Monaspace family fonts (*experimental*) User's Guide for LaTeX (v0.1)

Cédric Pierquet
cpierquet@outlook.fr

2nd December 2025

1 What is Monaspace ?

1.1 Presentation (actual version is v1.300)

One superfamily / Five fonts / Texture healing

Since the earliest days of the teletype machine, code has been set in monospaced type—letters, on a grid. Monaspace is a new type system that advances the state of the art for the display of code on screen.

Texture healing preserves the monospace grid, and works in most editors without needing new software or editor plugins.

For further information, https://monaspace.githubnext.com is the best source!

1.2 License

Fonts are licensed under the SIL Open Font License, Version 1.1.

They require LuaTeX or XeTeX as engine and the `fontspec` package[1].

[1]Please read the documentation `fontspec.pdf`.

2 Usage

2.1 Loading mono font

Several weights are available:

ExtraLight / Light / Regular / **Medium** / **SemiBold**

Several styles are available:

Argon / Krypton / Neon / *Radon* / Xenon

Files `Monaspace<style>-<weight>.fontspec` are provided to ensure that Italic, Bold, BoldItalic are properly loaded.

A basic call for Monaspace mono font could be:

```
\usepackage{monaspace-otf}
```

It loads `MonaspaceArgon-Regular` as mono font.

```
def Fibonacci(n) :
  # Check if input is 0 then it will print incorrect input
  if n < 0 :
    print("Incorrect␣input")
  elif n == 0 :
    return 0
  elif n == 1 or n == 2 :
    return 1
  else :
    return Fibonacci(n-1) + Fibonacci(n-2)
```

2.2 Options

Options can be given within the package:

- `Weight=...`, within `ExtraLight/Light/Regular/Medium/SemiBold`;
- `Style=...`, within `Argon/Krypton/Neon/Radon/Xenon`;
- the boolean `nott` for not loading monospace as mono font;
- `StylisticSet=...`
- `RawFeature=...`
- `CharacterVariant=...`
- `Scale=...`

2.3 Manual loading

It's also possible to load mono font with `\setmonofont`, thanks to `fontspec` files for example.

```
%w/o package loaded
\usepackage{fontspec}
\usepackage{unicode-math}
\setmonofont{Monaspace<style>-<weight>}[options]
```

2.4 Fontfamily

Combination of *style/weight* `fontfamily` are provided:

`\monaspaceargonextralight`	o008 iIlL1 g9qCGQ <=>
`\monaspaceargonextralightcode`	o008 iIlL1 g9qCGQ ⟺
`\monaspaceargonlight`	o008 iIlL1 g9qCGQ <=>
`\monaspaceargonlightcode`	o008 iIlL1 g9qCGQ ⟺
`\monaspaceargon`	o008 iIlL1 g9qCGQ <=>
`\monaspaceargoncode`	o008 iIlL1 g9qCGQ ⟺
`\monaspaceargonmedium`	**o008 iIlL1 g9qCGQ <=>**
`\monaspaceargonmediumcode`	**o008 iIlL1 g9qCGQ ⟺**
`\monaspaceargonsemibold`	**o008 iIlL1 g9qCGQ <=>**
`\monaspaceargonsemiboldcode`	**o008 iIlL1 g9qCGQ ⟺**

`\monaspacekryptonextralight`	o008 iIlL1 g9qCGQ <=>
`\monaspacekryptonextralightcode`	o008 iIlL1 g9qCGQ ⟺
`\monaspacekryptonlight`	o008 iIlL1 g9qCGQ <=>
`\monaspacekryptonlightcode`	o008 iIlL1 g9qCGQ ⟺
`\monaspacekrypton`	o008 iIlL1 g9qCGQ <=>
`\monaspacekryptoncode`	o008 iIlL1 g9qCGQ ⟺
`\monaspacekryptonmedium`	**o008 iIlL1 g9qCGQ <=>**
`\monaspacekryptonmediumcode`	**o008 iIlL1 g9qCGQ ⟺**
`\monaspacekryptonsemibold`	**o008 iIlL1 g9qCGQ <=>**
`\monaspacekryptonsemiboldcode`	**o008 iIlL1 g9qCGQ ⟺**

\monaspaceneonextralight	oO08 iIlL1 g9qCGQ <=>
\monaspaceneonextralightcode	oO08 iIlL1 g9qCGQ ⟺
\monaspaceneonlight	oO08 iIlL1 g9qCGQ <=>
\monaspaceneonlightcode	oO08 iIlL1 g9qCGQ ⟺
\monaspaceneon	**oO08 iIlL1 g9qCGQ <=>**
\monaspaceneoncode	**oO08 iIlL1 g9qCGQ ⟺**
\monaspaceneonmedium	**oO08 iIlL1 g9qCGQ <=>**
\monaspaceneonmediumcode	**oO08 iIlL1 g9qCGQ ⟺**
\monaspaceneonsemibold	**oO08 iIlL1 g9qCGQ <=>**
\monaspaceneonsemiboldcode	**oO08 iIlL1 g9qCGQ ⟺**
\monaspaceradonextralight	*oO08 iIlL1 g9qCGQ <=>*
\monaspaceradonextralightcode	*oO08 iIlL1 g9qCGQ ⟺*
\monaspaceradonlight	*oO08 iIlL1 g9qCGQ <=>*
\monaspaceradonlightcode	*oO08 iIlL1 g9qCGQ ⟺*
\monaspaceradon	***oO08 iIlL1 g9qCGQ <=>***
\monaspaceradoncode	***oO08 iIlL1 g9qCGQ ⟺***
\monaspaceradonmedium	***oO08 iIlL1 g9qCGQ <=>***
\monaspaceradonmediumcode	***oO08 iIlL1 g9qCGQ ⟺***
\monaspaceradonsemibold	***oO08 iIlL1 g9qCGQ <=>***
\monaspaceradonsemiboldcode	***oO08 iIlL1 g9qCGQ ⟺***
\monaspacexenonextralight	oO08 iIlL1 g9qCGQ <=>
\monaspacexenonextralightcode	oO08 iIlL1 g9qCGQ ⟺
\monaspacexenonlight	oO08 iIlL1 g9qCGQ <=>
\monaspacexenonlightcode	oO08 iIlL1 g9qCGQ ⟺
\monaspacexenon	oO08 iIlL1 g9qCGQ <=>
\monaspacexenoncode	oO08 iIlL1 g9qCGQ ⟺
\monaspacexenonmedium	**oO08 iIlL1 g9qCGQ <=>**
\monaspacexenonmediumcode	**oO08 iIlL1 g9qCGQ ⟺**
\monaspacexenonsemibold	**oO08 iIlL1 g9qCGQ <=>**
\monaspacexenonsemiboldcode	**oO08 iIlL1 g9qCGQ ⟺**

code version are declared with ligatures and all *ssXX*.

3 Openfeatures

See https://monaspace.githubnext.com/#code-ligatures.

3.1 Ligatures

The `liga` font feature enables customized spacing of repeating characters, like /// or ||. It is designed to avoid activating inside longer sequences like ////.

There are ten groups of coding ligatures, separated into stylistic sets. You may be able to enable or disable individual sets selectively:

- `ss01`: ligatures related to the equals glyph
- `ss02`: ligatures for greater/less or equal
- `ss03`: ligatures related to arrows
- `ss04`: ligatures related to markup
- `ss05`: ligatures related to the F# programming language
- `ss06`: ligatures related to repeated uses of #, +, and &.
- `ss07`: ligatures related to colons
- `ss08`: ligatures related to combinations of periods with other glyphs
- `ss09`: ligatures related to combinations of the greater/less than and equals signs
- `ss10`: other tags

You can see an interactive display of all the ligatures on the Monaspace website

3.2 Character Variants

Specific characters have variants that you can optionally enable using the `cvNN` opentype feature:

- `cv01-cv09`: figure variants
- `cv01`: 0 alternates (`1/2/3/4` for plain/slash/reverse slash/cut-out slash)
- `cv02`: 1 alternate (no serif)
- `cv10-cv29`: letter variants
- `cv10`: l i alternates (Neon, Argon, Xenon, Radon)
- `cv11`: j f r t alternates (Neon, Argon)
- `cv30-cv59`: symbol variants

- `cv30`: * vertically aligned closer to the top of the space, similar to how the asterisk was in Monaspace 1.0
- `cv31`: * 6-pointed asterisk
- `cv32`: leq/geq angled lower line
- `cv60-cv79`: optional ligatures
- `cv60`: forces the <= pair to render in a fashion that matches => instead of swapping for leq.
- `cv61`: enables the optional closed square ligature for [].
- `cv62`: @_ ligature

4 Samples (mono)

4.1 Argo(Mono)

```
\lstset{
  language=python,
  basicstyle=\footnotesize\monaspaceargon,
  commentstyle=\itshape\color{gray},
  keywordstyle=\bfseries\color{magenta},
  tabsize=4,
  frame=single,
  columns=flexible,
  showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 <= n <= 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

4.2 Krypton(Mono)

```
\lstset{
  language=python,
  basicstyle=\footnotesize\monaspacekrypton,
  commentstyle=\itshape\color{gray},
  keywordstyle=\bfseries\color{magenta},
  tabsize=4,
  frame=single,
  columns=flexible,
  showstringspaces=false
}
```

```
const similar = "o008 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 <= n <= 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

4.3 Neon(Mono)

```
\lstset{
  language=python,
  basicstyle=\footnotesize\monaspaceneon,
  commentstyle=\itshape\color{gray},
  keywordstyle=\bfseries\color{magenta},
  tabsize=4,
  frame=single,
  columns=flexible,
  showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 <= n <= 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

4.4 Radon(Mono)

```
\lstset{
language=python,
basicstyle=\footnotesize\monaspaceradon,
commentstyle=\itshape\color{gray},
keywordstyle=\bfseries\color{magenta},
tabsize=4,
frame=single,
columns=flexible,
showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 <= n <= 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

4.5 Xenon(Mono)

```
\lstset{
  language=python,
  basicstyle=\footnotesize\monaspacexenon,
  commentstyle=\itshape\color{gray},
  keywordstyle=\bfseries\color{magenta},
  tabsize=4,
  frame=single,
  columns=flexible,
  showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 <= n <= 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

5 Samples (code)

5.1 Argo(Code)

```
\lstset{
  language=python,
  basicstyle=\footnotesize\monaspaceargoncode,
  commentstyle=\itshape\color{gray},
  keywordstyle=\bfseries\color{magenta},
  tabsize=4,
  frame=single,
  columns=flexible,
  showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 ≤ n ≤ 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

5.2 Krypton(Code)

```
\lstset{
  language=python,
  basicstyle=\footnotesize\monaspacekryptoncode,
  commentstyle=\itshape\color{gray},
  keywordstyle=\bfseries\color{magenta},
  tabsize=4,
  frame=single,
  columns=flexible,
  showstringspaces=false
}
```

```
const similar = "o008 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) ⇒ {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) ⇒ {
        return v ≠≠ alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 ≤ n ≤ 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

5.3 Neon(Code)

```
\lstset{
language=python,
basicstyle=\footnotesize\monaspaceneoncode,
commentstyle=\itshape\color{gray},
keywordstyle=\bfseries\color{magenta},
tabsize=4,
frame=single,
columns=flexible,
showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) ⇒ {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) ⇒ {
        return v ≠= alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
    # Check if input is 0 then it will print incorrect input
    if n < 0 :
        print("Incorrect input")
    elif n == 0 :
        return 0
    elif 1 ≤ n ≤ 2 :
        return 1
    else :
        return Fibonacci(n-1) + Fibonacci(n-2)
```

5.4 Radon(Code)

```
\lstset{
language=python,
basicstyle=\footnotesize\monaspaceradoncode,
commentstyle=\itshape\color{gray},
keywordstyle=\bfseries\color{magenta},
tabsize=4,
frame=single,
columns=flexible,
showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) => {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) => {
        return v !== alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 ≤ n ≤ 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```

5.5 Xenon(Code)

```
\lstset{
language=python,
basicstyle=\footnotesize\monaspacexenoncode,
commentstyle=\itshape\color{gray},
keywordstyle=\bfseries\color{magenta},
tabsize=4,
frame=single,
columns=flexible,
showstringspaces=false
}
```

```
const similar = "oO08 iIlL1 g9qCGQ"
const diacritics_etc = "â é ù ï ø ç Ã ĒÆ œ"

window.toggleFavorite = (alias) ⇒ {
  try {
    let favorites = JSON.parse(localStorage.getItem('favorites')) || []
    if (favorites.indexOf(alias) > -1) {
      favorites = favorites.filter((v) ⇒ {
        return v ≠= alias
      })
    } else {
      favorites.push(alias)
    }
    localStorage.setItem('favorites', JSON.stringify(Array.from(new Set(favorites))))
  } catch (err) {
    // eslint-disable-next-line no-console
    console.error('could not save favorite', err)
  }
  renderSelectList()
  return false
}
```

```
def Fibonacci(n: int) -> int :
   # Check if input is 0 then it will print incorrect input
   if n < 0 :
      print("Incorrect input")
   elif n == 0 :
      return 0
   elif 1 ≤ n ≤ 2 :
      return 1
   else :
      return Fibonacci(n-1) + Fibonacci(n-2)
```