Fira Code: free monospaced font with programming ligatures

Fira Code

Read in Español | 简体中文

Problem

Programmers use a lot of symbols, often encoded with several characters. For the human brain, sequences like `->`, `<=` or `:=` are single logical tokens, even if they take two or three characters on the screen. Your eye spends a non-zero amount of energy to scan, parse and join multiple characters into a single logical one. Ideally, all programming languages should be designed with full-fledged Unicode symbols for operators, but that's not the case yet.

Solution

Fira Code is a free monospaced font containing ligatures for common programming multi-character combinations. This is just a font rendering feature: underlying code remains ASCII-compatible. This helps to read and understand code faster. For some frequent sequences like `..` or `//`, ligatures allow us to correct spacing.

Download & Install

Fira_Code_v6.2.zip - December 6, 2021 - 2.5 MB

Then:

- How to Install
- Troubleshooting
- News & Updates

Sponsors

Fira Code is a personal, free-time project with no funding and a huge feature request backlog. If you love it, consider supporting its development via GitHub Sponsors or Patreon. Any help counts!

Huge thanks to:

WorkOS

Your app, enterprise-ready.

Start selling to enterprise customers with just a few lines of code. Add Single Sign-On (and more) in minutes instead of months.

What's in the box?

Left: ligatures as rendered in Fira Code. Right: same character sequences without ligatures.

Fira Code comes with a huge variety of arrows. Even better: you can make them as long as you like and combine start/middle/end fragments however you want!

Fira Code is not only about ligatures. Some fine-tuning is done for punctuation and frequent letter pairs.

Fira Code comes with a few different character variants, so that everyone can choose what's best for them. [How to enable]

Some ligatures can be altered or enabled using stylistic sets/character variants:

Being a programming font, Fira Code has fantastic support for ASCII/box drawing, powerline and other forms of console UIs:

Fira Code is the first programming font to offer dedicated glyphs to render progress bars:

In action:

We hope more programming fonts will adopt this convention and ship their own versions.

Unicode coverage makes Fira Code a great choice for mathematical writing:

How does it look?

Editor compatibility list

Works	Doesn't work
Abricotine	**Arduino IDE**

Android Studio (2.3+, instructions)	**Adobe Dreamweaver**
Anjuta (unless at the EOF)	**Delphi IDE**
AppCode (2016.2+, instructions)	Standalone **Emacs** (workaround)
Atom 1.1 or newer (instructions)	**Godot** (issue)
BBEdit/TextWrangler (v. 11 only, instructions)	**IDLE**
Brackets (with this plugin)	**KDevelop 4**
Chocolat	**Monkey Studio IDE**
CLion (2016.2+, instructions)	**UltraEdit**
Cloud9 (instructions)	
Coda 2	
CodeLite	
CodeRunner	
CotEditor	
Eclipse	
elementary Code	
Geany (1.37+)	
gEdit / Pluma	
GNOME Builder	
GoormIDE (instructions)	
gVim (Windows, GTK)	
IntelliJ IDEA (2016.2+, instructions)	
Kate, KWrite	
KDevelop 5+	
Komodo	
Leafpad	
LibreOffice	
LightTable (instructions)	
LINQPad	
MacVim 7.4 or newer (instructions)	
Mancy	
MATLAB (instructions)	

Meld	
Mousepad	
NeoVim-gtk	
NetBeans	
Notepad (Windows)	
Notepad++ (with a workaround)	
Notepad3 (instructions)	
Nova	
PhpStorm (2016.2+, instructions)	
PyCharm (2016.2+, instructions)	
QOwnNotes (21.16.6+)	
QtCreator	
Rider	
RStudio (instructions)	
RubyMine (2016.2+, instructions)	
Scratch	
Scribus (1.5.3+)	
SublimeText (3146+)	
Spyder IDE (only with Qt5)	
SuperCollider 3	
TeXShop	
TextAdept (Linux, macOS)	
TextEdit	
TextMate 2	
VimR (instructions)	
Visual Studio (2015+, instructions)	
Visual Studio Code (instructions)	
WebStorm (2016.2+, instructions)	
Xamarin Studio/Monodevelop	
Xcode (8.0+, otherwise with plugin)	
Xi	

Probably work: **Smultron, Vico**	Under question: **Code::Blocks IDE**

Terminal compatibility list

Works	Doesn’t work
crosh (instructions)	Alacritty
Hyper (see #3607)	Cmder
iTerm 2	ConEmu
Kitty	GNOME Terminal
Konsole	gtkterm (ticket)
Mintty	guake (ticket)
QTerminal	LXTerminal (ticket)
st (patch)	mate-terminal
Terminal.app	PuTTY
Termux	rxvt
Token2Shell	sakura (ticket)
Wez’s terminal	Terminator (ticket)
Windows Terminal	terminology
ZOC (macOS)	Windows Console
	xfce4-terminal (ticket)
	xterm
	ZOC (Windows)

Browser support

```html
<!-- HTML -->
<link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/firacode@6.2.0/distr/fira_code.css">
```

```css
/* CSS */
@import url(https://cdn.jsdelivr.net/npm/firacode@6.2.0/distr/fira_code.css);
```

```css
/* Specify in CSS */
code { font-family: 'Fira Code', monospace; }

@supports (font-variation-settings: normal) {
  code { font-family: 'Fira Code VF', monospace; }
}
```

- IE 10+, Edge Legacy: enable with `font-feature-settings: "calt";`
- Firefox
- Safari
- Chromium-based browsers (Chrome, Opera)
- ACE
- CodeMirror (enable with `font-variant-ligatures: contextual;`)

Projects using Fira Code

- CodePen
- Blink Shell
- Klipse
- IlyaBirman.net
- EvilMartians.com
- Web Maker
- FromScratch
- PEP20.org

Alternatives

Free monospaced fonts with ligatures:

- Hasklig
- Monoid
- Fixedsys Excelsior
- Iosevka
- DejaVu Sans Code
- Victor Mono
- Cascadia Code
- JetBrains Mono

Paid monospaced fonts with ligatures:

- PragmataPro
- Mono Lisa

Building Fira Code locally

In case you want to alter FiraCode.glyphs and build OTF/TTF/WOFF files yourself, this is the setup I use on macOS:

```
# install all required build tools
./script/bootstrap_macos.sh

# build the font files
./script/build.sh

# install OTFs to ~/Library/Fonts
cp distr/otf/*.otf ~/Library/Fonts
```

Alternatively, you can build Fira Code using Docker:

```
# install dependencies in a container and build the font files
make

# package the font files from dist/ into a zip
make package
```

If you want to *permanently enable* certain style sets or character variations, maybe because your editor of choice does not allow you to toggle these individually, you can provide the desired features as a comma separated list to the build script via the `-f / --features` flag.
Default: none.

To separate different versions of your font you can specify the desired font family name with the `-n / --family-name` flag. The special value 'features' will append a sorted, space separated list of enabled features to the default family name.
Default: "Fira Code"

You can also limit the font weights that will be created with the `-w / --weights` option.
Default: "Light,Regular,Retina,Medium,SemiBold,Bold"

```
# locally in your shell
./script/build.sh --features "ss02,ss08,ss10,cv03,cv07,cv14" --family-name "Fira
Code straight" --weights "Regular,Bold"

# or via a docker container (creates the family name 'Fira Code cv01 cv02 cv06 cv31
onum ss01 ss03 ss04 zero')
docker run --rm -v "${PWD}":/opt tonsky/firacode:latest ./script/build.sh -f
"cv01,cv02,cv06,ss01,zero,onum,ss03,ss04,cv31" -n "features"
```

Credits

- Author: Nikita Prokopov @nikitonsky
- Based on: Fira Mono
- Inspired by: Hasklig