = 0)

all values EXCEPT pdf_id1 and pdf_id2 must be provided

7.53.2.7 HepMC::PdfInfo::~PdfInfo ( ) [inline]

Definition at line 60 of file 2.06.09/HepMC/PdfInfo.h.

7.53.2.8 HepMC::PdfInfo::PdfInfo (PdfInfo const & *orig*)

copy constructor

7.53.3 Member Function Documentation

7.53.3.1 int HepMC::PdfInfo::id1 ( ) const [inline]

flavour code of first parton

Definition at line 75 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.2 int HepMC::PdfInfo::id1 ( ) const [inline]

flavour code of first parton

Definition at line 75 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.3 int HepMC::PdfInfo::id2 ( ) const [inline]

flavour code of second parton

Definition at line 77 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.4 int HepMC::PdfInfo::id2 ( ) const [inline]

flavour code of second parton

Definition at line 77 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.5 bool HepMC::PdfInfo::is_valid ( ) const

verify that the instance contains non-zero information

7.53.3.6 bool HepMC::PdfInfo::is_valid ( ) const [inline]

verify that the instance contains non-zero information

Definition at line 202 of file PdfInfo.h.

Referenced by HepMC::GenEvent::read().

7.53.3.7 bool HepMC::PdfInfo::operator!= (const PdfInfo & *a*) const [inline]

check for inequality

any nonmatching member generates inequality

any nonmatching member generates inequality

Definition at line 196 of file PdfInfo.h.

7.53.3.8 bool HepMC::PdfInfo::operator!= (const PdfInfo &) const

check for inequality

7.53.3.9 PdfInfo & HepMC::PdfInfo::operator= (PdfInfo const & *rhs*) [inline]

make a copy

Definition at line 162 of file PdfInfo.h.

References swap().

7.53.3.10 PdfInfo& HepMC::PdfInfo::operator= (PdfInfo const & *rhs*)

make a copy

7.53.3.11 bool HepMC::PdfInfo::operator== (const PdfInfo & *a*) const [inline]

check for equality

equality requires that each member match

equality requires that each member match

Definition at line 182 of file PdfInfo.h.

References id1(), id2(), pdf1(), pdf2(), pdf_id1(), pdf_id2(), scalePDF(), x1(), and x2().

7.53.3.12 bool HepMC::PdfInfo::operator== (const PdfInfo &) const

check for equality

7.53.3.13 double HepMC::PdfInfo::pdf1 ( ) const [inline]

PDF (id1, x1, Q) - x∗f(x)

Definition at line 89 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.14 double HepMC::PdfInfo::pdf1 ( ) const [inline]

PDF (id1, x1, Q) - x∗f(x)

Definition at line 89 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.15 double HepMC::PdfInfo::pdf2 ( ) const [inline]

PDF (id2, x2, Q) - x∗f(x)

Definition at line 91 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.16 double HepMC::PdfInfo::pdf2 ( ) const [inline]

PDF (id2, x2, Q) - x∗f(x)

Definition at line 91 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.17 int HepMC::PdfInfo::pdf_id1 ( ) const [inline]

LHAPDF set id of first parton.

Definition at line 79 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.18 int HepMC::PdfInfo::pdf_id1 ( ) const [inline]

LHAPDF set id of first parton.

Definition at line 79 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.19 int HepMC::PdfInfo::pdf_id2 ( ) const [inline]

LHAPDF set id of second parton.

Definition at line 81 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.20 int HepMC::PdfInfo::pdf_id2 ( ) const [inline]

LHAPDF set id of second parton.

Definition at line 81 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.21 double HepMC::PdfInfo::scalePDF ( ) const [inline]

Q-scale used in evaluation of PDF's (in GeV)

Definition at line 87 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.22 double HepMC::PdfInfo::scalePDF ( ) const [inline]

Q-scale used in evaluation of PDF's (in GeV)

Definition at line 87 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.23 void HepMC::PdfInfo::set_id1 (const int & *i*) [inline]

set flavour code of first parton

Definition at line 98 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.24 void HepMC::PdfInfo::set_id1 (const int & *i*) [inline]

set flavour code of first parton

Definition at line 98 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.25 void HepMC::PdfInfo::set_id2 (const int & *i*) [inline]

set flavour code of second parton

Definition at line 100 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.26 void HepMC::PdfInfo::set_id2 (const int & *i*) [inline]

set flavour code of second parton

Definition at line 100 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.27 void HepMC::PdfInfo::set_pdf1 (const double & *f*) [inline]

set x∗f(x) of first parton

Definition at line 112 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.28 void HepMC::PdfInfo::set_pdf1 (const double & *f*) [inline]

set x∗f(x) of first parton

Definition at line 112 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.29 void HepMC::PdfInfo::set_pdf2 (const double & *f*) [inline]

set x∗f(x) of second parton

Definition at line 114 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.30 void HepMC::PdfInfo::set_pdf2 (const double & *f*) [inline]

set x∗f(x) of second parton

Definition at line 114 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.31 void HepMC::PdfInfo::set_pdf_id1 (const int & *i*) [inline]

set LHAPDF set id of first parton

Definition at line 102 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.32 void HepMC::PdfInfo::set_pdf_id1 (const int & *i*) [inline]

set LHAPDF set id of first parton

Definition at line 102 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.33 void HepMC::PdfInfo::set_pdf_id2 (const int & *i*) [inline]

set LHAPDF set id of second parton

Definition at line 104 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.34 void HepMC::PdfInfo::set_pdf_id2 (const int & *i*) [inline]

set LHAPDF set id of second parton

Definition at line 104 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.35 void HepMC::PdfInfo::set_scalePDF (const double & *f*) [inline]

set Q-scale used in evaluation of PDF's (in GeV)

Definition at line 110 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.36 void HepMC::PdfInfo::set_scalePDF (const double & *f*) [inline]

set Q-scale used in evaluation of PDF's (in GeV)

Definition at line 110 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.37 void HepMC::PdfInfo::set_x1 (const double & *f*) [inline]

set fraction of beam momentum carried by first parton ("beam side")

Definition at line 106 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.38 void HepMC::PdfInfo::set_x1 (const double & *f*) [inline]

set fraction of beam momentum carried by first parton ("beam side")

Definition at line 106 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.39 void HepMC::PdfInfo::set_x2 (const double & *f*) [inline]

set fraction of beam momentum carried by second parton ("target side")

Definition at line 108 of file PdfInfo.h.

Referenced by HepMC::operator>>().

7.53.3.40 void HepMC::PdfInfo::set_x2 (const double & *f*) [inline]

set fraction of beam momentum carried by second parton ("target side")

Definition at line 108 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.41 void HepMC::PdfInfo::swap (PdfInfo & *other*) [inline]

swap two **PdfInfo** (p. 233) objects

Definition at line 169 of file PdfInfo.h.

Referenced by operator=().

7.53.3.42 void HepMC::PdfInfo::swap (PdfInfo & *other*)

swap two **PdfInfo** (p. 233) objects

7.53.3.43 double HepMC::PdfInfo::x1 ( ) const [inline]

fraction of beam momentum carried by first parton ("beam side")

Definition at line 83 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

7.53.3.44 double HepMC::PdfInfo::x1 ( ) const [inline]

fraction of beam momentum carried by first parton ("beam side")

Definition at line 83 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.45 double HepMC::PdfInfo::x2 ( ) const [inline]

fraction of beam momentum carried by second parton ("target side")

Definition at line 85 of file 2.06.09/HepMC/PdfInfo.h.

7.53.3.46 double HepMC::PdfInfo::x2 ( ) const [inline]

fraction of beam momentum carried by second parton ("target side")

Definition at line 85 of file PdfInfo.h.

Referenced by HepMC::operator<<(), and operator==().

The documentation for this class was generated from the following files:

- **PdfInfo.h**
- **2.06.09/HepMC/PdfInfo.h**

7.54 pin3 Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- double **xsfx** [81][2]
- int **isig** [3][1000]
- double **sigh** [1000]

7.54.1 Detailed Description

Definition at line 115 of file PythiaWrapper6_4.h.

7.54.2 Member Data Documentation

7.54.2.1 int pin3::isig

Definition at line 117 of file PythiaWrapper6_4.h.

7.54.2.2 double pin3::sigh

Definition at line 118 of file PythiaWrapper6_4.h.

7.54.2.3 double pin3::xsfx

Definition at line 116 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.55 pin5 Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- int **ngenpd**
- int **ngen** [3][501]
- double **xsec** [3][501]

7.55.1 Detailed Description

Definition at line 132 of file PythiaWrapper6_4.h.

7.55.2 Member Data Documentation

7.55.2.1 int pin5::ngen

Definition at line 133 of file PythiaWrapper6_4.h.

7.55.2.2 int pin5::ngenpd

Definition at line 133 of file PythiaWrapper6_4.h.

7.55.2.3 double pin5::xsec

Definition at line 134 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.56 pin7 Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- double **sigt** [6][7][7]

7.56.1 Detailed Description

Definition at line 140 of file PythiaWrapper6_4.h.

7.56.2 Member Data Documentation

7.56.2.1 double pin7::sigt

Definition at line 141 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.57 pin8 Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- double **xpvmd** [13]
- double **xpanl** [13]
- double **xpanh** [13]
- double **xpbeh** [13]
- double **xpdir** [13]

7.57.1 Detailed Description

Definition at line 147 of file PythiaWrapper6_4.h.

7.57.2 Member Data Documentation

7.57.2.1 double pin8::xpanh

Definition at line 150 of file PythiaWrapper6_4.h.

7.57.2.2 double pin8::xpanl

Definition at line 149 of file PythiaWrapper6_4.h.

7.57.2.3 double pin8::xpbeh

Definition at line 151 of file PythiaWrapper6_4.h.

7.57.2.4 double pin8::xpdir

Definition at line 152 of file PythiaWrapper6_4.h.

7.57.2.5 double pin8::xpvmd

Definition at line 148 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.58 pin9 Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- double **vxpvmd** [13]
- double **vxpanl** [13]
- double **vxpanh** [13]
- double **vxpdgm** [13]

7.58.1 Detailed Description

Definition at line 158 of file PythiaWrapper6_4.h.

7.58.2 Member Data Documentation

7.58.2.1 double pin9::vxpanh

Definition at line 161 of file PythiaWrapper6_4.h.

7.58.2.2 double pin9::vxpanl

Definition at line 160 of file PythiaWrapper6_4.h.

7.58.2.3 double pin9::vxpdgm

Definition at line 162 of file PythiaWrapper6_4.h.

7.58.2.4 double pin9::vxpvmd

Definition at line 159 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.59 HepMC::Polarization Class Reference

The **Polarization** (p. 245) class stores theta and phi for a **GenParticle** (p. 106).

`#include <Polarization.h>`

Public Member Functions

- **Polarization** ()
  default constructor
- **Polarization** (double **theta**, double **phi**=0)
  constructor requiring at least one value
- **Polarization** (const **Polarization** &inpolar)
  construct from another polarization object
- **Polarization** (const **ThreeVector** &vec3in)
  *construct using the polar and azimuthal angles from a **ThreeVector** (p. 271)*
- virtual ~**Polarization** ()
- void **swap** (**Polarization** &other)
  swap
- **Polarization** & **operator=** (const **Polarization** &inpolar)
  make a copy
- bool **operator==** (const **Polarization** &) const
  equality requires that theta and phi are equal
- bool **operator!=** (const **Polarization** &) const
  inequality results if either theta or phi differ
- void **print** (std::ostream &ostr=std::cout) const
  print theta and phi
- double **theta** () const
  returns polar angle in radians
- double **phi** () const
  returns azimuthal angle in radians
- **ThreeVector normal3d** () const
  unit 3 vector for easy manipulation
- bool **is_defined** () const
  *returns true if the **Polarization** (p. 245) has been defined*
- double **set_theta** (double **theta**)
  set polar angle in radians
- double **set_phi** (double **phi**)
  set azimuthal angle in radians
- void **set_theta_phi** (double **theta**, double **phi**)
  set both polar and azimuthal angles in radians
- **ThreeVector set_normal3d** (const **ThreeVector** &vec3in)
  sets polarization according to direction of 3 vec
- void **set_undefined** ()
  *declares the **Polarization** (p. 245) as undefined and zeros the values*
- **Polarization** ()
  default constructor
- **Polarization** (double **theta**, double **phi**=0)
  constructor requiring at least one value
- **Polarization** (const **Polarization** &inpolar)
  construct from another polarization object
- **Polarization** (const **ThreeVector** &vec3in)
  *construct using the polar and azimuthal angles from a **ThreeVector** (p. 271)*
- virtual ~**Polarization** ()
- void **swap** (**Polarization** &other)
  swap
- **Polarization** & **operator=** (const **Polarization** &inpolar)

make a copy

- bool **operator==** (const **Polarization** &) const

  equality requires that theta and phi are equal

- bool **operator!=** (const **Polarization** &) const

  inequality results if either theta or phi differ

- void **print** (std::ostream &ostr=std::cout) const

  print theta and phi

- double **theta** () const

  returns polar angle in radians

- double **phi** () const

  returns azimuthal angle in radians

- **ThreeVector normal3d** () const

  unit 3 vector for easy manipulation

- bool **is_defined** () const

  *returns true if the **Polarization** (p. 245) has been defined*

- double **set_theta** (double **theta**)

  set polar angle in radians

- double **set_phi** (double **phi**)

  set azimuthal angle in radians

- void **set_theta_phi** (double **theta**, double **phi**)

  set both polar and azimuthal angles in radians

- **ThreeVector set_normal3d** (const **ThreeVector** &vec3in)

  sets polarization according to direction of 3 vec

- void **set_undefined** ()

  *declares the **Polarization** (p. 245) as undefined and zeros the values*

Friends

- std::ostream & **operator**<< (std::ostream &, const **Polarization** &)

  print polarization information

- std::ostream & **operator**<< (std::ostream &, const **Polarization** &)

  print polarization information

7.59.1 Detailed Description

The **Polarization** (p. 245) class stores theta and phi for a **GenParticle** (p. 106).

HepMC::Polarization (p. 245) stores a particle's theta and phi in radians. Use of this information is optional. By default, the polarization is set to zero.

Definition at line 29 of file Polarization.h.

7.59.2 Constructor & Destructor Documentation

7.59.2.1 HepMC::Polarization::Polarization ( )

default constructor

Definition at line 11 of file HepMC-2.06.09/src/Polarization.cc.

7.59.2.2 HepMC::Polarization::Polarization (double *theta,* double *phi* = 0)

constructor requiring at least one value

Definition at line 17 of file HepMC-2.06.09/src/Polarization.cc.

7.59.2.3 HepMC::Polarization::Polarization (const **Polarization** & *inpolar*)

construct from another polarization object

Definition at line 23 of file HepMC-2.06.09/src/Polarization.cc.

7.59.2.4 HepMC::Polarization::Polarization (const **ThreeVector** & *vec3in*)

construct using the polar and azimuthal angles from a **ThreeVector** (p. 271)

Definition at line 29 of file HepMC-2.06.09/src/Polarization.cc.

7.59.2.5 virtual HepMC::Polarization::~Polarization ( ) [inline], [virtual]

Definition at line 43 of file Polarization.h.

7.59.2.6 HepMC::Polarization::Polarization ( )

default constructor

7.59.2.7 HepMC::Polarization::Polarization (double *theta,* double *phi* = 0)

constructor requiring at least one value

7.59.2.8 HepMC::Polarization::Polarization (const **Polarization** & *inpolar*)

construct from another polarization object

7.59.2.9 HepMC::Polarization::Polarization (const **ThreeVector** & *vec3in*)

construct using the polar and azimuthal angles from a **ThreeVector** (p. 271)

7.59.2.10 virtual HepMC::Polarization::~Polarization ( ) [inline], [virtual]

Definition at line 43 of file 2.06.09/HepMC/Polarization.h.

7.59.3 Member Function Documentation

7.59.3.1 bool HepMC::Polarization::is_defined ( ) const

returns true if the **Polarization** (p. 245) has been defined

Definition at line 77 of file HepMC-2.06.09/src/Polarization.cc.

Referenced by operator==().

7.59.3.2 bool HepMC::Polarization::is_defined ( ) const

returns true if the **Polarization** (p. 245) has been defined

7.59.3.3 ThreeVector HepMC::Polarization::normal3d ( ) const

unit 3 vector for easy manipulation

Definition at line 57 of file HepMC-2.06.09/src/Polarization.cc.

References phi(), HepMC::ThreeVector::setPhi(), HepMC::ThreeVector::setTheta(), and theta().

7.59.3.4 ThreeVector HepMC::Polarization::normal3d ( ) const

unit 3 vector for easy manipulation

7.59.3.5 bool HepMC::Polarization::operator!= (const Polarization & *a*) const [inline]

inequality results if either theta or phi differ

Definition at line 104 of file Polarization.h.

7.59.3.6 bool HepMC::Polarization::operator!= (const Polarization &) const

inequality results if either theta or phi differ

7.59.3.7 Polarization& HepMC::Polarization::operator= (const Polarization & *inpolar*)

make a copy

7.59.3.8 Polarization & HepMC::Polarization::operator= (const Polarization & *inpolar*)

make a copy

best practices implementation

best practices implementation

Definition at line 42 of file HepMC-2.06.09/src/Polarization.cc.

References swap().

7.59.3.9 bool HepMC::Polarization::operator== (const Polarization &) const

equality requires that theta and phi are equal

7.59.3.10 bool HepMC::Polarization::operator== (const Polarization & *a*) const [inline]

equality requires that theta and phi are equal

Definition at line 99 of file Polarization.h.

References is_defined(), phi(), and theta().

7.59.3.11 double HepMC::Polarization::phi ( ) const

returns azimuthal angle in radians

7.59.3.12 double HepMC::Polarization::phi ( ) const [inline]

returns azimuthal angle in radians

Definition at line 93 of file Polarization.h.

Referenced by normal3d(), HepMC::operator<<(), and operator==().

7.59.3.13 void HepMC::Polarization::print (std::ostream & *ostr* = std::cout) const

print theta and phi

7.59.3.14 void HepMC::Polarization::print (std::ostream & *ostr* = std::cout) const

print theta and phi

Definition at line 49 of file HepMC-2.06.09/src/Polarization.cc.

7.59.3.15 ThreeVector HepMC::Polarization::set_normal3d (const ThreeVector & *vec3in*)

sets polarization according to direction of 3 vec

Definition at line 93 of file HepMC-2.06.09/src/Polarization.cc.

References HepMC::ThreeVector::phi(), set_phi(), set_theta(), and HepMC::ThreeVector::theta().

7.59.3.16 ThreeVector HepMC::Polarization::set_normal3d (const ThreeVector & *vec3in*)

sets polarization according to direction of 3 vec

7.59.3.17 double HepMC::Polarization::set_phi (double *phi*)

set azimuthal angle in radians

Phi is restricted to be between 0 –> 2pi if an out of range value is given, it is translated to this range.

Phi is restricted to be between 0 –> 2pi if an out of range value is given, it is translated to this range.

Definition at line 71 of file HepMC-2.06.09/src/Polarization.cc.

Referenced by set_normal3d(), and set_theta_phi().

7.59.3.18 double HepMC::Polarization::set_phi (double *phi*)

set azimuthal angle in radians

7.59.3.19 double HepMC::Polarization::set_theta (double *theta*)

set polar angle in radians

7.59.3.20 double HepMC::Polarization::set_theta (double *theta*)

set polar angle in radians

Theta is restricted to be between 0 –> pi if an out of range value is given, it is translated to this range.

Theta is restricted to be between 0 –> pi if an out of range value is given, it is translated to this range.

Definition at line 65 of file HepMC-2.06.09/src/Polarization.cc.

Referenced by set_normal3d(), and set_theta_phi().

7.59.3.21 void HepMC::Polarization::set_theta_phi (double *theta,* double *phi*)

set both polar and azimuthal angles in radians

Definition at line 87 of file HepMC-2.06.09/src/Polarization.cc.

References set_phi(), and set_theta().

7.59.3.22 void HepMC::Polarization::set_theta_phi (double *theta,* double *phi*)

set both polar and azimuthal angles in radians

7.59.3.23 void HepMC::Polarization::set_undefined ( )

declares the **Polarization** (p. 245) as undefined and zeros the values

7.59.3.24 void HepMC::Polarization::set_undefined ( )

declares the **Polarization** (p. 245) as undefined and zeros the values

Definition at line 81 of file HepMC-2.06.09/src/Polarization.cc.

7.59.3.25 void HepMC::Polarization::swap (**Polarization** & *other*)

swap

Definition at line 35 of file HepMC-2.06.09/src/Polarization.cc.

Referenced by operator=(), and HepMC::GenParticle::swap().

7.59.3.26 void HepMC::Polarization::swap (**Polarization** & *other*)

swap

7.59.3.27 double HepMC::Polarization::theta ( ) const

returns polar angle in radians

7.59.3.28 double HepMC::Polarization::theta ( ) const [inline]

returns polar angle in radians

Definition at line 92 of file Polarization.h.

Referenced by normal3d(), HepMC::operator<<(), and operator==().

7.59.4 Friends And Related Function Documentation

7.59.4.1 std::ostream& operator<< (std::ostream & *ostr,* const Polarization & *polar*) [friend]

print polarization information

Definition at line 129 of file HepMC-2.06.09/src/Polarization.cc.

7.59.4.2 std::ostream& operator<< (std::ostream & *ostr,* const Polarization & *polar*) [friend]

print polarization information

print polarization information

Definition at line 129 of file HepMC-2.06.09/src/Polarization.cc.

The documentation for this class was generated from the following files:

- **Polarization.h**
- **2.06.09/HepMC/Polarization.h**
- **HepMC-2.06.09/src/Polarization.cc**
- **src/Polarization.cc**

7.60 PrintChildren Class Reference

test class

`#include <testHepMCIteration.h>`

Public Member Functions

- **PrintChildren** (std::ostream &os)
- void **operator()** (**HepMC::GenParticle** ∗**p**)

7.60.1 Detailed Description

test class

prints the particle

Examples:

testHepMCIteration.cc.in.

Definition at line 62 of file testHepMCIteration.h.

7.60.2 Constructor & Destructor Documentation

7.60.2.1 PrintChildren::PrintChildren (std::ostream & *os*) [inline]

Definition at line 64 of file testHepMCIteration.h.

7.60.3 Member Function Documentation

7.60.3.1 void PrintChildren::operator() (HepMC::GenParticle * p) [inline]

Definition at line 65 of file testHepMCIteration.h.

References HepMC::GenParticle::barcode(), p, HepMC::GenParticle::pdg_id(), and HepMC::GenParticle::status().

The documentation for this class was generated from the following file:

- **testHepMCIteration.h**

7.61 PrintConstW Class Reference

Public Member Functions

- **PrintConstW** (std::ostream &os, int num)
- void **operator()** (**HepMC::GenParticle** ∗**p**)
- **PrintConstW** (std::ostream &os, int num)
- void **operator()** (**HepMC::GenParticle** ∗**p**)

7.61.1 Detailed Description

This class is designed for use with std::for_each This class also illustrates the use of the new GenEventIterator classes. **PrintConstW** (p. 253) duplicates the funtionality within findW's particle loop.

Examples:

testHepMCIteration.cc.in.

Definition at line 79 of file HepMC-2.06.09/test/testHepMCIteration.cc.

7.61.2 Constructor & Destructor Documentation

7.61.2.1 PrintConstW::PrintConstW (std::ostream & *os,* int *num*) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 81 of file HepMC-2.06.09/test/testHepMCIteration.cc.

7.61.2.2 PrintConstW::PrintConstW (std::ostream & *os,* int *num*) [inline]

Definition at line 81 of file test/testHepMCIteration.cc.

7.61.3 Member Function Documentation

7.61.3.1 void PrintConstW::operator() (HepMC::GenParticle * p) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 82 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::GenParticleProductionRange::begin(), HepMC::GenParticleEndRange::begin(), HepMC::children, HepMC::descendants, HepMC::GenParticleProductionRange::end(), HepMC::GenParticleEndRange::end(), HepMC::GenParticle::end_vertex(), IsWBoson(), HepMC::parents, HepMC::GenParticle::particles_in(), HepMC::GenParticle::particles_out(), HepMC::GenParticle::print(), and HepMC::GenParticle::production_vertex().

7.61.3.2 void PrintConstW::operator() (HepMC::GenParticle * p) [inline]

Definition at line 82 of file test/testHepMCIteration.cc.

References HepMC::GenParticleProductionRange::begin(), HepMC::GenParticleEndRange::begin(), HepMC::children, HepMC::descendants, HepMC::GenParticleProductionRange::end(), HepMC::GenParticleEndRange::end(), HepMC::GenParticle::end_vertex(), IsWBoson(), HepMC::parents, HepMC::GenParticle::particles_in(), HepMC::GenParticle::particles_out(), HepMC::GenParticle::print(), and HepMC::GenParticle::production_vertex().

The documentation for this class was generated from the following files:

- **HepMC-2.06.09/test/testHepMCIteration.cc**
- **test/testHepMCIteration.cc**

7.62 PrintDescendants Class Reference

test class

`#include <testHepMCIteration.h>`

Public Member Functions

- **PrintDescendants** (std::ostream &os)
- void **operator()** (const **HepMC::GenParticle** *p)

7.62.1 Detailed Description

test class

prints the particle

Examples:

testHepMCIteration.cc.in.

Definition at line 82 of file testHepMCIteration.h.

7.62.2 Constructor & Destructor Documentation

7.62.2.1 PrintDescendants::PrintDescendants (std::ostream & os) [inline]

Definition at line 84 of file testHepMCIteration.h.

7.62.3 Member Function Documentation

7.62.3.1 void PrintDescendants::operator() (const HepMC::GenParticle * p) [inline]

Definition at line 85 of file testHepMCIteration.h.

References HepMC::GenParticle::print().

The documentation for this class was generated from the following file:

- **testHepMCIteration.h**

7.63 PrintParticle Class Reference

`#include <testHepMCIteration.h>`

Public Member Functions

- **PrintParticle** (std::ostream &os)
- void **operator()** (const **HepMC::GenParticle** ∗**p**)

7.63.1 Detailed Description

prints the particle

Examples:

testHepMCIteration.cc.in.

Definition at line 47 of file testHepMCIteration.h.

7.63.2 Constructor & Destructor Documentation

7.63.2.1 PrintParticle::PrintParticle (std::ostream & *os*) [inline]

Definition at line 49 of file testHepMCIteration.h.

7.63.3 Member Function Documentation

7.63.3.1 void PrintParticle::operator() (const HepMC::GenParticle ∗ *p*) [inline]

Definition at line 50 of file testHepMCIteration.h.

References HepMC::GenParticle::print().

The documentation for this class was generated from the following file:

- **testHepMCIteration.h**

7.64 PrintPhoton Class Reference

`#include <testHepMCIteration.h>`

Public Member Functions

- **PrintPhoton** (std::ostream &os)
- void **operator()** (const **HepMC::GenParticle** ∗**p**)

7.64.1 Detailed Description

prints the particle if it is a photon

Examples:

testHepMCIteration.cc.in.

Definition at line 35 of file testHepMCIteration.h.

7.64.2 Constructor & Destructor Documentation

7.64.2.1 PrintPhoton::PrintPhoton (std::ostream & *os*) [inline]

Definition at line 37 of file testHepMCIteration.h.

7.64.3 Member Function Documentation

7.64.3.1 void PrintPhoton::operator() (const HepMC::GenParticle * *p*) [inline]

Definition at line 38 of file testHepMCIteration.h.

References IsPhoton(), and HepMC::GenParticle::print().

The documentation for this class was generated from the following file:

- **testHepMCIteration.h**

7.65 PrintW Class Reference

Public Member Functions

- **PrintW** (std::ostream &os, int num)
- void **operator()** (**HepMC::GenParticle** ∗**p**)
- **PrintW** (std::ostream &os, int num)
- void **operator()** (**HepMC::GenParticle** ∗**p**)

7.65.1 Detailed Description

This class is designed for use with std::for_each This class also illustrates the use of the new GenEventIterator classes. **PrintW** (p. 256) duplicates the funtionality within findW's particle loop.

Examples:

testHepMCIteration.cc.in.

Definition at line 33 of file HepMC-2.06.09/test/testHepMCIteration.cc.

7.65.2 Constructor & Destructor Documentation

7.65.2.1 PrintW::PrintW (std::ostream & *os,* int *num*) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 35 of file HepMC-2.06.09/test/testHepMCIteration.cc.

7.65.2.2 PrintW::PrintW (std::ostream & *os,* int *num*) [inline]

Definition at line 35 of file test/testHepMCIteration.cc.

7.65.3 Member Function Documentation

7.65.3.1 void PrintW::operator() (HepMC::GenParticle * *p*) [inline]

Examples:

testHepMCIteration.cc.in.

Definition at line 36 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::GenParticleProductionRange::begin(), HepMC::GenParticleEndRange::begin(), HepMC::children, HepMC::descendants, HepMC::GenParticleProductionRange::end(), HepMC::GenParticleEndRange::end(), HepMC::GenParticle::end_vertex(), IsWBoson(), HepMC::parents, HepMC::GenParticle::particles_in(), HepMC::GenParticle::particles_out(), HepMC::GenParticle::print(), and HepMC::GenParticle::production_vertex().

7.65.3.2 void PrintW::operator() (HepMC::GenParticle * *p*) [inline]

Definition at line 36 of file test/testHepMCIteration.cc.

References HepMC::GenParticleProductionRange::begin(), HepMC::GenParticleEndRange::begin(), HepMC::children, HepMC::descendants, HepMC::GenParticleProductionRange::end(), HepMC::GenParticleEndRange::end(), HepMC::GenParticle::end_vertex(), IsWBoson(), HepMC::parents, HepMC::GenParticle::particles_in(), HepMC::GenParticle::particles_out(), HepMC::GenParticle::print(), and HepMC::GenParticle::production_vertex().

The documentation for this class was generated from the following files:

- **HepMC-2.06.09/test/testHepMCIteration.cc**
- **test/testHepMCIteration.cc**

7.66 prvnv Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- double **ab** [2][16][2]
- double **rms** [4]
- double **res** [5][6]
- int **idr**
- int **idr2**
- double **dcmass**
- int **kfr** [3]

7.66.1 Detailed Description

Definition at line 200 of file PythiaWrapper6_4.h.

7.66.2 Member Data Documentation

7.66.2.1 double prvnv::ab

Definition at line 201 of file PythiaWrapper6_4.h.

7.66.2.2 double prvnv::dcmass

Definition at line 206 of file PythiaWrapper6_4.h.

7.66.2.3 int prvnv::idr

Definition at line 204 of file PythiaWrapper6_4.h.

7.66.2.4 int prvnv::idr2

Definition at line 205 of file PythiaWrapper6_4.h.

7.66.2.5 int prvnv::kfr

Definition at line 207 of file PythiaWrapper6_4.h.

7.66.2.6 double prvnv::res

Definition at line 203 of file PythiaWrapper6_4.h.

7.66.2.7 double prvnv::rms

Definition at line 202 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.67 prvpm Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- double **rm** [4]
- double **a** [2]
- double **b** [2]
- double **resm** [2]
- double **resw** [2]
- bool **mflag**

7.67.1 Detailed Description

Definition at line 213 of file PythiaWrapper6_4.h.

7.67.2 Member Data Documentation

7.67.2.1 double prvpm::a

Definition at line 215 of file PythiaWrapper6_4.h.

7.67.2.2 double prvpm::b

Definition at line 216 of file PythiaWrapper6_4.h.

7.67.2.3 bool prvpm::mflag

Definition at line 219 of file PythiaWrapper6_4.h.

7.67.2.4 double prvpm::resm

Definition at line 217 of file PythiaWrapper6_4.h.

7.67.2.5 double prvpm::resw

Definition at line 218 of file PythiaWrapper6_4.h.

7.67.2.6 double prvpm::rm

Definition at line 214 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.68 pssm Struct Reference

`#include <PythiaWrapper6_4.h>`

Public Attributes

- int **imss** [100]
- double **rmss** [100]

7.68.1 Detailed Description

Definition at line 168 of file PythiaWrapper6_4.h.

7.68.2 Member Data Documentation

7.68.2.1 int pssm::imss

Definition at line 169 of file PythiaWrapper6_4.h.

7.68.2.2 double pssm::rmss

Definition at line 170 of file PythiaWrapper6_4.h.

The documentation for this struct was generated from the following files:

- **PythiaWrapper6_4.h**
- **2.06.09/HepMC/PythiaWrapper6_4.h**

7.69 HepMC::StreamInfo Class Reference

StreamInfo (p. 260) contains extra information needed when using streaming IO.

`#include <StreamInfo.h>`

Public Member Functions

- **StreamInfo** ()
  default constructor
- ~**StreamInfo** ()
  destructor
- std::string **IO_GenEvent_Key** () const
  IO_GenEvent *(p. 186) begin event block key.*
- std::string **IO_GenEvent_End** () const
  IO_GenEvent *(p. 186) end event block key.*
- std::string **IO_Ascii_Key** () const
- std::string **IO_Ascii_End** () const
  IO_Ascii end event block key.
- std::string **IO_Ascii_PDT_Key** () const
  IO_Ascii begin particle data block key.
- std::string **IO_Ascii_PDT_End** () const
  IO_Ascii end particle data block key.
- std::string **IO_ExtendedAscii_Key** () const
- std::string **IO_ExtendedAscii_End** () const
  IO_ExtendedAscii end event block key.
- std::string **IO_ExtendedAscii_PDT_Key** () const
  IO_ExtendedAscii begin particle data block key.
- std::string **IO_ExtendedAscii_PDT_End** () const
  IO_ExtendedAscii end particle data block key.
- int **io_type** () const
  get IO type
- void **set_io_type** (int)
  set IO type
- bool **has_key** () const
- void **set_has_key** (bool)
  set to false if the stream does not have a file type key
- **Units::MomentumUnit io_momentum_unit** () const
  get the I/O momentum units
- **Units::LengthUnit io_position_unit** () const
  get the I/O length units
- int **stream_id** () const
- bool **finished_first_event** () const

Special information is processed the first time we use the IO.

- void **set_finished_first_event** (bool b)

  Special information is processed the first time we use the IO.
- void **use_input_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- bool **reading_event_header** ()
- void **set_reading_event_header** (bool)

  set the reading_event_header flag
- **StreamInfo** ()

  default constructor
- ∼**StreamInfo** ()

  destructor
- std::string **IO_GenEvent_Key** () const

  IO_GenEvent *(*p. *186) begin event block key.*
- std::string **IO_GenEvent_End** () const

  IO_GenEvent *(*p. *186) end event block key.*
- std::string **IO_Ascii_Key** () const
- std::string **IO_Ascii_End** () const

  IO_Ascii end event block key.
- std::string **IO_Ascii_PDT_Key** () const

  IO_Ascii begin particle data block key.
- std::string **IO_Ascii_PDT_End** () const

  IO_Ascii end particle data block key.
- std::string **IO_ExtendedAscii_Key** () const
- std::string **IO_ExtendedAscii_End** () const

  IO_ExtendedAscii end event block key.
- std::string **IO_ExtendedAscii_PDT_Key** () const

  IO_ExtendedAscii begin particle data block key.
- std::string **IO_ExtendedAscii_PDT_End** () const

  IO_ExtendedAscii end particle data block key.
- int **io_type** () const

  get IO type
- void **set_io_type** (int)

  set IO type
- bool **has_key** () const
- void **set_has_key** (bool)

  set to false if the stream does not have a file type key
- **Units::MomentumUnit io_momentum_unit** () const

  get the I/O momentum units
- **Units::LengthUnit io_position_unit** () const

  get the I/O length units
- int **stream_id** () const
- bool **finished_first_event** () const

  Special information is processed the first time we use the IO.
- void **set_finished_first_event** (bool b)

  Special information is processed the first time we use the IO.
- void **use_input_units** (**Units::MomentumUnit**, **Units::LengthUnit**)
- bool **reading_event_header** ()
- void **set_reading_event_header** (bool)

  set the reading_event_header flag

7.69.1 Detailed Description

StreamInfo (p. 260) contains extra information needed when using streaming IO.

This class contains the extra information needed when using streaming IO to process **HepMC** (p. 15) GenEvents

Definition at line 26 of file StreamInfo.h.

7.69.2 Constructor & Destructor Documentation

7.69.2.1 HepMC::StreamInfo::StreamInfo ( )

default constructor

Definition at line 13 of file HepMC-2.06.09/src/StreamInfo.cc.

7.69.2.2 HepMC::StreamInfo::~StreamInfo ( ) [inline]

destructor

Definition at line 31 of file StreamInfo.h.

7.69.2.3 HepMC::StreamInfo::StreamInfo ( )

default constructor

7.69.2.4 HepMC::StreamInfo::~StreamInfo ( ) [inline]

destructor

Definition at line 31 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3 Member Function Documentation

7.69.3.1 bool HepMC::StreamInfo::finished_first_event ( ) const [inline]

Special information is processed the first time we use the IO.

Definition at line 81 of file StreamInfo.h.

Referenced by HepMC::detail::establish_input_stream_info(), HepMC::establish_input_stream_info(), HepMC::detail::establish_output_stream_info(), HepMC::establish_output_stream_info(), HepMC::GenEvent::read(), HepMC::GenEvent::write(), HepMC::write_HepMC_IO_block_begin(), and HepMC::write_HepMC_IO_block_end().

7.69.3.2 bool HepMC::StreamInfo::finished_first_event ( ) const [inline]

Special information is processed the first time we use the IO.

Definition at line 81 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.3 bool HepMC::StreamInfo::has_key ( ) const [inline]

true if the stream has a file type key has_key is true by default

Definition at line 67 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.4 bool HepMC::StreamInfo::has_key ( ) const [inline]

true if the stream has a file type key has_key is true by default

Definition at line 67 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read().

7.69.3.5 std::string HepMC::StreamInfo::IO_Ascii_End ( ) const [inline]

IO_Ascii end event block key.

Definition at line 43 of file StreamInfo.h.

7.69.3.6 std::string HepMC::StreamInfo::IO_Ascii_End ( ) const [inline]

IO_Ascii end event block key.

Definition at line 43 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.7 std::string HepMC::StreamInfo::IO_Ascii_Key ( ) const [inline]

IO_Ascii begin event block key IO_Ascii has been removed, but we want to be able to read existing files written by IO_Ascii

Definition at line 41 of file StreamInfo.h.

7.69.3.8 std::string HepMC::StreamInfo::IO_Ascii_Key ( ) const [inline]

IO_Ascii begin event block key IO_Ascii has been removed, but we want to be able to read existing files written by IO_Ascii

Definition at line 41 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.9 std::string HepMC::StreamInfo::IO_Ascii_PDT_End ( ) const [inline]

IO_Ascii end particle data block key.

Definition at line 47 of file StreamInfo.h.

7.69.3.10 std::string HepMC::StreamInfo::IO_Ascii_PDT_End ( ) const [inline]

IO_Ascii end particle data block key.

Definition at line 47 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.11 std::string HepMC::StreamInfo::IO_Ascii_PDT_Key ( ) const [inline]

IO_Ascii begin particle data block key.

Definition at line 45 of file StreamInfo.h.

7.69.3.12 std::string HepMC::StreamInfo::IO_Ascii_PDT_Key ( ) const [inline]

IO_Ascii begin particle data block key.

Definition at line 45 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.13 std::string HepMC::StreamInfo::IO_ExtendedAscii_End ( ) const [inline]

IO_ExtendedAscii end event block key.

Definition at line 54 of file StreamInfo.h.

7.69.3.14 std::string HepMC::StreamInfo::IO_ExtendedAscii_End ( ) const [inline]

IO_ExtendedAscii end event block key.

Definition at line 54 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.15 std::string HepMC::StreamInfo::IO_ExtendedAscii_Key ( ) const [inline]

IO_ExtendedAscii begin event block key IO_ExtendedAscii has been removed, but we want to be able to read existing files written by IO_ExtendedAscii

Definition at line 52 of file StreamInfo.h.

7.69.3.16 std::string HepMC::StreamInfo::IO_ExtendedAscii_Key ( ) const [inline]

IO_ExtendedAscii begin event block key IO_ExtendedAscii has been removed, but we want to be able to read existing files written by IO_ExtendedAscii

Definition at line 52 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.17 std::string HepMC::StreamInfo::IO_ExtendedAscii_PDT_End ( ) const [inline]

IO_ExtendedAscii end particle data block key.

Definition at line 58 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.18 std::string HepMC::StreamInfo::IO_ExtendedAscii_PDT_End ( ) const [inline]

IO_ExtendedAscii end particle data block key.

Definition at line 58 of file StreamInfo.h.

7.69.3.19 std::string HepMC::StreamInfo::IO_ExtendedAscii_PDT_Key ( ) const [inline]

IO_ExtendedAscii begin particle data block key.

Definition at line 56 of file StreamInfo.h.

7.69.3.20 std::string HepMC::StreamInfo::IO_ExtendedAscii_PDT_Key ( ) const [inline]

IO_ExtendedAscii begin particle data block key.

Definition at line 56 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.21 std::string HepMC::StreamInfo::IO_GenEvent_End ( ) const [inline]

IO_GenEvent (p. 186) end event block key.

Definition at line 36 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.22 std::string HepMC::StreamInfo::IO_GenEvent_End ( ) const [inline]

IO_GenEvent (p. 186) end event block key.

Definition at line 36 of file StreamInfo.h.

Referenced by HepMC::write_HepMC_IO_block_end().

7.69.3.23 std::string HepMC::StreamInfo::IO_GenEvent_Key ( ) const [inline]

IO_GenEvent (p. 186) begin event block key.

Definition at line 34 of file StreamInfo.h.

Referenced by HepMC::write_HepMC_IO_block_begin().

7.69.3.24 std::string HepMC::StreamInfo::IO_GenEvent_Key ( ) const [inline]

IO_GenEvent (p. 186) begin event block key.

Definition at line 34 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.25 Units::MomentumUnit HepMC::StreamInfo::io_momentum_unit ( ) const [inline]

get the I/O momentum units

Definition at line 72 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.26 Units::MomentumUnit HepMC::StreamInfo::io_momentum_unit ( ) const [inline]

get the I/O momentum units

Definition at line 72 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read().

7.69.3.27 Units::LengthUnit HepMC::StreamInfo::io_position_unit ( ) const [inline]

get the I/O length units

Definition at line 74 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read().

7.69.3.28 Units::LengthUnit HepMC::StreamInfo::io_position_unit ( ) const [inline]

get the I/O length units

Definition at line 74 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.29 int HepMC::StreamInfo::io_type ( ) const [inline]

get IO type

Definition at line 61 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.30 int HepMC::StreamInfo::io_type ( ) const [inline]

get IO type

Definition at line 61 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read(), and HepMC::detail::read_particle().

7.69.3.31 bool HepMC::StreamInfo::reading_event_header ( )

reading_event_header will return true when streaming input is processing the **GenEvent** (p. 72) header information

Definition at line 51 of file HepMC-2.06.09/src/StreamInfo.cc.

Referenced by HepMC::GenEvent::read().

7.69.3.32 bool HepMC::StreamInfo::reading_event_header ( )

reading_event_header will return true when streaming input is processing the **GenEvent** (p. 72) header information

7.69.3.33 void HepMC::StreamInfo::set_finished_first_event (bool *b*) [inline]

Special information is processed the first time we use the IO.

Definition at line 83 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.34 void HepMC::StreamInfo::set_finished_first_event (bool *b*) [inline]

Special information is processed the first time we use the IO.

Definition at line 83 of file StreamInfo.h.

Referenced by HepMC::GenEvent::read(), and HepMC::GenEvent::write().

7.69.3.35 void HepMC::StreamInfo::set_has_key (bool)

set to false if the stream does not have a file type key

7.69.3.36 void HepMC::StreamInfo::set_has_key (bool *io*)

set to false if the stream does not have a file type key

Definition at line 47 of file HepMC-2.06.09/src/StreamInfo.cc.

7.69.3.37 void HepMC::StreamInfo::set_io_type (int *io*)

set IO type

Definition at line 43 of file HepMC-2.06.09/src/StreamInfo.cc.

7.69.3.38 void HepMC::StreamInfo::set_io_type (int)

set IO type

7.69.3.39 void HepMC::StreamInfo::set_reading_event_header (bool)

set the reading_event_header flag

7.69.3.40 void HepMC::StreamInfo::set_reading_event_header (bool *tf*)

set the reading_event_header flag

Definition at line 55 of file HepMC-2.06.09/src/StreamInfo.cc.

Referenced by HepMC::GenEvent::read().

7.69.3.41 int HepMC::StreamInfo::stream_id ( ) const [inline]

get the I/O stream id This is used for sanity checking.

Definition at line 78 of file 2.06.09/HepMC/StreamInfo.h.

7.69.3.42 int HepMC::StreamInfo::stream_id ( ) const [inline]

get the I/O stream id This is used for sanity checking.

Definition at line 78 of file StreamInfo.h.

Referenced by HepMC::HepMCStreamCallback().

7.69.3.43 void HepMC::StreamInfo::use_input_units (Units::MomentumUnit *mom,* Units::LengthUnit *len*)

needed when reading a file without units if those units are different than the declared default units (e.g., the default units are MeV, but the file was written with GeV) This method is not necessary if the units are written in the file

Definition at line 38 of file HepMC-2.06.09/src/StreamInfo.cc.

Referenced by HepMC::set_input_units().

7.69.3.44 void HepMC::StreamInfo::use_input_units (Units::MomentumUnit , Units::LengthUnit)

needed when reading a file without units if those units are different than the declared default units (e.g., the default units are MeV, but the file was written with GeV) This method is not necessary if the units are written in the file

The documentation for this class was generated from the following files:

- **StreamInfo.h**
- **2.06.09/HepMC/StreamInfo.h**
- **HepMC-2.06.09/src/StreamInfo.cc**
- **src/StreamInfo.cc**

7.70 HepMC::TempParticleMap Class Reference

TempParticleMap (p. 267) is a temporary GenParticle∗ container used during input.

`#include <TempParticleMap.h>`

Public Types

- typedef std::map
  < **HepMC::GenParticle** ∗, int > **TempMap**

- typedef std::map< int, **HepMC::GenParticle** * > **TempOrderMap**
- typedef TempMap::iterator **TempMapIterator**
- typedef TempOrderMap::iterator **orderIterator**
- typedef std::map < **HepMC::GenParticle** *, int > **TempMap**
- typedef std::map< int, **HepMC::GenParticle** * > **TempOrderMap**
- typedef TempMap::iterator **TempMapIterator**
- typedef TempOrderMap::iterator **orderIterator**

Public Member Functions

- **TempParticleMap** ()
- ~**TempParticleMap** ()
- **TempMapIterator begin** ()
- **TempMapIterator end** ()
- **orderIterator order_begin** ()
- **orderIterator order_end** ()
- int **end_vertex** (**GenParticle** *)
- void **addEndParticle** (**GenParticle** *, int &)
- **TempParticleMap** ()
- ~**TempParticleMap** ()
- **TempMapIterator begin** ()
- **TempMapIterator end** ()
- **orderIterator order_begin** ()
- **orderIterator order_end** ()
- int **end_vertex** (**GenParticle** *)
- void **addEndParticle** (**GenParticle** *, int &)

7.70.1 Detailed Description

TempParticleMap (p. 267) is a temporary GenParticle* container used during input.

Used by IO classes for recoverable particle ordering. Map GenParticle* against both outgoing vertex and particle order.

Definition at line 24 of file TempParticleMap.h.

7.70.2 Member Typedef Documentation

7.70.2.1 typedef TempOrderMap::iterator HepMC::TempParticleMap::orderIterator

Definition at line 29 of file TempParticleMap.h.

7.70.2.2 typedef TempOrderMap::iterator HepMC::TempParticleMap::orderIterator

Definition at line 29 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.2.3 typedef std::map<HepMC::GenParticle*,int> HepMC::TempParticleMap::TempMap

Definition at line 26 of file TempParticleMap.h.

7.70.2.4 typedef std::map<HepMC::GenParticle*,int> HepMC::TempParticleMap::TempMap

Definition at line 26 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.2.5 typedef TempMap::iterator HepMC::TempParticleMap::TempMapIterator

Definition at line 28 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.2.6 typedef TempMap::iterator HepMC::TempParticleMap::TempMapIterator

Definition at line 28 of file TempParticleMap.h.

7.70.2.7 typedef std::map<int,HepMC::GenParticle*> HepMC::TempParticleMap::TempOrderMap

Definition at line 27 of file TempParticleMap.h.

7.70.2.8 typedef std::map<int,HepMC::GenParticle*> HepMC::TempParticleMap::TempOrderMap

Definition at line 27 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.3 Constructor & Destructor Documentation

7.70.3.1 HepMC::TempParticleMap::TempParticleMap ( ) [inline]

Definition at line 31 of file TempParticleMap.h.

7.70.3.2 HepMC::TempParticleMap::~TempParticleMap ( ) [inline]

Definition at line 34 of file TempParticleMap.h.

7.70.3.3 HepMC::TempParticleMap::TempParticleMap ( ) [inline]

Definition at line 31 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.3.4 HepMC::TempParticleMap::~TempParticleMap ( ) [inline]

Definition at line 34 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.4 Member Function Documentation

7.70.4.1 void HepMC::TempParticleMap::addEndParticle (GenParticle * *p*, int & *end_vtx_code*) [inline]

Definition at line 58 of file TempParticleMap.h.

References HepMC::GenParticle::barcode(), and p.

Referenced by HepMC::detail::read_particle().

7.70.4.2 void HepMC::TempParticleMap::addEndParticle (GenParticle *, int &)

7.70.4.3 TempMapIterator HepMC::TempParticleMap::begin ( ) [inline]

Definition at line 36 of file TempParticleMap.h.

7.70.4.4 TempMapIterator HepMC::TempParticleMap::begin ( ) [inline]

Definition at line 36 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.4.5 TempMapIterator HepMC::TempParticleMap::end ( ) [inline]

Definition at line 37 of file TempParticleMap.h.

Referenced by end_vertex().

7.70.4.6 TempMapIterator HepMC::TempParticleMap::end ( ) [inline]

Definition at line 37 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.4.7 int HepMC::TempParticleMap::end_vertex (GenParticle * p) [inline]

Definition at line 50 of file TempParticleMap.h.

References end(), and p.

Referenced by HepMC::GenEvent::read().

7.70.4.8 int HepMC::TempParticleMap::end_vertex (GenParticle *)

7.70.4.9 orderIterator HepMC::TempParticleMap::order_begin ( ) [inline]

Definition at line 38 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.4.10 orderIterator HepMC::TempParticleMap::order_begin ( ) [inline]

Definition at line 38 of file TempParticleMap.h.

Referenced by HepMC::GenEvent::read().

7.70.4.11 orderIterator HepMC::TempParticleMap::order_end ( ) [inline]

Definition at line 39 of file 2.06.09/HepMC/TempParticleMap.h.

7.70.4.12 orderIterator HepMC::TempParticleMap::order_end ( ) [inline]

Definition at line 39 of file TempParticleMap.h.

Referenced by HepMC::GenEvent::read().

The documentation for this class was generated from the following files:

- **TempParticleMap.h**
- **2.06.09/HepMC/TempParticleMap.h**

7.71 HepMC::ThreeVector Class Reference

ThreeVector (p. 271) is a simple representation of a position or displacement 3 vector.

`#include <SimpleVector.h>`

Public Member Functions

- **ThreeVector** (double xin, double yin=0, double zin=0)
  construct using x, y, and z (only x is required)
- **ThreeVector** ()
- template<class T >
  ThreeVector (const T &**v**, typename **detail::disable_if**< **detail::is_arithmetic**< T >::value, void >::type *=0)
- **ThreeVector** (const **ThreeVector** &**v**)
  copy constructor
- void **swap** (**ThreeVector** &other)
  swap
- double **x** () const
  return x
- double **y** () const
  return y
- double **z** () const
  return z
- void **setX** (double xin)
  set x
- void **setY** (double yin)
  set y
- void **setZ** (double zin)
  set z
- void **set** (double **x**, double **y**, double **z**)
  set x, y, and z
- double **phi** () const
  The azimuth angle.
- double **theta** () const
  The polar angle.
- double **r** () const
  The magnitude.
- void **setPhi** (double)
  Set phi keeping magnitude and theta constant (BaBar).
- void **setTheta** (double)
  Set theta keeping magnitude and phi constant (BaBar).
- double **perp2** () const
  The transverse component squared (rho^2 in cylindrical coordinate system).
- double **perp** () const
  The transverse component (rho in cylindrical coordinate system).
- **ThreeVector** & **operator=** (const **ThreeVector** &)
  make a copy
- bool **operator==** (const **ThreeVector** &) const
  equality
- bool **operator!=** (const **ThreeVector** &) const

inequality

- **ThreeVector** (double xin, double yin=0, double zin=0)

  construct using x, y, and z (only x is required)

- **ThreeVector** ()

- template<class T >
  ThreeVector (const T &**v**, typename **detail::disable_if**< **detail::is_arithmetic**< T >::value, void >::type *=0)

- **ThreeVector** (const **ThreeVector** &**v**)

  copy constructor

- void **swap** (**ThreeVector** &other)

  swap

- double **x** () const

  return x

- double **y** () const

  return y

- double **z** () const

  return z

- void **setX** (double xin)

  set x

- void **setY** (double yin)

  set y

- void **setZ** (double zin)

  set z

- void **set** (double **x**, double **y**, double **z**)

  set x, y, and z

- double **phi** () const

  The azimuth angle.

- double **theta** () const

  The polar angle.

- double **r** () const

  The magnitude.

- void **setPhi** (double)

  Set phi keeping magnitude and theta constant (BaBar).

- void **setTheta** (double)

  Set theta keeping magnitude and phi constant (BaBar).

- double **perp2** () const

  The transverse component squared (rho^2 in cylindrical coordinate system).

- double **perp** () const

  The transverse component (rho in cylindrical coordinate system).

- **ThreeVector** & **operator=** (const **ThreeVector** &)

  make a copy

- bool **operator==** (const **ThreeVector** &) const

  equality

- bool **operator!=** (const **ThreeVector** &) const

  inequality

7.71.1 Detailed Description

ThreeVector (p. 271) is a simple representation of a position or displacement 3 vector.

For compatibility with existing code, the basic expected geometrical access methods are povided. Also, there is a templated constructor that will take another vector (HepLorentzVector, GenVector, ...) which must have the following methods: **x()** (p. 278), **y()** (p. 278), **z()** (p. 278).

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 131 of file SimpleVector.h.

7.71.2 Constructor & Destructor Documentation

7.71.2.1 HepMC::ThreeVector::ThreeVector (double *xin,* double *yin* = 0, double *zin* = 0) [inline]

construct using x, y, and z (only x is required)

Definition at line 136 of file SimpleVector.h.

7.71.2.2 HepMC::ThreeVector::ThreeVector ( ) [inline]

Definition at line 139 of file SimpleVector.h.

7.71.2.3 template<class T > HepMC::ThreeVector::ThreeVector (const T & *v,* typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type * = 0) [inline]

templated constructor this is used ONLY if T is not arithmetic

Definition at line 145 of file SimpleVector.h.

7.71.2.4 HepMC::ThreeVector::ThreeVector (const ThreeVector & *v*) [inline]

copy constructor

Definition at line 150 of file SimpleVector.h.

7.71.2.5 HepMC::ThreeVector::ThreeVector (double *xin,* double *yin* = 0, double *zin* = 0) [inline]

construct using x, y, and z (only x is required)

Definition at line 136 of file 2.06.09/HepMC/SimpleVector.h.

7.71.2.6 HepMC::ThreeVector::ThreeVector ( ) [inline]

Definition at line 139 of file 2.06.09/HepMC/SimpleVector.h.

7.71.2.7 template<class T > HepMC::ThreeVector::ThreeVector (const T & *v,* typename detail::disable_if< detail::is_arithmetic< T >::value, void >::type * = 0) [inline]

templated constructor this is used ONLY if T is not arithmetic

Definition at line 145 of file 2.06.09/HepMC/SimpleVector.h.

7.71.2.8 HepMC::ThreeVector::ThreeVector (const ThreeVector & *v*) [inline]

copy constructor

Definition at line 150 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3 Member Function Documentation

7.71.3.1 bool HepMC::ThreeVector::operator!= (const ThreeVector &) const

inequality

7.71.3.2 bool HepMC::ThreeVector::operator!= (const ThreeVector &) const

inequality

7.71.3.3 ThreeVector& HepMC::ThreeVector::operator= (const ThreeVector &)

make a copy

7.71.3.4 ThreeVector& HepMC::ThreeVector::operator= (const ThreeVector &)

make a copy

7.71.3.5 bool HepMC::ThreeVector::operator== (const ThreeVector &) const

equality

7.71.3.6 bool HepMC::ThreeVector::operator== (const ThreeVector &) const

equality

7.71.3.7 double HepMC::ThreeVector::perp ( ) const

The transverse component (rho in cylindrical coordinate system).

7.71.3.8 double HepMC::ThreeVector::perp ( ) const

The transverse component (rho in cylindrical coordinate system).

Examples:

testSimpleVector.cc.

Referenced by main().

7.71.3.9 double HepMC::ThreeVector::perp2 ( ) const

The transverse component squared (rho^2 in cylindrical coordinate system).

7.71.3.10 double HepMC::ThreeVector::perp2 ( ) const

The transverse component squared (rho^2 in cylindrical coordinate system).

Examples:

testSimpleVector.cc.

Referenced by main().

7.71.3.11 double HepMC::ThreeVector::phi ( ) const

The azimuth angle.

7.71.3.12 double HepMC::ThreeVector::phi ( ) const

The azimuth angle.

Examples:

testSimpleVector.cc.

Referenced by main(), and HepMC::Polarization::set_normal3d().

7.71.3.13 double HepMC::ThreeVector::r ( ) const

The magnitude.

Examples:

testSimpleVector.cc.

Referenced by main().

7.71.3.14 double HepMC::ThreeVector::r ( ) const

The magnitude.

7.71.3.15 void HepMC::ThreeVector::set (double *x,* double *y,* double *z*)

set x, y, and z

Examples:

testSimpleVector.cc.

Referenced by main().

7.71.3.16 void HepMC::ThreeVector::set (double *x,* double *y,* double *z*)

set x, y, and z

7.71.3.17 void HepMC::ThreeVector::setPhi (double)

Set phi keeping magnitude and theta constant (BaBar).

7.71.3.18 void HepMC::ThreeVector::setPhi (double)

Set phi keeping magnitude and theta constant (BaBar).

Examples:

testSimpleVector.cc.

Referenced by main(), and HepMC::Polarization::normal3d().

7.71.3.19 void HepMC::ThreeVector::setTheta (double)

Set theta keeping magnitude and phi constant (BaBar).

7.71.3.20 void HepMC::ThreeVector::setTheta (double)

Set theta keeping magnitude and phi constant (BaBar).

Examples:

testSimpleVector.cc.

Referenced by main(), and HepMC::Polarization::normal3d().

7.71.3.21 void HepMC::ThreeVector::setX (double *xin*) [inline]

set x

Examples:

testSimpleVector.cc.

Definition at line 159 of file SimpleVector.h.

Referenced by main().

7.71.3.22 void HepMC::ThreeVector::setX (double *xin*) [inline]

set x

Definition at line 159 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3.23 void HepMC::ThreeVector::setY (double *yin*) [inline]

set y

Definition at line 160 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3.24 void HepMC::ThreeVector::setY (double *yin*) [inline]

set y

Examples:

testSimpleVector.cc.

Definition at line 160 of file SimpleVector.h.

Referenced by main().

7.71.3.25 void HepMC::ThreeVector::setZ (double *zin*) [inline]

set z

Examples:

testSimpleVector.cc.

Definition at line 161 of file SimpleVector.h.

Referenced by main().

7.71.3.26 void HepMC::ThreeVector::setZ (double *zin*) [inline]

set z

Definition at line 161 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3.27 void HepMC::ThreeVector::swap (ThreeVector & *other*)

swap

7.71.3.28 void HepMC::ThreeVector::swap (ThreeVector & *other*)

swap

7.71.3.29 double HepMC::ThreeVector::theta ( ) const

The polar angle.

Examples:

testSimpleVector.cc.

Referenced by main(), and HepMC::Polarization::set_normal3d().

7.71.3.30 double HepMC::ThreeVector::theta ( ) const

The polar angle.

7.71.3.31 double HepMC::ThreeVector::x ( ) const [inline]

return x

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 155 of file SimpleVector.h.

Referenced by convertTo(), and main().

7.71.3.32 double HepMC::ThreeVector::x ( ) const [inline]

return x

Definition at line 155 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3.33 double HepMC::ThreeVector::y ( ) const [inline]

return y

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 156 of file SimpleVector.h.

Referenced by convertTo(), and main().

7.71.3.34 double HepMC::ThreeVector::y ( ) const [inline]

return y

Definition at line 156 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3.35 double HepMC::ThreeVector::z ( ) const [inline]

return z

Definition at line 157 of file 2.06.09/HepMC/SimpleVector.h.

7.71.3.36 double HepMC::ThreeVector::z ( ) const [inline]

return z

Examples:

testSimpleVector.cc, and **VectorConversion.h**.

Definition at line 157 of file SimpleVector.h.

Referenced by convertTo(), and main().

The documentation for this class was generated from the following files:

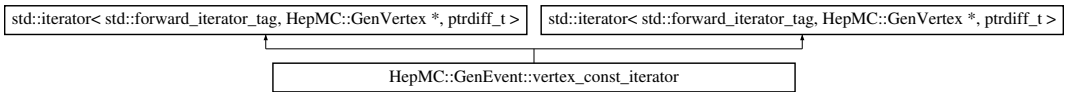
- **SimpleVector.h**
- **2.06.09/HepMC/SimpleVector.h**

7.72 HepMC::GenEvent::vertex_const_iterator Class Reference

const vertex iterator

`#include <GenEvent.h>`

Inheritance diagram for HepMC::GenEvent::vertex_const_iterator:



Public Member Functions

- **vertex_const_iterator** (const std::map< int, **HepMC::GenVertex** *, std::greater< int > >::const_iterator &i)
  constructor requiring vertex information
- **vertex_const_iterator** ()
- **vertex_const_iterator** (const **vertex_const_iterator** &i)
  copy constructor
- virtual ~**vertex_const_iterator** ()
- **vertex_const_iterator** & **operator=** (const **vertex_const_iterator** &i)
  make a copy
- **GenVertex** * **operator*** (void) const
  *return a pointer to a **GenVertex** (p. 125)*
- **vertex_const_iterator** & **operator++** (void)
  Pre-fix increment.
- **vertex_const_iterator operator++** (int)
  Post-fix increment.
- bool **operator==** (const **vertex_const_iterator** &a) const
  equality
- bool **operator!=** (const **vertex_const_iterator** &a) const
  inequality
- **vertex_const_iterator** (const std::map< int, **HepMC::GenVertex** *, std::greater< int > >::const_iterator &i)
  constructor requiring vertex information
- **vertex_const_iterator** ()
- **vertex_const_iterator** (const **vertex_const_iterator** &i)
  copy constructor
- virtual ~**vertex_const_iterator** ()
- **vertex_const_iterator** & **operator=** (const **vertex_const_iterator** &i)
  make a copy
- **GenVertex** * **operator*** (void) const
  *return a pointer to a **GenVertex** (p. 125)*
- **vertex_const_iterator** & **operator++** (void)
  Pre-fix increment.
- **vertex_const_iterator operator++** (int)
  Post-fix increment.
- bool **operator==** (const **vertex_const_iterator** &a) const
  equality
- bool **operator!=** (const **vertex_const_iterator** &a) const
  inequality

Protected Attributes

- std::map< int, **HepMC::GenVertex** *, std::greater< int > >::const_iterator **m_map_iterator**

  const iterator to a vertex map

7.72.1 Detailed Description

const vertex iterator

HepMC::GenEvent::vertex_const_iterator (p. 279) is used to iterate over all vertices in the event.

Definition at line 334 of file GenEvent.h.

7.72.2 Constructor & Destructor Documentation

7.72.2.1 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator (const std::map< int, HepMC::GenVertex *, std::greater< int > >::const_iterator & *i*) [inline]

constructor requiring vertex information

Definition at line 339 of file GenEvent.h.

7.72.2.2 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator ( ) [inline]

Definition at line 343 of file GenEvent.h.

7.72.2.3 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator (const vertex_const_iterator & *i*) [inline]

copy constructor

Definition at line 345 of file GenEvent.h.

7.72.2.4 virtual HepMC::GenEvent::vertex_const_iterator::~vertex_const_iterator ( ) [inline], [virtual]

Definition at line 347 of file GenEvent.h.

7.72.2.5 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator (const std::map< int, HepMC::GenVertex *, std::greater< int > >::const_iterator & *i*) [inline]

constructor requiring vertex information

Definition at line 339 of file 2.06.09/HepMC/GenEvent.h.

7.72.2.6 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator ( ) [inline]

Definition at line 343 of file 2.06.09/HepMC/GenEvent.h.

7.72.2.7 HepMC::GenEvent::vertex_const_iterator::vertex_const_iterator (const vertex_const_iterator & *i*) [inline]

copy constructor

Definition at line 345 of file 2.06.09/HepMC/GenEvent.h.

7.72.2.8 virtual HepMC::GenEvent::vertex_const_iterator::~vertex_const_iterator ( ) [inline], [virtual]

Definition at line 347 of file 2.06.09/HepMC/GenEvent.h.

7.72.3 Member Function Documentation

7.72.3.1 bool HepMC::GenEvent::vertex_const_iterator::operator!= (const vertex_const_iterator & *a*) const [inline]

inequality

Definition at line 363 of file GenEvent.h.

References m_map_iterator.

7.72.3.2 bool HepMC::GenEvent::vertex_const_iterator::operator!= (const vertex_const_iterator & *a*) const [inline]

inequality

Definition at line 363 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.72.3.3 GenVertex* HepMC::GenEvent::vertex_const_iterator::operator* (void) const [inline]

return a pointer to a **GenVertex** (p. 125)

Definition at line 352 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.72.3.4 GenVertex* HepMC::GenEvent::vertex_const_iterator::operator* (void) const [inline]

return a pointer to a **GenVertex** (p. 125)

Definition at line 352 of file GenEvent.h.

References m_map_iterator.

7.72.3.5 vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 354 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.72.3.6 vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 354 of file GenEvent.h.

References m_map_iterator.

7.72.3.7 vertex_const_iterator HepMC::GenEvent::vertex_const_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 357 of file GenEvent.h.

7.72.3.8 vertex_const_iterator HepMC::GenEvent::vertex_const_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 357 of file 2.06.09/HepMC/GenEvent.h.

7.72.3.9 vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator= (const vertex_const_iterator & *i*) [inline]

make a copy

Definition at line 349 of file GenEvent.h.

References m_map_iterator.

7.72.3.10 vertex_const_iterator& HepMC::GenEvent::vertex_const_iterator::operator= (const vertex_const_iterator & *i*) [inline]

make a copy

Definition at line 349 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.72.3.11 bool HepMC::GenEvent::vertex_const_iterator::operator== (const vertex_const_iterator & *a*) const [inline]

equality

Definition at line 360 of file GenEvent.h.

References m_map_iterator.

7.72.3.12 bool HepMC::GenEvent::vertex_const_iterator::operator== (const vertex_const_iterator & *a*) const [inline]

equality

Definition at line 360 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.72.4 Member Data Documentation

7.72.4.1 std::map< int, HepMC::GenVertex *, std::greater< int > >::const_iterator HepMC::GenEvent::vertex_const_iterator::m_map_iterator [protected]

const iterator to a vertex map

Definition at line 368 of file GenEvent.h.

Referenced by operator!=(), operator∗(), operator++(), operator=(), and operator==().

The documentation for this class was generated from the following files:

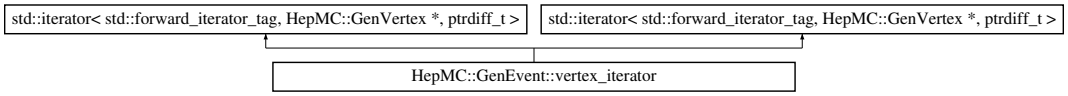
- **GenEvent.h**
- **2.06.09/HepMC/GenEvent.h**

7.73 HepMC::GenEvent::vertex_iterator Class Reference

non-const vertex iterator

`#include <GenEvent.h>`

Inheritance diagram for HepMC::GenEvent::vertex_iterator:



Public Member Functions

- **vertex_iterator** (const std::map< int, **HepMC::GenVertex** ∗, std::greater< int > >::iterator &i)
  constructor requiring vertex information
- **vertex_iterator** ()
- **vertex_iterator** (const **vertex_iterator** &i)
  copy constructor
- virtual ∼**vertex_iterator** ()
- **vertex_iterator** & **operator=** (const **vertex_iterator** &i)
  make a copy
- **operator vertex_const_iterator** () const
  const vertex iterator
- **GenVertex** ∗ **operator**∗ (void) const
  *return a pointer to a **GenVertex** (p. 125)*
- **vertex_iterator** & **operator++** (void)
  Pre-fix increment.
- **vertex_iterator operator++** (int)
  Post-fix increment.
- bool **operator==** (const **vertex_iterator** &a) const
  equality
- bool **operator!=** (const **vertex_iterator** &a) const
  inequality
- **vertex_iterator** (const std::map< int, **HepMC::GenVertex** ∗, std::greater< int > >::iterator &i)
  constructor requiring vertex information
- **vertex_iterator** ()
- **vertex_iterator** (const **vertex_iterator** &i)
  copy constructor
- virtual ∼**vertex_iterator** ()
- **vertex_iterator** & **operator=** (const **vertex_iterator** &i)
  make a copy
- **operator vertex_const_iterator** () const

const vertex iterator

- **GenVertex** ∗ **operator**∗ (void) const

  *return a pointer to a **GenVertex** (p. 125)*
- **vertex_iterator** & **operator++** (void)

  Pre-fix increment.
- **vertex_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **vertex_iterator** &a) const

  equality
- bool **operator!=** (const **vertex_iterator** &a) const

  inequality

Protected Attributes

- std::map< int, **HepMC::GenVertex** ∗, std::greater< int > >::iterator **m_map_iterator**

  iterator to the vertex map

7.73.1 Detailed Description

non-const vertex iterator

HepMC::GenEvent::vertex_iterator (p. 283) is used to iterate over all vertices in the event.

Examples:

example_UsingIterators.cc, and **testHepMCIteration.cc.in**.

Definition at line 391 of file GenEvent.h.

7.73.2 Constructor & Destructor Documentation

7.73.2.1 HepMC::GenEvent::vertex_iterator::vertex_iterator (const std::map< int, HepMC::GenVertex ∗, std::greater< int > >::iterator & *i*) [inline]

constructor requiring vertex information

Definition at line 396 of file GenEvent.h.

7.73.2.2 HepMC::GenEvent::vertex_iterator::vertex_iterator ( ) [inline]

Definition at line 400 of file GenEvent.h.

7.73.2.3 HepMC::GenEvent::vertex_iterator::vertex_iterator (const vertex_iterator & *i*) [inline]

copy constructor

Definition at line 402 of file GenEvent.h.

7.73.2.4 virtual HepMC::GenEvent::vertex_iterator::~vertex_iterator ( ) [inline], [virtual]

Definition at line 403 of file GenEvent.h.

7.73.2.5 HepMC::GenEvent::vertex_iterator::vertex_iterator (const std::map< int, HepMC::GenVertex *, std::greater< int > >::iterator & *i*) [inline]

constructor requiring vertex information

Definition at line 396 of file 2.06.09/HepMC/GenEvent.h.

7.73.2.6 HepMC::GenEvent::vertex_iterator::vertex_iterator ( ) [inline]

Definition at line 400 of file 2.06.09/HepMC/GenEvent.h.

7.73.2.7 HepMC::GenEvent::vertex_iterator::vertex_iterator (const vertex_iterator & *i*) [inline]

copy constructor

Definition at line 402 of file 2.06.09/HepMC/GenEvent.h.

7.73.2.8 virtual HepMC::GenEvent::vertex_iterator::~vertex_iterator ( ) [inline], [virtual]

Definition at line 403 of file 2.06.09/HepMC/GenEvent.h.

7.73.3 Member Function Documentation

7.73.3.1 HepMC::GenEvent::vertex_iterator::operator vertex_const_iterator ( ) const [inline]

const vertex iterator

Definition at line 410 of file GenEvent.h.

References m_map_iterator, and HepMC::GenEvent::vertex_const_iterator.

7.73.3.2 HepMC::GenEvent::vertex_iterator::operator vertex_const_iterator ( ) const [inline]

const vertex iterator

Definition at line 410 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator, and HepMC::GenEvent::vertex_const_iterator.

7.73.3.3 bool HepMC::GenEvent::vertex_iterator::operator!= (const vertex_iterator & *a*) const [inline]

inequality

Definition at line 425 of file GenEvent.h.

References m_map_iterator.

7.73.3.4 bool HepMC::GenEvent::vertex_iterator::operator!= (const vertex_iterator & *a*) const [inline]

inequality

Definition at line 425 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.73.3.5 GenVertex* HepMC::GenEvent::vertex_iterator::operator* (void) const [inline]

return a pointer to a **GenVertex** (p. 125)

Definition at line 413 of file GenEvent.h.

References m_map_iterator.

7.73.3.6 GenVertex* HepMC::GenEvent::vertex_iterator::operator* (void) const [inline]

return a pointer to a **GenVertex** (p. 125)

Definition at line 413 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.73.3.7 vertex_iterator& HepMC::GenEvent::vertex_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 416 of file GenEvent.h.

References m_map_iterator.

7.73.3.8 vertex_iterator& HepMC::GenEvent::vertex_iterator::operator++ (void) [inline]

Pre-fix increment.

Definition at line 416 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.73.3.9 vertex_iterator HepMC::GenEvent::vertex_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 419 of file GenEvent.h.

7.73.3.10 vertex_iterator HepMC::GenEvent::vertex_iterator::operator++ (int) [inline]

Post-fix increment.

Definition at line 419 of file 2.06.09/HepMC/GenEvent.h.

7.73.3.11 vertex_iterator& HepMC::GenEvent::vertex_iterator::operator= (const vertex_iterator & *i*) [inline]

make a copy

Definition at line 405 of file GenEvent.h.

References m_map_iterator.

7.73.3.12 vertex_iterator& HepMC::GenEvent::vertex_iterator::operator= (const vertex_iterator & *i*) [inline]

make a copy

Definition at line 405 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.73.3.13 bool HepMC::GenEvent::vertex_iterator::operator== (const vertex_iterator & *a*) const [inline]

equality

Definition at line 422 of file GenEvent.h.

References m_map_iterator.

7.73.3.14 bool HepMC::GenEvent::vertex_iterator::operator== (const vertex_iterator & *a*) const [inline]

equality

Definition at line 422 of file 2.06.09/HepMC/GenEvent.h.

References m_map_iterator.

7.73.4 Member Data Documentation

7.73.4.1 std::map< int, HepMC::GenVertex *, std::greater< int > >::iterator HepMC::GenEvent::vertex_iterator::m_map_iterator [protected]

iterator to the vertex map

Definition at line 430 of file GenEvent.h.

Referenced by operator vertex_const_iterator(), operator!=(), operator*(), operator++(), operator=(), and operator==().

The documentation for this class was generated from the following files:

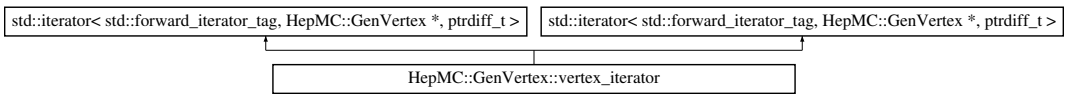
- **GenEvent.h**
- **2.06.09/HepMC/GenEvent.h**

7.74 HepMC::GenVertex::vertex_iterator Class Reference

vertex iterator

`#include <GenVertex.h>`

Inheritance diagram for HepMC::GenVertex::vertex_iterator:



Public Member Functions

- **vertex_iterator** ()
- **vertex_iterator** (**GenVertex** &vtx_root, **IteratorRange range**)

  used to set limits on the iteration
- **vertex_iterator** (**GenVertex** &vtx_root, **IteratorRange range**, std::set< const **HepMC::GenVertex** * > &visited_vertices)

  next constructor is intended for internal use only
- **vertex_iterator** (const **vertex_iterator** &v_iter)

  copy
- virtual ~**vertex_iterator** ()

- **vertex_iterator** & **operator=** (const **vertex_iterator** &)

  make a copy
- **GenVertex** ∗ **operator**∗ (void) const

  return a pointer to a vertex
- **vertex_iterator** & **operator++** (void)

  Pre-fix increment.
- **vertex_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **vertex_iterator** &) const

  equality
- bool **operator!=** (const **vertex_iterator** &) const

  inequality
- **GenVertex** ∗ **vertex_root** () const

  vertex that this iterator begins from
- **IteratorRange range** () const

  iterator range
- void **copy_with_own_set** (const **vertex_iterator** &v_iter, std::set< const **HepMC::GenVertex** ∗ > &visited_vertices)

  intended for internal use only.
- **vertex_iterator** ()
- **vertex_iterator** (**GenVertex** &vtx_root, **IteratorRange range**)

  used to set limits on the iteration
- **vertex_iterator** (**GenVertex** &vtx_root, **IteratorRange range**, std::set< const **HepMC::GenVertex** ∗ > &visited_vertices)

  next constructor is intended for internal use only
- **vertex_iterator** (const **vertex_iterator** &v_iter)

  copy
- virtual ∼**vertex_iterator** ()
- **vertex_iterator** & **operator=** (const **vertex_iterator** &)

  make a copy
- **GenVertex** ∗ **operator**∗ (void) const

  return a pointer to a vertex
- **vertex_iterator** & **operator++** (void)

  Pre-fix increment.
- **vertex_iterator operator++** (int)

  Post-fix increment.
- bool **operator==** (const **vertex_iterator** &) const

  equality
- bool **operator!=** (const **vertex_iterator** &) const

  inequality
- **GenVertex** ∗ **vertex_root** () const

  vertex that this iterator begins from
- **IteratorRange range** () const

  iterator range
- void **copy_with_own_set** (const **vertex_iterator** &v_iter, std::set< const **HepMC::GenVertex** ∗ > &visited_vertices)

  intended for internal use only.

Protected Member Functions

- **GenVertex** ∗ **follow_edge_** ()

  non-null if recursive iter. created
- void **copy_recursive_iterator_** (const **vertex_iterator** ∗recursive_v_iter)

  copy recursive iterator
- **GenVertex** ∗ **follow_edge_** ()

  non-null if recursive iter. created
- void **copy_recursive_iterator_** (const **vertex_iterator** ∗recursive_v_iter)

  copy recursive iterator

7.74.1 Detailed Description

vertex iterator

Iterates over all vertices connected via a graph to this vertex. this is made friend to that it can access protected edge iterator the range can be IteratorRange= (parents, children, family, ancestors, descendants, relatives) example for range=descendants the iterator will return all vertices which are children (connected by an outgoing particle edge), grandchildren, great-grandchildren, etc. of this vertex In all cases the iterator always returns this vertex (returned last). The algorithm is accomplished by converting the graph to a tree (by "chopping" the edges connecting to an already visited vertex) and returning the vertices in POST ORDER traversal.

Definition at line 263 of file GenVertex.h.

7.74.2 Constructor & Destructor Documentation

7.74.2.1 HepMC::GenVertex::vertex_iterator::vertex_iterator ( )

Definition at line 607 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.2.2 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & *vtx_root,* IteratorRange *range*)

used to set limits on the iteration

Definition at line 612 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenVertex::edges_begin(), HepMC::GenVertex::edges_end(), and follow_edge_().

7.74.2.3 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & *vtx_root,* IteratorRange *range,* std::set< const HepMC::GenVertex ∗ > & *visited_vertices*)

next constructor is intended for internal use only

Definition at line 628 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenVertex::edges_begin(), HepMC::GenVertex::edges_end(), and follow_edge_().

7.74.2.4 HepMC::GenVertex::vertex_iterator::vertex_iterator (const vertex_iterator & *v_iter*)

copy

Definition at line 645 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.2.5 HepMC::GenVertex::vertex_iterator::~vertex_iterator ( ) [virtual]

Definition at line 652 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.2.6 HepMC::GenVertex::vertex_iterator::vertex_iterator ( )

7.74.2.7 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & *vtx_root,* IteratorRange *range*)

used to set limits on the iteration

7.74.2.8 HepMC::GenVertex::vertex_iterator::vertex_iterator (GenVertex & *vtx_root,* IteratorRange *range,* std::set< const HepMC::GenVertex * > & *visited_vertices*)

next constructor is intended for internal use only

7.74.2.9 HepMC::GenVertex::vertex_iterator::vertex_iterator (const vertex_iterator & *v_iter*)

copy

7.74.2.10 virtual HepMC::GenVertex::vertex_iterator::~vertex_iterator ( ) [virtual]

7.74.3 Member Function Documentation

7.74.3.1 void HepMC::GenVertex::vertex_iterator::copy_recursive_iterator_ (const vertex_iterator * *recursive_v_iter*) [protected]

copy recursive iterator

Definition at line 817 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::GenVertex::vertex_iterator.

7.74.3.2 void HepMC::GenVertex::vertex_iterator::copy_recursive_iterator_ (const vertex_iterator * *recursive_v_iter*) [protected]

copy recursive iterator

7.74.3.3 void HepMC::GenVertex::vertex_iterator::copy_with_own_set (const vertex_iterator & *v_iter,* std::set< const HepMC::GenVertex * > & *visited_vertices*)

intended for internal use only.

intended for internal use only. (use with care!) this is the same as the operator= method, but it allows the user to specify which set container m_visited_vertices points to. in all cases, this vertex will NOT own its set.

intended for internal use only. (use with care!) this is the same as the operator= method, but it allows the user to specify which set container m_visited_vertices points to. in all cases, this vertex will NOT own its set.

Definition at line 758 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.3.4 void HepMC::GenVertex::vertex_iterator::copy_with_own_set (const vertex_iterator & *v_iter,* std::set< const HepMC::GenVertex * > & *visited_vertices*)

intended for internal use only.

7.74.3.5 GenVertex* HepMC::GenVertex::vertex_iterator::follow_edge_ ( ) [protected]

non-null if recursive iter. created

7.74.3.6 GenVertex * HepMC::GenVertex::vertex_iterator::follow_edge_ ( ) [protected]

non-null if recursive iter. created

Definition at line 781 of file HepMC-2.06.09/src/GenVertex.cc.

References HepMC::family, and HepMC::GenVertex::vertex_iterator.

Referenced by vertex_iterator().

7.74.3.7 bool HepMC::GenVertex::vertex_iterator::operator!= (const vertex_iterator &) const

inequality

7.74.3.8 bool HepMC::GenVertex::vertex_iterator::operator!= (const vertex_iterator & *a*) const [inline]

inequality

Definition at line 491 of file GenVertex.h.

7.74.3.9 GenVertex * HepMC::GenVertex::vertex_iterator::operator* (void) const

return a pointer to a vertex

Definition at line 694 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.3.10 GenVertex* HepMC::GenVertex::vertex_iterator::operator* (void) const

return a pointer to a vertex

7.74.3.11 GenVertex::vertex_iterator & HepMC::GenVertex::vertex_iterator::operator++ (void)

Pre-fix increment.

Definition at line 709 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.3.12 vertex_iterator& HepMC::GenVertex::vertex_iterator::operator++ (void)

Pre-fix increment.

7.74.3.13 GenVertex::vertex_iterator HepMC::GenVertex::vertex_iterator::operator++ (int)

Post-fix increment.

Definition at line 751 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.3.14 vertex_iterator HepMC::GenVertex::vertex_iterator::operator++ (int)

Post-fix increment.

7.74.3.15 GenVertex::vertex_iterator & HepMC::GenVertex::vertex_iterator::operator= (const vertex_iterator & *v_iter*)

make a copy

Definition at line 657 of file HepMC-2.06.09/src/GenVertex.cc.

7.74.3.16 vertex_iterator& HepMC::GenVertex::vertex_iterator::operator= (const vertex_iterator &)

make a copy

7.74.3.17 bool HepMC::GenVertex::vertex_iterator::operator== (const vertex_iterator &) const

equality

7.74.3.18 bool HepMC::GenVertex::vertex_iterator::operator== (const vertex_iterator & *a*) const [inline]

equality

Definition at line 486 of file GenVertex.h.

7.74.3.19 IteratorRange HepMC::GenVertex::vertex_iterator::range ( ) const [inline]

iterator range

Definition at line 500 of file GenVertex.h.

7.74.3.20 IteratorRange HepMC::GenVertex::vertex_iterator::range ( ) const

iterator range

7.74.3.21 GenVertex* HepMC::GenVertex::vertex_iterator::vertex_root ( ) const

vertex that this iterator begins from

7.74.3.22 GenVertex * HepMC::GenVertex::vertex_iterator::vertex_root ( ) const [inline]

vertex that this iterator begins from

Definition at line 496 of file GenVertex.h.

The documentation for this class was generated from the following files:

- **GenVertex.h**
- **2.06.09/HepMC/GenVertex.h**
- **HepMC-2.06.09/src/GenVertex.cc**
- **src/GenVertex.cc**

7.75 HepMC::WeightContainer Class Reference

Container for the Weights associated with an event or vertex.

`#include <WeightContainer.h>`

Public Types

- typedef std::size_t **size_type**
  defining the size type used by vector and map
- typedef std::vector< double >
  ::**iterator iterator**

iterator for the weight container

- typedef std::vector< double > ::**const_iterator const_iterator**

  const iterator for the weight container
- typedef std::size_t **size_type**

  defining the size type used by vector and map
- typedef std::vector< double > ::**iterator iterator**

  iterator for the weight container
- typedef std::vector< double > ::**const_iterator const_iterator**

  const iterator for the weight container

Public Member Functions

- **WeightContainer** (**size_type n**=0, double value=0.)

  default constructor
- **WeightContainer** (const std::vector< double > &weights)

  construct from a vector of weights
- **WeightContainer** (const **WeightContainer** &in)

  copy
- ~**WeightContainer** ()
- void **swap** (**WeightContainer** &other)

  swap
- **WeightContainer** & **operator=** (const **WeightContainer** &)

  copy assignment
- **WeightContainer** & **operator=** (const std::vector< double > &in)

  alternate assignment using a vector of doubles
- void **print** (std::ostream &ostr=std::cout) const

  print weights
- void **write** (std::ostream &ostr=std::cout) const

  write weights in a readable table
- **size_type size** () const

  size of weight container
- bool **empty** () const

  return true if weight container is empty
- void **push_back** (const double &)

  push onto weight container
- void **pop_back** ()

  pop from weight container
- void **clear** ()

  clear the weight container
- bool **has_key** (const std::string &s) const

  check to see if a name exists in the map
- double & **operator[]** (**size_type n**)

  access the weight container
- const double & **operator[]** (**size_type n**) const

  access the weight container
- double & **operator[]** (const std::string &s)

  access the weight container
- const double & **operator[]** (const std::string &s) const

access the weight container

- bool **operator==** (const **WeightContainer** &) const

  equality
- bool **operator!=** (const **WeightContainer** &) const

  inequality
- double & **front** ()

  returns the first element
- const double & **front** () const

  returns the first element
- double & **back** ()

  returns the last element
- const double & **back** () const

  returns the last element
- **iterator begin** ()

  begining of the weight container
- **iterator end** ()

  end of the weight container
- **const_iterator begin** () const

  begining of the weight container
- **const_iterator end** () const

  end of the weight container
- **WeightContainer** (**size_type n**=0, double value=0.)

  default constructor
- **WeightContainer** (const std::vector< double > &weights)

  construct from a vector of weights
- **WeightContainer** (const **WeightContainer** &in)

  copy
- ~**WeightContainer** ()
- void **swap** (**WeightContainer** &other)

  swap
- **WeightContainer** & **operator=** (const **WeightContainer** &)

  copy assignment
- **WeightContainer** & **operator=** (const std::vector< double > &in)

  alternate assignment using a vector of doubles
- void **print** (std::ostream &ostr=std::cout) const

  print weights
- void **write** (std::ostream &ostr=std::cout) const

  write weights in a readable table
- **size_type size** () const

  size of weight container
- bool **empty** () const

  return true if weight container is empty
- void **push_back** (const double &)

  push onto weight container
- void **pop_back** ()

  pop from weight container
- void **clear** ()

  clear the weight container
- bool **has_key** (const std::string &s) const

  check to see if a name exists in the map
- double & **operator[]** (**size_type n**)

access the weight container

- const double & **operator[]** (**size_type n**) const

  access the weight container
- double & **operator[]** (const std::string &s)

  access the weight container
- const double & **operator[]** (const std::string &s) const

  access the weight container
- bool **operator==** (const **WeightContainer** &) const

  equality
- bool **operator!=** (const **WeightContainer** &) const

  inequality
- double & **front** ()

  returns the first element
- const double & **front** () const

  returns the first element
- double & **back** ()

  returns the last element
- const double & **back** () const

  returns the last element
- **iterator begin** ()

  begining of the weight container
- **iterator end** ()

  end of the weight container
- **const_iterator begin** () const

  begining of the weight container
- **const_iterator end** () const

  end of the weight container

Friends

- class **GenEvent**

7.75.1 Detailed Description

Container for the Weights associated with an event or vertex.

This class has both map-like and vector-like functionality. Named weights are now supported.

Definition at line 29 of file WeightContainer.h.

7.75.2 Member Typedef Documentation

7.75.2.1 typedef std::vector<double>::const_iterator HepMC::WeightContainer::const_iterator

const iterator for the weight container

Definition at line 38 of file WeightContainer.h.

7.75.2.2 typedef std::vector<double>::const_iterator HepMC::WeightContainer::const_iterator

const iterator for the weight container

Definition at line 38 of file 2.06.09/HepMC/WeightContainer.h.

7.75.2.3 typedef std::vector<double>::iterator HepMC::WeightContainer::iterator

iterator for the weight container

Definition at line 36 of file 2.06.09/HepMC/WeightContainer.h.

7.75.2.4 typedef std::vector<double>::iterator HepMC::WeightContainer::iterator

iterator for the weight container

Definition at line 36 of file WeightContainer.h.

7.75.2.5 typedef std::size_t HepMC::WeightContainer::size_type

defining the size type used by vector and map

Definition at line 34 of file 2.06.09/HepMC/WeightContainer.h.

7.75.2.6 typedef std::size_t HepMC::WeightContainer::size_type

defining the size type used by vector and map

Definition at line 34 of file WeightContainer.h.

7.75.3 Constructor & Destructor Documentation

7.75.3.1 HepMC::WeightContainer::WeightContainer (size_type *n* = 0, double *value* = 0.) [explicit]

default constructor

Definition at line 22 of file HepMC-2.06.09/src/WeightContainer.cc.

7.75.3.2 HepMC::WeightContainer::WeightContainer (const std::vector< double > & *weights*)

construct from a vector of weights

Definition at line 26 of file HepMC-2.06.09/src/WeightContainer.cc.

References size().

7.75.3.3 HepMC::WeightContainer::WeightContainer (const WeightContainer & *in*) [inline]

copy

Definition at line 141 of file WeightContainer.h.

7.75.3.4 HepMC::WeightContainer::~WeightContainer ( ) [inline]

Definition at line 145 of file WeightContainer.h.

7.75.3.5 HepMC::WeightContainer::WeightContainer (size_type *n* = 0, double *value* = 0.) [explicit]

default constructor

7.75.3.6 HepMC::WeightContainer::WeightContainer (const std::vector< double > & *weights*)

construct from a vector of weights

7.75.3.7 HepMC::WeightContainer::WeightContainer (const WeightContainer & *in*)

copy

7.75.3.8 HepMC::WeightContainer::~WeightContainer ( )

7.75.4 Member Function Documentation

7.75.4.1 double & HepMC::WeightContainer::back ( ) [inline]

returns the last element

Definition at line 190 of file WeightContainer.h.

7.75.4.2 double& HepMC::WeightContainer::back ( )

returns the last element

7.75.4.3 const double & HepMC::WeightContainer::back ( ) const [inline]

returns the last element

Definition at line 192 of file WeightContainer.h.

7.75.4.4 const double& HepMC::WeightContainer::back ( ) const

returns the last element

7.75.4.5 WeightContainer::iterator HepMC::WeightContainer::begin ( ) [inline]

begining of the weight container

Definition at line 195 of file WeightContainer.h.

Referenced by write(), and HepMC::IO_AsciiParticles::write_event().

7.75.4.6 iterator HepMC::WeightContainer::begin ( )

begining of the weight container

7.75.4.7 WeightContainer::const_iterator HepMC::WeightContainer::begin ( ) const [inline]

begining of the weight container

Definition at line 201 of file WeightContainer.h.

7.75.4.8 const_iterator HepMC::WeightContainer::begin ( ) const

begining of the weight container

7.75.4.9 void HepMC::WeightContainer::clear ( )

clear the weight container

7.75.4.10 void HepMC::WeightContainer::clear ( ) [inline]

clear the weight container

Definition at line 173 of file WeightContainer.h.

7.75.4.11 bool HepMC::WeightContainer::empty ( ) const

return true if weight container is empty

7.75.4.12 bool HepMC::WeightContainer::empty ( ) const [inline]

return true if weight container is empty

Definition at line 171 of file WeightContainer.h.

Referenced by main().

7.75.4.13 WeightContainer::iterator HepMC::WeightContainer::end ( ) [inline]

end of the weight container

Definition at line 198 of file WeightContainer.h.

Referenced by HepMC::GenVertex::print(), write(), and HepMC::IO_AsciiParticles::write_event().

7.75.4.14 iterator HepMC::WeightContainer::end ( )

end of the weight container

7.75.4.15 WeightContainer::const_iterator HepMC::WeightContainer::end ( ) const [inline]

end of the weight container

Definition at line 204 of file WeightContainer.h.

7.75.4.16 const_iterator HepMC::WeightContainer::end ( ) const

end of the weight container

7.75.4.17 double & HepMC::WeightContainer::front ( ) [inline]

returns the first element

Definition at line 185 of file WeightContainer.h.

7.75.4.18 double& HepMC::WeightContainer::front ( )

returns the first element

7.75.4.19 const double & HepMC::WeightContainer::front ( ) const [inline]

returns the first element

Definition at line 187 of file WeightContainer.h.

7.75.4.20 const double& HepMC::WeightContainer::front ( ) const

returns the first element

7.75.4.21 bool HepMC::WeightContainer::has_key (const std::string & *s*) const

check to see if a name exists in the map

7.75.4.22 bool HepMC::WeightContainer::has_key (const std::string & *s*) const

check to see if a name exists in the map

Definition at line 105 of file HepMC-2.06.09/src/WeightContainer.cc.

Referenced by main().

7.75.4.23 bool HepMC::WeightContainer::operator!= (const WeightContainer &) const

inequality

7.75.4.24 bool HepMC::WeightContainer::operator!= (const WeightContainer & *other*) const

inequality

Definition at line 100 of file HepMC-2.06.09/src/WeightContainer.cc.

7.75.4.25 WeightContainer & HepMC::WeightContainer::operator= (const WeightContainer & *in*) [inline]

copy assignment

best practices implementation

best practices implementation

Definition at line 154 of file WeightContainer.h.

7.75.4.26 WeightContainer& HepMC::WeightContainer::operator= (const WeightContainer &)

copy assignment

7.75.4.27 WeightContainer & HepMC::WeightContainer::operator= (const std::vector< double > & *in*) [inline]

alternate assignment using a vector of doubles

best practices implementation

best practices implementation

Definition at line 162 of file WeightContainer.h.

7.75.4.28 WeightContainer& HepMC::WeightContainer::operator= (const std::vector< double > & *in*)

alternate assignment using a vector of doubles

7.75.4.29 bool HepMC::WeightContainer::operator== (const WeightContainer & *other*) const

equality

Definition at line 92 of file HepMC-2.06.09/src/WeightContainer.cc.

References size().

7.75.4.30 bool HepMC::WeightContainer::operator== (const WeightContainer &) const

equality

7.75.4.31 double & HepMC::WeightContainer::operator[] (size_type *n*) [inline]

access the weight container

Definition at line 179 of file WeightContainer.h.

References n.

7.75.4.32 double& HepMC::WeightContainer::operator[] (size_type *n*)

access the weight container

7.75.4.33 const double& HepMC::WeightContainer::operator[] (size_type *n*) const

access the weight container

7.75.4.34 const double & HepMC::WeightContainer::operator[] (size_type *n*) const [inline]

access the weight container

Definition at line 182 of file WeightContainer.h.

References n.

7.75.4.35 double& HepMC::WeightContainer::operator[] (const std::string & *s*)

access the weight container

7.75.4.36 double & HepMC::WeightContainer::operator[] (const std::string & *s*)

access the weight container

Definition at line 66 of file HepMC-2.06.09/src/WeightContainer.cc.

7.75.4.37 const double& HepMC::WeightContainer::operator[] (const std::string & *s*) const

access the weight container

7.75.4.38 const double & HepMC::WeightContainer::operator[] (const std::string & *s*) const

access the weight container

Definition at line 80 of file HepMC-2.06.09/src/WeightContainer.cc.

7.75.4.39 void HepMC::WeightContainer::pop_back ( )

pop from weight container

7.75.4.40 void HepMC::WeightContainer::pop_back ( )

pop from weight container

Definition at line 51 of file HepMC-2.06.09/src/WeightContainer.cc.

References size().

Referenced by main().

7.75.4.41 void HepMC::WeightContainer::print (std::ostream & *ostr* = `std::cout`) const

print weights

7.75.4.42 void HepMC::WeightContainer::print (std::ostream & *ostr* = `std::cout`) const

print weights

Examples:

testMass.cc.in.

Definition at line 111 of file HepMC-2.06.09/src/WeightContainer.cc.

Referenced by main(), and HepMC::GenEvent::print().

7.75.4.43 void HepMC::WeightContainer::push_back (const double &)

push onto weight container

7.75.4.44 void HepMC::WeightContainer::push_back (const double & *value*)

push onto weight container

Examples:

fio/testPythiaCopies.cc, and **testHepMC.cc.in**.

Definition at line 42 of file HepMC-2.06.09/src/WeightContainer.cc.

References HepMC::Units::name().

Referenced by main(), writeWithCrossSection(), and writeWithWeight().

7.75.4.45 WeightContainer::size_type HepMC::WeightContainer::size ( ) const [inline]

size of weight container

Examples:

testMass.cc.in.

Definition at line 169 of file WeightContainer.h.

Referenced by main(), operator==(), pop_back(), HepMC::GenVertex::print(), HepMC::GenEvent::print(), WeightContainer(), HepMC::GenEvent::write(), and HepMC::IO_AsciiParticles::write_event().

7.75.4.46 size_type HepMC::WeightContainer::size ( ) const

size of weight container

7.75.4.47 void HepMC::WeightContainer::swap (WeightContainer & *other*) [inline]

swap

Definition at line 147 of file WeightContainer.h.

Referenced by HepMC::GenVertex::swap(), and HepMC::GenEvent::swap().

7.75.4.48 void HepMC::WeightContainer::swap (WeightContainer & *other*)

swap

7.75.4.49 void HepMC::WeightContainer::write (std::ostream & *ostr* = std::cout) const

write weights in a readable table

Definition at line 121 of file HepMC-2.06.09/src/WeightContainer.cc.

References begin(), end(), and HepMC::Units::name().

Referenced by main().

7.75.4.50 void HepMC::WeightContainer::write (std::ostream & *ostr* = std::cout) const

write weights in a readable table

7.75.5 Friends And Related Function Documentation

7.75.5.1 GenEvent [friend]

Definition at line 30 of file WeightContainer.h.

The documentation for this class was generated from the following files:

- **WeightContainer.h**
- **2.06.09/HepMC/WeightContainer.h**
- **HepMC-2.06.09/src/WeightContainer.cc**
- **src/WeightContainer.cc**

Chapter 8

File Documentation

8.1 CompareGenEvent.cc File Reference

```
#include <iostream>
#include "HepMC/CompareGenEvent.h"
#include "HepMC/GenEvent.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::compareGenEvent** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareSignalProcessVertex** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareBeamParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareWeights** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertices** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertex** (GenVertex ∗v1, GenVertex ∗v2)

8.2 CompareGenEvent.cc File Reference

```
#include <iostream>
#include "HepMC/CompareGenEvent.h"
#include "HepMC/GenEvent.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::compareGenEvent** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareSignalProcessVertex** (GenEvent ∗, GenEvent ∗)

- bool **HepMC::compareBeamParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareWeights** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertices** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertex** (GenVertex ∗v1, GenVertex ∗v2)

8.3 CompareGenEvent.h File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::compareGenEvent** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareSignalProcessVertex** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareBeamParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareWeights** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertices** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertex** (GenVertex ∗v1, GenVertex ∗v2)

8.4 CompareGenEvent.h File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::compareGenEvent** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareSignalProcessVertex** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareBeamParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareWeights** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertices** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareParticles** (GenEvent ∗, GenEvent ∗)
- bool **HepMC::compareVertex** (GenVertex ∗v1, GenVertex ∗v2)

8.5 defs.h File Reference

Macros

- #define **HAVE_DLFCN_H** 1
- #define **HAVE_INTTYPES_H** 1
- #define **HAVE_MEMORY_H** 1
- #define **HAVE_PTRDIFF_T** 1
- #define **HAVE_STDBOOL_H** 1
- #define **HAVE_STDINT_H** 1
- #define **HAVE_STDLIB_H** 1
- #define **HAVE_STRINGS_H** 1
- #define **HAVE_STRING_H** 1
- #define **HAVE_SYS_STAT_H** 1
- #define **HAVE_SYS_TYPES_H** 1
- #define **HAVE_UNISTD_H** 1
- #define **HAVE__BOOL** 1
- #define **LT_OBJDIR** ".libs/"
- #define **PACKAGE** "HepMC"
- #define **PACKAGE_BUGREPORT** "https://savannah.cern.ch/projects/hepmc/"
- #define **PACKAGE_NAME** "HepMC"
- #define **PACKAGE_STRING** "HepMC 2.06.09"
- #define **PACKAGE_TARNAME** "HepMC"
- #define **PACKAGE_URL** ""
- #define **PACKAGE_VERSION** "2.06.09"
- #define **STDC_HEADERS** 1
- #define **VERSION** "2.06.09"

8.5.1 Macro Definition Documentation

8.5.1.1 #define HAVE__BOOL 1

Definition at line 41 of file defs.h.

8.5.1.2 #define HAVE_DLFCN_H 1

Definition at line 5 of file defs.h.

8.5.1.3 #define HAVE_INTTYPES_H 1

Definition at line 8 of file defs.h.

8.5.1.4 #define HAVE_MEMORY_H 1

Definition at line 11 of file defs.h.

8.5.1.5 #define HAVE_PTRDIFF_T 1

Definition at line 14 of file defs.h.

8.5.1.6 #define HAVE_STDBOOL_H 1

Definition at line 17 of file defs.h.

8.5.1.7 #define HAVE_STDINT_H 1

Definition at line 20 of file defs.h.

8.5.1.8 #define HAVE_STDLIB_H 1

Definition at line 23 of file defs.h.

8.5.1.9 #define HAVE_STRING_H 1

Definition at line 29 of file defs.h.

8.5.1.10 #define HAVE_STRINGS_H 1

Definition at line 26 of file defs.h.

8.5.1.11 #define HAVE_SYS_STAT_H 1

Definition at line 32 of file defs.h.

8.5.1.12 #define HAVE_SYS_TYPES_H 1

Definition at line 35 of file defs.h.

8.5.1.13 #define HAVE_UNISTD_H 1

Definition at line 38 of file defs.h.

8.5.1.14 #define LT_OBJDIR ".libs/"

Definition at line 45 of file defs.h.

8.5.1.15 #define PACKAGE "HepMC"

Definition at line 48 of file defs.h.

8.5.1.16 #define PACKAGE_BUGREPORT "https://savannah.cern.ch/projects/hepmc/"

Definition at line 51 of file defs.h.

8.5.1.17 #define PACKAGE_NAME "HepMC"

Definition at line 54 of file defs.h.

8.5.1.18 #define PACKAGE_STRING "HepMC 2.06.09"

Definition at line 57 of file defs.h.

8.5.1.19 #define PACKAGE_TARNAME "HepMC"

Definition at line 60 of file defs.h.

8.5.1.20 #define PACKAGE_URL ""

Definition at line 63 of file defs.h.

8.5.1.21 #define PACKAGE_VERSION "2.06.09"

Definition at line 66 of file defs.h.

8.5.1.22 #define STDC_HEADERS 1

Definition at line 69 of file defs.h.

8.5.1.23 #define VERSION "2.06.09"

Definition at line 72 of file defs.h.

8.6 enable_if.h File Reference

Classes

- struct **HepMC::detail::enable_if< bool, class >**
  internal - used to decide if a class is arithmetic
- struct **HepMC::detail::enable_if< true, T >**
  internal - use if class T is arithmetic
- struct **HepMC::detail::disable_if< bool, class >**
  internal - used by SimpleVector to decide if a class is arithmetic
- struct **HepMC::detail::disable_if< false, T >**
  internal - used by SimpleVector to decide if a class is arithmetic

Namespaces

- **HepMC**
- **HepMC::detail**

8.7 enable_if.h File Reference

Classes

- struct **HepMC::detail::enable_if< bool, class >**
  internal - used to decide if a class is arithmetic

- struct **HepMC::detail::enable_if< true, T >**

  internal - use if class T is arithmetic
- struct **HepMC::detail::disable_if< bool, class >**

  internal - used by SimpleVector to decide if a class is arithmetic
- struct **HepMC::detail::disable_if< false, T >**

  internal - used by SimpleVector to decide if a class is arithmetic

Namespaces

- **HepMC**
- **HepMC::detail**

8.8 example_BuildEventFromScratch.cc File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Functions

- int **main** ()

8.8.1 Function Documentation

8.8.1.1 int main ( )

Examples:

example_BuildEventFromScratch.cc, **example_EventSelection.cc**, **example_MyPythiaOnlyToHepMC.cc**, **example_UsingIterators.cc**, **example_VectorConversion.cc**, **fio/example_MyHerwig.cc**, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, **fio/testHerwigCopies.cc**, **fio/testPythiaCopies.cc**, **testFlow.cc**, **testHepMC.cc.in**, **testHepMCIteration.cc.in**, **testMass.cc.in**, **testMultipleCopies.cc.in**, **testPrintBug.cc**, **testSimpleVector.cc**, **testStreamIO.cc.in**, and **testUnits.cc**.

Definition at line 22 of file examples/example_BuildEventFromScratch.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::Units::GEV, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::GenEvent::use_units().

8.9 example_BuildEventFromScratch.cc File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Functions

- int **main** ()

8.9.1 Function Documentation

8.9.1.1 int main ( )

Definition at line 22 of file HepMC-2.06.09/examples/example_BuildEventFromScratch.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::Units::GEV, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::GenEvent::use_units().

8.10 example_EventSelection.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
```

Classes

- class **IsEventGood**

  example class

Functions

- int **main** ()

8.10.1 Function Documentation

8.10.1.1 int main ( )

Definition at line 37 of file examples/example_EventSelection.cc.

References HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

8.11 example_EventSelection.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
```

Classes

- class **IsEventGood**

  example class

Functions

- int **main** ()

8.11.1 Function Documentation

8.11.1.1 int main ( )

Definition at line 37 of file HepMC-2.06.09/examples/example_EventSelection.cc.

References HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

8.12 example_MyHerwig.cc File Reference

```
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Functions

- void **hwaend_** ()
- int **main** ()

8.12.1 Function Documentation

8.12.1.1 void hwaend_ ( )

To Compile: go to the **HepMC** (p. 15) directory and type: gmake examples/example_MyHerwig.exe

In this example the precision and number of entries for the HEPEVT fortran common block are explicitly defined to correspond to those used in the Herwig version of the HEPEVT common block. If you get funny output from HEPEVT in your own code, probably you have set these values incorrectly!

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 24 of file examples/fio/example_MyHerwig.cc.

8.12.1.2 int main ( )

Definition at line 26 of file examples/fio/example_MyHerwig.cc.

References HepMC::getHerwigCrossSection(), HepMC::Units::GEV, hwbgen, hwbmch, hwcdec, hwcfor, hwdhad, hwdhob, hwdhvy, hwefin, hweini, hwepro, hwevnt, hwigin, hwmevt, hwproc, hwufne, hwuinc, hwuine, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::HEPEVT_Wrapper::print_hepevt(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.13 example_MyHerwig.cc File Reference

```
#include <iostream>
```

```
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Functions

- void **hwaend_** ()
- int **main** ()

8.13.1 Function Documentation

8.13.1.1 void hwaend_ ( )

To Compile: go to the **HepMC** (p. 15) directory and type: gmake examples/example_MyHerwig.exe

In this example the precision and number of entries for the HEPEVT fortran common block are explicitly defined to correspond to those used in the Herwig version of the HEPEVT common block. If you get funny output from HEPEVT in your own code, probably you have set these values incorrectly!

Definition at line 24 of file HepMC-2.06.09/examples/fio/example_MyHerwig.cc.

8.13.1.2 int main ( )

Definition at line 26 of file HepMC-2.06.09/examples/fio/example_MyHerwig.cc.

References HepMC::getHerwigCrossSection(), HepMC::Units::GEV, hwbgen, hwbmch, hwcdec, hwcfor, hwdhad, hwdhob, hwdhvy, hwefin, hweini, hwepro, hwevnt, hwigin, hwmevt, hwproc, hwufne, hwuinc, hwuine, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::HEPEVT_Wrapper::print_hepevt(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.14 example_MyPythia.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Classes

- class **IsGoodEventMyPythia**

  example class

Functions

- void **pythia_out** ()

- void **pythia_in** ()
- void **pythia_in_out** ()
- void **event_selection** ()
- void **pythia_particle_out** ()
- int **main** ()

8.14.1 Function Documentation

8.14.1.1 void event_selection ( )

Examples:

fio/example_MyPythia.cc.

Definition at line 152 of file examples/fio/example_MyPythia.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

Referenced by main().

8.14.1.2 int main ( )

Definition at line 85 of file examples/fio/example_MyPythia.cc.

References event_selection(), pythia_in(), pythia_in_out(), and pythia_out().

8.14.1.3 void pythia_in ( )

Examples:

fio/example_MyPythia.cc.

Definition at line 205 of file examples/fio/example_MyPythia.cc.

References HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

Referenced by main().

8.14.1.4 void pythia_in_out ( )

Examples:

fio/example_MyPythia.cc.

Definition at line 239 of file examples/fio/example_MyPythia.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::GenEvent::event_number(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

Referenced by main().

8.14.1.5 void pythia_out ( )

Examples:

fio/example_MyPythia.cc.

Definition at line 99 of file examples/fio/example_MyPythia.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

Referenced by main().

8.14.1.6 void pythia_particle_out ( )

Examples:

fio/example_MyPythia.cc.

Definition at line 311 of file examples/fio/example_MyPythia.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.15 example_MyPythia.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Classes

- class **IsGoodEventMyPythia**

  example class

Functions

- void **pythia_out** ()
- void **pythia_in** ()
- void **pythia_in_out** ()
- void **event_selection** ()
- void **pythia_particle_out** ()
- int **main** ()

8.15.1 Function Documentation

8.15.1.1 void event_selection ( )

8.15.1.2 int main ( )

Definition at line 85 of file HepMC-2.06.09/examples/fio/example_MyPythia.cc.

References event_selection(), pythia_in(), pythia_in_out(), and pythia_out().

8.15.1.3 void pythia_in ( )

8.15.1.4 void pythia_in_out ( )

8.15.1.5 void pythia_out ( )

8.15.1.6 void pythia_particle_out ( )

8.16 example_MyPythiaOnlyToHepMC.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

8.16.1 Function Documentation

8.16.1.1 int main ( )

Definition at line 23 of file examples/fio/example_MyPythiaOnlyToHepMC.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.17 example_MyPythiaOnlyToHepMC.cc File Reference

```
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

8.17.1 Function Documentation

8.17.1.1 int main ( )

Definition at line 23 of file HepMC-2.06.09/examples/fio/example_MyPythiaOnlyToHepMC.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.18 example_PythiaStreamIO.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- void **writePythiaStreamIO** ()
- void **readPythiaStreamIO** ()
- int **main** ()

8.18.1 Function Documentation

8.18.1.1 int main ( )

Definition at line 31 of file examples/fio/example_PythiaStreamIO.cc.

References readPythiaStreamIO(), and writePythiaStreamIO().

8.18.1.2 void readPythiaStreamIO ( )

Examples:

fio/example_PythiaStreamIO.cc.

Definition at line 103 of file examples/fio/example_PythiaStreamIO.cc.

References HepMC::GenCrossSection::cross_section(), HepMC::GenEvent::cross_section(), HepMC::GenEvent::is_valid(), HepMC::GenEvent::read(), HepMC::GenEvent::write(), HepMC::write_HepMC_IO_block_begin(), and HepMC::write_HepMC_IO_block_end().

Referenced by main().

8.18.1.3 void writePythiaStreamIO ( )

example of generating events with Pythia using HepMC/PythiaWrapper.h Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy

To Compile: go to the **HepMC** (p. 15) example directory and type: make example_PythiaStreamIO.exe

This example uses streaming I/O **writePythiaStreamIO()** (p. 316) sets the cross section in GenRun **readPythiaStreamIO()** (p. 315) reads the file written by **writePythiaStreamIO()** (p. 316)

Examples:

fio/example_PythiaStreamIO.cc.

Definition at line 40 of file examples/fio/example_PythiaStreamIO.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), HepMC::GenEvent::use_units(), HepMC::write_HepMC_IO_block_begin(), and HepMC::write_HepMC_IO_block_end().

Referenced by main().

8.19 example_PythiaStreamIO.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"
```

Functions

- void **writePythiaStreamIO** ()
- void **readPythiaStreamIO** ()
- int **main** ()

8.19.1 Function Documentation

8.19.1.1 int main ( )

Definition at line 31 of file HepMC-2.06.09/examples/fio/example_PythiaStreamIO.cc.

References readPythiaStreamIO(), and writePythiaStreamIO().

8.19.1.2 void readPythiaStreamIO ( )

8.19.1.3 void writePythiaStreamIO ( )

example of generating events with Pythia using HepMC/PythiaWrapper.h Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy

To Compile: go to the **HepMC** (p. 15) example directory and type: make example_PythiaStreamIO.exe

This example uses streaming I/O **writePythiaStreamIO()** (p. 316) sets the cross section in GenRun **readPythiaStreamIO()** (p. 315) reads the file written by **writePythiaStreamIO()** (p. 316)

8.20 example_UsingIterators.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include <math.h>
#include <algorithm>
#include <list>
```

Classes

- class **IsPhoton**
  example class
- class **IsW_Boson**
  example class
- class **IsStateFinal**
  example class

Functions

- int **main** ()

8.20.1 Function Documentation

8.20.1.1 int main ( )

Definition at line 56 of file examples/example_UsingIterators.cc.

References HepMC::copy_if(), HepMC::descendants, p, HepMC::parents, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::IO_GenEvent::rdstate(), HepMC::IO_BaseClass::read_next_event(), v, HepMC::GenEvent::vertices_begin(), and HepMC::GenEvent::vertices_end().

8.21 example_UsingIterators.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include <math.h>
#include <algorithm>
#include <list>
```

Classes

- class **IsPhoton**
  example class
- class **IsW_Boson**
  example class

- class **IsStateFinal**

  example class

Functions

- int **main** ()

8.21.1 Function Documentation

8.21.1.1 int main ( )

Definition at line 56 of file HepMC-2.06.09/examples/example_UsingIterators.cc.

References HepMC::copy_if(), HepMC::descendants, p, HepMC::parents, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::IO_GenEvent::rdstate(), HepMC::IO_BaseClass::read_next_event(), v, HepMC::GenEvent::vertices_begin(), and HepMC::GenEvent::vertices_end().

8.22 example_VectorConversion.cc File Reference

```
#include <iostream>
#include "VectorConversion.h"
#include "HepMC/GenEvent.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Functions

- int **main** ()

8.22.1 Function Documentation

8.22.1.1 int main ( )

Definition at line 25 of file examples/example_VectorConversion.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), convertTo(), HepMC::Units::GEV, HepMC::Units::MM, p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::print(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::GenEvent::use_units().

8.23 example_VectorConversion.cc File Reference

```
#include <iostream>
#include "VectorConversion.h"
#include "HepMC/GenEvent.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Functions

- int **main** ()

8.23.1 Function Documentation

8.23.1.1 int main ( )

Definition at line 25 of file HepMC-2.06.09/examples/example_VectorConversion.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), convertTo(), HepMC::Units::GEV, HepMC::Units::MM, p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::print(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::GenEvent::use_units().

8.24 filterEvent.cc File Reference

```
#include "HepMC/GenEvent.h"
```

Functions

- void **filterEvent** (**HepMC::GenEvent** ∗ge)

8.24.1 Function Documentation

8.24.1.1 void filterEvent (HepMC::GenEvent ∗ *ge*)

Todo Have to build a list, since the GV::add_particle_out method modifies the end vertex!

Todo Why does this cause an error?

Definition at line 5 of file filterEvent.cc.

References HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::beam_particles(), HepMC::GenParticle::end_vertex(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenVertex::particles_out_const_begin(), HepMC::GenVertex::particles_out_const_end(), HepMC::GenVertex::particles_out_size(), HepMC::GenParticle::production_vertex(), HepMC::GenVertex::remove_particle(), HepMC::GenEvent::vertices_begin(), and HepMC::GenEvent::vertices_end().

8.25 Flow.cc File Reference

```
#include "HepMC/Flow.h"
#include "HepMC/GenParticle.h"
#include "HepMC/GenVertex.h"
#include "HepMC/SearchVector.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const Flow &f)

  *send **Flow*** (p. *44) informatin to ostr for printing*

8.26 Flow.cc File Reference

```
#include "HepMC/Flow.h"
#include "HepMC/GenParticle.h"
#include "HepMC/GenVertex.h"
#include "HepMC/SearchVector.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const Flow &f)

  *send **Flow*** (p. *44) informatin to ostr for printing*

8.27 Flow.h File Reference

```
#include <iostream>
#include <map>
#include <vector>
```

Classes

- class **HepMC::Flow**

  The flow object.

Namespaces

- **HepMC**

8.28 Flow.h File Reference

```
#include <iostream>
#include <map>
#include <vector>
```

Classes

- class **HepMC::Flow**

  The flow object.

Namespaces

- **HepMC**

8.29 GenCrossSection.cc File Reference

```
#include <iostream>
#include <sstream>
#include "HepMC/GenCrossSection.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**

8.30 GenCrossSection.cc File Reference

```
#include <iostream>
#include <sstream>
#include "HepMC/GenCrossSection.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**

8.31 GenCrossSection.h File Reference

```
#include <iostream>
```

Classes

- class **HepMC::GenCrossSection**

  *The **GenCrossSection*** *(p. 67) class stores the generated cross section.*

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &os, GenCrossSection &xs)
- std::istream & **HepMC::operator**>> (std::istream &is, GenCrossSection &xs)

8.32 GenCrossSection.h File Reference

```
#include <iostream>
```

Classes

- class **HepMC::GenCrossSection**

  *The **GenCrossSection** (p. 67) class stores the generated cross section.*

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &os, GenCrossSection &xs)
- std::istream & **HepMC::operator**>> (std::istream &is, GenCrossSection &xs)

8.33 GenEvent.cc File Reference

```
#include <iomanip>
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/Version.h"
#include "HepMC/StreamHelpers.h"
```

Namespaces

- **HepMC**

8.34 GenEvent.cc File Reference

```
#include <iomanip>
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/Version.h"
#include "HepMC/StreamHelpers.h"
```

Namespaces

- **HepMC**

8.35 GenEvent.h File Reference

```
#include "HepMC/GenVertex.h"
```

```
#include "HepMC/GenParticle.h"
#include "HepMC/WeightContainer.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/HeavyIon.h"
#include "HepMC/PdfInfo.h"
#include "HepMC/Units.h"
#include "HepMC/HepMCDefs.h"
#include <map>
#include <string>
#include <vector>
#include <algorithm>
#include <iostream>
```

Classes

- class **HepMC::GenEvent**

  *The **GenEvent** (p. 72) class is the core of **HepMC** (p. 15).*
- class **HepMC::GenEvent::vertex_const_iterator**

  const vertex iterator
- class **HepMC::GenEvent::vertex_iterator**

  non-const vertex iterator
- class **HepMC::GenEvent::particle_const_iterator**

  const particle iterator
- class **HepMC::GenEvent::particle_iterator**

  non-const particle iterator

Namespaces

- **HepMC**

Functions

- template<class InputIterator , class OutputIterator , class Predicate >
  void **HepMC::copy_if** (InputIterator first, InputIterator last, OutputIterator out, Predicate pred)

  define the type of iterator to use
- std::ostream & **HepMC::operator<<** (std::ostream &, GenEvent &)

  standard streaming IO output operator
- std::istream & **HepMC::operator>>** (std::istream &, GenEvent &)

  standard streaming IO input operator
- std::istream & **HepMC::set_input_units** (std::istream &, Units::MomentumUnit, Units::LengthUnit)

  set the units for this input stream
- std::ostream & **HepMC::write_HepMC_IO_block_begin** (std::ostream &)

  *Explicitly write the begin block lines that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **HepMC::write_HepMC_IO_block_end** (std::ostream &)

  *Explicitly write the end block line that **IO_GenEvent** (p. 186) uses.*
- GenEvent & **HepMC::convert_units** (GenEvent &evt, Units::MomentumUnit m, Units::LengthUnit l)

8.36 GenEvent.h File Reference

```
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include "HepMC/WeightContainer.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/HeavyIon.h"
#include "HepMC/PdfInfo.h"
#include "HepMC/Units.h"
#include "HepMC/HepMCDefs.h"
#include <map>
#include <string>
#include <vector>
#include <algorithm>
#include <iostream>
```

Classes

- class **HepMC::GenEvent**

  *The **GenEvent** (p. 72) class is the core of **HepMC** (p. 15).*
- class **HepMC::GenEvent::vertex_const_iterator**

  const vertex iterator
- class **HepMC::GenEvent::vertex_iterator**

  non-const vertex iterator
- class **HepMC::GenEvent::particle_const_iterator**

  const particle iterator
- class **HepMC::GenEvent::particle_iterator**

  non-const particle iterator

Namespaces

- **HepMC**

Functions

- template<class InputIterator , class OutputIterator , class Predicate >
  void **HepMC::copy_if** (InputIterator first, InputIterator last, OutputIterator out, Predicate pred)

  define the type of iterator to use
- std::ostream & **HepMC::operator**<< (std::ostream &, GenEvent &)

  standard streaming IO output operator
- std::istream & **HepMC::operator**>> (std::istream &, GenEvent &)

  standard streaming IO input operator
- std::istream & **HepMC::set_input_units** (std::istream &, Units::MomentumUnit, Units::LengthUnit)

  set the units for this input stream
- std::ostream & **HepMC::write_HepMC_IO_block_begin** (std::ostream &)

  *Explicitly write the begin block lines that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **HepMC::write_HepMC_IO_block_end** (std::ostream &)

  *Explicitly write the end block line that **IO_GenEvent** (p. 186) uses.*
- GenEvent & **HepMC::convert_units** (GenEvent &evt, Units::MomentumUnit m, Units::LengthUnit l)

8.37 GenEventStreamIO.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/StreamInfo.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/Version.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**
- **HepMC::detail**

Functions

- void **HepMC::HepMCStreamCallback** (std::ios_base::event e, std::ios_base &b, int i)
- template<class IO >
  StreamInfo & **HepMC::get_stream_info** (IO &iost)
- std::ostream & **HepMC::operator<<** (std::ostream &, GenEvent &)
  standard streaming IO output operator
- std::istream & **HepMC::operator>>** (std::istream &, GenEvent &)
  standard streaming IO input operator
- std::istream & **HepMC::set_input_units** (std::istream &, Units::MomentumUnit, Units::LengthUnit)
  set the units for this input stream
- std::ostream & **HepMC::write_HepMC_IO_block_begin** (std::ostream &)
  *Explicitly write the begin block lines that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **HepMC::write_HepMC_IO_block_end** (std::ostream &)
  *Explicitly write the end block line that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **HepMC::establish_output_stream_info** (std::ostream &os)
- std::istream & **HepMC::establish_input_stream_info** (std::istream &is)
- std::istream & **HepMC::detail::read_particle** (std::istream &, TempParticleMap &, GenParticle *)
- std::ostream & **HepMC::detail::establish_output_stream_info** (std::ostream &)
  *used by **IO_GenEvent** (p. 186) constructor*
- std::istream & **HepMC::detail::establish_input_stream_info** (std::istream &)
  *used by **IO_GenEvent** (p. 186) constructor*

8.38 GenEventStreamIO.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/StreamInfo.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/Version.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**
- **HepMC::detail**

Functions

- void **HepMC::HepMCStreamCallback** (std::ios_base::event e, std::ios_base &b, int i)
- template<class IO >
  StreamInfo & **HepMC::get_stream_info** (IO &iost)
- std::ostream & **HepMC::operator**<< (std::ostream &, GenEvent &)

  standard streaming IO output operator
- std::istream & **HepMC::operator**>> (std::istream &, GenEvent &)

  standard streaming IO input operator
- std::istream & **HepMC::set_input_units** (std::istream &, Units::MomentumUnit, Units::LengthUnit)

  set the units for this input stream
- std::ostream & **HepMC::write_HepMC_IO_block_begin** (std::ostream &)

  *Explicitly write the begin block lines that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **HepMC::write_HepMC_IO_block_end** (std::ostream &)

  *Explicitly write the end block line that **IO_GenEvent** (p. 186) uses.*
- std::ostream & **HepMC::establish_output_stream_info** (std::ostream &os)
- std::istream & **HepMC::establish_input_stream_info** (std::istream &is)
- std::istream & **HepMC::detail::read_particle** (std::istream &, TempParticleMap &, GenParticle *)
- std::ostream & **HepMC::detail::establish_output_stream_info** (std::ostream &)

  *used by **IO_GenEvent** (p. 186) constructor*
- std::istream & **HepMC::detail::establish_input_stream_info** (std::istream &)

  *used by **IO_GenEvent** (p. 186) constructor*

8.39 GenParticle.cc File Reference

```
#include "HepMC/GenEvent.h"
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include <iomanip>
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const GenParticle &part)

  Dump this particle's full info to ostr.

8.40 GenParticle.cc File Reference

```
#include "HepMC/GenEvent.h"
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include <iomanip>
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const GenParticle &part)

  Dump this particle's full info to ostr.

8.41 GenParticle.h File Reference

```
#include "HepMC/Flow.h"
#include "HepMC/Polarization.h"
#include "HepMC/SimpleVector.h"
#include "HepMC/IteratorRange.h"
#include <iostream>
#include <stdint.h>
```

Classes

- class **HepMC::GenParticle**

  *The **GenParticle*** (p. *106*) *class contains information about generated particles.*

Namespaces

- **HepMC**

Macros

- #define **hepmc_uint64_t** uint64_t

8.41.1 Macro Definition Documentation

8.41.1.1 #define hepmc_uint64_t uint64_t

Definition at line 38 of file GenParticle.h.

8.42 GenParticle.h File Reference

```
#include "HepMC/Flow.h"
#include "HepMC/Polarization.h"
#include "HepMC/SimpleVector.h"
#include "HepMC/IteratorRange.h"
#include <iostream>
#include <stdint.h>
```

Classes

- class **HepMC::GenParticle**

  *The **GenParticle** (p. 106) class contains information about generated particles.*

Namespaces

- **HepMC**

Macros

- #define **hepmc_uint64_t** uint64_t

8.42.1 Macro Definition Documentation

8.42.1.1 #define hepmc_uint64_t uint64_t

Definition at line 38 of file 2.06.09/HepMC/GenParticle.h.

8.43 GenRanges.cc File Reference

```
#include <iostream>
#include "HepMC/GenRanges.h"
#include "HepMC/GenEvent.h"
#include "HepMC/GenVertex.h"
```

Namespaces

- **HepMC**

8.44 GenRanges.cc File Reference

```
#include <iostream>
#include "HepMC/GenRanges.h"
#include "HepMC/GenEvent.h"
#include "HepMC/GenVertex.h"
```

Namespaces

- **HepMC**

8.45 GenRanges.h File Reference

```
#include <stdexcept>
#include "HepMC/GenEvent.h"
#include "HepMC/GenVertex.h"
```

Classes

- class **HepMC::GenEventVertexRange**

  GenEventVertexRange *(*p. *104) acts like a collection of vertices.*
- class **HepMC::ConstGenEventVertexRange**

  ConstGenEventVertexRange *(*p. *34) acts like a collection of vertices.*
- class **HepMC::GenEventParticleRange**

  GenEventParticleRange *(*p. *103) acts like a collection of particles.*
- class **HepMC::ConstGenEventParticleRange**

  ConstGenEventParticleRange *(*p. *33) acts like a collection of particles.*
- class **HepMC::GenVertexParticleRange**

  GenVertexParticleRange *(*p. *144) acts like a collection of particles.*
- class **HepMC::GenParticleProductionRange**

  GenParticleProductionRange *(*p. *123) acts like a collection of particles.*
- class **HepMC::ConstGenParticleProductionRange**
- class **HepMC::GenParticleEndRange**

  GenParticleEndRange *(*p. *122) acts like a collection of particles.*
- class **HepMC::ConstGenParticleEndRange**

Namespaces

- **HepMC**

8.46 GenRanges.h File Reference

```
#include <stdexcept>
#include "HepMC/GenEvent.h"
#include "HepMC/GenVertex.h"
```

Classes

- class **HepMC::GenEventVertexRange**

  GenEventVertexRange *(*p. *104) acts like a collection of vertices.*
- class **HepMC::ConstGenEventVertexRange**

  ConstGenEventVertexRange *(*p. *34) acts like a collection of vertices.*
- class **HepMC::GenEventParticleRange**

  GenEventParticleRange *(*p. *103) acts like a collection of particles.*

- class **HepMC::ConstGenEventParticleRange**

  ConstGenEventParticleRange *(*p. *33) acts like a collection of particles.*
- class **HepMC::GenVertexParticleRange**

  GenVertexParticleRange *(*p. *144) acts like a collection of particles.*
- class **HepMC::GenParticleProductionRange**

  GenParticleProductionRange *(*p. *123) acts like a collection of particles.*
- class **HepMC::ConstGenParticleProductionRange**
- class **HepMC::GenParticleEndRange**

  GenParticleEndRange *(*p. *122) acts like a collection of particles.*
- class **HepMC::ConstGenParticleEndRange**

Namespaces

- **HepMC**

8.47 GenVertex.cc File Reference

```
#include "HepMC/GenParticle.h"
#include "HepMC/GenVertex.h"
#include "HepMC/GenEvent.h"
#include "HepMC/SearchVector.h"
#include <iomanip>
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const GenVertex &vtx)

  send vertex information to ostr for printing

8.48 GenVertex.cc File Reference

```
#include "HepMC/GenParticle.h"
#include "HepMC/GenVertex.h"
#include "HepMC/GenEvent.h"
#include "HepMC/SearchVector.h"
#include <iomanip>
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const GenVertex &vtx)

  send vertex information to ostr for printing

8.49 GenVertex.h File Reference

```
#include "HepMC/WeightContainer.h"
#include "HepMC/SimpleVector.h"
#include "HepMC/IteratorRange.h"
#include <iostream>
#include <iterator>
#include <vector>
#include <set>
#include <algorithm>
#include <cstddef>
```

Classes

- class **HepMC::GenVertex**

  GenVertex *(*p. *125) contains information about decay vertices.*
- class **HepMC::GenVertex::edge_iterator**

  edge iterator
- class **HepMC::GenVertex::vertex_iterator**

  vertex iterator
- class **HepMC::GenVertex::particle_iterator**

  particle iterator

Namespaces

- **HepMC**

8.50 GenVertex.h File Reference

```
#include "HepMC/WeightContainer.h"
#include "HepMC/SimpleVector.h"
#include "HepMC/IteratorRange.h"
#include <iostream>
#include <iterator>
#include <vector>
#include <set>
#include <algorithm>
#include <cstddef>
```

Classes

- class **HepMC::GenVertex**

  GenVertex *(*p. *125) contains information about decay vertices.*
- class **HepMC::GenVertex::edge_iterator**

  edge iterator
- class **HepMC::GenVertex::vertex_iterator**

  vertex iterator
- class **HepMC::GenVertex::particle_iterator**

  particle iterator

Namespaces

- **HepMC**

8.51 HeavyIon.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/HeavyIon.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, HeavyIon const *)

  *Write the contents of **HeavyIon*** (p. *146*) *to an output stream.*
- std::istream & **HepMC::operator**>> (std::istream &, HeavyIon *)

  *Read the contents of **HeavyIon*** (p. *146*) *from an input stream.*

8.52 HeavyIon.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/HeavyIon.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, HeavyIon const *)

  *Write the contents of **HeavyIon*** (p. *146*) *to an output stream.*
- std::istream & **HepMC::operator**>> (std::istream &, HeavyIon *)

  *Read the contents of **HeavyIon*** (p. *146*) *from an input stream.*

8.53 Heavylon.h File Reference

```
#include <ostream>
#include <istream>
#include <utility>
```

Classes

- class **HepMC::Heavylon**

  *The **Heavylon** (p. 146) class stores information about heavy ions.*

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, Heavylon const *)

  *Write the contents of **Heavylon** (p. 146) to an output stream.*
- std::istream & **HepMC::operator**>> (std::istream &, Heavylon *)

  *Read the contents of **Heavylon** (p. 146) from an input stream.*

8.54 Heavylon.h File Reference

```
#include <ostream>
#include <istream>
#include <utility>
```

Classes

- class **HepMC::Heavylon**

  *The **Heavylon** (p. 146) class stores information about heavy ions.*

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, Heavylon const *)

  *Write the contents of **Heavylon** (p. 146) to an output stream.*
- std::istream & **HepMC::operator**>> (std::istream &, Heavylon *)

  *Read the contents of **Heavylon** (p. 146) from an input stream.*

8.55 HEPEVT_Wrapper.cc File Reference

```
#include "HepMC/HEPEVT_Wrapper.h"
```

Namespaces

- **HepMC**

8.56 HEPEVT_Wrapper.cc File Reference

```
#include "HepMC/HEPEVT_Wrapper.h"
```

Namespaces

- **HepMC**

8.57 HEPEVT_Wrapper.h File Reference

```
#include <ctype.h>
#include <iostream>
#include <cstdio>
```

Classes

- class **HepMC::HEPEVT_Wrapper**

  Generic Wrapper for the fortran HEPEVT common block.

Namespaces

- **HepMC**

Macros

- #define **HEPEVT_EntriesAllocation** 10000
- #define **hepevt hepevt_**
- #define **HEPMC_HEPEVT_WRAPPER_H**

Variables

- const unsigned int **hepevt_bytes_allocation**
- struct {
  char **data** [**hepevt_bytes_allocation**]
  } **hepevt_**

8.57.1 Macro Definition Documentation

8.57.1.1 #define hepevt hepevt_

Definition at line 84 of file HEPEVT_Wrapper.h.

Referenced by HepMC::HEPEVT_Wrapper::byte_num_to_double(), HepMC::HEPEVT_Wrapper::byte_num_to_int(), and HepMC::HEPEVT_Wrapper::write_byte_num().

8.57.1.2 #define HEPEVT_EntriesAllocation 10000

Definition at line 4 of file HEPEVT_Wrapper.h.

8.57.1.3 #define HEPMC_HEPEVT_WRAPPER_H

Definition at line 92 of file HEPEVT_Wrapper.h.

8.57.2 Variable Documentation

8.57.2.1 char data[hepevt_bytes_allocation]

Definition at line 81 of file HEPEVT_Wrapper.h.

8.57.2.2 struct { ... } hepevt_

8.57.2.3 const unsigned int hepevt_bytes_allocation

Initial value:

```
=
                sizeof(long int) * ( 2 + 6 * HEPEVT_EntriesAllocation )
                + sizeof(double) * ( 9 * HEPEVT_EntriesAllocation )
```

Definition at line 66 of file HEPEVT_Wrapper.h.

Referenced by HepMC::HEPEVT_Wrapper::byte_num_to_double(), HepMC::HEPEVT_Wrapper::byte_num_to_int(), and HepMC::HEPEVT_Wrapper::write_byte_num().

8.58 HEPEVT_Wrapper.h File Reference

```
#include <ctype.h>
#include <iostream>
#include <cstdio>
```

Classes

- class **HepMC::HEPEVT_Wrapper**

  Generic Wrapper for the fortran HEPEVT common block.

Namespaces

- **HepMC**

Macros

- #define **HEPEVT_EntriesAllocation** 10000
- #define **hepevt hepevt_**
- #define **HEPMC_HEPEVT_WRAPPER_H**

Variables

- const unsigned int **hepevt_bytes_allocation**
- struct {
  char **data** [**hepevt_bytes_allocation**]
  } **hepevt_**

8.58.1 Macro Definition Documentation

8.58.1.1 #define hepevt hepevt_

Definition at line 84 of file 2.06.09/HepMC/HEPEVT_Wrapper.h.

8.58.1.2 #define HEPEVT_EntriesAllocation 10000

Definition at line 4 of file 2.06.09/HepMC/HEPEVT_Wrapper.h.

8.58.1.3 #define HEPMC_HEPEVT_WRAPPER_H

Definition at line 92 of file 2.06.09/HepMC/HEPEVT_Wrapper.h.

8.58.2 Variable Documentation

8.58.2.1 char data[hepevt_bytes_allocation]

Definition at line 81 of file 2.06.09/HepMC/HEPEVT_Wrapper.h.

8.58.2.2 struct { ... } hepevt_

8.58.2.3 const unsigned int hepevt_bytes_allocation

Initial value:

```
=
                sizeof(long int) * ( 2 + 6 * HEPEVT_EntriesAllocation )
                + sizeof(double) * ( 9 * HEPEVT_EntriesAllocation )
```

Definition at line 66 of file 2.06.09/HepMC/HEPEVT_Wrapper.h.

8.59 HepMCDefs.h File Reference

Macros

- #define **HEPMC_HAS_HEAVY_ION**

- #define **HEPMC_IO_ASCII_REMOVED**
- #define **HEPMC_PARTICLE_DATA_REMOVED**
- #define **HEPMC_HAS_IO_GENEVENT**
- #define **HEPMC_HAS_PDF_INFO**
- #define **HEPMC_HAS_SIMPLE_VECTOR**
- #define **HEPMC_HAS_UNITS**
- #define **HEPMC_HAS_CROSS_SECTION**
- #define **HEPMC_HAS_ITERATOR_RANGES**
- #define **HEPMC_HAS_NAMED_WEIGHTS**
- #define **HEPMC_VERSION** "2.06.09"

8.59.1 Macro Definition Documentation

8.59.1.1 #define HEPMC_HAS_CROSS_SECTION

Definition at line 50 of file HepMCDefs.h.

8.59.1.2 #define HEPMC_HAS_HEAVY_ION

Definition at line 15 of file HepMCDefs.h.

8.59.1.3 #define HEPMC_HAS_IO_GENEVENT

Definition at line 30 of file HepMCDefs.h.

8.59.1.4 #define HEPMC_HAS_ITERATOR_RANGES

Definition at line 55 of file HepMCDefs.h.

8.59.1.5 #define HEPMC_HAS_NAMED_WEIGHTS

Definition at line 60 of file HepMCDefs.h.

8.59.1.6 #define HEPMC_HAS_PDF_INFO

Definition at line 35 of file HepMCDefs.h.

8.59.1.7 #define HEPMC_HAS_SIMPLE_VECTOR

Definition at line 40 of file HepMCDefs.h.

8.59.1.8 #define HEPMC_HAS_UNITS

Definition at line 45 of file HepMCDefs.h.

8.59.1.9 #define HEPMC_IO_ASCII_REMOVED

Definition at line 20 of file HepMCDefs.h.

8.59.1.10 #define HEPMC_PARTICLE_DATA_REMOVED

Definition at line 25 of file HepMCDefs.h.

8.59.1.11 #define HEPMC_VERSION "2.06.09"

Definition at line 65 of file HepMCDefs.h.

Referenced by HepMC::versionName().

8.60 HepMCDefs.h File Reference

Macros

- #define **HEPMC_HAS_HEAVY_ION**
- #define **HEPMC_IO_ASCII_REMOVED**
- #define **HEPMC_PARTICLE_DATA_REMOVED**
- #define **HEPMC_HAS_IO_GENEVENT**
- #define **HEPMC_HAS_PDF_INFO**
- #define **HEPMC_HAS_SIMPLE_VECTOR**
- #define **HEPMC_HAS_UNITS**
- #define **HEPMC_HAS_CROSS_SECTION**
- #define **HEPMC_HAS_ITERATOR_RANGES**
- #define **HEPMC_HAS_NAMED_WEIGHTS**
- #define **HEPMC_VERSION** "2.06.09"

8.60.1 Macro Definition Documentation

8.60.1.1 #define HEPMC_HAS_CROSS_SECTION

Definition at line 50 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.2 #define HEPMC_HAS_HEAVY_ION

Definition at line 15 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.3 #define HEPMC_HAS_IO_GENEVENT

Definition at line 30 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.4 #define HEPMC_HAS_ITERATOR_RANGES

Definition at line 55 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.5 #define HEPMC_HAS_NAMED_WEIGHTS

Definition at line 60 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.6 #define HEPMC_HAS_PDF_INFO

Definition at line 35 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.7 #define HEPMC_HAS_SIMPLE_VECTOR

Definition at line 40 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.8 #define HEPMC_HAS_UNITS

Definition at line 45 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.9 #define HEPMC_IO_ASCII_REMOVED

Definition at line 20 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.10 #define HEPMC_PARTICLE_DATA_REMOVED

Definition at line 25 of file 2.06.09/HepMC/HepMCDefs.h.

8.60.1.11 #define HEPMC_VERSION "2.06.09"

Definition at line 65 of file 2.06.09/HepMC/HepMCDefs.h.

8.61 HerwigWrapper.cc File Reference

```
#include <cmath>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/GenCrossSection.h"
```

Namespaces

- **HepMC**

Functions

- GenCrossSection **HepMC::getHerwigCrossSection** (int ngen)

  calculate the Herwig cross section and statistical error

Variables

- struct **hwgev hwevnt_**

8.61.1 Variable Documentation

8.61.1.1 struct hwgev hwevnt_

Definition at line 20 of file fio/HerwigWrapper.cc.

8.62 HerwigWrapper.cc File Reference

```
#include <cmath>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/GenCrossSection.h"
```

Namespaces

- **HepMC**

Functions

- GenCrossSection **HepMC::getHerwigCrossSection** (int ngen)

  calculate the Herwig cross section and statistical error

Variables

- struct **hwgev hwevnt_**

8.62.1 Variable Documentation

8.62.1.1 struct hwgev hwevnt_

Definition at line 20 of file HepMC-2.06.09/fio/HerwigWrapper.cc.

8.63 HerwigWrapper.h File Reference

```
#include <ctype.h>
#include "HepMC/GenCrossSection.h"
```

Classes

- struct **hwgev**

Namespaces

- **HepMC**

Macros

- #define **hwproc hwproc_**
- #define **hwbeam hwbeam_**
- #define **hwbmch hwbmch_**
- #define **hwevnt hwevnt_**
- #define **hwpram hwpram_**
- #define **hwigin** hwigin_
- #define **hwigup** hwigup_
- #define **hwuinc** hwuinc_

- #define **hwusta** hwusta_
- #define **hweini** hweini_
- #define **hwuine** hwuine_
- #define **hwepro** hwepro_
- #define **hwupro** hwupro_
- #define **hwbgen** hwbgen_
- #define **hwdhob** hwdhob_
- #define **hwcfor** hwcfor_
- #define **hwcdec** hwcdec_
- #define **hwdhad** hwdhad_
- #define **hwdhvy** hwdhvy_
- #define **hwmevt** hwmevt_
- #define **hwufne** hwufne_
- #define **hwefin** hwefin_
- #define **hwudpr** hwudpr_
- #define **hwuepr** hwuepr_
- #define **hwupup** hwupup_
- #define **hwegup** hwegup_
- #define **hwudat** hwudat_

Functions

- C void **hwigin** (void)
- void **hwigup** (void)
- void **hwuinc** (void)
- void **hwusta** (const char ∗, int)
- void **hweini** (void)
- void **hwuine** (void)
- void **hwepro** (void)
- void **hwupro** (void)
- void **hwbgen** (void)
- void **hwdhob** (void)
- void **hwcfor** (void)
- void **hwcdec** (void)
- void **hwdhad** (void)
- void **hwdhvy** (void)
- void **hwmevt** (void)
- void **hwufne** (void)
- void **hwefin** (void)
- void **hwudpr** (void)
- void **hwuepr** (void)
- void **hwupup** (void)
- void **hwegup** (void)
- void **hwudat** (void)
- GenCrossSection **HepMC::getHerwigCrossSection** (int ngen)

  calculate the Herwig cross section and statistical error

Variables

- struct {
  double **EBEAM1**
  double **EBEAM2**
  double **PBEAM1**
  double **PBEAM2**
  int **IPROC**
  int **MAXEV**
  } **hwproc_**

- struct {
  int **IPART1**
  int **IPART2**
  } **hwbeam_**

- struct {
  char **PART1** [8]
  char **PART2** [8]
  } **hwbmch_**

- const int **herwig_hepevt_size** = 4000
- struct **hwgev hwevnt_**
- struct {
  double **AFCH** [2][16]
  double **ALPHEM**
  double **B1LIM**
  double **BETAF**
  double **BTCLM**
  double **CAFAC**
  double **CFFAC**
  double **CLMAX**
  double **CLPOW**
  double **CLSMR** [2]
  double **CSPEED**
  double **ENSOF**
  double **ETAMIX**
  double **F0MIX**
  double **F1MIX**
  double **F2MIX**
  double **GAMH**
  double **GAMW**
  double **GAMZ**
  double **GAMZP**
  double **GEV2NB**
  double **H1MIX**
  double **PDIQK**
  double **PGSMX**
  double **PGSPL** [4]
  double **PHIMIX**
  double **PIFAC**
  double **PRSOF**
  double **PSPLT** [2]
  double **PTRMS**
  double **PXRMS**
  double **QCDL3**
  double **QCDL5**
  double **QCDLAM**

double **QDIQK**
double **QFCH** [16]
double **QG**
double **QSPAC**
double **QV**
double **SCABI**
double **SWEIN**
double **TMTOP**
double **VFCH** [2][16]
double **VCKM** [3][3]
double **VGCUT**
double **VQCUT**
double **VPCUT**
double **ZBINM**
double **EFFMIN**
double **OMHMIX**
double **ET2MIX**
double **PH3MIX**
double **GCUTME**
int **IOPREM**
int **IPRINT**
int **ISPAC**
int **LRSUD**
int **LWSUD**
int **MODPDF** [2]
int **NBTRY**
int **NCOLO**
int **NCTRY**
int **NDTRY**
int **NETRY**
int **NFLAV**
int **NGSPL**
int **NSTRU**
int **NSTRY**
int **NZBIN**
int **IOP4JT** [2]
int **NPRFMT**
int **AZSOFT**
int **AZSPIN**
int **CLDIR** [2]
int **HARDME**
int **NOSPAC**
int **PRNDEC**
int **PRVTX**
int **SOFTME**
int **ZPRIME**
int **PRNDEF**
int **PRNTEX**
int **PRNWEB**
} **hwpram_**

8.63.1 Macro Definition Documentation

8.63.1.1 #define hwbeam hwbeam_

Definition at line 40 of file HerwigWrapper.h.

8.63.1.2 #define hwbgen hwbgen_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 104 of file HerwigWrapper.h.

Referenced by main().

8.63.1.3 #define hwbmch hwbmch_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 49 of file HerwigWrapper.h.

Referenced by main().

8.63.1.4 #define hwcdec hwcdec_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 107 of file HerwigWrapper.h.

Referenced by main().

8.63.1.5 #define hwcfor hwcfor_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 106 of file HerwigWrapper.h.

Referenced by main().

8.63.1.6 #define hwdhad hwdhad_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 108 of file HerwigWrapper.h.

Referenced by main().

8.63.1.7 #define hwdhob hwdhob_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 105 of file HerwigWrapper.h.

Referenced by main().

8.63.1.8 #define hwdhvy hwdhvy_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 109 of file HerwigWrapper.h.

Referenced by main().

8.63.1.9 #define hwefin hwefin_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 112 of file HerwigWrapper.h.

Referenced by main().

8.63.1.10 #define hwegup hwegup_

Definition at line 117 of file HerwigWrapper.h.

8.63.1.11 #define hweini hweini_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 100 of file HerwigWrapper.h.

Referenced by main().

8.63.1.12 #define hwepro hwepro_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 102 of file HerwigWrapper.h.

Referenced by main().

8.63.1.13 #define hwevnt hwevnt_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 63 of file HerwigWrapper.h.

Referenced by HepMC::getHerwigCrossSection(), and main().

8.63.1.14 #define hwigin hwigin_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 96 of file HerwigWrapper.h.

Referenced by main().

8.63.1.15 #define hwigup hwigup_

Definition at line 97 of file HerwigWrapper.h.

8.63.1.16 #define hwmevt hwmevt_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 110 of file HerwigWrapper.h.

Referenced by main().

8.63.1.17 #define hwpram hwpram_

Definition at line 91 of file HerwigWrapper.h.

8.63.1.18 #define hwproc hwproc_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 32 of file HerwigWrapper.h.

Referenced by main().

8.63.1.19 #define hwudat hwudat_

Definition at line 147 of file HerwigWrapper.h.

8.63.1.20 #define hwudpr hwudpr_

Definition at line 114 of file HerwigWrapper.h.

8.63.1.21 #define hwuepr hwuepr_

Definition at line 115 of file HerwigWrapper.h.

8.63.1.22 #define hwufne hwufne_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 111 of file HerwigWrapper.h.

Referenced by main().

8.63.1.23 #define hwuinc hwuinc_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 98 of file HerwigWrapper.h.

Referenced by main().

8.63.1.24 #define hwuine hwuine_

Examples:

fio/example_MyHerwig.cc, and **fio/testHerwigCopies.cc**.

Definition at line 101 of file HerwigWrapper.h.

Referenced by main().

8.63.1.25 #define hwupro hwupro_

Definition at line 103 of file HerwigWrapper.h.

8.63.1.26 #define hwupup hwupup_

Definition at line 116 of file HerwigWrapper.h.

8.63.1.27 #define hwusta hwusta_

Definition at line 99 of file HerwigWrapper.h.

8.63.2 Function Documentation

8.63.2.1 void hwbgen (void)

8.63.2.2 void hwcdec (void)

8.63.2.3 void hwcfor (void)

8.63.2.4 void hwdhad (void)

8.63.2.5 void hwdhob (void)

8.63.2.6 void hwdhvy (void)

8.63.2.7 void hwefin (void)

8.63.2.8 void hwegup (void)

8.63.2.9 void hweini (void)

8.63.2.10 void hwepro (void)

8.63.2.11 C void hwigin (void)

8.63.2.12 void hwigup (void)

8.63.2.13 void hwmevt (void)

8.63.2.14 void hwudat (void)

8.63.2.15 void hwudpr (void)

8.63.2.16 void hwuepr (void)

8.63.2.17 void hwufne (void)

8.63.2.18 void hwuinc (void)

8.63.2.19 void hwuine (void)

8.63.2.20 void hwupro (void)

8.63.2.21 void hwupup (void)

8.63.2.22 void hwusta (const char * , int)

8.63.3 Variable Documentation

8.63.3.1 double AFCH[2][16]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.2 double ALPHEM

Definition at line 79 of file HerwigWrapper.h.

8.63.3.3 int AZSOFT

Definition at line 87 of file HerwigWrapper.h.

8.63.3.4 int AZSPIN

Definition at line 87 of file HerwigWrapper.h.

8.63.3.5 double B1LIM

Definition at line 79 of file HerwigWrapper.h.

8.63.3.6 double BETAF

Definition at line 79 of file HerwigWrapper.h.

8.63.3.7 double BTCLM

Definition at line 79 of file HerwigWrapper.h.

8.63.3.8 double CAFAC

Definition at line 79 of file HerwigWrapper.h.

8.63.3.9 double CFFAC

Definition at line 79 of file HerwigWrapper.h.

8.63.3.10 int CLDIR[2]

Definition at line 87 of file HerwigWrapper.h.

8.63.3.11 double CLMAX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.12 double CLPOW

Definition at line 79 of file HerwigWrapper.h.

8.63.3.13 double CLSMR[2]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.14 double CSPEED

Definition at line 79 of file HerwigWrapper.h.

8.63.3.15 double EBEAM1

Definition at line 28 of file HerwigWrapper.h.

8.63.3.16 double EBEAM2

Definition at line 28 of file HerwigWrapper.h.

8.63.3.17 double EFFMIN

Definition at line 79 of file HerwigWrapper.h.

8.63.3.18 double ENSOF

Definition at line 79 of file HerwigWrapper.h.

8.63.3.19 double ET2MIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.20 double ETAMIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.21 double F0MIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.22 double F1MIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.23 double F2MIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.24 double GAMH

Definition at line 79 of file HerwigWrapper.h.

8.63.3.25 double GAMW

Definition at line 79 of file HerwigWrapper.h.

8.63.3.26 double GAMZ

Definition at line 79 of file HerwigWrapper.h.

8.63.3.27 double GAMZP

Definition at line 79 of file HerwigWrapper.h.

8.63.3.28 double GCUTME

Definition at line 79 of file HerwigWrapper.h.

8.63.3.29 double GEV2NB

Definition at line 79 of file HerwigWrapper.h.

8.63.3.30 double H1MIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.31 int HARDME

Definition at line 87 of file HerwigWrapper.h.

8.63.3.32 const int herwig_hepevt_size = 4000

Definition at line 54 of file HerwigWrapper.h.

8.63.3.33 struct { ... } hwbeam_

8.63.3.34 struct { ... } hwbmch_

8.63.3.35 struct hwgev hwevnt_

8.63.3.36 struct { ... } hwpram_

8.63.3.37 struct { ... } hwproc_

8.63.3.38 int IOP4JT[2]

Definition at line 85 of file HerwigWrapper.h.

8.63.3.39 int IOPREM

Definition at line 85 of file HerwigWrapper.h.

8.63.3.40 int IPART1

Definition at line 37 of file HerwigWrapper.h.

8.63.3.41 int IPART2

Definition at line 37 of file HerwigWrapper.h.

8.63.3.42 int IPRINT

Definition at line 85 of file HerwigWrapper.h.

8.63.3.43 int IPROC

Definition at line 29 of file HerwigWrapper.h.

8.63.3.44 int ISPAC

Definition at line 85 of file HerwigWrapper.h.

8.63.3.45 int LRSUD

Definition at line 85 of file HerwigWrapper.h.

8.63.3.46 int LWSUD

Definition at line 85 of file HerwigWrapper.h.

8.63.3.47 int MAXEV

Definition at line 29 of file HerwigWrapper.h.

8.63.3.48 int MODPDF[2]

Definition at line 85 of file HerwigWrapper.h.

8.63.3.49 int NBTRY

Definition at line 85 of file HerwigWrapper.h.

8.63.3.50 int NCOLO

Definition at line 85 of file HerwigWrapper.h.

8.63.3.51 int NCTRY

Definition at line 85 of file HerwigWrapper.h.

8.63.3.52 int NDTRY

Definition at line 85 of file HerwigWrapper.h.

8.63.3.53 int NETRY

Definition at line 85 of file HerwigWrapper.h.

8.63.3.54 int NFLAV

Definition at line 85 of file HerwigWrapper.h.

8.63.3.55 int NGSPL

Definition at line 85 of file HerwigWrapper.h.

8.63.3.56 int NOSPAC

Definition at line 87 of file HerwigWrapper.h.

8.63.3.57 int NPRFMT

Definition at line 85 of file HerwigWrapper.h.

8.63.3.58 int NSTRU

Definition at line 85 of file HerwigWrapper.h.

8.63.3.59 int NSTRY

Definition at line 85 of file HerwigWrapper.h.

8.63.3.60 int NZBIN

Definition at line 85 of file HerwigWrapper.h.

8.63.3.61 double OMHMIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.62 char PART1[8]

Definition at line 46 of file HerwigWrapper.h.

8.63.3.63 char PART2[8]

Definition at line 46 of file HerwigWrapper.h.

8.63.3.64 double PBEAM1

Definition at line 28 of file HerwigWrapper.h.

8.63.3.65 double PBEAM2

Definition at line 28 of file HerwigWrapper.h.

8.63.3.66 double PDIQK

Definition at line 79 of file HerwigWrapper.h.

8.63.3.67 double PGSMX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.68 double PGSPL[4]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.69 double PH3MIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.70 double PHIMIX

Definition at line 79 of file HerwigWrapper.h.

8.63.3.71 double PIFAC

Definition at line 79 of file HerwigWrapper.h.

8.63.3.72 int PRNDEC

Definition at line 87 of file HerwigWrapper.h.

8.63.3.73 int PRNDEF

Definition at line 87 of file HerwigWrapper.h.

8.63.3.74 int PRNTEX

Definition at line 87 of file HerwigWrapper.h.

8.63.3.75 int PRNWEB

Definition at line 87 of file HerwigWrapper.h.

8.63.3.76 double PRSOF

Definition at line 79 of file HerwigWrapper.h.

8.63.3.77 int PRVTX

Definition at line 87 of file HerwigWrapper.h.

8.63.3.78 double PSPLT[2]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.79 double PTRMS

Definition at line 79 of file HerwigWrapper.h.

8.63.3.80 double PXRMS

Definition at line 79 of file HerwigWrapper.h.

8.63.3.81 double QCDL3

Definition at line 79 of file HerwigWrapper.h.

8.63.3.82 double QCDL5

Definition at line 79 of file HerwigWrapper.h.

8.63.3.83 double QCDLAM

Definition at line 79 of file HerwigWrapper.h.

8.63.3.84 double QDIQK

Definition at line 79 of file HerwigWrapper.h.

8.63.3.85 double QFCH[16]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.86 double QG

Definition at line 79 of file HerwigWrapper.h.

8.63.3.87 double QSPAC

Definition at line 79 of file HerwigWrapper.h.

8.63.3.88 double QV

Definition at line 79 of file HerwigWrapper.h.

8.63.3.89 double SCABI

Definition at line 79 of file HerwigWrapper.h.

8.63.3.90 int SOFTME

Definition at line 87 of file HerwigWrapper.h.

8.63.3.91 double SWEIN

Definition at line 79 of file HerwigWrapper.h.

8.63.3.92 double TMTOP

Definition at line 79 of file HerwigWrapper.h.

8.63.3.93 double VCKM[3][3]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.94 double VFCH[2][16]

Definition at line 79 of file HerwigWrapper.h.

8.63.3.95 double VGCUT

Definition at line 79 of file HerwigWrapper.h.

8.63.3.96 double VPCUT

Definition at line 79 of file HerwigWrapper.h.

8.63.3.97 double VQCUT

Definition at line 79 of file HerwigWrapper.h.

8.63.3.98 double ZBINM

Definition at line 79 of file HerwigWrapper.h.

8.63.3.99 int ZPRIME

Definition at line 87 of file HerwigWrapper.h.

8.64 HerwigWrapper.h File Reference

```
#include <ctype.h>
#include "HepMC/GenCrossSection.h"
```

Classes

- struct **hwgev**

Namespaces

- **HepMC**

Macros

- #define **hwproc hwproc_**
- #define **hwbeam hwbeam_**
- #define **hwbmch hwbmch_**
- #define **hwevnt hwevnt_**
- #define **hwpram hwpram_**
- #define **hwigin** hwigin_
- #define **hwigup** hwigup_
- #define **hwuinc** hwuinc_
- #define **hwusta** hwusta_
- #define **hweini** hweini_
- #define **hwuine** hwuine_
- #define **hwepro** hwepro_
- #define **hwupro** hwupro_
- #define **hwbgen** hwbgen_

- #define **hwdhob** hwdhob_
- #define **hwcfor** hwcfor_
- #define **hwcdec** hwcdec_
- #define **hwdhad** hwdhad_
- #define **hwdhvy** hwdhvy_
- #define **hwmevt** hwmevt_
- #define **hwufne** hwufne_
- #define **hwefin** hwefin_
- #define **hwudpr** hwudpr_
- #define **hwuepr** hwuepr_
- #define **hwupup** hwupup_
- #define **hwegup** hwegup_
- #define **hwudat** hwudat_

Functions

- C void **hwigin** (void)
- void **hwigup** (void)
- void **hwuinc** (void)
- void **hwusta** (const char ∗, int)
- void **hweini** (void)
- void **hwuine** (void)
- void **hwepro** (void)
- void **hwupro** (void)
- void **hwbgen** (void)
- void **hwdhob** (void)
- void **hwcfor** (void)
- void **hwcdec** (void)
- void **hwdhad** (void)
- void **hwdhvy** (void)
- void **hwmevt** (void)
- void **hwufne** (void)
- void **hwefin** (void)
- void **hwudpr** (void)
- void **hwuepr** (void)
- void **hwupup** (void)
- void **hwegup** (void)
- void **hwudat** (void)
- GenCrossSection **HepMC::getHerwigCrossSection** (int ngen)

  calculate the Herwig cross section and statistical error

Variables

- struct {
  double **EBEAM1**
  double **EBEAM2**
  double **PBEAM1**
  double **PBEAM2**
  int **IPROC**
  int **MAXEV**
  } **hwproc_**

- struct {
  int **IPART1**
  int **IPART2**
  } **hwbeam_**

- struct {
  char **PART1** [8]
  char **PART2** [8]
  } **hwbmch_**

- const int **herwig_hepevt_size** = 4000
- struct **hwgev hwevnt_**
- struct {
  double **AFCH** [2][16]
  double **ALPHEM**
  double **B1LIM**
  double **BETAF**
  double **BTCLM**
  double **CAFAC**
  double **CFFAC**
  double **CLMAX**
  double **CLPOW**
  double **CLSMR** [2]
  double **CSPEED**
  double **ENSOF**
  double **ETAMIX**
  double **F0MIX**
  double **F1MIX**
  double **F2MIX**
  double **GAMH**
  double **GAMW**
  double **GAMZ**
  double **GAMZP**
  double **GEV2NB**
  double **H1MIX**
  double **PDIQK**
  double **PGSMX**
  double **PGSPL** [4]
  double **PHIMIX**
  double **PIFAC**
  double **PRSOF**
  double **PSPLT** [2]
  double **PTRMS**
  double **PXRMS**
  double **QCDL3**
  double **QCDL5**
  double **QCDLAM**
  double **QDIQK**
  double **QFCH** [16]
  double **QG**
  double **QSPAC**
  double **QV**
  double **SCABI**
  double **SWEIN**
  double **TMTOP**
  double **VFCH** [2][16]
  double **VCKM** [3][3]
  double **VGCUT**

double **VQCUT**
double **VPCUT**
double **ZBINM**
double **EFFMIN**
double **OMHMIX**
double **ET2MIX**
double **PH3MIX**
double **GCUTME**
int **IOPREM**
int **IPRINT**
int **ISPAC**
int **LRSUD**
int **LWSUD**
int **MODPDF** [2]
int **NBTRY**
int **NCOLO**
int **NCTRY**
int **NDTRY**
int **NETRY**
int **NFLAV**
int **NGSPL**
int **NSTRU**
int **NSTRY**
int **NZBIN**
int **IOP4JT** [2]
int **NPRFMT**
int **AZSOFT**
int **AZSPIN**
int **CLDIR** [2]
int **HARDME**
int **NOSPAC**
int **PRNDEC**
int **PRVTX**
int **SOFTME**
int **ZPRIME**
int **PRNDEF**
int **PRNTEX**
int **PRNWEB**
} **hwpram_**

8.64.1 Macro Definition Documentation

8.64.1.1 #define hwbeam hwbeam_

Definition at line 40 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.2 #define hwbgen hwbgen_

Definition at line 104 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.3 #define hwbmch hwbmch_

Definition at line 49 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.4 #define hwcdec hwcdec_

Definition at line 107 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.5 #define hwcfor hwcfor_

Definition at line 106 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.6 #define hwdhad hwdhad_

Definition at line 108 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.7 #define hwdhob hwdhob_

Definition at line 105 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.8 #define hwdhvy hwdhvy_

Definition at line 109 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.9 #define hwefin hwefin_

Definition at line 112 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.10 #define hwegup hwegup_

Definition at line 117 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.11 #define hweini hweini_

Definition at line 100 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.12 #define hwepro hwepro_

Definition at line 102 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.13 #define hwevnt hwevnt_

Definition at line 63 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.14 #define hwigin hwigin_

Definition at line 96 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.15 #define hwigup hwigup_

Definition at line 97 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.16 #define hwmevt hwmevt_

Definition at line 110 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.17 #define hwpram hwpram_

Definition at line 91 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.18 #define hwproc hwproc_

Definition at line 32 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.19 #define hwudat hwudat_

Definition at line 147 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.20 #define hwudpr hwudpr_

Definition at line 114 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.21 #define hwuepr hwuepr_

Definition at line 115 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.22 #define hwufne hwufne_

Definition at line 111 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.23 #define hwuinc hwuinc_

Definition at line 98 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.24 #define hwuine hwuine_

Definition at line 101 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.25 #define hwupro hwupro_

Definition at line 103 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.26 #define hwupup hwupup_

Definition at line 116 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.1.27 #define hwusta hwusta_

Definition at line 99 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.2 Function Documentation

8.64.2.1 void hwbgen (void)

8.64.2.2 void hwcdec (void)

8.64.2.3 void hwcfor (void)

8.64.2.4 void hwdhad (void)

8.64.2.5 void hwdhob (void)

8.64.2.6 void hwdhvy (void)

8.64.2.7 void hwefin (void)

8.64.2.8 void hwegup (void)

8.64.2.9 void hweini (void)

8.64.2.10 void hwepro (void)

8.64.2.11 C void hwigin (void)

8.64.2.12 void hwigup (void)

8.64.2.13 void hwmevt (void)

8.64.2.14 void hwudat (void)

8.64.2.15 void hwudpr (void)

8.64.2.16 void hwuepr (void)

8.64.2.17 void hwufne (void)

8.64.2.18 void hwuinc (void)

8.64.2.19 void hwuine (void)

8.64.2.20 void hwupro (void)

8.64.2.21 void hwupup (void)

8.64.2.22 void hwusta (const char * , int)

8.64.3 Variable Documentation

8.64.3.1 double AFCH[2][16]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.2 double ALPHEM

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.3 int AZSOFT

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.4 int AZSPIN

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.5 double B1LIM

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.6 double BETAF

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.7 double BTCLM

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.8 double CAFAC

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.9 double CFFAC

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.10 int CLDIR[2]

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.11 double CLMAX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.12 double CLPOW

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.13 double CLSMR[2]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.14 double CSPEED

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.15 double EBEAM1

Definition at line 28 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.16 double EBEAM2

Definition at line 28 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.17 double EFFMIN

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.18 double ENSOF

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.19 double ET2MIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.20 double ETAMIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.21 double F0MIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.22 double F1MIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.23 double F2MIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.24 double GAMH

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.25 double GAMW

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.26 double GAMZ

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.27 double GAMZP

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.28 double GCUTME

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.29 double GEV2NB

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.30 double H1MIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.31 int HARDME

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.32 const int herwig_hepevt_size = 4000

Definition at line 54 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.33 struct { ... } hwbeam_

8.64.3.34 struct { ... } hwbmch_

8.64.3.35 struct hwgev hwevnt_

8.64.3.36 struct { ... } hwpram_

8.64.3.37 struct { ... } hwproc_

8.64.3.38 int IOP4JT[2]

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.39 int IOPREM

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.40 int IPART1

Definition at line 37 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.41 int IPART2

Definition at line 37 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.42 int IPRINT

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.43 int IPROC

Definition at line 29 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.44 int ISPAC

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.45 int LRSUD

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.46 int LWSUD

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.47 int MAXEV

Definition at line 29 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.48 int MODPDF[2]

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.49 int NBTRY

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.50 int NCOLO

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.51 int NCTRY

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.52 int NDTRY

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.53 int NETRY

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.54 int NFLAV

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.55 int NGSPL

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.56 int NOSPAC

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.57 int NPRFMT

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.58 int NSTRU

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.59 int NSTRY

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.60 int NZBIN

Definition at line 85 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.61 double OMHMIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.62 char PART1[8]

Definition at line 46 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.63 char PART2[8]

Definition at line 46 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.64 double PBEAM1

Definition at line 28 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.65 double PBEAM2

Definition at line 28 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.66 double PDIQK

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.67 double PGSMX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.68 double PGSPL[4]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.69 double PH3MIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.70 double PHIMIX

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.71 double PIFAC

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.72 int PRNDEC

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.73 int PRNDEF

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.74 int PRNTEX

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.75 int PRNWEB

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.76 double PRSOF

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.77 int PRVTX

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.78 double PSPLT[2]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.79 double PTRMS

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.80 double PXRMS

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.81 double QCDL3

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.82 double QCDL5

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.83 double QCDLAM

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.84 double QDIQK

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.85 double QFCH[16]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.86 double QG

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.87 double QSPAC

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.88 double QV

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.89 double SCABI

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.90 int SOFTME

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.91 double SWEIN

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.92 double TMTOP

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.93 double VCKM[3][3]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.94 double VFCH[2][16]

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.95 double VGCUT

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.96 double VPCUT

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.97 double VQCUT

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.98 double ZBINM

Definition at line 79 of file 2.06.09/HepMC/HerwigWrapper.h.

8.64.3.99 int ZPRIME

Definition at line 87 of file 2.06.09/HepMC/HerwigWrapper.h.

8.65 initpydata.f File Reference

Functions/Subroutines

- subroutine **initpydata**

8.65.1 Function/Subroutine Documentation

8.65.1.1 subroutine initpydata (void)

Definition at line 6 of file examples/fio/initpydata.f.

8.66 initpydata.f File Reference

Functions/Subroutines

- subroutine **initpydata**

8.66.1 Function/Subroutine Documentation

8.66.1.1 subroutine initpydata (void)

Definition at line 6 of file HepMC-2.06.09/examples/fio/initpydata.f.

References pydata.

8.67 initPythia.cc File Reference

```
#include "HepMC/PythiaWrapper.h"
#include "PythiaHelper.h"
```

Functions

- void **initPythia** ()

8.67.1 Function Documentation

8.67.1.1 void initPythia ( )

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 12 of file examples/fio/initPythia.cc.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

8.68 initPythia.cc File Reference

```
#include "HepMC/PythiaWrapper.h"
#include "PythiaHelper.h"
```

Functions

- void **initPythia** ()

8.68.1 Function Documentation

8.68.1.1 void initPythia ( )

Definition at line 12 of file HepMC-2.06.09/examples/fio/initPythia.cc.

References call_pyinit(), pydat2, pydatr, pypars, and pysubs.

8.69 IO_AsciiParticles.cc File Reference

```
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "HepMC/Version.h"
```

Namespaces

- **HepMC**

8.70 IO_AsciiParticles.cc File Reference

```
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "HepMC/Version.h"
```

Namespaces

- **HepMC**

8.71 IO_AsciiParticles.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
```

Classes

- class **HepMC::IO_AsciiParticles**

  event input/output in ascii format for eye and machine reading

Namespaces

- **HepMC**

8.72 IO_AsciiParticles.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
```

Classes

- class **HepMC::IO_AsciiParticles**

  event input/output in ascii format for eye and machine reading

Namespaces

- **HepMC**

8.73 IO_BaseClass.h File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Classes

- class **HepMC::IO_BaseClass**

  *all input/output classes inherit from **IO_BaseClass*** (p. *180)*

Namespaces

- **HepMC**

8.74 IO_BaseClass.h File Reference

```
#include <iostream>
#include "HepMC/GenEvent.h"
```

Classes

- class **HepMC::IO_BaseClass**

  *all input/output classes inherit from **IO_BaseClass*** (p. *180)*

Namespaces

- **HepMC**

8.75 IO_Exception.h File Reference

```
#include <stdexcept>
```

Classes

- class **HepMC::IO_Exception**

  IO exception handling.

Namespaces

- **HepMC**

8.76 IO_Exception.h File Reference

```
#include <stdexcept>
```

Classes

- class **HepMC::IO_Exception**

  IO exception handling.

Namespaces

- **HepMC**

8.77 IO_GenEvent.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_Exception.h"
#include "HepMC/GenEvent.h"
#include "HepMC/StreamHelpers.h"
```

Namespaces

- **HepMC**

8.78 IO_GenEvent.cc File Reference

```
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_Exception.h"
#include "HepMC/GenEvent.h"
#include "HepMC/StreamHelpers.h"
```

Namespaces

- **HepMC**

8.79 IO_GenEvent.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/IO_Exception.h"
#include "HepMC/Units.h"
```

Classes

- class **HepMC::IO_GenEvent**
  IO_GenEvent *(p. 186) also deals with* ***Heavylon*** *(p. 146) and* ***PdfInfo*** *(p. 233).*

Namespaces

- **HepMC**

8.80 IO_GenEvent.h File Reference

```
#include <fstream>
#include <string>
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/IO_Exception.h"
#include "HepMC/Units.h"
```

Classes

- class **HepMC::IO_GenEvent**
  IO_GenEvent *(p. 186) also deals with* ***Heavylon*** *(p. 146) and* ***PdfInfo*** *(p. 233).*

Namespaces

- **HepMC**

8.81 IO_HEPEVT.cc File Reference

```
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include <cstdio>
```

Namespaces

- **HepMC**

8.82 IO_HEPEVT.cc File Reference

```
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include <cstdio>
```

Namespaces

- **HepMC**

8.83 IO_HEPEVT.h File Reference

```
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Classes

- class **HepMC::IO_HEPEVT**

  HEPEVT IO class.

Namespaces

- **HepMC**

8.84 IO_HEPEVT.h File Reference

```
#include <map>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Classes

- class **HepMC::IO_HEPEVT**

  HEPEVT IO class.

Namespaces

- **HepMC**

8.85 IO_HERWIG.cc File Reference

```
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include <cstdio>
```

Namespaces

- **HepMC**

8.86 IO_HERWIG.cc File Reference

```
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include <cstdio>
```

Namespaces

- **HepMC**

8.87 IO_HERWIG.h File Reference

```
#include <set>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Classes

- class **HepMC::IO_HERWIG**

  IO_HERWIG *(*p. *199) is used to get Herwig information.*

Namespaces

- **HepMC**

8.88 IO_HERWIG.h File Reference

```
#include <set>
#include <vector>
#include "HepMC/IO_BaseClass.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Classes

- class **HepMC::IO_HERWIG**

  IO_HERWIG *(*p. *199) is used to get Herwig information.*

Namespaces

- **HepMC**

8.89 is_arithmetic.h File Reference

Classes

- struct **HepMC::detail::is_arithmetic**< **T** >

  undefined and therefore non-arithmetic
- struct **HepMC::detail::is_arithmetic**< **char** >

  character is arithmetic
- struct **HepMC::detail::is_arithmetic**< **unsigned char** >

  unsigned character is arithmetic
- struct **HepMC::detail::is_arithmetic**< **signed char** >

  signed character is arithmetic
- struct **HepMC::detail::is_arithmetic**< **short** >

  short is arithmetic
- struct **HepMC::detail::is_arithmetic**< **unsigned short** >

  unsigned short is arithmetic
- struct **HepMC::detail::is_arithmetic**< **int** >

  int is arithmetic
- struct **HepMC::detail::is_arithmetic**< **unsigned int** >

  unsigned int is arithmetic
- struct **HepMC::detail::is_arithmetic**< **long** >

  long is arithmetic
- struct **HepMC::detail::is_arithmetic**< **unsigned long** >

  unsigned long is arithmetic
- struct **HepMC::detail::is_arithmetic**< **float** >

  float is arithmetic
- struct **HepMC::detail::is_arithmetic**< **double** >

  double is arithmetic
- struct **HepMC::detail::is_arithmetic**< **long double** >

  long double is arithmetic

Namespaces

- **HepMC**
- **detail**
- **HepMC::detail**

8.90 is_arithmetic.h File Reference

Classes

- struct **HepMC::detail::is_arithmetic< T >**
  undefined and therefore non-arithmetic
- struct **HepMC::detail::is_arithmetic< char >**
  character is arithmetic
- struct **HepMC::detail::is_arithmetic< unsigned char >**
  unsigned character is arithmetic
- struct **HepMC::detail::is_arithmetic< signed char >**
  signed character is arithmetic
- struct **HepMC::detail::is_arithmetic< short >**
  short is arithmetic
- struct **HepMC::detail::is_arithmetic< unsigned short >**
  unsigned short is arithmetic
- struct **HepMC::detail::is_arithmetic< int >**
  int is arithmetic
- struct **HepMC::detail::is_arithmetic< unsigned int >**
  unsigned int is arithmetic
- struct **HepMC::detail::is_arithmetic< long >**
  long is arithmetic
- struct **HepMC::detail::is_arithmetic< unsigned long >**
  unsigned long is arithmetic
- struct **HepMC::detail::is_arithmetic< float >**
  float is arithmetic
- struct **HepMC::detail::is_arithmetic< double >**
  double is arithmetic
- struct **HepMC::detail::is_arithmetic< long double >**
  long double is arithmetic

Namespaces

- **HepMC**
- **detail**
- **HepMC::detail**

8.91 IsGoodEvent.h File Reference

Classes

- class **IsGoodEvent**
  used in the tests

8.92 IteratorRange.h File Reference

Namespaces

- **HepMC**

Enumerations

- enum **HepMC::IteratorRange** {
  HepMC::parents, **HepMC::children**, **HepMC::family**, **HepMC::ancestors**,
  HepMC::descendants, **HepMC::relatives**, **HepMC::parents**, **HepMC::children**,
  HepMC::family, **HepMC::ancestors**, **HepMC::descendants**, **HepMC::relatives** }

  type of iteration

8.93 IteratorRange.h File Reference

Namespaces

- **HepMC**

Enumerations

- enum **HepMC::IteratorRange** {
  HepMC::parents, **HepMC::children**, **HepMC::family**, **HepMC::ancestors**,
  HepMC::descendants, **HepMC::relatives**, **HepMC::parents**, **HepMC::children**,
  HepMC::family, **HepMC::ancestors**, **HepMC::descendants**, **HepMC::relatives** }

  type of iteration

8.94 list_of_examples.cc File Reference

8.95 list_of_examples.cc File Reference

8.96 main31.cc File Reference

```
#include "Pythia.h"
#include "HepMCInterface.h"
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Functions

- int **main** ()

8.96.1 Function Documentation

8.96.1.1 int main ( )

Definition at line 32 of file examples/pythia8/main31.cc.

References HepMC::Units::GEV, and HepMC::Units::MM.

8.97 main31.cc File Reference

```
#include "Pythia.h"
#include "HepMCInterface.h"
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Functions

- int **main** ()

8.97.1 Function Documentation

8.97.1.1 int main ( )

Definition at line 32 of file HepMC-2.06.09/examples/pythia8/main31.cc.

References HepMC::Units::GEV, and HepMC::Units::MM.

8.98 main32.cc File Reference

```
#include "Pythia.h"
#include "HepMCInterface.h"
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Functions

- int **main** (int argc, char ∗argv[])

8.98.1 Function Documentation

8.98.1.1 int main (int *argc,* char ∗ *argv[]*)

Definition at line 33 of file examples/pythia8/main32.cc.

References HepMC::Units::GEV, and HepMC::Units::MM.

8.99 main32.cc File Reference

```
#include "Pythia.h"
#include "HepMCInterface.h"
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Functions

- int **main** (int argc, char ∗argv[])

8.99.1 Function Documentation

8.99.1.1 int main (int *argc,* char ∗ *argv[]*)

Definition at line 33 of file HepMC-2.06.09/examples/pythia8/main32.cc.

References HepMC::Units::GEV, and HepMC::Units::MM.

8.100 PdfInfo.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/PdfInfo.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, PdfInfo const ∗)
- std::istream & **HepMC::operator**>> (std::istream &, PdfInfo ∗)

8.101 PdfInfo.cc File Reference

```
#include <iostream>
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/PdfInfo.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, PdfInfo const ∗)
- std::istream & **HepMC::operator**>> (std::istream &, PdfInfo ∗)

8.102 PdfInfo.h File Reference

Classes

- class **HepMC::PdfInfo**

  *The **PdfInfo** (p. 233) class stores PDF information.*

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, PdfInfo const *)
- std::istream & **HepMC::operator**>> (std::istream &, PdfInfo *)

8.103 PdfInfo.h File Reference

Classes

- class **HepMC::PdfInfo**

  *The **PdfInfo** (p. 233) class stores PDF information.*

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &, PdfInfo const *)
- std::istream & **HepMC::operator**>> (std::istream &, PdfInfo *)

8.104 Polarization.cc File Reference

```
#include "HepMC/Polarization.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const Polarization &polar)

  write theta and phi to the output stream

8.105 Polarization.cc File Reference

```
#include "HepMC/Polarization.h"
```

Namespaces

- **HepMC**

Functions

- std::ostream & **HepMC::operator**<< (std::ostream &ostr, const Polarization &polar)
  write theta and phi to the output stream

8.106 Polarization.h File Reference

```
#include "HepMC/SimpleVector.h"
#include <iostream>
#include <cmath>
```

Classes

- class **HepMC::Polarization**
  *The **Polarization** (p. 245) class stores theta and phi for a **GenParticle** (p. 106).*

Namespaces

- **HepMC**

Variables

- static const double **HepMC::HepMC_pi** = 3.14159265358979323846

8.107 Polarization.h File Reference

```
#include "HepMC/SimpleVector.h"
#include <iostream>
#include <cmath>
```

Classes

- class **HepMC::Polarization**
  *The **Polarization** (p. 245) class stores theta and phi for a **GenParticle** (p. 106).*

Namespaces

- **HepMC**

Variables

- static const double **HepMC::HepMC_pi** = 3.14159265358979323846

8.108 PythiaHelper.h File Reference

```
#include "HepMC/PythiaWrapper.h"
```

Functions

- void **initPythia** ()

8.108.1 Function Documentation

8.108.1.1 void initPythia ( )

Definition at line 12 of file examples/fio/initPythia.cc.

8.109 PythiaHelper.h File Reference

```
#include "HepMC/PythiaWrapper.h"
```

Functions

- void **initPythia** ()

8.109.1 Function Documentation

8.109.1.1 void initPythia ( )

Definition at line 12 of file examples/fio/initPythia.cc.

References call_pyinit(), pydat2, pydatr, pypars, and pysubs.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

8.110 PythiaWrapper.h File Reference

```
#include "HepMC/PythiaWrapper6_4.h"
#include <cmath>
#include "HepMC/GenCrossSection.h"
```

Namespaces

- **HepMC**

Functions

- GenCrossSection **HepMC::getPythiaCrossSection** ()
  calculate the Pythia cross section and statistical error

8.111 PythiaWrapper.h File Reference

```
#include "HepMC/PythiaWrapper6_4.h"
#include <cmath>
#include "HepMC/GenCrossSection.h"
```

Namespaces

- **HepMC**

Functions

- GenCrossSection **HepMC::getPythiaCrossSection** ()
  calculate the Pythia cross section and statistical error

8.112 PythiaWrapper6_4.h File Reference

```
#include <ctype.h>
#include <cstring>
```

Classes

- struct **pin3**
- struct **pin5**
- struct **pin7**
- struct **pin8**
- struct **pin9**
- struct **pssm**
- struct **prvnv**
- struct **prvpm**

Macros

- #define **initpydata** initpydata_
- #define **pyjets pyjets_**
- #define **pydat1 pydat1_**
- #define **pydat2 pydat2_**
- #define **pydat3 pydat3_**
- #define **pydatr pydatr_**
- #define **pysubs pysubs_**
- #define **pypars pypars_**
- #define **pyint1 pyint1_**
- #define **pyint2 pyint2_**

- #define **pyint3 pyint3_**
- #define **pyint4 pyint4_**
- #define **pyint5 pyint5_**
- #define **pyint7 pyint7_**
- #define **pyint8 pyint8_**
- #define **pyint9 pyint9_**
- #define **pyssm pyssm_**
- #define **pyssmt pyssmt_**
- #define **pymsrv pymsrv_**
- #define **pyrvnv pyrvnv_**
- #define **pyrvpm pyrvpm_**
- #define **pyints pyints_**
- #define **pyg2dx pyg2dx_**
- #define **pyhepc** pyhepc_
- #define **pyinit** pyinit_
- #define **pylist** pylist_
- #define **pystat** pystat_
- #define **pyevnt** pyevnt_
- #define **upinit** upinit_
- #define **upevnt** upevnt_
- #define **pydata** pydata_

Functions

- void **initpydata** (void)
- C void **pyhepc** (int ∗)
- void **pyinit** (const char ∗, const char ∗, const char ∗, double ∗, int, int, int)
- void **pylist** (int ∗)
- void **pystat** (int ∗)
- void **pyevnt** ()
- void **upinit** ()
- void **upevnt** ()
- void **call_pyhepc** (int mode)
- void **call_pyinit** (const char ∗frame, const char ∗beam, const char ∗target, double win)
- void **call_pylist** (int mode)
- void **call_pystat** (int mode)
- void **call_pyevnt** ()
- void **pydata** (void)

Variables

- const int **pyjets_maxn** =4000
- struct {
  int **n**
  int **npad**
  int **k** [5][**pyjets_maxn**]
  double **p** [5][**pyjets_maxn**]
  double **v** [5][**pyjets_maxn**]
  } **pyjets_**

- struct {
  int **mstu** [200]
  double **paru** [200]
  int **mstj** [200]
  double **parj** [200]
  } **pydat1_**

- struct {
  int **kchg** [4][500]
  double **pmas** [4][500]
  double **parf** [2000]
  double **vckm** [4][4]
  } **pydat2_**

- struct {
  int **mdcy** [3][500]
  int **mdme** [2][8000]
  double **brat** [8000]
  int **kfdp** [5][8000]
  } **pydat3_**

- struct {
  int **mrpy** [6]
  double **rrpy** [100]
  } **pydatr_**

- struct {
  int **msel**
  int **mselpd**
  int **msub** [500]
  int **kfin** [81][2]
  double **ckin** [200]
  } **pysubs_**

- struct {
  int **mstp** [200]
  double **parp** [200]
  int **msti** [200]
  double **pari** [200]
  } **pypars_**

- struct {
  int **mint** [400]
  double **vint** [400]
  } **pyint1_**

- struct {
  int **iset** [500]
  int **kfpr** [2][500]
  double **coef** [20][500]
  int **icol** [2][4][40]
  } **pyint2_**

- struct **pin3 pyint3_**
- struct {
  int **mwid** [500]
  double **wids** [5][500]
  } **pyint4_**

- struct **pin5 pyint5_**
- struct **pin7 pyint7_**
- struct **pin8 pyint8_**
- struct **pin9 pyint9_**
- struct **pssm pyssm_**
- struct {
  double **zmix** [4][4]
  double **umix** [2][2]
  double **vmix** [2][2]
  double **smz** [4]
  double **smw** [2]
  double **sfmix** [4][16]
  double **zmixi** [4][4]
  double **umixi** [2][2]
  double **vmixi** [2][2]
  } **pyssmt_**

- struct {
  double **rvlam** [3][3][3]
  double **rvlamp** [3][3][3]
  double **rvlamb** [3][3][3]
  } **pymsrv_**

- struct **prvnv pyrvnv_**
- struct **prvpm pyrvpm_**
- struct {
  double **xxm** [20]
  } **pyints_**

- struct {
  double **x1**
  } **pyg2dx_**

8.112.1 Macro Definition Documentation

8.112.1.1 #define initpydata initpydata_

Definition at line 30 of file PythiaWrapper6_4.h.

8.112.1.2 #define pydat1 pydat1_

Definition at line 52 of file PythiaWrapper6_4.h.

8.112.1.3 #define pydat2 pydat2_

Definition at line 60 of file PythiaWrapper6_4.h.

Referenced by initPythia().

8.112.1.4 #define pydat3 pydat3_

Definition at line 69 of file PythiaWrapper6_4.h.

8.112.1.5 #define pydata pydata_

Definition at line 275 of file PythiaWrapper6_4.h.

Referenced by initpydata().

8.112.1.6 #define pydatr pydatr_

Definition at line 77 of file PythiaWrapper6_4.h.

Referenced by initPythia().

8.112.1.7 #define pyevnt pyevnt_

Definition at line 245 of file PythiaWrapper6_4.h.

Referenced by call_pyevnt().

8.112.1.8 #define pyg2dx pyg2dx_

Definition at line 236 of file PythiaWrapper6_4.h.

8.112.1.9 #define pyhepc pyhepc_

Definition at line 241 of file PythiaWrapper6_4.h.

Referenced by call_pyhepc().

8.112.1.10 #define pyinit pyinit_

Definition at line 242 of file PythiaWrapper6_4.h.

Referenced by call_pyinit().

8.112.1.11 #define pyint1 pyint1_

Definition at line 103 of file PythiaWrapper6_4.h.

8.112.1.12 #define pyint2 pyint2_

Definition at line 112 of file PythiaWrapper6_4.h.

8.112.1.13 #define pyint3 pyint3_

Definition at line 121 of file PythiaWrapper6_4.h.

8.112.1.14 #define pyint4 pyint4_

Definition at line 129 of file PythiaWrapper6_4.h.

8.112.1.15 #define pyint5 pyint5_

Definition at line 137 of file PythiaWrapper6_4.h.

Referenced by HepMC::getPythiaCrossSection().

8.112.1.16 #define pyint7 pyint7_

Definition at line 144 of file PythiaWrapper6_4.h.

8.112.1.17 #define pyint8 pyint8_

Definition at line 155 of file PythiaWrapper6_4.h.

8.112.1.18 #define pyint9 pyint9_

Definition at line 165 of file PythiaWrapper6_4.h.

8.112.1.19 #define pyints pyints_

Definition at line 229 of file PythiaWrapper6_4.h.

8.112.1.20 #define pyjets pyjets_

Definition at line 42 of file PythiaWrapper6_4.h.

8.112.1.21 #define pylist pylist_

Definition at line 243 of file PythiaWrapper6_4.h.

Referenced by call_pylist().

8.112.1.22 #define pymsrv pymsrv_

Definition at line 197 of file PythiaWrapper6_4.h.

8.112.1.23 #define pypars pypars_

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 95 of file PythiaWrapper6_4.h.

Referenced by event_selection(), initPythia(), main(), pythia_out(), and writePythiaStreamIO().

8.112.1.24 #define pyrvnv pyrvnv_

Definition at line 210 of file PythiaWrapper6_4.h.

8.112.1.25 #define pyrvpm pyrvpm_

Definition at line 222 of file PythiaWrapper6_4.h.

8.112.1.26 #define pyssm pyssm_

Definition at line 173 of file PythiaWrapper6_4.h.

8.112.1.27 #define pyssmt pyssmt_

Definition at line 188 of file PythiaWrapper6_4.h.

8.112.1.28 #define pystat pystat_

Definition at line 244 of file PythiaWrapper6_4.h.

Referenced by call_pystat().

8.112.1.29 #define pysubs pysubs_

Definition at line 85 of file PythiaWrapper6_4.h.

Referenced by initPythia().

8.112.1.30 #define upevnt upevnt_

Definition at line 247 of file PythiaWrapper6_4.h.

8.112.1.31 #define upinit upinit_

Definition at line 246 of file PythiaWrapper6_4.h.

8.112.2 Function Documentation

8.112.2.1 void call_pyevnt ( ) [inline]

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 265 of file PythiaWrapper6_4.h.

References pyevnt.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

8.112.2.2 void call_pyhepc (int *mode*) [inline]

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 259 of file PythiaWrapper6_4.h.

References pyhepc.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

8.112.2.3 void call_pyinit (const char * *frame,* const char * *beam,* const char * *target,* double *win*) [inline]

Definition at line 260 of file PythiaWrapper6_4.h.

References pyinit.

Referenced by initPythia().

8.112.2.4 void call_pylist (int *mode*) [inline]

Definition at line 263 of file PythiaWrapper6_4.h.

References pylist.

8.112.2.5 void call_pystat (int *mode*) [inline]

Examples:

example_MyPythiaOnlyToHepMC.cc, **fio/example_MyPythia.cc**, **fio/example_PythiaStreamIO.cc**, and **fio/testPythiaCopies.cc**.

Definition at line 264 of file PythiaWrapper6_4.h.

References pystat.

Referenced by event_selection(), main(), pythia_in_out(), pythia_out(), pythia_particle_out(), and writePythiaStreamIO().

8.112.2.6 void initpydata (void)

Definition at line 6 of file examples/fio/initpydata.f.

8.112.2.7 void pydata (void)

8.112.2.8 void pyevnt ( )

8.112.2.9 C void pyhepc (int *)

8.112.2.10 void pyinit (const char * , const char * , const char * , double * , int , int , int)

8.112.2.11 void pylist (int *)

8.112.2.12 void pystat (int *)

8.112.2.13 void upevnt ( )

8.112.2.14 void upinit ( )

8.112.3 Variable Documentation

8.112.3.1 double brat[8000]

Definition at line 65 of file PythiaWrapper6_4.h.

8.112.3.2 double ckin[200]

Definition at line 82 of file PythiaWrapper6_4.h.

8.112.3.3 double coef[20][500]

Definition at line 108 of file PythiaWrapper6_4.h.

8.112.3.4 int icol[2][4][40]

Definition at line 109 of file PythiaWrapper6_4.h.

8.112.3.5 int iset[500]

Definition at line 107 of file PythiaWrapper6_4.h.

8.112.3.6 int k[5][pyjets_maxn]

Definition at line 38 of file PythiaWrapper6_4.h.

8.112.3.7 int kchg[4][500]

Definition at line 56 of file PythiaWrapper6_4.h.

8.112.3.8 int kfdp[5][8000]

Definition at line 66 of file PythiaWrapper6_4.h.

8.112.3.9 int kfin[81][2]

Definition at line 81 of file PythiaWrapper6_4.h.

8.112.3.10 int kfpr[2][500]

Definition at line 107 of file PythiaWrapper6_4.h.

8.112.3.11 int mdcy[3][500]

Definition at line 64 of file PythiaWrapper6_4.h.

8.112.3.12 int mdme[2][8000]

Definition at line 64 of file PythiaWrapper6_4.h.

8.112.3.13 int mint[400]

Definition at line 99 of file PythiaWrapper6_4.h.

8.112.3.14 int mrpy[6]

Definition at line 73 of file PythiaWrapper6_4.h.

8.112.3.15 int msel

Definition at line 81 of file PythiaWrapper6_4.h.

8.112.3.16 int mselpd

Definition at line 81 of file PythiaWrapper6_4.h.

8.112.3.17 int msti[200]

Definition at line 91 of file PythiaWrapper6_4.h.

8.112.3.18 int mstj[200]

Definition at line 48 of file PythiaWrapper6_4.h.

8.112.3.19 int mstp[200]

Definition at line 89 of file PythiaWrapper6_4.h.

8.112.3.20 int mstu[200]

Definition at line 46 of file PythiaWrapper6_4.h.

8.112.3.21 int msub[500]

Definition at line 81 of file PythiaWrapper6_4.h.

8.112.3.22 int mwid[500]

Definition at line 125 of file PythiaWrapper6_4.h.

8.112.3.23 int n

Definition at line 38 of file PythiaWrapper6_4.h.

Referenced by HepMC::WeightContainer::operator[]().

8.112.3.24 int npad

Definition at line 38 of file PythiaWrapper6_4.h.

8.112.3.25 double p[5][pyjets_maxn]

Examples:

example_EventSelection.cc, **example_UsingIterators.cc**, **example_VectorConversion.cc**, **fio/example_MyPythia.cc**, **testHepMCIteration.cc.in**, and **testMass.cc.in**.

Definition at line 39 of file PythiaWrapper6_4.h.

Referenced by HepMC::TempParticleMap::addEndParticle(), HepMC::already_in_vector(), HepMC::IO_HEPEVT::build_end_vertex(), HepMC::IO_HERWIG::build_end_vertex(), HepMC::IO_HEPEVT::build_particle(), HepMC::IO_HERWIG::build_particle(), HepMC::IO_HEPEVT::build_production_vertex(), HepMC::IO_HERWIG::build_production_vertex(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::GenVertex::edge_iterator::edge_iterator(), HepMC::TempParticleMap::end_vertex(), findPiZero(), findW(), HepMC::GenEvent::GenEvent(), main(), massInfo(), IsGoodEvent::operator()(), IsEventGood::operator()(), IsGoodEventMyPythia::operator()(), PrintChildren::operator()(), particleTypes(), HepMC::GenEvent::read(), HepMC::detail::read_particle(), repairUnits(), HepMC::GenEvent::set_barcode(), simpleIter(), simpleIter3(), simpleIter4(), HepMC::GenEvent::valid_beam_particles(), and HepMC::IO_HEPEVT::write_event().

8.112.3.26 double parf[2000]

Definition at line 57 of file PythiaWrapper6_4.h.

8.112.3.27 double pari[200]

Definition at line 92 of file PythiaWrapper6_4.h.

8.112.3.28 double parj[200]

Definition at line 49 of file PythiaWrapper6_4.h.

8.112.3.29 double parp[200]

Definition at line 90 of file PythiaWrapper6_4.h.

8.112.3.30 double paru[200]

Definition at line 47 of file PythiaWrapper6_4.h.

8.112.3.31 double pmas[4][500]

Definition at line 57 of file PythiaWrapper6_4.h.

8.112.3.32 struct { ... } pydat1_

8.112.3.33 struct { ... } pydat2_

8.112.3.34 struct { ... } pydat3_

8.112.3.35 struct { ... } pydatr_

8.112.3.36 struct { ... } pyg2dx_

8.112.3.37 struct { ... } pyint1_

8.112.3.38 struct { ... } pyint2_

8.112.3.39 struct pin3 pyint3_

8.112.3.40 struct { ... } pyint4_

8.112.3.41 struct pin5 pyint5_

8.112.3.42 struct pin7 pyint7_

8.112.3.43 struct pin8 pyint8_

8.112.3.44 struct pin9 pyint9_

8.112.3.45 struct { ... } pyints_

8.112.3.46 struct { ... } pyjets_

8.112.3.47 const int pyjets_maxn =4000

Definition at line 35 of file PythiaWrapper6_4.h.

8.112.3.48 struct { ... } pymsrv_

8.112.3.49 struct { ... } pypars_

8.112.3.50 struct prvnv pyrvnv_

8.112.3.51 struct prvpm pyrvpm_

8.112.3.52 struct pssm pyssm_

8.112.3.53 struct { ... } pyssmt_

8.112.3.54 struct { ... } pysubs_

8.112.3.55 double rrpy[100]

Definition at line 74 of file PythiaWrapper6_4.h.

8.112.3.56 double rvlam[3][3][3]

Definition at line 192 of file PythiaWrapper6_4.h.

8.112.3.57 double rvlamb[3][3][3]

Definition at line 194 of file PythiaWrapper6_4.h.

8.112.3.58 double rvlamp[3][3][3]

Definition at line 193 of file PythiaWrapper6_4.h.

8.112.3.59 double sfmix[4][16]

Definition at line 182 of file PythiaWrapper6_4.h.

8.112.3.60 double smw[2]

Definition at line 181 of file PythiaWrapper6_4.h.

8.112.3.61 double smz[4]

Definition at line 180 of file PythiaWrapper6_4.h.

8.112.3.62 double umix[2][2]

Definition at line 178 of file PythiaWrapper6_4.h.

8.112.3.63 double umixi[2][2]

Definition at line 184 of file PythiaWrapper6_4.h.

8.112.3.64 double v[5][pyjets_maxn]

Examples:

example_UsingIterators.cc, **testHepMCIteration.cc.in**, and **VectorConversion.h**.

Definition at line 39 of file PythiaWrapper6_4.h.

Referenced by HepMC::compareVertices(), HepMC::GenEvent::GenEvent(), main(), HepMC::GenEvent::read(), HepMC::GenEvent::set_barcode(), simpleIter(), simpleIter3(), simpleIter4(), HepMC::GenEvent::write(), and HepMC::IO_HEPEVT::write_event().

8.112.3.65 double vckm[4][4]

Definition at line 57 of file PythiaWrapper6_4.h.

8.112.3.66 double vint[400]

Definition at line 100 of file PythiaWrapper6_4.h.

8.112.3.67 double vmix[2][2]

Definition at line 179 of file PythiaWrapper6_4.h.

8.112.3.68 double vmixi[2][2]

Definition at line 185 of file PythiaWrapper6_4.h.

8.112.3.69 double wids[5][500]

Definition at line 126 of file PythiaWrapper6_4.h.

8.112.3.70 double x1

Examples:

testMass.cc.in.

Definition at line 233 of file PythiaWrapper6_4.h.

Referenced by main(), and HepMC::operator>>().

8.112.3.71 double xxm[20]

Definition at line 226 of file PythiaWrapper6_4.h.

8.112.3.72 double zmix[4][4]

Definition at line 177 of file PythiaWrapper6_4.h.

8.112.3.73 double zmixi[4][4]

Definition at line 183 of file PythiaWrapper6_4.h.

8.113 PythiaWrapper6_4.h File Reference

```
#include <ctype.h>
#include <cstring>
```

Classes

- struct **pin3**
- struct **pin5**
- struct **pin7**
- struct **pin8**
- struct **pin9**
- struct **pssm**
- struct **prvnv**
- struct **prvpm**

Macros

- #define **initpydata** initpydata_
- #define **pyjets pyjets_**
- #define **pydat1 pydat1_**
- #define **pydat2 pydat2_**
- #define **pydat3 pydat3_**
- #define **pydatr pydatr_**
- #define **pysubs pysubs_**
- #define **pypars pypars_**
- #define **pyint1 pyint1_**
- #define **pyint2 pyint2_**
- #define **pyint3 pyint3_**

- #define **pyint4 pyint4_**
- #define **pyint5 pyint5_**
- #define **pyint7 pyint7_**
- #define **pyint8 pyint8_**
- #define **pyint9 pyint9_**
- #define **pyssm pyssm_**
- #define **pyssmt pyssmt_**
- #define **pymsrv pymsrv_**
- #define **pyrvnv pyrvnv_**
- #define **pyrvpm pyrvpm_**
- #define **pyints pyints_**
- #define **pyg2dx pyg2dx_**
- #define **pyhepc** pyhepc_
- #define **pyinit** pyinit_
- #define **pylist** pylist_
- #define **pystat** pystat_
- #define **pyevnt** pyevnt_
- #define **upinit** upinit_
- #define **upevnt** upevnt_
- #define **pydata** pydata_

Functions

- void **initpydata** (void)
- C void **pyhepc** (int ∗)
- void **pyinit** (const char ∗, const char ∗, const char ∗, double ∗, int, int, int)
- void **pylist** (int ∗)
- void **pystat** (int ∗)
- void **pyevnt** ()
- void **upinit** ()
- void **upevnt** ()
- void **call_pyhepc** (int mode)
- void **call_pyinit** (const char ∗frame, const char ∗beam, const char ∗target, double win)
- void **call_pylist** (int mode)
- void **call_pystat** (int mode)
- void **call_pyevnt** ()
- void **pydata** (void)

Variables

- const int **pyjets_maxn** =4000
- struct {
  int **n**
  int **npad**
  int **k** [5][**pyjets_maxn**]
  double **p** [5][**pyjets_maxn**]
  double **v** [5][**pyjets_maxn**]
  } **pyjets_**

- struct {
  int **mstu** [200]
  double **paru** [200]
  int **mstj** [200]
  double **parj** [200]
  } **pydat1_**

- struct {
  int **kchg** [4][500]
  double **pmas** [4][500]
  double **parf** [2000]
  double **vckm** [4][4]
  } **pydat2_**

- struct {
  int **mdcy** [3][500]
  int **mdme** [2][8000]
  double **brat** [8000]
  int **kfdp** [5][8000]
  } **pydat3_**

- struct {
  int **mrpy** [6]
  double **rrpy** [100]
  } **pydatr_**

- struct {
  int **msel**
  int **mselpd**
  int **msub** [500]
  int **kfin** [81][2]
  double **ckin** [200]
  } **pysubs_**

- struct {
  int **mstp** [200]
  double **parp** [200]
  int **msti** [200]
  double **pari** [200]
  } **pypars_**

- struct {
  int **mint** [400]
  double **vint** [400]
  } **pyint1_**

- struct {
  int **iset** [500]
  int **kfpr** [2][500]
  double **coef** [20][500]
  int **icol** [2][4][40]
  } **pyint2_**

- struct **pin3 pyint3_**
- struct {
  int **mwid** [500]
  double **wids** [5][500]
  } **pyint4_**

- struct **pin5 pyint5_**
- struct **pin7 pyint7_**
- struct **pin8 pyint8_**
- struct **pin9 pyint9_**
- struct **pssm pyssm_**

- struct {
  double **zmix** [4][4]
  double **umix** [2][2]
  double **vmix** [2][2]
  double **smz** [4]
  double **smw** [2]
  double **sfmix** [4][16]
  double **zmixi** [4][4]
  double **umixi** [2][2]
  double **vmixi** [2][2]
  } **pyssmt_**

- struct {
  double **rvlam** [3][3][3]
  double **rvlamp** [3][3][3]
  double **rvlamb** [3][3][3]
  } **pymsrv_**

- struct **prvnv pyrvnv_**
- struct **prvpm pyrvpm_**
- struct {
  double **xxm** [20]
  } **pyints_**

- struct {
  double **x1**
  } **pyg2dx_**

8.113.1 Macro Definition Documentation

8.113.1.1 #define initpydata initpydata_

Definition at line 30 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.2 #define pydat1 pydat1_

Definition at line 52 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.3 #define pydat2 pydat2_

Definition at line 60 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.4 #define pydat3 pydat3_

Definition at line 69 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.5 #define pydata pydata_

Definition at line 275 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.6 #define pydatr pydatr_

Definition at line 77 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.7 #define pyevnt pyevnt_

Definition at line 245 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.8 #define pyg2dx pyg2dx_

Definition at line 236 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.9 #define pyhepc pyhepc_

Definition at line 241 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.10 #define pyinit pyinit_

Definition at line 242 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.11 #define pyint1 pyint1_

Definition at line 103 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.12 #define pyint2 pyint2_

Definition at line 112 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.13 #define pyint3 pyint3_

Definition at line 121 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.14 #define pyint4 pyint4_

Definition at line 129 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.15 #define pyint5 pyint5_

Definition at line 137 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.16 #define pyint7 pyint7_

Definition at line 144 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.17 #define pyint8 pyint8_

Definition at line 155 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.18 #define pyint9 pyint9_

Definition at line 165 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.19 #define pyints pyints_

Definition at line 229 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.20 #define pyjets pyjets_

Definition at line 42 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.21 #define pylist pylist_

Definition at line 243 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.22 #define pymsrv pymsrv_

Definition at line 197 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.23 #define pypars pypars_

Definition at line 95 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.24 #define pyrvnv pyrvnv_

Definition at line 210 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.25 #define pyrvpm pyrvpm_

Definition at line 222 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.26 #define pyssm pyssm_

Definition at line 173 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.27 #define pyssmt pyssmt_

Definition at line 188 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.28 #define pystat pystat_

Definition at line 244 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.29 #define pysubs pysubs_

Definition at line 85 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.30 #define upevnt upevnt_

Definition at line 247 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.1.31 #define upinit upinit_

Definition at line 246 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.2 Function Documentation

8.113.2.1 void call_pyevnt ( ) [inline]

Definition at line 265 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

References pyevnt.

8.113.2.2 void call_pyhepc (int *mode*) [inline]

Definition at line 259 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

References pyhepc.

8.113.2.3 void call_pyinit (const char * *frame*, const char * *beam*, const char * *target*, double *win*) [inline]

Definition at line 260 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

References pyinit.

8.113.2.4 void call_pylist (int *mode*) [inline]

Definition at line 263 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

References pylist.

8.113.2.5 void call_pystat (int *mode*) [inline]

Definition at line 264 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

References pystat.

8.113.2.6 void initpydata (void)

Definition at line 6 of file examples/fio/initpydata.f.

References pydata.

8.113.2.7 void pydata (void)

8.113.2.8 void pyevnt ( )

8.113.2.9 C void pyhepc (int *)

8.113.2.10 void pyinit (const char *, const char *, const char *, double *, int, int, int)

8.113.2.11 void pylist (int *)

8.113.2.12 void pystat (int *)

8.113.2.13 void upevnt ( )

8.113.2.14 void upinit ( )

8.113.3 Variable Documentation

8.113.3.1 double brat[8000]

Definition at line 65 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.2 double ckin[200]

Definition at line 82 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.3 double coef[20][500]

Definition at line 108 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.4 int icol[2][4][40]

Definition at line 109 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.5 int iset[500]

Definition at line 107 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.6 int k[5][pyjets_maxn]

Definition at line 38 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.7 int kchg[4][500]

Definition at line 56 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.8 int kfdp[5][8000]

Definition at line 66 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.9 int kfin[81][2]

Definition at line 81 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.10 int kfpr[2][500]

Definition at line 107 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.11 int mdcy[3][500]

Definition at line 64 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.12 int mdme[2][8000]

Definition at line 64 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.13 int mint[400]

Definition at line 99 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.14 int mrpy[6]

Definition at line 73 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.15 int msel

Definition at line 81 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.16 int mselpd

Definition at line 81 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.17 int msti[200]

Definition at line 91 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.18 int mstj[200]

Definition at line 48 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.19 int mstp[200]

Definition at line 89 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.20 int mstu[200]

Definition at line 46 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.21 int msub[500]

Definition at line 81 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.22 int mwid[500]

Definition at line 125 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.23 int n

Definition at line 38 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.24 int npad

Definition at line 38 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.25 double p[5][pyjets_maxn]

Definition at line 39 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.26 double parf[2000]

Definition at line 57 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.27 double pari[200]

Definition at line 92 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.28 double parj[200]

Definition at line 49 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.29 double parp[200]

Definition at line 90 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.30 double paru[200]

Definition at line 47 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.31 double pmas[4][500]

Definition at line 57 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.32 struct { ... } pydat1_

8.113.3.33 struct { ... } pydat2_

8.113.3.34 struct { ... } pydat3_

8.113.3.35 struct { ... } pydatr_

8.113.3.36 struct { ... } pyg2dx_

8.113.3.37 struct { ... } pyint1_

8.113.3.38 struct { ... } pyint2_

8.113.3.39 struct pin3 pyint3_

8.113.3.40 struct { ... } pyint4_

8.113.3.41 struct pin5 pyint5_

8.113.3.42 struct pin7 pyint7_

8.113.3.43 struct pin8 pyint8_

8.113.3.44 struct pin9 pyint9_

8.113.3.45 struct { ... } pyints_

8.113.3.46 struct { ... } pyjets_

8.113.3.47 const int pyjets_maxn =4000

Definition at line 35 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.48 struct { ... } pymsrv_

8.113.3.49 struct { ... } pypars_

8.113.3.50 struct prvnv pyrvnv_

8.113.3.51 struct prvpm pyrvpm_

8.113.3.52 struct pssm pyssm_

8.113.3.53 struct { ... } pyssmt_

8.113.3.54 struct { ... } pysubs_

8.113.3.55 double rrpy[100]

Definition at line 74 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.56 double rvlam[3][3][3]

Definition at line 192 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.57 double rvlamb[3][3][3]

Definition at line 194 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.58 double rvlamp[3][3][3]

Definition at line 193 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.59 double sfmix[4][16]

Definition at line 182 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.60 double smw[2]

Definition at line 181 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.61 double smz[4]

Definition at line 180 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.62 double umix[2][2]

Definition at line 178 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.63 double umixi[2][2]

Definition at line 184 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.64 double v[5][pyjets_maxn]

Definition at line 39 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.65 double vckm[4][4]

Definition at line 57 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.66 double vint[400]

Definition at line 100 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.67 double vmix[2][2]

Definition at line 179 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.68 double vmixi[2][2]

Definition at line 185 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.69 double wids[5][500]

Definition at line 126 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.70 double x1

Definition at line 233 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.71 double xxm[20]

Definition at line 226 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.72 double zmix[4][4]

Definition at line 177 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.113.3.73 double zmixi[4][4]

Definition at line 183 of file 2.06.09/HepMC/PythiaWrapper6_4.h.

8.114 PythiaWrapper6_4_WIN32.h File Reference

8.115 PythiaWrapper6_4_WIN32.h File Reference

8.116 SearchVector.cc File Reference

```
#include "HepMC/SearchVector.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::not_in_vector** (std::vector< **HepMC::GenParticle** * > *, GenParticle *)

  returns true if it cannot find GenParticle in the vector*

- std::vector
  < **HepMC::GenParticle** * >
  ::iterator **HepMC::already_in_vector** (std::vector< GenParticle * > ***v**, GenParticle ***p**)

  *returns true if **GenParticle** (*p. *106) is in the vector*

8.117 SearchVector.cc File Reference

```
#include "HepMC/SearchVector.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::not_in_vector** (std::vector< **HepMC::GenParticle** * > *, GenParticle *)

  returns true if it cannot find GenParticle in the vector*

- std::vector
  < **HepMC::GenParticle** * >
  ::iterator **HepMC::already_in_vector** (std::vector< GenParticle * > ***v**, GenParticle ***p**)

  *returns true if **GenParticle** (*p. *106) is in the vector*

8.118 SearchVector.h File Reference

```
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::not_in_vector** (std::vector< **HepMC::GenParticle** * > *, GenParticle *)

  returns true if it cannot find GenParticle in the vector*
- std::vector
  < **HepMC::GenParticle** * >
  ::iterator **HepMC::already_in_vector** (std::vector< GenParticle * > ***v**, GenParticle ***p**)

  *returns true if **GenParticle** (*p. *106) is in the vector*

8.119 SearchVector.h File Reference

```
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
```

Namespaces

- **HepMC**

Functions

- bool **HepMC::not_in_vector** (std::vector< **HepMC::GenParticle** * > *, GenParticle *)

  returns true if it cannot find GenParticle in the vector*
- std::vector
  < **HepMC::GenParticle** * >
  ::iterator **HepMC::already_in_vector** (std::vector< GenParticle * > ***v**, GenParticle ***p**)

  *returns true if **GenParticle** (*p. *106) is in the vector*

8.120 SimpleVector.h File Reference

```
#include "HepMC/enable_if.h"
#include "HepMC/is_arithmetic.h"
#include "HepMC/SimpleVector.icc"
```

Classes

- class **HepMC::FourVector**

  ***FourVector** (*p. *54) is a simple representation of a physics 4 vector.*

- class **HepMC::ThreeVector**

  ThreeVector *(*p. *271) is a simple representation of a position or displacement 3 vector.*

Namespaces

- **HepMC**

8.121 SimpleVector.h File Reference

```
#include "HepMC/enable_if.h"
#include "HepMC/is_arithmetic.h"
#include "HepMC/SimpleVector.icc"
```

Classes

- class **HepMC::FourVector**

  FourVector *(*p. *54) is a simple representation of a physics 4 vector.*
- class **HepMC::ThreeVector**

  ThreeVector *(*p. *271) is a simple representation of a position or displacement 3 vector.*

Namespaces

- **HepMC**

8.122 StreamHelpers.cc File Reference

```
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**
- **HepMC::detail**

Functions

- std::istream & **HepMC::detail::read_vertex** (std::istream &, TempParticleMap &, GenVertex ∗)
- std::istream & **HepMC::detail::find_event_end** (std::istream &)

  used to read to the end of a bad event

8.123 StreamHelpers.cc File Reference

```
#include <ostream>
#include <istream>
#include <sstream>
#include "HepMC/GenVertex.h"
#include "HepMC/GenParticle.h"
#include "HepMC/StreamHelpers.h"
#include "HepMC/IO_Exception.h"
```

Namespaces

- **HepMC**
- **HepMC::detail**

Functions

- std::istream & **HepMC::detail::read_vertex** (std::istream &, TempParticleMap &, GenVertex ∗)
- std::istream & **HepMC::detail::find_event_end** (std::istream &)

  used to read to the end of a bad event

8.124 StreamHelpers.h File Reference

```
#include <ostream>
#include <istream>
#include "HepMC/GenEvent.h"
#include "HepMC/TempParticleMap.h"
```

Namespaces

- **HepMC**
- **HepMC::detail**

Functions

- std::ostream & **HepMC::detail::establish_output_stream_info** (std::ostream &)

  *used by **IO_GenEvent** (p. 186) constructor*
- std::istream & **HepMC::detail::establish_input_stream_info** (std::istream &)

  *used by **IO_GenEvent** (p. 186) constructor*
- std::istream & **HepMC::detail::read_vertex** (std::istream &, TempParticleMap &, GenVertex ∗)
- std::istream & **HepMC::detail::read_particle** (std::istream &, TempParticleMap &, GenParticle ∗)
- std::ostream & **HepMC::detail::output** (std::ostream &os, const double &d)

  write a double - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const float &d)

  write a float - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const int &i)

  write an int - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const long &i)

  write a long - for internal use by streaming IO

- std::ostream & **HepMC::detail::output** (std::ostream &os, const char &c)

  write a single char - for internal use by streaming IO
- std::istream & **HepMC::detail::find_event_end** (std::istream &)

  used to read to the end of a bad event

8.125 StreamHelpers.h File Reference

```
#include <ostream>
#include <istream>
#include "HepMC/GenEvent.h"
#include "HepMC/TempParticleMap.h"
```

Namespaces

- **HepMC**
- **HepMC::detail**

Functions

- std::ostream & **HepMC::detail::establish_output_stream_info** (std::ostream &)

  *used by **IO_GenEvent*** (p. *186) constructor*
- std::istream & **HepMC::detail::establish_input_stream_info** (std::istream &)

  *used by **IO_GenEvent*** (p. *186) constructor*
- std::istream & **HepMC::detail::read_vertex** (std::istream &, TempParticleMap &, GenVertex ∗)
- std::istream & **HepMC::detail::read_particle** (std::istream &, TempParticleMap &, GenParticle ∗)
- std::ostream & **HepMC::detail::output** (std::ostream &os, const double &d)

  write a double - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const float &d)

  write a float - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const int &i)

  write an int - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const long &i)

  write a long - for internal use by streaming IO
- std::ostream & **HepMC::detail::output** (std::ostream &os, const char &c)

  write a single char - for internal use by streaming IO
- std::istream & **HepMC::detail::find_event_end** (std::istream &)

  used to read to the end of a bad event

8.126 StreamInfo.cc File Reference

```
#include <string>
#include "HepMC/StreamInfo.h"
```

Namespaces

- **HepMC**

8.127 StreamInfo.cc File Reference

```
#include <string>
#include "HepMC/StreamInfo.h"
```

Namespaces

- **HepMC**

8.128 StreamInfo.h File Reference

```
#include <string>
#include "HepMC/Units.h"
```

Classes

- class **HepMC::StreamInfo**

  StreamInfo *(*p. *260) contains extra information needed when using streaming IO.*

Namespaces

- **HepMC**

Enumerations

- enum **HepMC::known_io** {
  HepMC::gen =1, **HepMC::ascii**, **HepMC::extascii**, **HepMC::ascii_pdt**,
  HepMC::extascii_pdt, **HepMC::gen** =1, **HepMC::ascii**, **HepMC::extascii**,
  HepMC::ascii_pdt, **HepMC::extascii_pdt** }

  The known_io enum is used to track which type of input is being read.

8.129 StreamInfo.h File Reference

```
#include <string>
#include "HepMC/Units.h"
```

Classes

- class **HepMC::StreamInfo**

  StreamInfo *(*p. *260) contains extra information needed when using streaming IO.*

Namespaces

- **HepMC**

Enumerations

- enum **HepMC::known_io** {
  HepMC::gen =1, **HepMC::ascii**, **HepMC::extascii**, **HepMC::ascii_pdt**,
  HepMC::extascii_pdt, **HepMC::gen** =1, **HepMC::ascii**, **HepMC::extascii**,
  HepMC::ascii_pdt, **HepMC::extascii_pdt** }

  The known_io enum is used to track which type of input is being read.

8.130 TempParticleMap.h File Reference

```
#include <map>
```

Classes

- class **HepMC::TempParticleMap**

  TempParticleMap *(*p. *267) is a temporary GenParticle∗ container used during input.*

Namespaces

- **HepMC**

8.131 TempParticleMap.h File Reference

```
#include <map>
```

Classes

- class **HepMC::TempParticleMap**

  TempParticleMap *(*p. *267) is a temporary GenParticle∗ container used during input.*

Namespaces

- **HepMC**

8.132 testFlow.cc File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Typedefs

- typedef std::vector
  < **HepMC::GenParticle** ∗ > **FlowVec**

Functions

- int **main** ()

8.132.1 Typedef Documentation

8.132.1.1 typedef std::vector<HepMC::GenParticle*> FlowVec

Examples:

testFlow.cc.

Definition at line 15 of file HepMC-2.06.09/test/testFlow.cc.

8.132.2 Function Documentation

8.132.2.1 int main ( )

Definition at line 17 of file HepMC-2.06.09/test/testFlow.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::GenParticle::barcode(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::Flow::erase(), HepMC::GenParticle::flow(), HepMC::Units::GEV, HepMC::Flow::icode(), HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::GenParticle::set_flow(), HepMC::GenEvent::set_signal_process_vertex(), HepMC::GenEvent::use_units(), and HepMC::GenEvent::write().

8.133 testFlow.cc File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Typedefs

- typedef std::vector
  < **HepMC::GenParticle** * > **FlowVec**

Functions

- int **main** ()

8.133.1 Typedef Documentation

8.133.1.1 typedef std::vector<HepMC::GenParticle*> FlowVec

Definition at line 15 of file test/testFlow.cc.

8.133.2 Function Documentation

8.133.2.1 int main ( )

Definition at line 17 of file test/testFlow.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::GenParticle::barcode(), HepMC::Flow::connected_partners(), HepMC::Flow::dangling_connected_partners(), HepMC::Flow::erase(), HepMC::GenParticle::flow(), HepMC::Units::GEV, HepMC::Flow::icode(), HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::GenParticle::set_flow(), HepMC::GenEvent::set_signal_process_vertex(), HepMC::GenEvent::use_units(), and HepMC::GenEvent::write().

8.134 testHepMC.cc File Reference

```
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/IO_Ascii.h"
#include "HepMC/IO_AsciiParticles.h"
#include "IsGoodEvent.h"
#include "testHepMCMethods.h"
```

Functions

- void **read_testIOGenEvent** (std::ostream &os)
- void **read_testUnits** (std::ostream &os)
- void **read_variousFormats** (std::ostream &os)
- void **writeWithCrossSection** (std::ostream &os)
- void **readWithCrossSection** (std::ostream &os)
- void **writeWithWeight** (std::ostream &os)
- void **readWithWeight** (std::ostream &os)
- void **read_nan** (std::ostream &os)
- int **main** ()

8.134.1 Function Documentation

8.134.1.1 int main ( )

Definition at line 36 of file HepMC-2.06.09/test/testHepMC.cc.

References read_nan(), read_testIOGenEvent(), read_testUnits(), read_variousFormats(), readWithCrossSection(), readWithWeight(), writeWithCrossSection(), and writeWithWeight().

8.134.1.2 void read_nan (std::ostream & *os*)

Examples:

testHepMC.cc.in.

Definition at line 338 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::IO_GenEvent::error_message(), HepMC::IO_GenEvent::error_type(), HepMC::GenEvent::event_number(), HepMC::Units::GEV, HepMC::IO_Exception::InvalidData, HepMC::Units::MM, HepMC::IO_BaseClass::read_next_event(), and HepMC::IO_GenEvent::use_input_units().

Referenced by main().

8.134.1.3 void read_testIOGenEvent (std::ostream & *os*)

Examples:

testHepMC.cc.in, and **testStreamIO.cc.in**.

Definition at line 50 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenEvent::event_number(), findPiZero(), HepMC::Units::GEV, HepMC::Units::MM, particleTypes(), HepMC::IO_GenEvent::precision(), HepMC::IO_BaseClass::read_next_event(), HepMC::IO_GenEvent::use_input_units(), and HepMC::GenEvent::write_units().

Referenced by main().

8.134.1.4 void read_testUnits (std::ostream & *os*)

Examples:

testHepMC.cc.in.

Definition at line 98 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenEvent::define_units(), HepMC::GenEvent::event_number(), findPiZero(), HepMC::Units::GEV, HepMC::Units::MM, particleTypes(), HepMC::IO_BaseClass::read_next_event(), and HepMC::GenEvent::write_units().

Referenced by main().

8.134.1.5 void read_variousFormats (std::ostream & *os*)

Examples:

testHepMC.cc.in, and **testStreamIO.cc.in**.

Definition at line 140 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenEvent::event_number(), findPiZero(), HepMC::Units::GEV, HepMC::Units::MEV, HepMC::Units::MM, HepMC::Units::name(), HepMC::IO_BaseClass::read_next_event(), repairUnits(), HepMC::IO_GenEvent::use_input_units(), and HepMC::GenEvent::write_units().

Referenced by main().

8.134.1.6 void readWithCrossSection (std::ostream & *os*)

Examples:

testHepMC.cc.in.

Definition at line 311 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenCrossSection::cross_section(), HepMC::GenEvent::cross_section(), HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

Referenced by main().

8.134.1.7 void readWithWeight (std::ostream & *os*)

Examples:

testHepMC.cc.in.

Definition at line 431 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenEvent::cross_section(), HepMC::GenEvent::event_number(), and HepMC::IO_BaseClass::read_next_event().

Referenced by main().

8.134.1.8 void writeWithCrossSection (std::ostream & *os*)

Examples:

testHepMC.cc.in.

Definition at line 245 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenEvent::event_number(), HepMC::Units::GEV, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::WeightContainer::push_back(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenCrossSection::set_cross_section(), HepMC::GenEvent::set_cross_section(), HepMC::IO_GenEvent::use_input_units(), HepMC::GenEvent::weights(), and HepMC::GenEvent::write_cross_section().

Referenced by main().

8.134.1.9 void writeWithWeight (std::ostream & *os*)

Examples:

testHepMC.cc.in.

Definition at line 393 of file HepMC-2.06.09/test/testHepMC.cc.

References HepMC::GenEvent::event_number(), HepMC::Units::GEV, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::WeightContainer::push_back(), HepMC::IO_BaseClass::read_next_event(), HepMC::IO_GenEvent::use_input_units(), and HepMC::GenEvent::weights().

Referenced by main().

8.135 testHepMC.cc File Reference

```
#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#include "HepMC/IO_Ascii.h"
#include "HepMC/IO_AsciiParticles.h"
#include "IsGoodEvent.h"
#include "testHepMCMethods.h"
```

Functions

- void **read_testIOGenEvent** (std::ostream &os)
- void **read_testUnits** (std::ostream &os)
- void **read_variousFormats** (std::ostream &os)
- void **writeWithCrossSection** (std::ostream &os)
- void **readWithCrossSection** (std::ostream &os)
- void **writeWithWeight** (std::ostream &os)
- void **readWithWeight** (std::ostream &os)
- void **read_nan** (std::ostream &os)
- int **main** ()

8.135.1 Function Documentation

8.135.1.1 int main ( )

Definition at line 36 of file test/testHepMC.cc.

References read_nan(), read_testIOGenEvent(), read_testUnits(), read_variousFormats(), readWithCrossSection(), readWithWeight(), writeWithCrossSection(), and writeWithWeight().

8.135.1.2 void read_nan (std::ostream & *os*)

8.135.1.3 void read_testIOGenEvent (std::ostream & *os*)

8.135.1.4 void read_testUnits (std::ostream & *os*)

8.135.1.5 void read_variousFormats (std::ostream & *os*)

8.135.1.6 void readWithCrossSection (std::ostream & *os*)

8.135.1.7 void readWithWeight (std::ostream & *os*)

8.135.1.8 void writeWithCrossSection (std::ostream & *os*)

8.135.1.9 void writeWithWeight (std::ostream & *os*)

8.136 testHepMCIteration.cc File Reference

```
#include <list>
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "HepMC/GenRanges.h"
#include "IsGoodEvent.h"
#include "testHepMCIteration.h"
```

Classes

- class **PrintW**
- class **PrintConstW**

Functions

- bool **findW** (**HepMC::GenEvent** ∗evt, std::ofstream &os)
- bool **simpleIter** (**HepMC::GenEvent** ∗evt, std::ostream &os=std::cout)
- bool **simpleIter2** (**HepMC::GenEvent** ∗evt, std::ostream &os=std::cout)
- bool **simpleIter3** (**HepMC::GenEvent** ∗evt, std::ostream &os=std::cout)
- bool **simpleIter4** (**HepMC::GenEvent** ∗evt, std::ostream &os=std::cout)
- int **main** ()

8.136.1 Function Documentation

8.136.1.1 bool findW (HepMC::GenEvent * *evt*, std::ofstream & *os*)

Examples:

testHepMCIteration.cc.in.

Definition at line 295 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::children, HepMC::descendants, HepMC::GenEvent::event_number(), IsWBoson(), p, HepMC::parents, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

Referenced by main().

8.136.1.2 int main ( )

Definition at line 121 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::GenEvent::clear(), HepMC::GenEvent::event_number(), findW(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::IO_BaseClass::read_next_event(), simpleIter(), simpleIter2(), simpleIter3(), and simpleIter4().

8.136.1.3 bool simpleIter (HepMC::GenEvent * *evt*, std::ostream & *os* = `std::cout`)

Examples:

testHepMCIteration.cc.in.

Definition at line 178 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::GenEvent::event_number(), IsPhoton(), p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), v, HepMC::GenEvent::vertices_begin(), and HepMC::GenEvent::vertices_end().

Referenced by main().

8.136.1.4 bool simpleIter2 (HepMC::GenEvent * *evt*, std::ostream & *os* = `std::cout`)

Examples:

testHepMCIteration.cc.in.

Definition at line 207 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::copy_if(), HepMC::GenEvent::event_number(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::vertices_begin(), and HepMC::GenEvent::vertices_end().

Referenced by main().

8.136.1.5 bool simpleIter3 (HepMC::GenEvent * *evt*, std::ostream & *os* = `std::cout`)

Examples:

testHepMCIteration.cc.in.

Definition at line 234 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::GenEventVertexRange::begin(), HepMC::GenEventParticleRange::begin(), HepMC::GenEventVertexRange::end(), HepMC::GenEventParticleRange::end(), HepMC::GenEvent::event_number(), p, and v.

Referenced by main().

8.136.1.6 bool simpleIter4 (HepMC::GenEvent * *evt*, std::ostream & *os* = `std::cout`)

Examples:

testHepMCIteration.cc.in.

Definition at line 262 of file HepMC-2.06.09/test/testHepMCIteration.cc.

References HepMC::GenEventVertexRange::begin(), HepMC::GenEventParticleRange::begin(), HepMC::GenEventVertexRange::end(), HepMC::GenEventParticleRange::end(), HepMC::GenEvent::event_number(), p, HepMC::GenEvent::particle_range(), v, and HepMC::GenEvent::vertex_range().

Referenced by main().

8.137 testHepMCIteration.cc File Reference

```
#include <list>
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "HepMC/GenRanges.h"
#include "IsGoodEvent.h"
#include "testHepMCIteration.h"
```

Classes

- class **PrintW**
- class **PrintConstW**

Functions

- bool **findW** (**HepMC::GenEvent** *evt, std::ofstream &os)
- bool **simpleIter** (**HepMC::GenEvent** *evt, std::ostream &os=std::cout)
- bool **simpleIter2** (**HepMC::GenEvent** *evt, std::ostream &os=std::cout)
- bool **simpleIter3** (**HepMC::GenEvent** *evt, std::ostream &os=std::cout)
- bool **simpleIter4** (**HepMC::GenEvent** *evt, std::ostream &os=std::cout)
- int **main** ()

8.137.1 Function Documentation

8.137.1.1 bool findW (HepMC::GenEvent * *evt*, std::ofstream & *os*)

8.137.1.2 int main ( )

Definition at line 121 of file test/testHepMCIteration.cc.

References HepMC::GenEvent::clear(), HepMC::GenEvent::event_number(), findW(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::IO_BaseClass::read_next_event(), simpleIter(), simpleIter2(), simpleIter3(), and simpleIter4().

8.137.1.3 bool simpleIter (HepMC::GenEvent * *evt*, std::ostream & *os* = `std::cout`)

8.137.1.4 bool simpleIter2 (HepMC::GenEvent * *evt*, std::ostream & *os* = `std::cout`)

8.137.1.5 bool simpleIter3 (HepMC::GenEvent * *evt*, std::ostream & *os* = std::cout)

8.137.1.6 bool simpleIter4 (HepMC::GenEvent * *evt*, std::ostream & *os* = std::cout)

8.138 testHepMCIteration.h File Reference

Classes

- class **IsFinalState**
- class **PrintPhoton**
- class **PrintParticle**
- class **PrintChildren**

  test class
- class **PrintDescendants**

  test class

Functions

- bool **IsPhoton** (const **HepMC::GenParticle** ∗**p**)

  returns true if the GenParticle particle is a photon with pT > 10 GeV
- bool **IsWBoson** (const **HepMC::GenParticle** ∗**p**)

  returns true if the GenParticle is a W+/W-

8.138.1 Function Documentation

8.138.1.1 bool IsPhoton (const HepMC::GenParticle * *p*)

returns true if the GenParticle particle is a photon with pT > 10 GeV

Examples:

testHepMCIteration.cc.in.

Definition at line 10 of file testHepMCIteration.h.

References HepMC::GenParticle::momentum(), HepMC::GenParticle::pdg_id(), and HepMC::FourVector::perp().

Referenced by PrintPhoton::operator()(), and simpleIter().

8.138.1.2 bool IsWBoson (const HepMC::GenParticle * *p*)

returns true if the GenParticle is a W+/W-

Examples:

testHepMCIteration.cc.in.

Definition at line 17 of file testHepMCIteration.h.

References HepMC::GenParticle::pdg_id().

Referenced by findW(), PrintW::operator()(), and PrintConstW::operator()().

8.139 testHepMCMethods.cc File Reference

```
#include "testHepMCMethods.h"
```

Functions

- double **findPiZero** (**HepMC::GenEvent** ∗evt)
- void **particleTypes** (**HepMC::GenEvent** ∗evt, std::ostream &os)
- void **repairUnits** (**HepMC::GenEvent** ∗evt, **HepMC::Units::MomentumUnit** from, **HepMC::Units::-MomentumUnit** to)

8.139.1 Function Documentation

8.139.1.1 double findPiZero (HepMC::GenEvent ∗ *evt*)

Examples:

testHepMC.cc.in, and **testStreamIO.cc.in**.

Definition at line 11 of file HepMC-2.06.09/test/testHepMCMethods.cc.

Referenced by read_testIOGenEvent(), read_testUnits(), and read_variousFormats().

8.139.1.2 void particleTypes (HepMC::GenEvent ∗ *evt,* std::ostream & *os*)

Examples:

testHepMC.cc.in, and **testStreamIO.cc.in**.

Definition at line 22 of file HepMC-2.06.09/test/testHepMCMethods.cc.

Referenced by read_from_stream4(), read_testIOGenEvent(), read_testUnits(), write_to_stream(), and write_to_-stream3().

8.139.1.3 void repairUnits (HepMC::GenEvent ∗ *evt,* HepMC::Units::MomentumUnit *from,* HepMC::Units::MomentumUnit *to*)

Examples:

testHepMC.cc.in.

Definition at line 78 of file HepMC-2.06.09/test/testHepMCMethods.cc.

Referenced by read_variousFormats().

8.140 testHepMCMethods.cc File Reference

```
#include "testHepMCMethods.h"
```

Functions

- double **findPiZero** (**HepMC::GenEvent** ∗evt)

- void **particleTypes** (**HepMC::GenEvent** ∗evt, std::ostream &os)
- void **repairUnits** (**HepMC::GenEvent** ∗evt, **HepMC::Units::MomentumUnit** from, **HepMC::Units::-MomentumUnit** to)

8.140.1 Function Documentation

8.140.1.1 double findPiZero (HepMC::GenEvent ∗ *evt*)

Definition at line 11 of file test/testHepMCMethods.cc.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

8.140.1.2 void particleTypes (HepMC::GenEvent ∗ *evt,* std::ostream & *os*)

Definition at line 22 of file test/testHepMCMethods.cc.

References HepMC::GenEvent::event_number(), p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent-::particles_end(), and HepMC::GenEvent::particles_size().

8.140.1.3 void repairUnits (HepMC::GenEvent ∗ *evt,* HepMC::Units::MomentumUnit *from,* HepMC::Units::MomentumUnit *to*)

Definition at line 78 of file test/testHepMCMethods.cc.

References HepMC::Units::conversion_factor(), HepMC::FourVector::e(), p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::FourVector::px(), HepMC::FourVector::py(), and HepMC::FourVector-::pz().

8.141 testHepMCMethods.h File Reference

```
#include "HepMC/GenEvent.h"
```

Functions

- double **findPiZero** (**HepMC::GenEvent** ∗)
- void **particleTypes** (**HepMC::GenEvent** ∗, std::ostream &os=std::cout)
- void **repairUnits** (**HepMC::GenEvent** ∗, **HepMC::Units::MomentumUnit**, **HepMC::Units::Momentum-Unit**)

8.141.1 Function Documentation

8.141.1.1 double findPiZero (HepMC::GenEvent ∗)

Definition at line 11 of file HepMC-2.06.09/test/testHepMCMethods.cc.

References p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

Referenced by read_testIOGenEvent(), read_testUnits(), and read_variousFormats().

8.141.1.2 void particleTypes (HepMC::GenEvent ∗ , std::ostream & *os* = `std::cout`)

Definition at line 22 of file HepMC-2.06.09/test/testHepMCMethods.cc.

References HepMC::GenEvent::event_number(), p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), and HepMC::GenEvent::particles_size().

Referenced by read_from_stream4(), read_testIOGenEvent(), read_testUnits(), write_to_stream(), and write_to_stream3().

8.141.1.3 void repairUnits (HepMC::GenEvent * , HepMC::Units::MomentumUnit , HepMC::Units::MomentumUnit)

Definition at line 78 of file HepMC-2.06.09/test/testHepMCMethods.cc.

References HepMC::Units::conversion_factor(), HepMC::FourVector::e(), p, HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::FourVector::px(), HepMC::FourVector::py(), and HepMC::FourVector::pz().

Referenced by read_variousFormats().

8.142 testHerwigCopies.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Functions

- void **hwaend_** ()
- int **main** ()

8.142.1 Function Documentation

8.142.1.1 void hwaend_ ( )

Definition at line 16 of file examples/fio/testHerwigCopies.cc.

8.142.1.2 int main ( )

Definition at line 18 of file examples/fio/testHerwigCopies.cc.

References HepMC::compareGenEvent(), HepMC::GenEvent::event_number(), HepMC::getHerwigCrossSection(), HepMC::Units::GEV, hwbgen, hwbmch, hwcdec, hwcfor, hwdhad, hwdhob, hwdhvy, hwefin, hweini, hwepro, hwevnt, hwigin, hwmevt, hwproc, hwufne, hwuinc, hwuine, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.143 testHerwigCopies.cc File Reference

```
#include <fstream>
```

```
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"
```

Functions

- void **hwaend_** ()
- int **main** ()

8.143.1 Function Documentation

8.143.1.1 void hwaend_ ( )

Definition at line 16 of file HepMC-2.06.09/examples/fio/testHerwigCopies.cc.

8.143.1.2 int main ( )

Definition at line 18 of file HepMC-2.06.09/examples/fio/testHerwigCopies.cc.

References HepMC::compareGenEvent(), HepMC::GenEvent::event_number(), HepMC::getHerwigCrossSection(), HepMC::Units::GEV, hwbgen, hwbmch, hwcdec, hwcfor, hwdhad, hwdhob, hwdhvy, hwefin, hweini, hwepro, hwevnt, hwigin, hwmevt, hwproc, hwufne, hwuinc, hwuine, HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::GenEvent::set_event_number(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_signal_process_id(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), and HepMC::GenEvent::use_units().

8.144 testMass.cc File Reference

```
#include <cmath>
#include <ostream>
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/Version.h"
#include "IsGoodEvent.h"
```

Functions

- void **massInfo** (const **HepMC::GenEvent** *, std::ostream &os)
- int **main** ()

8.144.1 Function Documentation

8.144.1.1 int main ( )

Definition at line 24 of file HepMC-2.06.09/test/testMass.cc.

References HepMC::GenEvent::beam_particles(), HepMC::GenEvent::event_number(), HepMC::Units::GEV, massInfo(), HepMC::Units::MM, HepMC::GenParticle::momentum(), HepMC::WeightContainer::print(), HepMC::IO_BaseClass::read_next_event(), HepMC::FourVector::rho(), HepMC::GenEvent::set_heavy_ion(), HepMC::GenEvent::set_pdf_info(), HepMC::WeightContainer::size(), HepMC::IO_GenEvent::use_input_units(), HepMC::GenEvent::valid_beam_particles(), HepMC::version(), HepMC::GenEvent::weights(), and x1.

8.144.1.2 void massInfo (const HepMC::GenEvent * e, std::ostream & os)

Examples:

testMass.cc.in.

Definition at line 124 of file HepMC-2.06.09/test/testMass.cc.

References HepMC::GenEvent::event_number(), p, HepMC::GenEvent::particles_begin(), and HepMC::GenEvent::particles_end().

Referenced by main().

8.145 testMass.cc File Reference

```
#include <cmath>
#include <ostream>
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/Version.h"
#include "IsGoodEvent.h"
```

Functions

- void **massInfo** (const **HepMC::GenEvent** *, std::ostream &os)
- int **main** ()

8.145.1 Function Documentation

8.145.1.1 int main ( )

Definition at line 24 of file test/testMass.cc.

References HepMC::GenEvent::beam_particles(), HepMC::GenEvent::event_number(), HepMC::Units::GEV, massInfo(), HepMC::Units::MM, HepMC::GenParticle::momentum(), HepMC::WeightContainer::print(), HepMC::IO_BaseClass::read_next_event(), HepMC::FourVector::rho(), HepMC::GenEvent::set_heavy_ion(), HepMC::GenEvent::set_pdf_info(), HepMC::WeightContainer::size(), HepMC::IO_GenEvent::use_input_units(), HepMC::GenEvent::valid_beam_particles(), HepMC::version(), HepMC::GenEvent::weights(), and x1.

8.145.1.2 void massInfo (const HepMC::GenEvent * e, std::ostream & os)

8.146 testMultipleCopies.cc File Reference

```
#include <fstream>
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "IsGoodEvent.h"
```

Functions

- int **main** ()

8.146.1 Function Documentation

8.146.1.1 int main ( )

Definition at line 19 of file HepMC-2.06.09/test/testMultipleCopies.cc.

References HepMC::compareGenEvent(), HepMC::GenEvent::event_number(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::particles_size(), HepMC::GenEvent::print(), HepMC::IO_BaseClass::read_next_event(), and HepMC::GenEvent::vertices_size().

8.147 testMultipleCopies.cc File Reference

```
#include <fstream>
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "IsGoodEvent.h"
```

Functions

- int **main** ()

8.147.1 Function Documentation

8.147.1.1 int main ( )

Definition at line 19 of file test/testMultipleCopies.cc.

References HepMC::compareGenEvent(), HepMC::GenEvent::event_number(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::particles_size(), HepMC::GenEvent::print(), HepMC::IO_BaseClass::read_next_event(), and HepMC::GenEvent::vertices_size().

8.148 testPolarization.cc File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Functions

- int **main** ()

8.148.1 Function Documentation

8.148.1.1 int main ( )

Definition at line 14 of file HepMC-2.06.09/test/testPolarization.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::print(), HepMC::GenParticle::set_flow(), HepMC::GenParticle::set_polarization(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::GenEvent::write().

8.149 testPolarization.cc File Reference

```
#include <iostream>
#include <fstream>
#include <vector>
#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"
```

Functions

- int **main** ()

8.149.1 Function Documentation

8.149.1.1 int main ( )

Definition at line 14 of file test/testPolarization.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::GenEvent::particles_begin(), HepMC::GenEvent::particles_end(), HepMC::GenEvent::print(), HepMC::GenParticle::set_flow(), HepMC::GenParticle::set_polarization(), HepMC::GenEvent::set_signal_process_vertex(), and HepMC::GenEvent::write().

8.150 testPrintBug.cc File Reference

```
#include <fstream>
#include "HepMC/GenEvent.h"
#include "HepMC/SimpleVector.h"
```

Functions

- int **main** ()

8.150.1 Function Documentation

8.150.1.1 int main ( )

Definition at line 10 of file HepMC-2.06.09/test/testPrintBug.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::Units::GEV, HepMC::Units::MM, HepMC::GenEvent::print(), and HepMC::GenEvent::use_units().

8.151 testPrintBug.cc File Reference

```
#include <fstream>
#include "HepMC/GenEvent.h"
#include "HepMC/SimpleVector.h"
```

Functions

- int **main** ()

8.151.1 Function Documentation

8.151.1.1 int main ( )

Definition at line 10 of file test/testPrintBug.cc.

References HepMC::GenVertex::add_particle_in(), HepMC::GenVertex::add_particle_out(), HepMC::GenEvent::add_vertex(), HepMC::Units::GEV, HepMC::Units::MM, HepMC::GenEvent::print(), and HepMC::GenEvent::use_units().

8.152 testPythiaCopies.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

8.152.1 Function Documentation

8.152.1.1 int main ( )

Definition at line 16 of file examples/fio/testPythiaCopies.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::compareGenEvent(), HepMC::GenEvent::event_number(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::WeightContainer::push_back(), pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), HepMC::GenEvent::use_units(), and HepMC::GenEvent::weights().

8.153 testPythiaCopies.cc File Reference

```
#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "PythiaHelper.h"
```

Functions

- int **main** ()

8.153.1 Function Documentation

8.153.1.1 int main ( )

Definition at line 16 of file HepMC-2.06.09/examples/fio/testPythiaCopies.cc.

References call_pyevnt(), call_pyhepc(), call_pystat(), HepMC::compareGenEvent(), HepMC::GenEvent::event_number(), HepMC::getPythiaCrossSection(), HepMC::Units::GEV, initPythia(), HepMC::Units::MM, HepMC::GenEvent::print(), HepMC::WeightContainer::push_back(), pypars, HepMC::IO_BaseClass::read_next_event(), HepMC::GenEvent::set_cross_section(), HepMC::HEPEVT_Wrapper::set_max_number_entries(), HepMC::GenEvent::set_mpi(), HepMC::HEPEVT_Wrapper::set_sizeof_real(), HepMC::GenEvent::use_units(), and HepMC::GenEvent::weights().

8.154 testSimpleVector.cc File Reference

```
#include <iostream>
#include "HepMC/SimpleVector.h"
```

Functions

- int **main** ()

8.154.1 Function Documentation

8.154.1.1 int main ( )

Definition at line 8 of file HepMC-2.06.09/test/testSimpleVector.cc.

References HepMC::FourVector::e(), HepMC::FourVector::eta(), HepMC::FourVector::m(), HepMC::FourVector::m2(), HepMC::FourVector::perp(), HepMC::ThreeVector::perp(), HepMC::FourVector::perp2(), HepMC::ThreeVector::perp2(), HepMC::FourVector::phi(), HepMC::ThreeVector::phi(), HepMC::FourVector::pseudoRapidity(), HepMC::FourVector::px(), HepMC::FourVector::py(), HepMC::FourVector::pz(), HepMC::ThreeVector::r(), HepMC::FourVector::rho(), HepMC::FourVector::set(), HepMC::ThreeVector::set(), HepMC::FourVector::setE(), HepMC::ThreeVector::setPhi(), HepMC::FourVector::setPx(), HepMC::FourVector::setPy(), HepMC::FourVector::setPz(), HepMC::FourVector::setT(), HepMC::ThreeVector::setTheta(), HepMC::FourVector::setX(), HepMC::ThreeVector::setX(), HepMC::FourVector::setY(), HepMC::ThreeVector::setY(), HepMC::FourVector::setZ(), HepMC::ThreeVector::setZ(), HepMC::FourVector::t(), HepMC::FourVector::theta(), HepMC::ThreeVector::theta(), HepMC::Four-

Vector::x(), HepMC::ThreeVector::x(), HepMC::FourVector::y(), HepMC::ThreeVector::y(), HepMC::FourVector::z(), and HepMC::ThreeVector::z().

8.155 testSimpleVector.cc File Reference

```
#include <iostream>
#include "HepMC/SimpleVector.h"
```

Functions

- int **main** ()

8.155.1 Function Documentation

8.155.1.1 int main ( )

Definition at line 8 of file test/testSimpleVector.cc.

References HepMC::FourVector::e(), HepMC::FourVector::eta(), HepMC::FourVector::m(), HepMC::FourVector::m2(), HepMC::FourVector::perp(), HepMC::ThreeVector::perp(), HepMC::FourVector::perp2(), HepMC::ThreeVector::perp2(), HepMC::FourVector::phi(), HepMC::ThreeVector::phi(), HepMC::FourVector::pseudoRapidity(), HepMC::FourVector::px(), HepMC::FourVector::py(), HepMC::FourVector::pz(), HepMC::ThreeVector::r(), HepMC::FourVector::rho(), HepMC::FourVector::set(), HepMC::ThreeVector::set(), HepMC::FourVector::setE(), HepMC::ThreeVector::setPhi(), HepMC::FourVector::setPx(), HepMC::FourVector::setPy(), HepMC::FourVector::setPz(), HepMC::FourVector::setT(), HepMC::ThreeVector::setTheta(), HepMC::FourVector::setX(), HepMC::ThreeVector::setX(), HepMC::FourVector::setY(), HepMC::ThreeVector::setY(), HepMC::FourVector::setZ(), HepMC::ThreeVector::setZ(), HepMC::FourVector::t(), HepMC::FourVector::theta(), HepMC::ThreeVector::theta(), HepMC::FourVector::x(), HepMC::ThreeVector::x(), HepMC::FourVector::y(), HepMC::ThreeVector::y(), HepMC::FourVector::z(), and HepMC::ThreeVector::z().

8.156 testStreamIO.cc File Reference

```
#include <fstream>
#include "HepMC/GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/Version.h"
#include "HepMC/IO_Exception.h"
#include "IsGoodEvent.h"
#include "testHepMCMethods.h"
```

Functions

- void **read_testIOGenEvent** (std::ostream &os)
- void **read_variousFormats** (std::ostream &os)
- void **write_to_stream** (std::ostream &os)
- void **write_to_stream3** (std::ostream &os)
- void **read_from_stream4** (std::ostream &os)
- int **main** ()

8.156.1 Function Documentation

8.156.1.1 int main ( )

Definition at line 30 of file HepMC-2.06.09/test/testStreamIO.cc.

References read_from_stream4(), read_testIOGenEvent(), read_variousFormats(), write_to_stream(), and write_to_stream3().

8.156.1.2 void read_from_stream4 (std::ostream & *os*)

Examples:

testStreamIO.cc.in.

Definition at line 260 of file HepMC-2.06.09/test/testStreamIO.cc.

References HepMC::GenEvent::event_number(), HepMC::GenEvent::is_valid(), particleTypes(), HepMC::GenEvent::read(), HepMC::GenEvent::write(), HepMC::write_HepMC_IO_block_begin(), and HepMC::write_HepMC_IO_block_end().

Referenced by main().

8.156.1.3 void read_testIOGenEvent (std::ostream & *os*)

8.156.1.4 void read_variousFormats (std::ostream & *os*)

8.156.1.5 void write_to_stream (std::ostream & *os*)

Examples:

testStreamIO.cc.in.

Definition at line 41 of file HepMC-2.06.09/test/testStreamIO.cc.

References HepMC::GenEvent::event_number(), HepMC::Units::GEV, HepMC::Units::MM, particleTypes(), HepMC::IO_BaseClass::read_next_event(), HepMC::IO_GenEvent::use_input_units(), HepMC::write_HepMC_IO_block_begin(), and HepMC::write_HepMC_IO_block_end().

Referenced by main().

8.156.1.6 void write_to_stream3 (std::ostream & *os*)

Examples:

testStreamIO.cc.in.

Definition at line 216 of file HepMC-2.06.09/test/testStreamIO.cc.

References HepMC::GenEvent::event_number(), HepMC::Units::GEV, HepMC::Units::MM, particleTypes(), HepMC::IO_BaseClass::read_next_event(), HepMC::IO_GenEvent::use_input_units(), HepMC::GenEvent::write(), HepMC::write_HepMC_IO_block_begin(), and HepMC::write_HepMC_IO_block_end().

Referenced by main().

8.157 testStreamIO.cc File Reference

```
#include <fstream>
```

```
#include "HepMC/GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/Version.h"
#include "HepMC/IO_Exception.h"
#include "IsGoodEvent.h"
#include "testHepMCMethods.h"
```

Functions

- void **read_testIOGenEvent** (std::ostream &os)
- void **read_variousFormats** (std::ostream &os)
- void **write_to_stream** (std::ostream &os)
- void **write_to_stream3** (std::ostream &os)
- void **read_from_stream4** (std::ostream &os)
- int **main** ()

8.157.1 Function Documentation

8.157.1.1 int main ( )

Definition at line 30 of file test/testStreamIO.cc.

References read_from_stream4(), read_testIOGenEvent(), read_variousFormats(), write_to_stream(), and write_to_stream3().

8.157.1.2 void read_from_stream4 (std::ostream & *os*)

8.157.1.3 void read_testIOGenEvent (std::ostream & *os*)

8.157.1.4 void read_variousFormats (std::ostream & *os*)

8.157.1.5 void write_to_stream (std::ostream & *os*)

8.157.1.6 void write_to_stream3 (std::ostream & *os*)

8.158 testUnits.cc File Reference

```
#include <iostream>
#include "HepMC/Units.h"
```

Functions

- int **main** ()

8.158.1 Function Documentation

8.158.1.1 int main ( )

Definition at line 8 of file HepMC-2.06.09/test/testUnits.cc.

References HepMC::Units::CM, HepMC::Units::conversion_factor(), HepMC::Units::default_length_unit(), HepMC::Units::default_momentum_unit(), HepMC::Units::GEV, HepMC::Units::MEV, HepMC::Units::MM, and HepMC::Units::name().

8.159 testUnits.cc File Reference

```
#include <iostream>
#include "HepMC/Units.h"
```

Functions

- int **main** ()

8.159.1 Function Documentation

8.159.1.1 int main ( )

Definition at line 8 of file test/testUnits.cc.

References HepMC::Units::CM, HepMC::Units::conversion_factor(), HepMC::Units::default_length_unit(), HepMC::Units::default_momentum_unit(), HepMC::Units::GEV, HepMC::Units::MEV, HepMC::Units::MM, and HepMC::Units::name().

8.160 testWeights.cc File Reference

```
#include <assert.h>
#include <iostream>
#include <string>
#include <vector>
#include "HepMC/WeightContainer.h"
#include <stdexcept>
```

Functions

- int **main** ()

8.160.1 Function Documentation

8.160.1.1 int main ( )

Definition at line 16 of file HepMC-2.06.09/test/testWeights.cc.

References HepMC::WeightContainer::empty(), HepMC::WeightContainer::has_key(), HepMC::WeightContainer::pop_back(), HepMC::WeightContainer::push_back(), HepMC::WeightContainer::size(), and HepMC::WeightContainer::write().

8.161 testWeights.cc File Reference

```
#include <assert.h>
#include <iostream>
#include <string>
#include <vector>
#include "HepMC/WeightContainer.h"
#include <stdexcept>
```

Functions

- int **main** ()

8.161.1 Function Documentation

8.161.1.1 int main ( )

Definition at line 16 of file test/testWeights.cc.

References HepMC::WeightContainer::empty(), HepMC::WeightContainer::has_key(), HepMC::WeightContainer::pop_back(), HepMC::WeightContainer::push_back(), HepMC::WeightContainer::size(), and HepMC::WeightContainer::write().

8.162 Units.cc File Reference

```
#include "HepMC/Units.h"
```

Namespaces

- **HepMC**
- **HepMC::Units**

Functions

- std::string **HepMC::Units::name** (MomentumUnit)

  convert enum to string
- std::string **HepMC::Units::name** (LengthUnit)

  convert enum to string
- double **HepMC::Units::conversion_factor** (MomentumUnit from, MomentumUnit to)

  scaling factor relative to MeV
- double **HepMC::Units::conversion_factor** (LengthUnit from, LengthUnit to)
- LengthUnit **HepMC::Units::default_length_unit** ()

  default unit is defined by configure
- MomentumUnit **HepMC::Units::default_momentum_unit** ()

  default unit is defined by configure

8.163 Units.cc File Reference

```
#include "HepMC/Units.h"
```

Namespaces

- **HepMC**
- **HepMC::Units**

Functions

- std::string **HepMC::Units::name** (MomentumUnit)

  convert enum to string
- std::string **HepMC::Units::name** (LengthUnit)

  convert enum to string
- double **HepMC::Units::conversion_factor** (MomentumUnit from, MomentumUnit to)

  scaling factor relative to MeV
- double **HepMC::Units::conversion_factor** (LengthUnit from, LengthUnit to)
- LengthUnit **HepMC::Units::default_length_unit** ()

  default unit is defined by configure
- MomentumUnit **HepMC::Units::default_momentum_unit** ()

  default unit is defined by configure

8.164 Units.h File Reference

```
#include <iostream>
#include <string>
```

Namespaces

- **HepMC**
- **Units**
- **HepMC::Units**

Enumerations

- enum **HepMC::Units::MomentumUnit** { **HepMC::Units::MEV**, **HepMC::Units::GEV**, **HepMC::Units::MEV**, **HepMC::Units::GEV** }
- enum **HepMC::Units::LengthUnit** { **HepMC::Units::MM**, **HepMC::Units::CM**, **HepMC::Units::MM**, **HepMC::Units::CM** }

Functions

- LengthUnit **HepMC::Units::default_length_unit** ()

  default unit is defined by configure
- MomentumUnit **HepMC::Units::default_momentum_unit** ()

  default unit is defined by configure
- std::string **HepMC::Units::name** (MomentumUnit)

convert enum to string

- std::string **HepMC::Units::name** (LengthUnit)

  convert enum to string
- double **HepMC::Units::conversion_factor** (MomentumUnit from, MomentumUnit to)

  scaling factor relative to MeV
- double **HepMC::Units::conversion_factor** (LengthUnit from, LengthUnit to)

8.165 Units.h File Reference

```
#include <iostream>
#include <string>
```

Namespaces

- **HepMC**
- **Units**
- **HepMC::Units**

Enumerations

- enum **HepMC::Units::MomentumUnit** { **HepMC::Units::MEV**, **HepMC::Units::GEV**, **HepMC::Units::MEV**, **HepMC::Units::GEV** }
- enum **HepMC::Units::LengthUnit** { **HepMC::Units::MM**, **HepMC::Units::CM**, **HepMC::Units::MM**, **HepMC::Units::CM** }

Functions

- LengthUnit **HepMC::Units::default_length_unit** ()

  default unit is defined by configure
- MomentumUnit **HepMC::Units::default_momentum_unit** ()

  default unit is defined by configure
- std::string **HepMC::Units::name** (MomentumUnit)

  convert enum to string
- std::string **HepMC::Units::name** (LengthUnit)

  convert enum to string
- double **HepMC::Units::conversion_factor** (MomentumUnit from, MomentumUnit to)

  scaling factor relative to MeV
- double **HepMC::Units::conversion_factor** (LengthUnit from, LengthUnit to)

8.166 VectorConversion.h File Reference

```
#include "HepMC/SimpleVector.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Namespaces

- **CLHEP**

Functions

- CLHEP::Hep3Vector **convertTo** (const **HepMC::ThreeVector** &**v**)

  *Convert from **HepMC::ThreeVector*** (p. *271*) *to CLHEP::Hep3Vector.*
- CLHEP::HepLorentzVector **convertTo** (const **HepMC::FourVector** &**v**)

  *Convert from **HepMC::FourVector*** (p. *54*) *to CLHEP::HepLorentzVector.*

8.166.1 Function Documentation

8.166.1.1 CLHEP::Hep3Vector convertTo (const HepMC::ThreeVector & *v*) [inline]

Convert from **HepMC::ThreeVector** (p. 271) to CLHEP::Hep3Vector.

Examples:

example_VectorConversion.cc, and **VectorConversion.h**.

Definition at line 21 of file examples/VectorConversion.h.

References HepMC::ThreeVector::x(), HepMC::ThreeVector::y(), and HepMC::ThreeVector::z().

Referenced by main().

8.166.1.2 CLHEP::HepLorentzVector convertTo (const HepMC::FourVector & *v*) [inline]

Convert from **HepMC::FourVector** (p. 54) to CLHEP::HepLorentzVector.

Definition at line 25 of file examples/VectorConversion.h.

References HepMC::FourVector::t(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

8.167 VectorConversion.h File Reference

```
#include "HepMC/SimpleVector.h"
#include "CLHEP/Vector/LorentzVector.h"
```

Namespaces

- **CLHEP**

Functions

- CLHEP::Hep3Vector **convertTo** (const **HepMC::ThreeVector** &**v**)

  *Convert from **HepMC::ThreeVector*** (p. *271*) *to CLHEP::Hep3Vector.*
- CLHEP::HepLorentzVector **convertTo** (const **HepMC::FourVector** &**v**)

  *Convert from **HepMC::FourVector*** (p. *54*) *to CLHEP::HepLorentzVector.*

8.167.1 Function Documentation

8.167.1.1 CLHEP::Hep3Vector convertTo (const HepMC::ThreeVector & *v*) [inline]

Convert from **HepMC::ThreeVector** (p. 271) to CLHEP::Hep3Vector.

Definition at line 21 of file HepMC-2.06.09/examples/VectorConversion.h.

References HepMC::ThreeVector::x(), HepMC::ThreeVector::y(), and HepMC::ThreeVector::z().

8.167.1.2 CLHEP::HepLorentzVector convertTo (const HepMC::FourVector & *v*) [inline]

Convert from **HepMC::FourVector** (p. 54) to CLHEP::HepLorentzVector.

Definition at line 25 of file HepMC-2.06.09/examples/VectorConversion.h.

References HepMC::FourVector::t(), HepMC::FourVector::x(), HepMC::FourVector::y(), and HepMC::FourVector::z().

8.168 Version.h File Reference

```
#include <string>
#include <iostream>
#include "HepMC/HepMCDefs.h"
```

Namespaces

- **HepMC**

Functions

- void **HepMC::version** (std::ostream &os=std::cout)

  *print **HepMC** (p. 15) version*
- void **HepMC::writeVersion** (std::ostream &os)

  *write **HepMC** (p. 15) version to os*
- std::string **HepMC::versionName** ()

  *return **HepMC** (p. 15) version*

8.169 Version.h File Reference

```
#include <string>
#include <iostream>
#include "HepMC/HepMCDefs.h"
```

Namespaces

- **HepMC**

Functions

- void **HepMC::version** (std::ostream &os=std::cout)

  *print **HepMC** (p. 15) version*
- void **HepMC::writeVersion** (std::ostream &os)

  *write **HepMC** (p. 15) version to os*
- std::string **HepMC::versionName** ()

  *return **HepMC** (p. 15) version*

8.170 WeightContainer.cc File Reference

```
#include <iostream>
#include <iomanip>
#include <sstream>
#include <vector>
#include <string>
#include <map>
#include <stdexcept>
#include "HepMC/WeightContainer.h"
```

Namespaces

- **HepMC**

8.171 WeightContainer.cc File Reference

```
#include <iostream>
#include <iomanip>
#include <sstream>
#include <vector>
#include <string>
#include <map>
#include <stdexcept>
#include "HepMC/WeightContainer.h"
```

Namespaces

- **HepMC**

8.172 WeightContainer.h File Reference

```
#include <iostream>
#include <vector>
#include <string>
#include <map>
```

Classes

- class **HepMC::WeightContainer**

  Container for the Weights associated with an event or vertex.

Namespaces

- **HepMC**

8.173 WeightContainer.h File Reference

```
#include <iostream>
#include <vector>
#include <string>
#include <map>
```

Classes

- class **HepMC::WeightContainer**

  Container for the Weights associated with an event or vertex.

Namespaces

- **HepMC**

Chapter 9

Example Documentation

9.1 example_BuildEventFromScratch.cc

Example of building an event and a particle data table from scratch This is meant to be of use for persons implementing **HepMC** (p. 15) inside a MC event generator

```
// Matt.Dobbs@Cern.CH, Feb 2000
// Example of building an event and a particle data table from scratch
// This is meant to be of use for persons implementing HepMC inside a MC
// event generator
// To Compile: go to the HepMC directory and type:
// gmake examples/example_BuildEventFromScratch.exe
//

#include <iostream>

#include "HepMC/GenEvent.h"

// in this example we use the HepMC namespace, so that we do not have to
// precede all HepMC classes with HepMC::

// This example also shows how to use the CLHEP Lorentz vector with HepMC2

using namespace HepMC;

int main() {
    //
    // In this example we will place the following event into HepMC "by hand"
    //
    //     name status pdg_id  parent Px       Py    Pz       Energy      Mass
    //  1  !p+!    3   2212    0,0    0.000    0.000 7000.000 7000.000    0.938
    //  2  !p+!    3   2212    0,0    0.000    0.000-7000.000 7000.000    0.938
    //=========================================================================
    //  3  !d!     3      1    1,1    0.750   -1.569   32.191   32.238    0.000
    //  4  !u~!    3     -2    2,2   -3.047  -19.000  -54.629   57.920    0.000
    //  5  !W-!    3    -24    1,2    1.517   -20.68  -20.605   85.925   80.799
    //  6  !gamma! 1     22    1,2   -3.813    0.113   -1.833    4.233    0.000
    //  7  !d!     1      1    5,5   -2.445   28.816    6.082   29.552    0.010
    //  8  !u~!    1     -2    5,5    3.962  -49.498  -26.687   56.373    0.006

    // now we build the graph, which will look like
    //                        p7                          #
    // p1                    /                            #
    //   \v1__p3      p5---v4                             #
    //         \_v3_/       \                             #
    //         /    \        p8                          #
    //    v2__p4     \                                    #
    //   /            p6                                  #
    // p2                                                 #
    //                                                    #

    // First create the event container, with Signal Process 20, event number 1
    //
    GenEvent* evt = new GenEvent( 20, 1 );
    // define the units
    evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
    //
    // create vertex 1 and vertex 2, together with their inparticles
    GenVertex* v1 = new GenVertex();
    evt->add_vertex( v1 );
    v1->add_particle_in( new GenParticle( FourVector(0,0,7000,7000),
```

```
                                        2212, 3 ) );
    GenVertex* v2 = new GenVertex();
    evt->add_vertex( v2 );
    v2->add_particle_in( new GenParticle( FourVector(0,0,-7000,7000),
                                        2212, 3 ) );
    //
    // create the outgoing particles of v1 and v2
    GenParticle* p3 =
        new GenParticle( FourVector(.750,-1.569,32.191,32.238), 1, 3 );
    v1->add_particle_out( p3 );
    GenParticle* p4 =
        new GenParticle( FourVector(-3.047,-19.,-54.629,57.920), -2, 3 );
    v2->add_particle_out( p4 );
    //
    // create v3
    GenVertex* v3 = new GenVertex();
    evt->add_vertex( v3 );
    v3->add_particle_in( p3 );
    v3->add_particle_in( p4 );
    v3->add_particle_out(
        new GenParticle( FourVector(-3.813,0.113,-1.833,4.233 ), 22, 1 )
        );
    GenParticle* p5 =
        new GenParticle( FourVector(1.517,-20.68,-20.605,85.925), -24,3);
    v3->add_particle_out( p5 );
    //
    // create v4
    GenVertex* v4 = new GenVertex(FourVector(0.12,-0.3,0.05,0.004));
    evt->add_vertex( v4 );
    v4->add_particle_in( p5 );
    v4->add_particle_out(
        new GenParticle( FourVector(-2.445,28.816,6.082,29.552), 1,1 )
        );
    v4->add_particle_out(
        new GenParticle( FourVector(3.962,-49.498,-26.687,56.373), -2,1 )
        );
    //
    // tell the event which vertex is the signal process vertex
    evt->set_signal_process_vertex( v3 );
    // the event is complete, we now print it out to the screen
    evt->print();

    // now clean-up by deleteing all objects from memory
    //
    // deleting the event deletes all contained vertices, and all particles
    // contained in those vertices
    delete evt;

    return 0;
}
```

9.2 example_EventSelection.cc

Example of applying an event selection to the events written to file using example_MyPythia.cxx Events containing a photon of pT $>$ 25 GeV pass the selection and are written to "example_EventSelection.dat"

```
// Matt.Dobbs@Cern.CH, Feb 2000
// Example of applying an event selection to the events written to file
// using example_MyPythia.cxx
// Events containing a photon of pT > 25 GeV pass the selection and are
// written to "example_EventSelection.dat"
// To Compile: go to the HepMC directory and type:
// gmake examples/example_EventSelection.exe
//

#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"


class IsEventGood {
public:
    bool operator()( const HepMC::GenEvent* evt ) {
        for ( HepMC::GenEvent::particle_const_iterator p
                  = evt->particles_begin(); p != evt->particles_end(); ++p ){
            if ( (*p)->pdg_id() == 22 && (*p)->momentum().perp() > 25. ) {
                //std::cout << "Event " << evt->event_number()
                //     << " is a good event." << std::endl;
                //(*p)->print();
                return 1;
            }
        }
```

```
        return 0;
    }
};

int main() {
    // declare an input strategy to read the data produced with the
    // example_MyPythia
    { // begin scope of ascii_in and ascii_out
        HepMC::IO_GenEvent ascii_in("example_MyPythia.dat",std::ios::in);
        // declare another IO_GenEvent for writing out the good events
        HepMC::IO_GenEvent ascii_out("example_EventSelection.dat",std::ios::out);
        // declare an instance of the event selection predicate
        IsEventGood is_good_event;
        //.....................................EVENT LOOP
        int icount=0;
        int num_good_events=0;
        HepMC::GenEvent* evt = ascii_in.read_next_event();
        while ( evt ) {
            icount++;
            if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
                                          << " its # " << evt->event_number()
                                          << std::endl;
            if ( is_good_event(evt) ) {
                ascii_out << evt;
                ++num_good_events;
            }
            delete evt;
            ascii_in >> evt;
        }
        //.....................................PRINT RESULT
        std::cout << num_good_events << " out of " << icount
                  << " processed events passed the cuts. Finished." << std::endl;
    } // end scope of ascii_in and ascii_out
    return 0;
}
```

9.3 example_MyPythiaOnlyToHepMC.cc

Example of generating events with Pythia using HepMC/PythiaWrapper.h Events are read into the **HepMC** (p. 15) event record from the FORTRAN HEPEVT common block using the IO_HEPEVT strategy – nothing is done with them. This program is just used to find the total time required to transfer from HEPEVT into the **HepMC** (p. 15) event record.

```
// Matt.Dobbs@Cern.CH, December 1999
// November 2000, updated to use Pythia 6.1
// example of generating events with Pythia
// using HepMC/PythiaWrapper.h
// Events are read into the HepMC event record from the FORTRAN HEPEVT
// common block using the IO_HEPEVT strategy -- nothing is done with them.
// This program is just used to find the total time required to transfer
// from HEPEVT into the HepMC event record.
// To Compile: go to the HepMC directory and type:
// gmake examples/example_MyPythiaOnlyTo HepMC.exe
//
// See comments in examples/example_MyPythia.cxx regarding the HEPEVT wrapper.
//

#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"

int main() {
    //
    //.....................................HEPEVT
    // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
```

```
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();
    //
    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    //
    //.....................................EVENT LOOP
    for ( int i = 1; i <= 100; i++ ) {
        if ( i%50==1 ) std::cout << "Processing Event Number "
                                 << i << std::endl;
        call_pyevnt();      // generate one event with Pythia
        // pythia pyhepc routine convert common PYJETS in common HEPEVT
        call_pyhepc( 1 );
        HepMC::GenEvent* evt = hepevtio.read_next_event();
        // define the units (Pythia uses GeV and mm)
        evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
        // set number of multi parton interactions
        evt->set_mpi( pypars.msti[31-1] );
        // set cross section information
        evt->set_cross_section( HepMC::getPythiaCrossSection() );
        //
        //...................USER WOULD PROCESS EVENT HERE
        //
        // we also need to delete the created event from memory
        delete evt;
    }
    //.....................................TERMINATION
    // write out some information from Pythia to the screen
    call_pystat( 1 );

    return 0;
}
```

9.4 example_UsingIterators.cc

This example shows low to use the particle and vertex iterators

```
// Matt.Dobbs@Cern.CH, Feb 2000
// This example shows low to use the particle and vertex iterators
// To Compile: go to the HepMC directory and type:
// gmake examples/example_UsingIterators.exe
//

#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include <math.h>
#include <algorithm>
#include <list>


class IsPhoton {
public:
    bool operator()( const HepMC::GenParticle* p ) {
        if ( p->pdg_id() == 22
             && p->momentum().perp() > 10. ) return 1;
        return 0;
    }
};


class IsW_Boson {
public:
    bool operator()( const HepMC::GenParticle* p ) {
        if ( abs(p->pdg_id()) == 24 ) return 1;
        return 0;
    }
};


class IsStateFinal {
public:
    bool operator()( const HepMC::GenParticle* p ) {
        if ( !p->end_vertex() && p->status()==1 ) return 1;
        return 0;
```

```
    }
};

int main() {
    { // begin scope of ascii_in
        // an event has been prepared in advance for this example, read it
        // into memory using the IO_GenEvent input strategy
        HepMC::IO_GenEvent ascii_in("example_UsingIterators.txt",std::ios::in);
        if ( ascii_in.rdstate() == std::ios::failbit ) {
            std::cerr << "ERROR input file example_UsingIterators.txt is needed "
                      << "and does not exist. "
                      << "\n Look for it in HepMC/examples, Exit." << std::endl;
            return 1;
        }

        HepMC::GenEvent* evt = ascii_in.read_next_event();

        // if you wish to have a look at the event, then use evt->print();

        // use GenEvent::vertex_iterator to fill a list of all
        // vertices in the event
        std::list<HepMC::GenVertex*> allvertices;
        for ( HepMC::GenEvent::vertex_iterator v = evt->vertices_begin();
              v != evt->vertices_end(); ++v ) {
            allvertices.push_back(*v);
        }

        // we could do the same thing with the STL algorithm copy
        std::list<HepMC::GenVertex*> allvertices2;
        copy( evt->vertices_begin(), evt->vertices_end(),
              back_inserter(allvertices2) );

        // fill a list of all final state particles in the event, by requiring
        // that each particle satisfyies the IsStateFinal predicate
        IsStateFinal isfinal;
        std::list<HepMC::GenParticle*> finalstateparticles;
        for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
              p != evt->particles_end(); ++p ) {
            if ( isfinal(*p) ) finalstateparticles.push_back(*p);
        }

        // an STL-like algorithm called HepMC::copy_if is provided in the
        // GenEvent.h header to do this sort of operation more easily,
        // you could get the identical results as above by using:
        std::list<HepMC::GenParticle*> finalstateparticles2;
        HepMC::copy_if( evt->particles_begin(), evt->particles_end(),
                        back_inserter(finalstateparticles2), IsStateFinal() );

        // lets print all photons in the event that satisfy the IsPhoton criteria
        IsPhoton isphoton;
        for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
              p != evt->particles_end(); ++p ) {
            if ( isphoton(*p) ) (*p)->print();
        }

        // the GenVertex::particle_iterator and GenVertex::vertex_iterator
        // are slightly different from the GenEvent:: versions, in that
        // the iterator starts at the given vertex, and walks through the attached
        // vertex returning particles/vertices.
        // Thus only particles/vertices which are in the same graph as the given
        // vertex will be returned. A range is specified with these iterators,
        // the choices are:
        //    parents, children, family, ancestors, descendants, relatives
        // here are some examples.

        // use GenEvent::particle_iterator to find all W's in the event,
        // then
        // (1) for each W user the GenVertex::particle_iterator with a range of
        //     parents to return and print the immediate mothers of these W's.
        // (2) for each W user the GenVertex::particle_iterator with a range of
        //     descendants to return and print all descendants of these W's.
        IsW_Boson isw;
        for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
              p != evt->particles_end(); ++p ) {
            if ( isw(*p) ) {
                std::cout << "A W boson has been found: " << std::endl;
                (*p)->print();
                // return all parents
                // we do this by pointing to the production vertex of the W
                // particle and asking for all particle parents of that vertex
                std::cout << "\t Its parents are: " << std::endl;
                if ( (*p)->production_vertex() ) {
                    for ( HepMC::GenVertex::particle_iterator mother
                              = (*p)->production_vertex()->
                              particles_begin(HepMC::parents);
                          mother != (*p)->production_vertex()->
                              particles_end(HepMC::parents);
```

```
                        ++mother ) {
                    std::cout << "\t";
                    (*mother)->print();
                }
            }
            // return all descendants
            // we do this by pointing to the end vertex of the W
            // particle and asking for all particle descendants of that vertex
            std::cout << "\t\t Its descendants are: " << std::endl;
            if ( (*p)->end_vertex() ) {
                for ( HepMC::GenVertex::particle_iterator des
                          =(*p)->end_vertex()->
                          particles_begin(HepMC::descendants);
                      des != (*p)->end_vertex()->
                          particles_end(HepMC::descendants);
                      ++des ) {
                    std::cout << "\t\t";
                    (*des)->print();
                }
            }
        }
    }
    // cleanup
    delete evt;
    // in analogy to the above, similar use can be made of the
    // HepMC::GenVertex::vertex_iterator, which also accepts a range.
} // end scope of ascii_in

return 0;
}
```

9.5 example_VectorConversion.cc

Example of how to convert from another vector class to a SimpleVector. This example uses CLHEP::HepLorentzVector

```
// Matt.Dobbs@Cern.CH, Feb 2000
// Example of building an event and a particle data table from scratch
// This is meant to be of use for persons implementing HepMC inside a MC
// event generator
// To Compile: go to the HepMC directory and type:
// gmake examples/example_BuildEventFromScratch.exe
//

#include <iostream>

#include "VectorConversion.h"
#include "HepMC/GenEvent.h"
#include "CLHEP/Vector/LorentzVector.h"

// in this example we use the HepMC namespace, so that we do not have to
// precede all HepMC classes with HepMC::

// This example also shows how to use the CLHEP Lorentz vector with HepMC2

using namespace HepMC;
using namespace CLHEP;

int main() {
    //
    // In this example we will place the following event into HepMC "by hand"
    //
    //     name status pdg_id  parent Px       Py    Pz       Energy      Mass
    //  1  !p+!    3   2212    0,0    0.000    0.000 7000.000 7000.000    0.938
    //  2  !p+!    3   2212    0,0    0.000    0.000-7000.000 7000.000    0.938
    //=========================================================================
    //  3  !d!     3      1    1,1    0.750   -1.569   32.191   32.238    0.000
    //  4  !u~!    3     -2    2,2   -3.047  -19.000  -54.629   57.920    0.000
    //  5  !W-!    3    -24    1,2    1.517   -20.68  -20.605   85.925   80.799
    //  6  !gamma! 1     22    1,2   -3.813    0.113   -1.833    4.233    0.000
    //  7  !d!     1      1    5,5   -2.445   28.816    6.082   29.552    0.010
    //  8  !u~!    1     -2    5,5    3.962  -49.498  -26.687   56.373    0.006

    // now we build the graph, which will look like
    //                       p7                          #
    // p1                   /                            #
    //   \v1__p3      p5---v4                            #
    //         \_v3_/       \                            #
    //         /    \        p8                          #
    //    v2__p4     \                                   #
    //   /            p6                                 #
```

```
    // p2                                                  #
    //                                                     #

    // First create the event container, with Signal Process 20, event number 1
    //
    // Note that the HepLorentzVectors will be automatically converted to
    // HepMC::FourVector within GenParticle and GenVertex
    GenEvent* evt = new GenEvent( 20, 1 );
    // define the units
    evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
    //
    // create vertex 1 and vertex 2, together with their inparticles
    GenVertex* v1 = new GenVertex();
    evt->add_vertex( v1 );
    v1->add_particle_in( new GenParticle( HepLorentzVector(0,0,7000,7000),
                                          2212, 3 ) );
    GenVertex* v2 = new GenVertex();
    evt->add_vertex( v2 );
    v2->add_particle_in( new GenParticle( HepLorentzVector(0,0,-7000,7000),
                                          2212, 3 ) );
    //
    // create the outgoing particles of v1 and v2
    GenParticle* p3 =
        new GenParticle( HepLorentzVector(.750,-1.569,32.191,32.238), 1, 3 );
    v1->add_particle_out( p3 );
    GenParticle* p4 =
        new GenParticle( HepLorentzVector(-3.047,-19.,-54.629,57.920), -2, 3 );
    v2->add_particle_out( p4 );
    //
    // create v3
    GenVertex* v3 = new GenVertex();
    evt->add_vertex( v3 );
    v3->add_particle_in( p3 );
    v3->add_particle_in( p4 );
    v3->add_particle_out(
        new GenParticle( HepLorentzVector(-3.813,0.113,-1.833,4.233 ), 22, 1 )
        );
    GenParticle* p5 =
        new GenParticle( HepLorentzVector(1.517,-20.68,-20.605,85.925), -24,3);
    v3->add_particle_out( p5 );
    //
    // create v4
    GenVertex* v4 = new GenVertex(HepLorentzVector(0.12,-0.3,0.05,0.004));
    evt->add_vertex( v4 );
    v4->add_particle_in( p5 );
    v4->add_particle_out(
        new GenParticle( HepLorentzVector(-2.445,28.816,6.082,29.552), 1,1 )
        );
    v4->add_particle_out(
        new GenParticle( HepLorentzVector(3.962,-49.498,-26.687,56.373), -2,1 )
        );
    //
    // tell the event which vertex is the signal process vertex
    evt->set_signal_process_vertex( v3 );
    // the event is complete, we now print it out to the screen
    evt->print();

    // example conversion back to Lorentz vector
    // add all outgoing momenta
    std::cout << std::endl;
    std::cout << " Add output momenta " << std::endl;
    HepLorentzVector sum;
    for ( GenEvent::particle_const_iterator p = evt->particles_begin();
              p != evt->particles_end(); ++p ){
        if( (*p)->status() == 1 ) {
            sum += convertTo( (*p)->momentum() );
            (*p)->print();
        }
    }
    std::cout << "Vector Sum: " << sum << std::endl;

    // now clean-up by deleteing all objects from memory
    //
    // deleting the event deletes all contained vertices, and all particles
    // contained in those vertices
    delete evt;

    return 0;
}
```

9.6 fio/example_MyHerwig.cc

```
// Matt.Dobbs@Cern.CH, October 2002
```

```
// example of generating events with Herwig using HepMC/HerwigWrapper.h
// Events are read into the HepMC event record from the FORTRAN HEPEVT
// common block using the IO_HERWIG strategy.

#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"

extern "C" void hwaend_() {}

int main() {
    //
    //.....................................HEPEVT
    // Herwig 6.4 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.................................INITIALIZATIONS

    hwproc.PBEAM1 = 7000.; // energy of beam1
    hwproc.PBEAM2 = 7000.; // energy of beam2
    // 1610 = gg->H--> WW, 1706 = qq-->ttbar, 2510 = ttH -> ttWW
    hwproc.IPROC = 1706; // qq -> ttbar production
    hwproc.MAXEV = 100; // number of events
    // tell it what the beam particles are:
    for ( unsigned int i = 0; i < 8; ++i ) {
        hwbmch.PART1[i] = (i < 1) ? 'P' : ' ';
        hwbmch.PART2[i] = (i < 1) ? 'P' : ' ';
    }
    hwigin();    // INITIALISE OTHER COMMON BLOCKS
    hwevnt.MAXPR = 1; // number of events to print
    hwuinc(); // compute parameter-dependent constants
    hweini(); // initialise elementary process

    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HERWIG hepevtio;
    // Instantiate an IO strategy to write the data to file
    HepMC::IO_GenEvent ascii_io("example_MyHerwig.dat",std::ios::out);
    //
    //.....................................EVENT LOOP
    for ( int i = 1; i <= hwproc.MAXEV; i++ ) {
        if ( i%50==1 ) std::cout << "Processing Event Number "
                                 << i << std::endl;
        // initialise event
        hwuine();
        // generate hard subprocess
        hwepro();
        // generate parton cascades
        hwbgen();
        // do heavy object decays
        hwdhob();
        // do cluster formation
        hwcfor();
        // do cluster decays
        hwcdec();
        // do unstable particle decays
        hwdhad();
        // do heavy flavour hadron decays
        hwdhvy();
        // add soft underlying event if needed
        hwmevt();
        // finish event
        hwufne();
        HepMC::GenEvent* evt = hepevtio.read_next_event();
        // define the units (Herwig uses GeV and mm)
        evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
        // set cross section information
        evt->set_cross_section( HepMC::getHerwigCrossSection(i) );
        // add some information to the event
        evt->set_event_number(i);
        evt->set_signal_process_id(20);
        if (i<=hwevnt.MAXPR) {
            std::cout << "\n\n This is the FIXED version of HEPEVT as "
                      << "coded in IO_HERWIG " << std::endl;
            HepMC::HEPEVT_Wrapper::print_hepevt();
            evt->print();
        }
        // write the event to the ascii file
        ascii_io << evt;
```

```
        // we also need to delete the created event from memory
        delete evt;
    }
    //.....................................TERMINATION
    hwefin();

    return 0;
}
```

9.7 fio/example_MyPythia.cc

example to generate events and write output example to generate events and perform simple event selection example to read the file written by pythia_out example to generate events, write them, and read them back

```
// Matt.Dobbs@Cern.CH, December 1999
// November 2000, updated to use Pythia 6.1
//


#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"


class IsGoodEventMyPythia {
public:
    bool operator()( const HepMC::GenEvent* evt ) {
        for ( HepMC::GenEvent::particle_const_iterator p
                  = evt->particles_begin(); p != evt->particles_end(); ++p ){
            if ( (*p)->pdg_id() == 22 && (*p)->momentum().perp() > 25. ) {
                //std::cout << "Event " << evt->event_number()
                //     << " is a good event." << std::endl;
                //(*p)->print();
                return 1;
            }
        }
        return 0;
    }
};


void pythia_out();
void pythia_in();
void pythia_in_out();
void event_selection();
void pythia_particle_out();

int main() {
    // example to generate events and write output
    pythia_out();
    // example to generate events and perform simple event selection
    event_selection();
    // example to read the file written by pythia_out
    pythia_in();
    // example to generate events, write them, and read them back
    pythia_in_out();

    return 0;
}


void pythia_out()
{
    std::cout << std::endl;
    std::cout << "Begin pythia_out()" << std::endl;
   //.....................................HEPEVT
    // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();
```

```
    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    //
    { // begin scope of ascii_io
        // Instantiate an IO strategy to write the data to file
        HepMC::IO_GenEvent ascii_io("example_MyPythia.dat",std::ios::out);
        //
        //.....................................EVENT LOOP
        for ( int i = 1; i <= 100; i++ ) {
            if ( i%50==1 ) std::cout << "Processing Event Number "
                                     << i << std::endl;
            call_pyevnt();      // generate one event with Pythia
            // pythia pyhepc routine converts common PYJETS in common HEPEVT
            call_pyhepc( 1 );
            HepMC::GenEvent* evt = hepevtio.read_next_event();
            // define the units (Pythia uses GeV and mm)
            evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
            // add some information to the event
            evt->set_event_number(i);
            evt->set_signal_process_id(20);
            // set number of multi parton interactions
            evt->set_mpi( pypars.msti[31-1] );
            // set cross section information
            evt->set_cross_section( HepMC::getPythiaCrossSection() );
            // write the event out to the ascii files
            ascii_io << evt;
            // we also need to delete the created event from memory
            delete evt;
        }
        //.....................................TERMINATION
        // write out some information from Pythia to the screen
        call_pystat( 1 );
    } // end scope of ascii_io
}


void event_selection()
{
    std::cout << std::endl;
    std::cout << "Begin event_selection()" << std::endl;
    //.....................................HEPEVT
    // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();
    //
    //.....................................HepMC INITIALIZATIONS
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    // declare an instance of the event selection predicate
    IsGoodEventMyPythia is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    for ( int i = 1; i <= 100; i++ ) {
        icount++;
        if ( i%50==1 ) std::cout << "Processing Event Number "
                                 << i << std::endl;
        call_pyevnt(); // generate one event with Pythia
        // pythia pyhepc routine convert common PYJETS in common HEPEVT
        call_pyhepc( 1 );
        HepMC::GenEvent* evt = hepevtio.read_next_event();
        // define the units (Pythia uses GeV and mm)
        evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
        // set number of multi parton interactions
        evt->set_mpi( pypars.msti[31-1] );
        // set cross section information
        evt->set_cross_section( HepMC::getPythiaCrossSection() );
        // do event selection
        if ( is_good_event(evt) ) {
            std::cout << "Good Event Number " << i << std::endl;
            ++num_good_events;
        }
        // we also need to delete the created event from memory
        delete evt;
    }
    //.....................................TERMINATION
    // write out some information from Pythia to the screen
    call_pystat( 1 );
    //.....................................PRINT RESULTS
```

```
    std::cout << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

void pythia_in()
{
    std::cout << std::endl;
    std::cout << "Begin pythia_in()" << std::endl;
    std::cout << "reading example_MyPythia.dat" << std::endl;
    //.....................................define an input scope
    {
        // open input stream
        std::ifstream istr( "example_MyPythia.dat" );
        if( !istr ) {
          std::cerr << "example_ReadMyPythia: cannot open example_MyPythia.dat" << std::endl;
          exit(-1);
        }
        HepMC::IO_GenEvent ascii_in(istr);
        // open output stream (alternate method)
        HepMC::IO_GenEvent ascii_out("example_MyPythia2.dat",std::ios::out);
        // now read the file
        int icount=0;
        HepMC::GenEvent* evt = ascii_in.read_next_event();
        while ( evt ) {
            icount++;
            if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
                                          << " its # " << evt->event_number()
                                          << std::endl;
            // write the event out to the ascii file
            ascii_out << evt;
            delete evt;
            ascii_in >> evt;
        }
        //.....................................PRINT RESULT
        std::cout << icount << " events found. Finished." << std::endl;
    } // ascii_out and istr destructors are called here
}

void pythia_in_out()
{
    std::cout << std::endl;
    std::cout << "Begin pythia_in_out()" << std::endl;
    //.....................................HEPEVT
    // Pythia 6.3 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();

    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    //
    //.....................................define the output scope
    {
        // Instantial an IO strategy to write the data to file
        HepMC::IO_GenEvent ascii_io("example_MyPythiaRead.dat",std::ios::out);
        //
        //.....................................EVENT LOOP
        for ( int i = 1; i <= 100; i++ ) {
            if ( i%50==1 ) std::cout << "Processing Event Number "
                                     << i << std::endl;
            call_pyevnt();      // generate one event with Pythia
            // pythia pyhepc routine converts common PYJETS in common HEPEVT
            call_pyhepc( 1 );
            HepMC::GenEvent* evt = hepevtio.read_next_event();
            // define the units (Pythia uses GeV and mm)
            evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
            // set cross section information
            evt->set_cross_section( HepMC::getPythiaCrossSection() );
            // add some information to the event
            evt->set_event_number(i);
            evt->set_signal_process_id(20);
            // write the event out to the ascii file
            ascii_io << evt;
            // we also need to delete the created event from memory
            delete evt;
        }
        //.....................................TERMINATION
        // write out some information from Pythia to the screen
        call_pystat( 1 );
    }  // ascii_io destructor is called here
```

```
    //
    //.....................................define an input scope
    {
        // now read the file we wrote
        HepMC::IO_GenEvent ascii_in("example_MyPythiaRead.dat",std::ios::in);
        HepMC::IO_GenEvent ascii_io2("example_MyPythiaRead2.dat",std::ios::out);
        int icount=0;
        HepMC::GenEvent* evt = ascii_in.read_next_event();
        while ( evt ) {
            icount++;
            if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
                                          << " its # " << evt->event_number()
                                          << std::endl;
            // write the event out to the ascii file
            ascii_io2 << evt;
            delete evt;
            ascii_in >> evt;
        }
        //.....................................PRINT RESULT
        std::cout << icount << " events found. Finished." << std::endl;
    } // ascii_io2 and ascii_in destructors are called here
}

void pythia_particle_out()
{
    std::cout << std::endl;
    std::cout << "Begin pythia_particle_out()" << std::endl;
    //.....................................HEPEVT
    // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();

    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    //
    { // begin scope of ascii_io
        // Instantiate an IO strategy to write the data to file
        HepMC::IO_AsciiParticles ascii_io("example_PythiaParticle.dat",std::ios::out);
        //
        //.....................................EVENT LOOP
        for ( int i = 1; i <= 100; i++ ) {
            if ( i%50==1 ) std::cout << "Processing Event Number "
                                     << i << std::endl;
            call_pyevnt();      // generate one event with Pythia
            // pythia pyhepc routine converts common PYJETS in common HEPEVT
            call_pyhepc( 1 );
            HepMC::GenEvent* evt = hepevtio.read_next_event();
            // define the units (Pythia uses GeV and mm)
            evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
            // set cross section information
            evt->set_cross_section( HepMC::getPythiaCrossSection() );
            // add some information to the event
            evt->set_event_number(i);
            evt->set_signal_process_id(20);
            // write the event out to the ascii file
            ascii_io << evt;
            // we also need to delete the created event from memory
            delete evt;
        }
        //.....................................TERMINATION
        // write out some information from Pythia to the screen
        call_pystat( 1 );
    } // end scope of ascii_io
}
```

9.8 fio/example_PythiaStreamIO.cc

This example generates Pythia events and fills cross section information from pyint5. The example uses streaming I/O to write a file and then read it.

```
// example_PythiaStreamIO.cc
//
```

```
// garren@fnal.gov, May 2009
//


#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "PythiaHelper.h"

void writePythiaStreamIO();
void readPythiaStreamIO();

int main() {

    writePythiaStreamIO();
    readPythiaStreamIO();

    return 0;
}


void writePythiaStreamIO() {
    // example to generate events and write output
    std::cout << std::endl;
    std::cout << "Begin pythia_out()" << std::endl;
   //.....................................HEPEVT
    // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();

    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    //
    { // begin scope of ascii_io
        // declare an output stream
        const char outfile[] = "example_PythiaStreamIO_write.dat";
        std::ofstream ascii_io( outfile );
        if( !ascii_io ) {
          std::cerr << "cannot open " << outfile << std::endl;
          exit(-1);
        }
        // use the default IO_GenEvent precision
        ascii_io.precision(16);
        // write the line that defines the beginning of a GenEvent block
        HepMC::write_HepMC_IO_block_begin( ascii_io );
        //
        //.....................................EVENT LOOP
        for ( int i = 1; i <= 100; i++ ) {
            if ( i%50==1 ) std::cout << "Processing Event Number "
                                     << i << std::endl;
            call_pyevnt();      // generate one event with Pythia
            // pythia pyhepc routine converts common PYJETS in common HEPEVT
            call_pyhepc( 1 );
            HepMC::GenEvent* evt = hepevtio.read_next_event();
            // define the units (Pythia uses GeV and mm)
            evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
            // add some information to the event
            evt->set_event_number(i);
            evt->set_signal_process_id(20);
            // set number of multi parton interactions
            evt->set_mpi( pypars.msti[31-1] );
            // set cross section information
            evt->set_cross_section( HepMC::getPythiaCrossSection() );
            // write the event out to the ascii files
            ascii_io << (*evt);;
            // we also need to delete the created event from memory
            delete evt;
        }
        // write the line that defines the end of a GenEvent block
        HepMC::write_HepMC_IO_block_end( ascii_io );
        //.....................................TERMINATION
        // write out some information from Pythia to the screen
        call_pystat( 1 );
    } // end scope of ascii_io
}

void readPythiaStreamIO() {
```

```
    // example to read events written by writePythiaStreamIO
    // and write them back out
    std::cout << std::endl;
    // input units are GeV and mm
    const char infile[] = "example_PythiaStreamIO_write.dat";
    std::ifstream is( infile );
    if( !is ) {
      std::cerr << "cannot open " << infile << std::endl;
      exit(-1);
    }
    //
    { // begin scope of ascii_io
        // declare an output stream
        const char outfile[] = "example_PythiaStreamIO_read.dat";
        std::ofstream ascii_io( outfile );
        if( !ascii_io ) {
          std::cerr << "cannot open " << outfile << std::endl;
          exit(-1);
        }
        ascii_io.precision(16);
        HepMC::write_HepMC_IO_block_begin( ascii_io );
        //
        //.....................................EVENT LOOP
        HepMC::GenEvent evt;
        int i = 0;
        while ( is ) {
            evt.read( is );
            // make sure we have a valid event
            if( evt.is_valid() ) {
                ++i;
                if ( i%50==1 ) std::cout << "Processing Event Number "
                                         << i << std::endl;
                if ( i%25==2 ) {
                    // write the cross section if it exists
                    if( evt.cross_section() ) {
                        std::cout << "cross section at event " << i << " is "
                                  << evt.cross_section()->cross_section()
                                  << std::endl;
                    }
                }
                // write the event out to the ascii files
                evt.write( ascii_io );
            }
        }
        //.....................................TERMINATION
        HepMC::write_HepMC_IO_block_end( ascii_io );
    } // end scope of ascii_io
}
```

9.9 fio/testHerwigCopies.cc

Multiple events in memory at the same time

```
// testHerwigCopies.cc
//
// garren@fnal.gov, January 2008
// Multiple events in memory at the same time

#include <fstream>
#include <iostream>
#include "HepMC/HerwigWrapper.h"
#include "HepMC/IO_HERWIG.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "HepMC/HEPEVT_Wrapper.h"

extern "C" void hwaend_() {}

int main() {
    //
    //.....................................HEPEVT
    // Herwig 6.4 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................INITIALIZATIONS

    hwproc.PBEAM1 = 7000.; // energy of beam1
    hwproc.PBEAM2 = 7000.; // energy of beam2
```

```
    // 1610 = gg->H--> WW, 1706 = qq-->ttbar, 2510 = ttH -> ttWW
    hwproc.IPROC = 1706; // qq -> ttbar production
    hwproc.MAXEV = 50; // number of events
    // tell it what the beam particles are:
    for ( unsigned int i = 0; i < 8; ++i ) {
        hwbmch.PART1[i] = (i < 1) ? 'P' : ' ';
        hwbmch.PART2[i] = (i < 1) ? 'P' : ' ';
    }
    hwigin();    // INITIALISE OTHER COMMON BLOCKS
    hwevnt.MAXPR = 0; // number of events to print
    hwuinc(); // compute parameter-dependent constants
    hweini(); // initialise elementary process

    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HERWIG hepevtio;
    //
    // open some output files
    std::ofstream out1( "testHerwigOriginals.dat" );
    std::ofstream out2( "testHerwigCopies1.dat" );
    std::ofstream out3( "testHerwigCopies2.dat" );
    //
    //.....................................EVENT LOOP
    for ( int i = 1; i <= hwproc.MAXEV; i++ ) {
        if ( i%50==1 ) std::cout << "Processing Event Number "
                                 << i << std::endl;
        // initialise event
        hwuine();
        // generate hard subprocess
        hwepro();
        // generate parton cascades
        hwbgen();
        // do heavy object decays
        hwdhob();
        // do cluster formation
        hwcfor();
        // do cluster decays
        hwcdec();
        // do unstable particle decays
        hwdhad();
        // do heavy flavour hadron decays
        hwdhvy();
        // add soft underlying event if needed
        hwmevt();
        // finish event
        hwufne();
        HepMC::GenEvent* evt = hepevtio.read_next_event();
        // herwig uses GeV and mm
        evt->use_units( HepMC::Units::GEV, HepMC::Units::MM);
        // set cross section information
        evt->set_cross_section( HepMC::getHerwigCrossSection(i) );
        // add some information to the event
        evt->set_event_number(i);
        evt->set_signal_process_id(20);
        //
        //.....................make some copies
        evt->print(out1);
        HepMC::GenEvent ec = (*evt);
        ec.print(out2);
        HepMC::GenEvent* evt4 = new HepMC::GenEvent(*evt);
        evt4->print(out3);
        if( !compareGenEvent(evt,evt4) ) {
           std::cerr << "testHerwigCopies: GenEvent comparison fails at event "
                     << evt->event_number() << std::endl;
           return -1;
        }

        // we also need to delete the created event from memory
        delete evt;
        delete evt4;
    }
    //.....................................TERMINATION
    hwefin();
    std::cout << "testHerwigCopies: event comparison is successful" << std::endl;

    return 0;
}
```

9.10 fio/testPythiaCopies.cc

Multiple events in memory at the same time

```
// testPythiaCopies.cc
//
// garren@fnal.gov, January 2008
// Multiple events in memory at the same time

#include <fstream>
#include <iostream>
#include "HepMC/PythiaWrapper.h"
#include "HepMC/IO_HEPEVT.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"
#include "PythiaHelper.h"

int main() {
    //
    //.....................................HEPEVT
    // Pythia 6.1 uses HEPEVT with 4000 entries and 8-byte floating point
    //  numbers. We need to explicitly pass this information to the
    //  HEPEVT_Wrapper.
    //
    HepMC::HEPEVT_Wrapper::set_max_number_entries(4000);
    HepMC::HEPEVT_Wrapper::set_sizeof_real(8);
    //
    //.....................................PYTHIA INITIALIZATIONS
    initPythia();
    //
    //.....................................HepMC INITIALIZATIONS
    //
    // Instantiate an IO strategy for reading from HEPEVT.
    HepMC::IO_HEPEVT hepevtio;
    //
    // open some output files
    std::ofstream out1( "testPythiaOriginals.dat" );
    std::ofstream out2( "testPythiaCopies1.dat" );
    std::ofstream out3( "testPythiaCopies2.dat" );
    //
    //.....................................EVENT LOOP
    for ( int i = 1; i <= 50; i++ ) {
        if ( i%50==1 ) std::cout << "Processing Event Number "
                                 << i << std::endl;
        call_pyevnt();      // generate one event with Pythia
        // pythia pyhepc routine convert common PYJETS in common HEPEVT
        call_pyhepc( 1 );
        HepMC::GenEvent* evt = hepevtio.read_next_event();
        // pythia uses GeV and mm
        evt->use_units( HepMC::Units::GEV, HepMC::Units::MM);
        // set a couple of arbitrary weights
        evt->weights().push_back(0.456);
        evt->weights()["test2"] = 0.8956;
        // set number of multi parton interactions
        evt->set_mpi( pypars.msti[31-1] );
        // set cross section information
        evt->set_cross_section( HepMC::getPythiaCrossSection() );
        //
        //....................make some copies
        evt->print(out1);
        HepMC::GenEvent ec = (*evt);
        ec.print(out2);
        HepMC::GenEvent* evt4 = new HepMC::GenEvent(*evt);
        evt4->print(out3);
        if( !compareGenEvent(evt,evt4) ) {
           std::cerr << "testPythiaCopies: GenEvent comparison fails at event "
                     << evt->event_number() << std::endl;
           return -1;
        }
        //
        // now delete the created events from memory
        delete evt;
        delete evt4;
    }
    //.....................................TERMINATION
    // write out some information from Pythia to the screen
    call_pystat( 1 );
    std::cout << "testPythiaCopies: event comparison is successful" << std::endl;

    return 0;
}
```

9.11 testFlow.cc

Use a modified example_BuildEventFromScratch to test Flow

```
// testFlow.cc
//
// garren@fnal.gov, June 2009
// based on example_BuildEventFromScratch.cc

#include <iostream>
#include <fstream>
#include <vector>

#include "HepMC/GenEvent.h"
#include "HepMC/IO_GenEvent.h"

typedef std::vector<HepMC::GenParticle*> FlowVec;

int main() {
    //
    // In this example we will place the following event into HepMC "by hand"
    //
    //     name status pdg_id  parent Px       Py    Pz       Energy      Mass
    //  1  !p+!    3   2212    0,0    0.000    0.000 7000.000 7000.000    0.938
    //  2  !p+!    3   2212    0,0    0.000    0.000-7000.000 7000.000    0.938
    //=========================================================================
    //  3  !d!     3      1    1,1    0.750   -1.569   32.191   32.238    0.000
    //  4  !u~!    3     -2    2,2   -3.047  -19.000  -54.629   57.920    0.000
    //  5  !W-!    3    -24    1,2    1.517   -20.68  -20.605   85.925   80.799
    //  6  !gamma! 1     22    1,2   -3.813    0.113   -1.833    4.233    0.000
    //  7  !d!     1      1    5,5   -2.445   28.816    6.082   29.552    0.010
    //  8  !u~!    1     -2    5,5    3.962  -49.498  -26.687   56.373    0.006

    // open an output file
    const char outfile[] = "testFlow.out";
    std::ofstream os( outfile );
    if( !os ) {
      std::cerr << "cannot open " << outfile << std::endl;
      exit(-1);
    }
    // declare several IO_GenEvent instances for comparison
    HepMC::IO_GenEvent xout1("testFlow.out1",std::ios::out);
    HepMC::IO_GenEvent xout2("testFlow.out2",std::ios::out);
    HepMC::IO_GenEvent xout3("testFlow.out3",std::ios::out);
    // output streams for copy test
    std::ofstream xout4( "testFlow.out4" );
    std::ofstream xout5( "testFlow.out5" );

    int numbad = 0;


    // build the graph, which will look like
    //                       p7                  #
    // p1                   /                    #
    //   \v1__p3      p5---v4                    #
    //         \_v3_/       \                    #
    //         /    \        p8                  #
    //    v2__p4     \                          #
    //   /            p6                         #
    // p2                                        #
    //
    // define a flow pattern as  p1 -> p3 -> p6
    //                       and p2 -> p4 -> p5
    //

    // First create the event container, with Signal Process 20, event number 1
    //
    HepMC::GenEvent* evt = new HepMC::GenEvent( 20, 1 );
    evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
    //
    // create vertex 1 and vertex 2, together with their inparticles
    HepMC::GenVertex* v1 = new HepMC::GenVertex();
    evt->add_vertex( v1 );
    HepMC::GenParticle* p1 = new HepMC::GenParticle( HepMC::FourVector(0,0,7000,7000),
                                     2212, 3 );
    p1->set_flow(1,231);
    v1->add_particle_in( p1 );
    HepMC::GenVertex* v2 = new HepMC::GenVertex();
    evt->add_vertex( v2 );
    HepMC::GenParticle* p2 = new HepMC::GenParticle( HepMC::FourVector(0,0,-7000,7000),
                                     2212, 3 );
    p2->set_flow(1,243);
    v2->add_particle_in( p2 );
    //
    // create the outgoing particles of v1 and v2
```

```
HepMC::GenParticle* p3 =
    new HepMC::GenParticle( HepMC::FourVector(.750,-1.569,32.191,32.238),
                            1, 3 );
p3->set_flow(1,231);
v1->add_particle_out( p3 );
HepMC::GenParticle* p4 =
    new HepMC::GenParticle( HepMC::FourVector(-3.047,-19.,-54.629,57.920),
                            -2, 3 );
p4->set_flow(1,243);
v2->add_particle_out( p4 );
//
// create v3
HepMC::GenVertex* v3 = new HepMC::GenVertex();
evt->add_vertex( v3 );
v3->add_particle_in( p3 );
v3->add_particle_in( p4 );
HepMC::GenParticle* p6 =
      new HepMC::GenParticle( HepMC::FourVector(-3.813,0.113,-1.833,4.233 ),
                              22, 1 );
p6->set_flow(1,231);
v3->add_particle_out( p6 );
HepMC::GenParticle* p5 =
    new HepMC::GenParticle( HepMC::FourVector(1.517,-20.68,-20.605,85.925),
                            -24, 3 );
p5->set_flow(1,243);
v3->add_particle_out( p5 );
//
// create v4
HepMC::GenVertex* v4 = new HepMC::GenVertex(HepMC::FourVector(0.12,-0.3,0.05,0.004));
evt->add_vertex( v4 );
v4->add_particle_in( p5 );
HepMC::GenParticle* p7 = new HepMC::GenParticle( HepMC::FourVector(-2.445,28.816,6.082,29.552), 1,1 );
v4->add_particle_out( p7 );
HepMC::GenParticle* p8 = new HepMC::GenParticle( HepMC::FourVector(3.962,-49.498,-26.687,56.373), -2,1
  );
v4->add_particle_out( p8 );
//
// tell the event which vertex is the signal process vertex
evt->set_signal_process_vertex( v3 );
// the event is complete, we now print it out
evt->print( os );

// look at the flow we created
os << std::endl;
FlowVec result1 = p1->flow().dangling_connected_partners( p1->flow().icode(1) );
FlowVec result2 = p1->flow().connected_partners( p1->flow().icode(1) );
FlowVec::iterator it;
os << "dangling partners of particle " << p1->barcode() << std::endl;
for( it = result1.begin(); it != result1.end(); ++it ) {
  os << (*it)->barcode() << " " ;
  os.width(8);
  os << (*it)->pdg_id() << " " << (*it)->flow(1)  << std::endl;
}
os << "all partners of particle " << p1->barcode() << std::endl;
for( it = result2.begin(); it != result2.end(); ++it ) {
  os << (*it)->barcode() << " " ;
  os.width(8);
  os << (*it)->pdg_id() << " " << (*it)->flow(1)  << std::endl;
}
FlowVec result3 = p2->flow().dangling_connected_partners( p2->flow().icode(1) );
FlowVec result4 = p2->flow().connected_partners( p2->flow().icode(1) );
os << "dangling partners of particle " << p2->barcode() << std::endl;
for( it = result3.begin(); it != result3.end(); ++it ) {
  os << (*it)->barcode() << " " ;
  os.width(8);
  os << (*it)->pdg_id() << " " << (*it)->flow(1)  << std::endl;
}
os << "all partners of particle " << p2->barcode() << std::endl;
for( it = result4.begin(); it != result4.end(); ++it ) {
  os << (*it)->barcode() << " " ;
  os.width(8);
  os << (*it)->pdg_id() << " " << (*it)->flow(1)  << std::endl;
}
// write event
xout1 << evt;
// testing bug #73987 - flow not copied
// call the write method directly
evt->write(xout4);
// make a copy and write it
HepMC::GenEvent(*evt).write(xout5);

// try changing and erasing flow
p2->set_flow(2,345);
        xout2 << evt;
FlowVec result5 = p2->flow().connected_partners( p2->flow().icode(1) );
if ( result4 != result5 ) {
    std::cerr << "ERROR: list of partners has changed after adding flow" << std::endl;
```

```
        ++numbad;
    }
    // the flow method returns a copy,
    // so we must set the flow again to change it
    HepMC::Flow f2 = p2->flow();
    if( f2.erase(2) ) {
        p2->set_flow( f2 );
    } else {
        std::cerr << "ERROR: first erase was NOT successful" << std::endl;
        ++numbad;
    }
    f2 = p2->flow();
    if( f2.erase(2) ) {
        std::cerr << "ERROR: second erase was successful" << std::endl;
    }
            xout3 << evt;
    FlowVec result6 = p2->flow().connected_partners( p2->flow().icode(1) );
    if ( result4 != result6 ) {
        std::cerr << "ERROR: list of partners has changed after removing flow" << std::endl;
        ++numbad;
    }

    // now clean-up by deleteing all objects from memory
    //
    // deleting the event deletes all contained vertices, and all particles
    // contained in those vertices
    delete evt;

    if( numbad > 0 ) std::cerr << numbad << " errors in testFlow" << std::endl;

    return numbad;
}
```

9.12 testHepMC.cc.in

The **HepMC** (p. 15) tests can also serve as useful examples based on example_EventSelection Apply an event selection to the events in testHepMC.input Events containing a photon of pT $>$ 25 GeV pass the selection and are written to "testHepMC.out" Add arbitrary PDF information to the good events Also write events using IO_AsciiParticles Test the new GenCrossSection class

```
//----------------------------------------------------------------------
// testHepMC.cc.in
//
// garren@fnal.gov, March 2006
// based on example_EventSelection
// Apply an event selection to the events in testHepMC.input
// Events containing a photon of pT > 25 GeV pass the selection
// and are written to "testHepMC.out"
// Also write events using IO_AsciiParticles
//----------------------------------------------------------------------
//

#include "HepMC/GenEvent.h"
#include "HepMC/GenCrossSection.h"
#ifndef HEPMC_IO_ASCII_REMOVED
#include "HepMC/IO_Ascii.h"
#endif
#ifdef HEPMC_HAS_IO_GENEVENT
#include "HepMC/IO_GenEvent.h"
#endif
#include "HepMC/IO_AsciiParticles.h"

// define methods and classes used by this test
#include "IsGoodEvent.h"
#include "testHepMCMethods.h"

void read_testIOGenEvent(std::ostream & os);
void read_testUnits(std::ostream & os);
void read_variousFormats(std::ostream & os);
void writeWithCrossSection(std::ostream & os);
void readWithCrossSection(std::ostream & os);
void writeWithWeight(std::ostream & os);
void readWithWeight(std::ostream & os);
void read_nan(std::ostream & os);

int main() {
    std::ofstream os( "testHepMC.cout" );
    std::ofstream osv( "testHepMCVarious.cout" );
    read_testIOGenEvent(os);
    read_testUnits(os);
```

```
    read_variousFormats(osv);
    read_nan(os);
    writeWithCrossSection(os);
    readWithCrossSection(os);
    writeWithWeight(os);
    readWithWeight(os);
    return 0;
}

void read_testIOGenEvent(std::ostream & os)
{
    os << std::endl;
    os << "basic IO_GenEvent input and output" << std::endl;
    // declare an input strategy to read the data produced with the
    // example_MyPythia - units are GeV and mm
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    // declare another IO_GenEvent for writing out the good events
    HepMC::IO_GenEvent ascii_out("testHepMC.out",std::ios::out);
    // declare an output IO_GenEvent for testing precision
    HepMC::IO_GenEvent prec_out("testHepMCprecision.out",std::ios::out);
    prec_out.precision(10);
    // declare an IO_AsciiParticle for output
    HepMC::IO_AsciiParticles particle_out("testHepMCParticle.out",std::ios::out);
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        ++icount;
        if ( icount%50==1 ) os << "Processing Event Number " << icount
                                   << " its # " << evt->event_number()
                                   << std::endl;
        if ( is_good_event(evt) ) {
            particleTypes(evt,os);
            // verify use_input_units()
            evt->write_units(os);
            double pim = findPiZero(evt);
            os << " pizero mass: " << pim << std::endl;
            //
            ascii_out << evt;
            particle_out << evt;
            prec_out << evt;
            ++num_good_events;
        }

        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    //.....................................PRINT RESULT
    os << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

void read_testUnits(std::ostream & os)
{
    os << std::endl;
    os << "IO_GenEvent input and output using define_units" << std::endl;
    // declare an input strategy to read the data produced with the
    // example_MyPythia - units are GeV and mm
    // we DO NOT define input units here, instead we use define_units
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    // declare another IO_GenEvent for writing out the good events
    HepMC::IO_GenEvent ascii_out("testDefineUnits.out",std::ios::out);
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        ++icount;
        evt->define_units( HepMC::Units::GEV, HepMC::Units::MM );
        if ( icount%50==1 ) os << "Processing Event Number " << icount
                                   << " its # " << evt->event_number()
                                   << std::endl;
        if ( is_good_event(evt) ) {
            // verify define_units()
            evt->write_units(os);
            double pim = findPiZero(evt);
            os << " pizero mass: " << pim << std::endl;
            //
            particleTypes(evt,os);
            ascii_out << evt;
```

```
            ++num_good_events;
        }

        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    //.....................................PRINT RESULT
    os << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

void read_variousFormats(std::ostream & os)
{
    os << std::endl;
    os << "process varied input" << std::endl;
    // declare an input strategy
    HepMC::IO_GenEvent ascii_in("@srcdir@/testHepMCVarious.input",std::ios::in);
    ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    // declare another IO_GenEvent for writing out the good events
    HepMC::IO_GenEvent ascii_out("testHepMCVarious.out",std::ios::out);
    //.....................................EVENT LOOP
    int icount=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        icount++;
        double pim;
        os << "Processing Event Number " << icount
                  << " its # " << evt->event_number()
                  << std::endl;
        ascii_out << evt;
        // units should be unknown
        evt->write_units(os);
        pim = findPiZero(evt);
        os << " pizero mass: " << pim << std::endl;
        if( HepMC::Units::name( evt->momentum_unit() ) == "GEV" ) {
            os << " GenEvent units are GeV" << std::endl;
            if( pim > 1.0 ) {
                // presume units are MEV and out of sync
                os << " pizero units are MeV" << std::endl;
                repairUnits(evt,HepMC::Units::MEV,HepMC::Units::GEV);
                // set units to MeV and mm
                evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
                // convert units to MeV
                evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
            } else if( pim > 0.1 ) {
                // presume units are GEV
                os << " pizero units are GeV" << std::endl;
                // set units to GeV and mm
                evt->use_units(HepMC::Units::GEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
                // convert units to MeV
                evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
            } else {
                os << " pizero mass: " << pim
                          << " is inconsistent with allowed units " << std::endl;
            }
        } else if( HepMC::Units::name( evt->momentum_unit() ) == "MEV" ) {
            os << " GenEvent units are MeV" << std::endl;
            if( pim > 1.0 ) {
                // presume units are MEV
                os << " pizero units are MeV" << std::endl;
                // set units to MeV and mm
                evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
                // convert units to MeV
                evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
```

```
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
            } else if( pim > 0.1 ) {
                // presume units are GEV and out of sync
                os << " pizero units are GeV" << std::endl;
                repairUnits(evt,HepMC::Units::GEV,HepMC::Units::MEV);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
                // convert units to MeV
                evt->use_units(HepMC::Units::MEV, HepMC::Units::MM);
                evt->write_units(os);
                pim = findPiZero(evt);
                os << " pizero mass: " << pim
                          << " " << HepMC::Units::name( evt->momentum_unit() ) << std::endl;
            } else {
                os << " pizero mass: " << pim
                          << " is inconsistent with allowed units " << std::endl;
            }
        }
        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    std::cout << "testHepMC: the HeavyIon and PdfInfo input stream errors are intentional" << std::endl;
    //.....................................PRINT RESULT
    os << icount << " events processed. Finished." << std::endl;
}

void writeWithCrossSection(std::ostream & os)
{
    // declare an input strategy to read input data
    // units are GeV and mm
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    // declare another IO_GenEvent for writing out some events
    HepMC::IO_GenEvent ascii_out("testCrossSection.out",std::ios::out);
    // declare an output stream for printing events
    std::ofstream xout( "testCrossSection.cout" );
    // create an empty GenCrossSection object
    HepMC::GenCrossSection cross;
    //.....................................EVENT LOOP
    int icount=0;
    const double xs0 = 0.00346;
    const double xs1 = 0.12;
    const double xs2 = 33.234;
    const double xs3 = 459.345;
    double xserr = 0.0001;
    double wgt1, wgt2;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        icount++;
        // use a variety of arbitrary cross section values
        if( icount < 10 ) {
            const double xs = xs0 - 1.34 * xserr;
            cross.set_cross_section( xs, xserr );
        } else if( icount < 20 ) {
            const double xs = xs1 - 1.34 * xserr;
            cross.set_cross_section( xs, xserr );
        } else if( icount < 30 ) {
            const double xs = xs2 - 1.34 * xserr;
            cross.set_cross_section( xs, xserr );
        } else {
            const double xs = xs3 - 1.34 * xserr;
            cross.set_cross_section( xs, xserr );
        }
        xserr *= 0.99;
        if ( icount == 10 ) xserr += 0.01;
        if ( icount == 20 ) xserr += 0.4;
        if ( icount == 30 ) xserr += 1.0;
        // attach this cross section to the event
        evt->set_cross_section( cross );
        evt->write_cross_section(os);
        // add weights
        wgt1 = 0.9853 + (double)icount * 0.00033;
        wgt2 = 0.9853 + (double)(icount+1) * 0.00033;
        evt->weights().push_back(0.3456);
        evt->weights()["weightName"] = wgt1;
        evt->weights()["second weight name"] = wgt2;
        if ( icount%20==1 ) {
            os << "writeWithCrossSection: Processing Event Number " << icount
                                      << " its # " << evt->event_number()
                                      << std::endl;
            ascii_out << evt;
            evt->print(xout);
        }
```

```
        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    //.....................................PRINT RESULT
    os << "writeWithCrossSection processed " << icount << " events. Finished." << std::endl;
}

void readWithCrossSection(std::ostream & os)
{
    // read the file we just wrote
    HepMC::IO_GenEvent ascii_in("testCrossSection.out",std::ios::in);
    // declare another IO_GenEvent for writing out some events
    HepMC::IO_GenEvent ascii_out("testCrossSection2.out",std::ios::out);
    //.....................................EVENT LOOP
    int icount=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        ++icount;
        os << "readWithCrossSection: Processing Event Number " << icount
                                << " its # " << evt->event_number()
                                << std::endl;
        if (evt->cross_section()->cross_section() <= 0) {
          os << "testReadCrossSection: invalid cross-section!" << std::endl;
        }
        ascii_out << evt;

        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    //.....................................PRINT RESULT
    os << "readWithCrossSection processed " << icount << " events. Finished." << std::endl;
}

void read_nan(std::ostream & os)
{
    // Read an input file that has corrupt information (nan's)
    //
    HepMC::IO_GenEvent xin("@srcdir@/testHepMCVarious.input",std::ios::in);
    HepMC::IO_GenEvent xout("testNaN.out",std::ios::out);
    // set input units
    xin.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    //.....................................EVENT LOOP
    int icount=0;
    int invaliddata=0;
    bool ok = true;
    os << "---------------------------------------- " << std::endl;
    os << "Begin NaN test " << std::endl;
    HepMC::GenEvent* evt = xin.read_next_event();
    //
    // To recover from corrupt input, replace "while(evt) {...}"
    // with  "while(ok) { if(evt) {... xin >> evt;} else {...} }"
    //
    while ( ok ) {
        if( evt ) {
            ++icount;
            os << "read_nan: Processing Event Number " << icount
                      << " its # " << evt->event_number()
                      << std::endl;
            xout << evt;
            // clean up and get next event
            delete evt;
            xin >> evt;
        } else if (xin.error_type() == HepMC::IO_Exception::InvalidData ) {
            ++invaliddata;
            os << "INPUT ERROR: " << xin.error_message() << std::endl;
            // clean up and get next event
            delete evt;
            xin >> evt;
        } else if (invaliddata > 50 ) {
            os << "INPUT ERROR: " << xin.error_message() << std::endl;
            ok = false;
        } else {
            ok = false;
        }
    }
    // print status of input stream
    if ( xin.error_type() != 0 ) {
        os << "processing of @srcdir@/testHepMCVarious.input ended with error "
                << xin.error_type() << std::endl;
        os << "  --- " << xin.error_message() << std::endl;
    }
    os << icount << " events processed and "
              << invaliddata << " events ignored. Finished."
              << std::endl;
```

```
    os << "End NaN test " << std::endl;
    os << "-------------------------------------- " << std::endl;
}

void writeWithWeight(std::ostream & os)
{
    // declare an input strategy to read input data
    // units are GeV and mm
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    // declare another IO_GenEvent for writing out some events
    HepMC::IO_GenEvent ascii_out("testWithWeight.out",std::ios::out);
    // declare an output stream for printing events
    std::ofstream xout( "testWithWeight.cout" );
    //.....................................EVENT LOOP
    int icount=0;
    double wgt1, wgt2;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        icount++;
        // add weights
        wgt1 = 0.9853 + (double)icount * 0.00033;
        wgt2 = 0.9853 + (double)(icount+1) * 0.00033;
        evt->weights().push_back(0.3456);
        evt->weights().push_back(wgt1);
        evt->weights().push_back(wgt2);
        if ( icount%20==1 ) {
            os << "writeWithWeight: Processing Event Number " << icount
                                    << " its # " << evt->event_number()
                                    << std::endl;
            ascii_out << evt;
            evt->print(xout);
        }

        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    //.....................................PRINT RESULT
    os << "writeWithWeight processed " << icount << " events. Finished." << std::endl;
}

void readWithWeight(std::ostream & os)
{
    // read the file we just wrote
    HepMC::IO_GenEvent ascii_in("testWithWeight.out",std::ios::in);
    // declare another IO_GenEvent for writing out some events
    HepMC::IO_GenEvent ascii_out("testWithWeight2.out",std::ios::out);
    //.....................................EVENT LOOP
    int icount=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        ++icount;
        os << "readWithWeight: Processing Event Number " << icount
                                << " its # " << evt->event_number()
                                << std::endl;
        if ( !evt->cross_section() ) {
          os << "testReadCrossSection: invalid cross-section!" << std::endl;
        }
        ascii_out << evt;

        // clean up and get next event
        delete evt;
        ascii_in >> evt;
    }
    //.....................................PRINT RESULT
    os << "readWithWeight processed " << icount << " events. Finished." << std::endl;
}
```

9.13 testHepMCIteration.cc.in

Use Matt's example_EventSelection along with example_UsingIterators to check **HepMC** (p. 15) iteration. Apply an event selection to the events in testHepMC.input Events containing a photon of pT $>$ 25 GeV pass the selection. Use iterators on these events.

```
// testHepMCIteration.cc.in
//
// garren@fnal.gov, May 2007
// Use Matt's example_EventSelection along with example_UsingIterators
// to check HepMC iteration.
// Apply an event selection to the events in testHepMC.input
```

```
// Events containing a photon of pT > 25 GeV pass the selection.
// Use iterators on these events.

#include <list>

#include "HepMC/IO_GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#include "HepMC/GenEvent.h"
#include "HepMC/GenRanges.h"

// define methods and classes used by this test
#include "IsGoodEvent.h"
#include "testHepMCIteration.h"

bool findW( HepMC::GenEvent* evt, std::ofstream& os);
bool simpleIter ( HepMC::GenEvent* evt, std::ostream& os = std::cout );
bool simpleIter2( HepMC::GenEvent* evt, std::ostream& os = std::cout );
bool simpleIter3( HepMC::GenEvent* evt, std::ostream& os = std::cout );
bool simpleIter4( HepMC::GenEvent* evt, std::ostream& os = std::cout );

class PrintW {
public:
    PrintW( std::ostream & os, int num ) : m_out( os ),m_event_num( num ) {}
    void operator()( HepMC::GenParticle* p ) {
        if ( IsWBoson(p) ) {
            m_out << std::endl;
            m_out << "A W boson has been found in event: " << m_event_num << std::endl;
            p->print( m_out );
            // return all parents
            // we do this by pointing to the production vertex of the W
            // particle and asking for all particle parents of that vertex
            m_out << "\t Its parents are: " << std::endl;
            if ( p->production_vertex() ) {
                std::for_each( p->particles_in(HepMC::parents).begin(),
                               p->particles_in(HepMC::parents).end(),
                               PrintParticle(m_out));
            }

            // return immediate children
            m_out << "\t\t" << "Its children are: " << std::endl;
            if ( p->end_vertex() ) {
                std::for_each( p->particles_out(HepMC::children).begin(),
                               p->particles_out(HepMC::children).end(),
                               PrintChildren(m_out));
            }

            // return all descendants
            // we do this by pointing to the end vertex of the W
            // particle and asking for all particle descendants of that vertex
            m_out << "\t\t Its descendants are: " << std::endl;
            if ( p->end_vertex() ) {
                std::for_each( p->particles_out(HepMC::descendants).begin(),
                               p->particles_out(HepMC::descendants).end(),
                               PrintDescendants(m_out));
            }
        }       // if IsWBoson
    }
private:
   std::ostream & m_out;
   int            m_event_num;
};

class PrintConstW {
public:
    PrintConstW( std::ostream & os, int num ) : m_out( os ),m_event_num( num ) {}
    void operator()( HepMC::GenParticle* p ) {
        if ( IsWBoson(p) ) {
            m_out << std::endl;
            m_out << "A W boson has been found in event: " << m_event_num << std::endl;
            p->print( m_out );
            // return all parents
            // we do this by pointing to the production vertex of the W
            // particle and asking for all particle parents of that vertex
            m_out << "\t Its parents are: " << std::endl;
            if ( p->production_vertex() ) {
                std::for_each( p->particles_in(HepMC::parents).begin(),
                               p->particles_in(HepMC::parents).end(),
                               PrintParticle(m_out));
            }

            // return immediate children
            m_out << "\t\t" << "Its children are: " << std::endl;
            if ( p->end_vertex() ) {
                std::for_each( p->particles_out(HepMC::children).begin(),
                               p->particles_out(HepMC::children).end(),
                               PrintChildren(m_out));
            }
```

```
            // return all descendants
            // we do this by pointing to the end vertex of the W
            // particle and asking for all particle descendants of that vertex
            m_out << "\t\t Its descendants are: " << std::endl;
            if ( p->end_vertex() ) {
                std::for_each( p->particles_out(HepMC::descendants).begin(),
                               p->particles_out(HepMC::descendants).end(),
                               PrintDescendants(m_out));
            }
        }       // if IsWBoson
    }
private:
   std::ostream & m_out;
   int            m_event_num;
};

int main() {
    // declare an input strategy to read the data produced with the
    // example_MyPythia
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    // define some output streams
    std::ofstream os( "testHepMCIteration.out" );
    std::ofstream os2( "testHepMCIteration2.out" );
    std::ofstream os3( "testHepMCIteration3.out" );
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    HepMC::GenEvent* evcopy;
    while ( evt ) {
        icount++;
        if ( icount%50==1 ) std::cout << "Processing Event Number " << icount
                                      << " its # " << evt->event_number()
                                      << std::endl;
        // icount of 100 should be the last event
        if ( icount==100 ) std::cout << "Processing Event Number " << icount
                                      << " its # " << evt->event_number()
                                      << std::endl;
        evcopy = evt;
        if ( is_good_event(evcopy) ) {
            ++num_good_events;
            // simple iteration several different ways
            os << "Event " << evcopy->event_number() << " is good " << std::endl;
            simpleIter( evcopy, os );
            os2 << "Event " << evcopy->event_number() << " is good " << std::endl;
            simpleIter2( evcopy, os2 );
            os3 << "Event " << evcopy->event_number() << " is good " << std::endl;
            simpleIter2( evcopy, os3 );
            std::cout << "Event " << evcopy->event_number() << " is good " << std::endl;
            simpleIter3( evcopy );
            simpleIter4( evcopy );
            // test iterators
            findW( evcopy, os );
            // this is the same as findW except that we use the STL for_each algorithm
            std::for_each( evt->particles_begin(), evt->particles_end(),
                           PrintW(os2,evcopy->event_number()));
            // repeat, using the const iterator
            std::for_each( evt->particles_begin(), evt->particles_end(),
                           PrintConstW(os3,evcopy->event_number()));
        }
        evcopy->clear();

        // clean up and get next event
        delete evt;
        evt = ascii_in.read_next_event();
    }
    //.....................................PRINT RESULT
    std::cout << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

bool simpleIter( HepMC::GenEvent* evt, std::ostream& os )
{
    // use GenEvent::vertex_iterator to fill a list of all
    // vertices in the event
    std::list<HepMC::GenVertex*> allvertices;
    for ( HepMC::GenEvent::vertex_iterator v = evt->vertices_begin();
          v != evt->vertices_end(); ++v ) {
        allvertices.push_back(*v);
    }

    // fill a list of all final state particles in the event, by requiring
    // that each particle satisfyies the IsFinalState predicate
    IsFinalState isfinal;
```

```
    std::list<HepMC::GenParticle*> finalstateparticles;
    for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
          p != evt->particles_end(); ++p ) {
        if ( isfinal(*p) ) finalstateparticles.push_back(*p);
    }

    // print all photons in the event that satisfy the IsPhoton criteria
    os << "photons in event " << evt->event_number() << ":" << std::endl;
    for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
          p != evt->particles_end(); ++p ) {
        if ( IsPhoton(*p) ) (*p)->print( os );
    }

    return true;
}

bool simpleIter2( HepMC::GenEvent* evt, std::ostream& os )
{
    // illustrates the use various helpful algorithms

    // use the STL copy algorithm to fill a list of all
    // vertices in the event
    std::list<HepMC::GenVertex*> allvertices2;
    copy( evt->vertices_begin(), evt->vertices_end(),
          back_inserter(allvertices2) );

    // fill a list of all final state particles in the event, by requiring
    // that each particle satisfyies the IsFinalState predicate
    // an STL-like algorithm called HepMC::copy_if is provided in the
    // GenEvent.h header to do this sort of operation more easily
    std::list<HepMC::GenParticle*> finalstateparticles2;
    HepMC::copy_if( evt->particles_begin(), evt->particles_end(),
                    back_inserter(finalstateparticles2), IsFinalState() );

    // use the STL for_each algorithm to
    // print all photons in the event that satisfy the IsPhoton criteria
    os << "photons in event " << evt->event_number() << ":" << std::endl;
    std::for_each(evt->particles_begin(), evt->particles_end(),
                  PrintPhoton(os));

    return true;
}

bool simpleIter3( HepMC::GenEvent* evt, std::ostream& os )
{
    // very simple illustration of using GenEventVertexRange
    // and GenEventParticleRange
    // NOTE that instead of creating this list,
    // you can just use GenEventVertexRange as if it were the list
    std::list<HepMC::GenVertex*> allvertices;
    HepMC::GenEventVertexRange vc(*evt);
    for ( HepMC::GenEvent::vertex_iterator v = vc.begin(); v != vc.end(); ++v ) {
        allvertices.push_back(*v);
    }

    // fill a list of all final state particles in the event, by requiring
    // that each particle satisfyies the IsFinalState predicate
    IsFinalState isfinal;
    std::list<HepMC::GenParticle*> finalstateparticles;
    HepMC::GenEventParticleRange pc(*evt);
    for ( HepMC::GenEvent::particle_iterator p = pc.begin(); p != pc.end(); ++p ) {
        if ( isfinal(*p) ) finalstateparticles.push_back(*p);
    }

    // print all photons in the event that satisfy the IsPhoton criteria
    os << "photons in event " << evt->event_number() << ":" << std::endl;
    std::for_each(pc.begin(), pc.end(), PrintPhoton(os));

    return true;
}

bool simpleIter4( HepMC::GenEvent* evt, std::ostream& os )
{
    // very simple illustration of using
    // GenEvent::vertex_range(), which returns GenEventVertexRange,
    // and GenEvent::particle_range(), which returns GenEventParticleRange
    // NOTE that instead of creating these lists,
    // you can just use GenEvent::vertex_range() and GenEvent::particle_range()
    // as if they were a list

    std::list<HepMC::GenVertex*> allvertices;
    for ( HepMC::GenEvent::vertex_iterator v = evt->vertex_range().begin();
          v != evt->vertex_range().end(); ++v ) {
        allvertices.push_back(*v);
    }

    // fill a list of all final state particles in the event, by requiring
```

```
    // that each particle satisfyies the IsFinalState predicate
    IsFinalState isfinal;
    std::list<HepMC::GenParticle*> finalstateparticles;
    for ( HepMC::GenEvent::particle_iterator p = evt->particle_range().begin();
          p != evt->particle_range().end(); ++p ) {
        if ( isfinal(*p) ) finalstateparticles.push_back(*p);
    }

    // print all photons in the event that satisfy the IsPhoton criteria
    os << "photons in event " << evt->event_number() << ":" << std::endl;
    std::for_each(evt->particle_range().begin(),
                  evt->particle_range().end(),
                  PrintPhoton(os));

    return true;
}

bool findW( HepMC::GenEvent* evt, std::ofstream& os )
{
    int num_W = 0;
    // use GenEvent::particle_iterator to find all W's in the event,
    // then
    // (1) for each W user the GenVertex::particle_iterator with a range of
    //     parents to return and print the immediate mothers of these W's.
    // (2) for each W user the GenVertex::particle_iterator with a range of
    //     descendants to return and print all descendants of these W's.
    for ( HepMC::GenEvent::particle_iterator p = evt->particles_begin();
          p != evt->particles_end(); ++p ) {
        if ( IsWBoson(*p) ) {
            ++num_W;
            os << std::endl;
            os << "A W boson has been found in event: " << evt->event_number() << std::endl;
            (*p)->print( os );
            // return all parents
            // we do this by pointing to the production vertex of the W
            // particle and asking for all particle parents of that vertex
            os << "\t Its parents are: " << std::endl;
            if ( (*p)->production_vertex() ) {
                for ( HepMC::GenVertex::particle_iterator mother
                          = (*p)->production_vertex()->
                          particles_begin(HepMC::parents);
                      mother != (*p)->production_vertex()->
                          particles_end(HepMC::parents);
                      ++mother ) {
                    os << "\t";
                    (*mother)->print( os );
                }
            }

            // return immediate children
            os << "\t\t" << "Its children are: " << std::endl;
            if ( (*p)->end_vertex() ) {
                for ( HepMC::GenVertex::particle_iterator child =
                        (*p)->end_vertex()->particles_begin(HepMC::children);
                      child != (*p)->end_vertex()->particles_end(HepMC::children);
                      ++child ) {
                    // make a copy
                    HepMC::GenVertex::particle_iterator cp = child;
                    // use the copy and the original
                    os << "\t\t\t (id,barcode,status) "
                       << (*cp)->pdg_id() << " "
                       << (*child)->barcode() << " "
                       << (*cp)->status() << std::endl;
                }
            }

            // return all descendants
            // we do this by pointing to the end vertex of the W
            // particle and asking for all particle descendants of that vertex
            os << "\t\t Its descendants are: " << std::endl;
            if ( (*p)->end_vertex() ) {
                for ( HepMC::GenVertex::particle_iterator des
                          =(*p)->end_vertex()->
                          particles_begin(HepMC::descendants);
                      des != (*p)->end_vertex()->
                          particles_end(HepMC::descendants);
                      ++des ) {
                    os << "\t\t";
                    (*des)->print( os );
                }
            }
        }       // if IsWBoson
    }   // end particle loop
    return true;
}
```

9.14 testMass.cc.in

Read events from testIOGenEvent.input Select events containing a photon of pT $>$ 25 GeV Add arbitrary PDF information to one of the good events Write the selected events and read them back in using an istream

```
//----------------------------------------------------------------------
// testMass.cc.in
//
// garren@fnal.gov, March 2006
// Read events written by example_MyPythia.cc
// Select events containing a photon of pT > 25 GeV
// Add arbitrary PDF information to one of the good events
// Add arbitrary HeavyIon information to one of the good events
// Write the selected events and read them back in using an istream
//----------------------------------------------------------------------

#include <cmath>        // for min()
#include <ostream>

#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/Version.h"

// define methods and classes used by this test
#include "IsGoodEvent.h"

void   massInfo( const HepMC::GenEvent*, std::ostream& os );

int main() {
    // output file
    std::ofstream os( "testMass.cout" );
    // read and process the input file
    {
        // declare an input strategy to read the data produced with the
        // example_MyPythia
        HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
        ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
        // declare another IO_GenEvent for output
        HepMC::IO_GenEvent ascii_out("testMass1.out",std::ios::out);
        // declare an instance of the event selection predicate
        IsGoodEvent is_good_event;
        // send version to output
        HepMC::version(os);
        //.....................................EVENT LOOP
        int icount=0;
        int num_good_events=0;
        double x1=0., x2=0., q=0., xf1=0., xf2=0.;
        HepMC::GenEvent* evt = ascii_in.read_next_event();
        while ( evt ) {
            icount++;
            if ( icount%50==1 ) os << "Processing Event Number " << icount
                                   << " its # " << evt->event_number()
                                   << std::endl;
            if ( is_good_event(evt) ) {
                if (num_good_events == 0 ) {
                    // add some arbitrary PDF information
                    x1 = std::min(0.8, 0.07 * icount);
                    x2 = 1-x1;
                    q = 1.69 * icount;
                    // use beam momentum
                    if( evt->valid_beam_particles() ) {
                        HepMC::GenParticle* bp1 = evt->beam_particles().first;
                        xf1 = x1*bp1->momentum().rho();
                        xf2 = x2*bp1->momentum().rho();
                    } else {
                        xf1 = x1*0.34;
                        xf2 = x2*0.34;
                    }
                    // provide optional pdf set id numbers
                    // (two ints at the end of the constructor)
                    HepMC::PdfInfo pdf( 2, 3, x1, x2, q, xf1, xf2, 230, 230);
                    evt->set_pdf_info(pdf);
                    // add some arbitrary HeavyIon information
                    HepMC::HeavyIon ion(23,11,12,15,3,5,0,0,0,0.0145);
                    evt->set_heavy_ion( ion );
                }
                os << "saving Event " << evt->event_number() << std::endl;
                if( evt->weights().size() > 0 ) {
                    os << "Weights: ";
                    evt->weights().print(os);
                }
                ascii_out << evt;
                ++num_good_events;
            }
```

```
            // clean up and get next event
            delete evt;
            ascii_in >> evt;
        }
        //.....................................PRINT RESULT
        os << num_good_events << " out of " << icount
           << " processed events passed the cuts. Finished." << std::endl;
    }
    // now read the file we just created
    {
        // declare an input strategy
        const char infile[] = "testMass1.out";
        std::ifstream istr( infile );
        if( !istr ) {
          std::cerr << "testMass: cannot open " << infile << std::endl;
          exit(-1);
        }
        HepMC::IO_GenEvent xin(istr);
        // declare another IO_GenEvent for output
        HepMC::IO_GenEvent xout("testMass2.out",std::ios::out);
        //.....................................EVENT LOOP
        int ixin=0;
        HepMC::GenEvent* evt = xin.read_next_event();
        while ( evt ) {
            ixin++;
            os << "reading Event " << evt->event_number() << std::endl;
            if( evt->weights().size() > 0 ) {
                os << "Weights: ";
                evt->weights().print(os);
            }
            xout << evt;
            // look at mass info
            massInfo(evt,os);

            // clean up and get next event
            delete evt;
            xin >> evt;
        }
        //.....................................PRINT RESULT
        os << ixin << " events in the second pass. Finished." << std::endl;
    }
}

void massInfo( const HepMC::GenEvent* e, std::ostream& os )
{
   double gm, m, d;
   for ( HepMC::GenEvent::particle_const_iterator p = e->particles_begin(); p != e->
      particles_end();
         ++p ) {

       gm = (*p)->generated_mass();
       m = (*p)->momentum().m();
       d = fabs(m-gm);
       if( d > 1.0e-5 ) {
            os << "Event " << e->event_number()
               << " Particle " << (*p)->barcode()
               << " " << (*p)->pdg_id()
               << " generated mass " << gm
               << " mass from momentum " << m
               << " difference " << d << std::endl;
       }
   }
}
```

9.15 testMultipleCopies.cc.in

Multiple events in memory at the same time run with valgrind or some other leak checker

```
// testMultipleCopies.cc.in
//
// garren@fnal.gov, January 2008
// Multiple events in memory at the same time
// run with valgrind or some other leak checker
//

#include <fstream>

#include "HepMC/IO_GenEvent.h"
#include "HepMC/GenEvent.h"
#include "HepMC/CompareGenEvent.h"

// define methods and classes used by this test
```

```
#include "IsGoodEvent.h"

int main() {
    // use output file
    std::ofstream os( "testMultipleCopies.out" );
    {
        // declare an input strategy
        HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
        // declare another input strategy
        HepMC::IO_GenEvent ascii_in2("@srcdir@/testHepMCVarious.input",std::ios::in);
        std::ofstream out1( "testMultipleOriginals.out" );
        std::ofstream out2( "testMultipleCopies1.out" );
        std::ofstream out3( "testMultipleCopies2.out" );
        // declare an instance of the event selection predicate
        IsGoodEvent is_good_event;

        //.....................................EVENT LOOP
        int icount=0;
        int num_good_events=0;
        int icnt;
        HepMC::GenEvent* evt1 = ascii_in.read_next_event();
        HepMC::GenEvent* evt2 = ascii_in2.read_next_event();
        HepMC::GenEvent* evt3 = ascii_in.read_next_event();

        while ( evt1 && evt2 ) {
            icount++;
            if ( icount%50==1 ) os << "Processing Event Number " << icount
                                   << " stream 1 # " << evt1->event_number()
                                   << " stream 2 # " << evt2->event_number()
                                   << std::endl;

            if ( is_good_event(evt1) ) {

                os << "good event in stream 1 # "
                   << evt1->event_number() << std::endl;
                evt1->print(out1);
                ++num_good_events;
                HepMC::GenEvent ec = (*evt1);
                ec.print(out3);
                icnt=0;
                for ( HepMC::GenEvent::particle_const_iterator p1 =  ec.
      particles_begin();
                      p1 !=  ec.particles_end(); ++p1 ) {
                      ++icnt;
                    os << "particle " << icnt << " barcode " <<(*p1)->barcode() << std::endl;
                }
                HepMC::GenEvent* evt4 = new HepMC::GenEvent(*evt1);
                evt4->print(out2);
                if( !compareGenEvent(evt1,evt4) ) { return -1; }
                delete evt4;
            }

            // clean up and get next events
            delete evt1;
            delete evt2;
            ascii_in >> evt1;
            ascii_in2 >> evt2;
        }
        // might have either evt1 or evt2 still in memory, cleanup here
        delete evt1;
        delete evt2;
        delete evt3;

        //.....................................PRINT RESULT
        os << std::endl;
        os << num_good_events << " out of " << icount
           << " processed events passed the cuts." << std::endl;
        os << std::endl;
        os << " GenEvent copy constructor passes the test" << std::endl;
        os << std::endl;
    }

    // test operator= and swap
    {
        // declare an input strategy
        HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
        //
        HepMC::GenEvent* evt5 = ascii_in.read_next_event();
        HepMC::GenEvent* evt6 = new HepMC::GenEvent();
        os << "event number for evt5: " << evt5->event_number() << std::endl;
        os << "event number for evt6: " << evt6->event_number() << std::endl;
        // copy  GenEvent object
        (*evt6) = (*evt5);
        if( !compareGenEvent(evt5,evt6) ) { return -4; }
        delete evt5;
        os << "event number for evt6 after copy: " << evt6->event_number() << std::endl;
        os << std::endl;
```

```
        delete evt6;
        os << " GenEvent operator= passes the test" << std::endl;
        os << std::endl;

        evt5 = ascii_in.read_next_event();
        evt6 = ascii_in.read_next_event();
        HepMC::GenEvent* evt7 = new HepMC::GenEvent(*evt5);
        HepMC::GenEvent* evt8 = new HepMC::GenEvent(*evt6);
        os << "event number for evt5: " << evt5->event_number() << std::endl;
        os << "event number for evt6: " << evt6->event_number() << std::endl;
        os << "before swap, evt5 has: " << evt5->vertices_size() << " vertices and "
           << evt5->particles_size() << " particles" << std::endl;
        os << "before swap, evt6 has: " << evt6->vertices_size() << " vertices and "
           << evt6->particles_size() << " particles" << std::endl;
        os << "before swap, evt7 has: " << evt7->vertices_size() << " vertices and "
           << evt7->particles_size() << " particles" << std::endl;
        os << "before swap, evt8 has: " << evt8->vertices_size() << " vertices and "
           << evt8->particles_size() << " particles" << std::endl;
        (*evt6).swap((*evt5));
        os << "event number for evt5 after swap: " << evt5->event_number() << std::endl;
        os << "event number for evt6 after swap: " << evt6->event_number() << std::endl;
        // evt6 should now match evt7
        os << "after swap, evt6 has: " << evt6->vertices_size() << " vertices and "
           << evt6->particles_size() << " particles" << std::endl;
        os << "after swap, evt7 has: " << evt7->vertices_size() << " vertices and "
           << evt7->particles_size() << " particles" << std::endl;
        if( !compareGenEvent(evt6,evt7) ) { return -6; }
        // evt5 should now match evt8
        os << "after swap, evt5 has: " << evt5->vertices_size() << " vertices and "
           << evt5->particles_size() << " particles" << std::endl;
        os << "after swap, evt8 has: " << evt8->vertices_size() << " vertices and "
           << evt8->particles_size() << " particles" << std::endl;
        if( !compareGenEvent(evt5,evt8) ) { return -5; }
        // cleanup
        delete evt5;
        delete evt6;
        delete evt7;
        delete evt8;
        os << std::endl;
        os << " GenEvent swap passes the test" << std::endl;
        os << std::endl;
    }
    return 0;
}
```

9.16 testPrintBug.cc

Thanks to Bob McElrath and Frank Siegert for this test

```
//
// Thanks to Bob McElrath and Frank Siegert for this test
//

#include <fstream>

#include "HepMC/GenEvent.h"
#include "HepMC/SimpleVector.h"

int main()
{
  HepMC::GenEvent* p_event;

  p_event = new HepMC::GenEvent();
  p_event->use_units(HepMC::Units::GEV, HepMC::Units::MM);

  // define an output stream
  std::ofstream os( "testPrintBug.out" );

  for(int i=0; i<10; i++) {
    HepMC::FourVector vector(1.0,1.0,1.0,1.0);
    HepMC::GenVertex* vertex = new HepMC::GenVertex(vector,i);
    for(int j=0; j<3; j++) {
      HepMC::GenParticle* particle = new HepMC::GenParticle(vector,1,2);
      vertex->add_particle_in(particle);
    }
    for(int j=0; j<3; j++) {
      HepMC::GenParticle* particle = new HepMC::GenParticle(vector,1,2);
      vertex->add_particle_out(particle);
    }
    p_event->add_vertex(vertex);
  }
  p_event->print(os);
```

```
  // cleanup
  delete p_event;
  return 0;
}
```

9.17 testSimpleVector.cc

Exercise all the vector methods

```
//
// First pass - simply exercise all the vector methods
//
#include <iostream>

#include "HepMC/SimpleVector.h"

int main()
{
  // ThreeVector
  HepMC::ThreeVector vector3;
  HepMC::ThreeVector v3(1.1,2.2,3.3);
  HepMC::ThreeVector vx(1.34);

  HepMC::ThreeVector v3copy( v3 );

  double eps = 4.e-15; // allowed difference between doubles
  int numbad = 0;

  double x = v3.x();
  double y = v3.y();
  double z = v3.z();
  double p2 = v3.perp2();
  double pt = v3.perp();
  double r = v3.r();
  double th = v3.theta();
  double ph = v3.phi();
  double mag = std::sqrt(x*x + y*y + z*z);
  double pperp = std::sqrt(x*x + y*y);

  vx.set(1., 2., 3.);
  vx.setX(1.1);
  vx.setY(2.3);
  vx.setZ(4.4);
  vx.setPhi(0.12);
  vx.setTheta(0.54);

  vector3 = v3;

  if( fabs( mag - r ) > eps ) {
     std::cout << "different ThreeVector magnitude: " << mag << " " << r << std::endl;
     std::cout << "difference is : " << ( mag - r ) << std::endl;
     ++numbad;
  }
  if( fabs( pperp - pt ) > eps ) {
     std::cout << "different ThreeVector Pt: " << pperp << " " << pt << std::endl;
     std::cout << "difference is : " << ( pperp - pt ) << std::endl;
     ++numbad;
  }

  if( v3 == vector3 ) {
  } else {
     ++numbad;
     std::cout << "vectors v3 and vector3 are different" << std::endl;
  }
  if( v3 != v3copy ) {
     ++numbad;
     std::cout << "vectors v3 and v3copy are different" << std::endl;
  }

  // FourVector
  HepMC::FourVector vector;
  HepMC::FourVector v4(1.1,2.2,3.3,4.4);
  HepMC::FourVector vt(1.34);

  HepMC::FourVector vectorcopy( v4 );
  vector = v4;

  double px = v4.px();
  double py = v4.py();
  double pz = v4.pz();
  double e  = v4.e();
   x = vectorcopy.x();
```

```
 y = vectorcopy.y();
 z = vectorcopy.z();
double t = vectorcopy.t();

 p2 = v4.perp2();
 pt = v4.perp();
 th = v4.theta();
 ph = v4.phi();
 r = v4.rho();
double masssq1 = v4.m2();
double mass1 = v4.m();
double pr1 = v4.pseudoRapidity();
double eta1 = v4.eta();
double masssq2 = vector.m2();
double mass2 = vector.m();
double pr2 = vector.pseudoRapidity();
double eta2 = vector.eta();

vt.set(1., 2., 3., 5.5);
vt.setX(1.1);
vt.setY(2.3);
vt.setZ(4.4);
vt.setT(6.5);
vt.setPx(3.1);
vt.setPy(2.2);
vt.setPz(-1.1);
vt.setE(5.4);

mag = std::sqrt(x*x + y*y + z*z);
pperp = std::sqrt(x*x + y*y);
if( fabs( mag - r ) > eps ) {
   std::cout << "different FourVector magnitude: " << mag << " " << r << std::endl;
   std::cout << "difference is : " << ( mag - r ) << std::endl;
   ++numbad;
}
if( fabs( pperp - pt ) > eps ) {
   std::cout << "different FourVector Pt: " << pperp << " " << pt << std::endl;
   std::cout << "difference is : " << ( pperp - pt ) << std::endl;
   ++numbad;
}

if( px != x ) {
   std::cout << "different X values: " << px << " " << x << std::endl;
   ++numbad;
}
if( py != y ) {
   std::cout << "different Y values: " << py << " " << y << std::endl;
   ++numbad;
}
if( pz != z ) {
   std::cout << "different Z values: " << pz << " " << z << std::endl;
   ++numbad;
}
if( e != t ) {
   std::cout << "different E values: " << e << " " << t << std::endl;
   ++numbad;
}
if( fabs( masssq1 - masssq2 ) > eps ) {
   std::cout << "different mass sq values: " << masssq1 << " " << masssq2 << std::endl;
   std::cout << "difference is : " << ( masssq1 - masssq2 ) << std::endl;
   ++numbad;
}
if( fabs( mass1 - mass2 ) > eps ) {
   std::cout << "different mass values: " << mass1 << " " << mass2 << std::endl;
   std::cout << "difference is : " << ( mass1 - mass2 ) << std::endl;
   ++numbad;
}
if( fabs( pr1 - pr2 ) > eps ) {
   std::cout << "different pseudorapidity values: " << pr1 << " " << pr2 << std::endl;
   std::cout << "difference is : " << ( pr1 - pr2 ) << std::endl;
   ++numbad;
}
if( fabs( eta1 - eta2 ) > eps ) {
   std::cout << "different eta values: " << eta1 << " " << eta2 << std::endl;
   std::cout << "difference is : " << ( eta1 - eta2 ) << std::endl;
   ++numbad;
}
if( v4 == vector ) {
} else {
   std::cout << "vectors v and vector are different" << std::endl;
   ++numbad;
}
if( v4 != vectorcopy ) {
   std::cout << "vectors v and vectorcopy are different" << std::endl;
   ++numbad;
}
```

```
  return numbad;
}
```

9.18 testStreamIO.cc.in

Use streaming IO to read and write a file

```
// testStreamIO.cc.in
//
// garren@fnal.gov, March 2006
//
// The same as testHepMC, but using the IO stream directly
//

#include <fstream>

#include "HepMC/GenEvent.h"
#include "HepMC/IO_AsciiParticles.h"
#ifdef HEPMC_HAS_IO_GENEVENT
#include "HepMC/IO_GenEvent.h"
#endif
#include "HepMC/Version.h"
#include "HepMC/IO_Exception.h"

// define methods and classes used by this test
#include "IsGoodEvent.h"
#include "testHepMCMethods.h"

void read_testIOGenEvent(std::ostream & os);
void read_variousFormats(std::ostream & os);
void write_to_stream(std::ostream & os);
void write_to_stream3(std::ostream & os);
void read_from_stream4(std::ostream & os);

int main() {
    std::ofstream os( "testStreamIO.cout" );
    std::ofstream osv( "testStreamIOVarious.cout" );
    write_to_stream(os);
    read_testIOGenEvent(os);
    read_variousFormats(osv);
    write_to_stream3(os);
    read_from_stream4(os);
    return 0;
}

void write_to_stream(std::ostream & os)
{
    os << std::endl;
    os << "basic IO_GenEvent input with streaming output" << std::endl;
    // declare an input strategy to read the data produced with the
    // example_MyPythia - units are GeV and mm
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    // declare an output stream
    const char outfile[] = "testStreamIO.out";
    std::ofstream ascii_out( outfile );
    if( !ascii_out ) {
      std::cerr << "cannot open " << outfile << std::endl;
      exit(-1);
    }
    ascii_out.precision(16);
    HepMC::write_HepMC_IO_block_begin( ascii_out );
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        icount++;
        if ( icount%50==1 ) os << "Processing Event Number " << icount
                                  << " its # " << evt->event_number()
                                  << std::endl;
        if ( is_good_event(evt) ) {
            ++num_good_events;
            particleTypes( evt, os );
            ascii_out << (*evt);
        }

        // clean up and get next event
        delete evt;
        ascii_in >> evt;
```

```
    }
    HepMC::write_HepMC_IO_block_end( ascii_out );
    //.....................................PRINT RESULT
    os << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

void read_testIOGenEvent(std::ostream & os)
{
    os << std::endl;
    os << "streaming input and output" << std::endl;
    // input units are GeV and mm
    const char infile[] = "@srcdir@/testIOGenEvent.input";
    std::ifstream is( infile );
    if( !is ) {
      std::cerr << "cannot open " << infile << std::endl;
      exit(-1);
    }
    // declare an output stream
    const char outfile[] = "testStreamIO2.out";
    std::ofstream ascii_out( outfile );
    if( !ascii_out ) {
      std::cerr << "cannot open " << outfile << std::endl;
      exit(-1);
    }
    ascii_out.precision(16);
    HepMC::write_HepMC_IO_block_begin( ascii_out );
    // declare another output stream to test precision
    const char poutfile[] = "testStreamIOprecision.out";
    std::ofstream pout( poutfile );
    if( !pout ) {
      std::cerr << "cannot open " << poutfile << std::endl;
      exit(-1);
    }
    pout.precision(10);
    // declare an IO_AsciiParticle for output
    HepMC::IO_AsciiParticles particle_out("testStreamIOParticle.out",std::ios::out);
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent evt;
    while ( is ) {
        // WARNING - we are not using pointers, so this could be an empty event
        is >> evt;
        // make sure this is a valid event
        if( evt.is_valid() ) {
            ++icount;
            if ( icount%50==1 ) os << "Processing Event Number " << icount
                                   << " its # " << evt.event_number()
                                   << std::endl;
            if ( is_good_event( &evt ) ) {
                ++num_good_events;
                particleTypes(&evt,os);
                ascii_out << evt;
                pout << evt;
                // We must explicitly create the pointer if we want to use this event
                // with any IO strategy (e.g., IO_AsciiParticles)
                HepMC::GenEvent* pevt= &evt;
                particle_out << pevt;
            }
        }
    }
    HepMC::write_HepMC_IO_block_end( ascii_out );
    //.....................................PRINT RESULT
    os << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

void read_variousFormats(std::ostream & os)
{
    os << std::endl;
    os << "process varied input" << std::endl;
    // declare an input stream
    const char infile[] = "@srcdir@/testHepMCVarious.input";
    std::ifstream is( infile );
    if( !is ) {
      std::cerr << "cannot open " << infile << std::endl;
      exit(-1);
    }
    // set input units
    HepMC::set_input_units( is, HepMC::Units::GEV, HepMC::Units::MM );
    // declare an output stream
    const char outfile[] = "testStreamIOVarious.out";
    std::ofstream ascii_out( outfile );
    if( !ascii_out ) {
```

```
      std::cerr << "cannot open " << outfile << std::endl;
      exit(-1);
    }
    ascii_out.precision(16);
    HepMC::write_HepMC_IO_block_begin( ascii_out );
    //.....................................EVENT LOOP
    int icount=0, ibad=0;
    HepMC::GenEvent evt;
    while ( is ) {
        // we have to do our own try/catch blocks
        try {
           is >> evt;
        }
        catch (HepMC::IO_Exception& e) {
            evt.clear();
            ++ibad;
        }
        // WARNING - we are not using pointers, so this could be an empty event
        // make sure this is a valid event
        if( evt.is_valid() ) {
            icount++;
            double pim;
            os << "Processing Event Number " << icount
                      << " its # " << evt.event_number()
                      << std::endl;
            ascii_out << evt;
            // units should be unknown
            evt.write_units(os);
            pim = findPiZero(&evt);
            os << " pizero mass: " << pim << std::endl;
            // set units to GeV and mm
            evt.use_units(HepMC::Units::GEV, HepMC::Units::MM);
            evt.write_units(os);
            pim = findPiZero(&evt);
            os << " pizero mass: " << pim
                      << " " << HepMC::Units::name( evt.momentum_unit() ) << std::endl;
            // convert units to MeV
            evt.use_units(HepMC::Units::MEV, HepMC::Units::MM);
            evt.write_units(os);
            pim = findPiZero(&evt);
            os << " pizero mass: " << pim
                      << " " << HepMC::Units::name( evt.momentum_unit() ) << std::endl;
        }
    }
    std::cout << "testSteamIO: the HeavyIon and PdfInfo input stream errors are intentional" << std::endl;
    HepMC::write_HepMC_IO_block_end( ascii_out );
    //.....................................PRINT RESULT
    os << icount << " valid events processed.  " ;
    os << ibad << " invalid events processed. Finished." << std::endl;
}

void write_to_stream3(std::ostream & os)
{
    os << std::endl;
    os << "basic IO_GenEvent input with streaming output using member function" << std::endl;
    // declare an input strategy to read the data produced with the
    // example_MyPythia - units are GeV and mm
    HepMC::IO_GenEvent ascii_in("@srcdir@/testIOGenEvent.input",std::ios::in);
    ascii_in.use_input_units( HepMC::Units::GEV, HepMC::Units::MM );
    // declare an output stream
    const char outfile[] = "testStreamIO3.out";
    std::ofstream ascii_out( outfile );
    if( !ascii_out ) {
      std::cerr << "cannot open " << outfile << std::endl;
      exit(-1);
    }
    ascii_out.precision(16);
    HepMC::write_HepMC_IO_block_begin( ascii_out );
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent* evt = ascii_in.read_next_event();
    while ( evt ) {
        icount++;
        if ( icount%50==1 ) os << "Processing Event Number " << icount
                                  << " its # " << evt->event_number()
                                  << std::endl;
        if ( is_good_event(evt) ) {
            ++num_good_events;
            particleTypes( evt, os );
            evt->write(ascii_out);
        }

        // clean up and get next event
        delete evt;
```

```
        ascii_in >> evt;
    }
    HepMC::write_HepMC_IO_block_end( ascii_out );
    //.....................................PRINT RESULT
    os << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}

void read_from_stream4(std::ostream & os)
{
    os << std::endl;
    os << "streaming input and output using member functions" << std::endl;
    // input units are GeV and mm
    const char infile[] = "@srcdir@/testIOGenEvent.input";
    std::ifstream is( infile );
    if( !is ) {
      std::cerr << "cannot open " << infile << std::endl;
      exit(-1);
    }
    // declare an output stream
    const char outfile[] = "testStreamIO4.out";
    std::ofstream ascii_out( outfile );
    if( !ascii_out ) {
      std::cerr << "cannot open " << outfile << std::endl;
      exit(-1);
    }
    ascii_out.precision(16);
    HepMC::write_HepMC_IO_block_begin( ascii_out );
    // declare an instance of the event selection predicate
    IsGoodEvent is_good_event;
    //.....................................EVENT LOOP
    int icount=0;
    int num_good_events=0;
    HepMC::GenEvent evt;
    while ( is ) {
        // WARNING - we are not using pointers, so this could be an empty event
        evt.read(is);
        // make sure this is a valid event
        if( evt.is_valid() ) {
            ++icount;
            if ( icount%50==1 ) os << "Processing Event Number " << icount
                                   << " its # " << evt.event_number()
                                   << std::endl;
            if ( is_good_event( &evt ) ) {
                ++num_good_events;
                particleTypes(&evt,os);
                evt.write(ascii_out);
            }
        }
    }
    HepMC::write_HepMC_IO_block_end( ascii_out );
    //.....................................PRINT RESULT
    os << num_good_events << " out of " << icount
              << " processed events passed the cuts. Finished." << std::endl;
}
```

9.19 testUnits.cc

Test MomentumUnits and PositionUnits Make sure set and change methods work as expected.

```
//
// Test Units
//
#include <iostream>

#include "HepMC/Units.h"

int main()
{

  int err = 0;
  double cf;

  std::cout << "Default units: " << HepMC::Units::name(
      HepMC::Units::default_momentum_unit())
     << " " << HepMC::Units::name(HepMC::Units::default_length_unit()) << std::endl;

  // check momentum conversion factors
  cf = conversion_factor( HepMC::Units::GEV, HepMC::Units::GEV );
  if( cf != 1 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
```

```
              << " for GEV to GEV - should be 1 \n";
  }
  cf = conversion_factor( HepMC::Units::MEV, HepMC::Units::MEV );
  if( cf != 1 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for MEV to MEV - should be 1 \n";
  }
  cf = conversion_factor( HepMC::Units::MEV, HepMC::Units::GEV );
  if( cf != 0.001 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for MEV to GEV - should be 0.001 \n";
  }
  cf = conversion_factor( HepMC::Units::GEV, HepMC::Units::MEV );
  if( cf != 1000.0 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for GEV to MEV - should be 1000 \n";
  }

  // check length conversion factors
  cf = conversion_factor( HepMC::Units::MM, HepMC::Units::MM );
  if( cf != 1 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for MM to MM - should be 1 \n";
  }
  cf = conversion_factor( HepMC::Units::CM, HepMC::Units::CM );
  if( cf != 1 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for CM to CM - should be 1 \n";
  }
  cf = conversion_factor( HepMC::Units::CM, HepMC::Units::MM );
  if( cf != 10.0 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for CM to MM - should be 10 \n";
  }
  cf = conversion_factor( HepMC::Units::MM, HepMC::Units::CM );
  if( cf != 0.1 ) {
      ++err;
      std::cerr << "wrong conversion factor " << cf
              << " for MM to CM - should be 0.1 \n";
  }

  return err;
}
```

9.20 VectorConversion.h

This example converts from ThreeVector and FourVector to CLHEP::Hep3Vector and CLHEP::HepLorentzVector Similar (or perhaps templated) conversion methods could be added to any vector class.

```
#ifndef VECTOR_CONVERSION_H
#define VECTOR_CONVERSION_H
// garren@fnal.gov, January 2007
//
// This example converts from ThreeVector and FourVector to
// CLHEP::Hep3Vector and CLHEP::HepLorentzVector
// Similar (or perhaps templated) conversion methods could be added to
// any vector class.
//

#include "HepMC/SimpleVector.h"
#include "CLHEP/Vector/LorentzVector.h"

inline CLHEP::Hep3Vector convertTo( const HepMC::ThreeVector& v )
     { return CLHEP::Hep3Vector( v.x(), v.y(), v.z() ); }

inline CLHEP::HepLorentzVector convertTo( const HepMC::FourVector& v )
     { return CLHEP::HepLorentzVector( v.x(), v.y(), v.z(), v.t() ); }

#endif  // VECTOR_CONVERSION_H
```

Index