SPŠE Ječná

IT

Ječná 30, Praha 2

Text Editor

Filip Heger
IT
2024

1. Goal

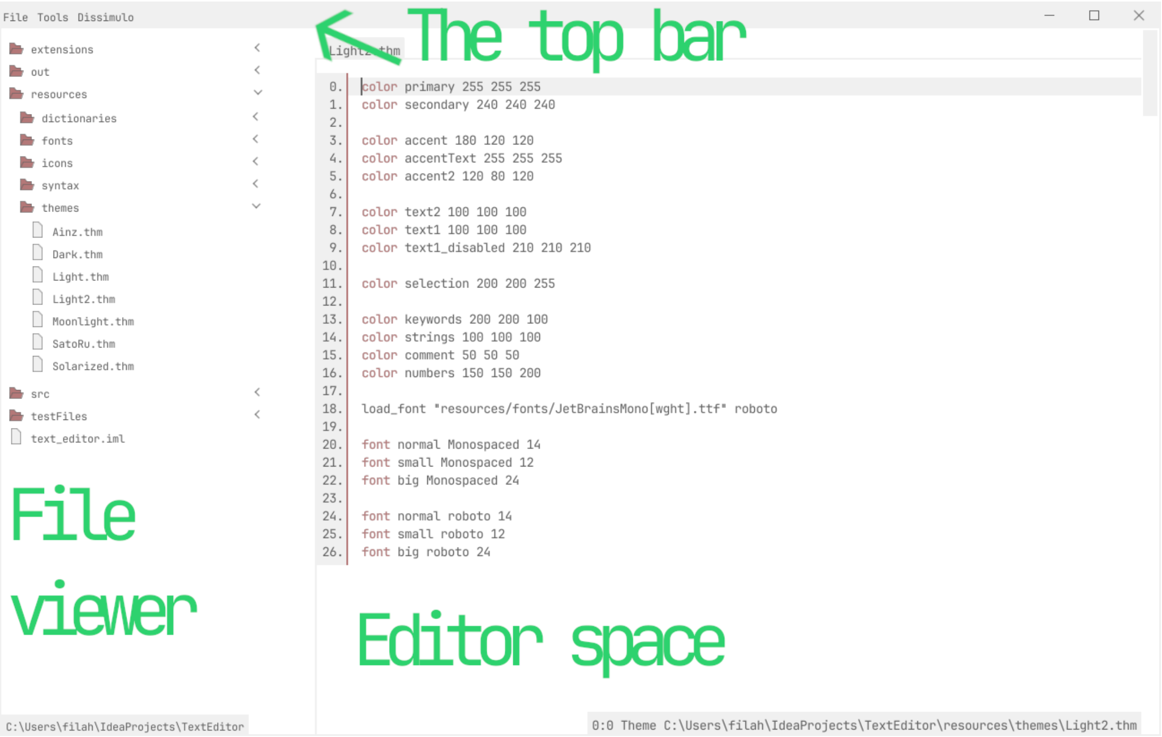
The goal of the project is to create a text editor akin to Visual Studio Code, it is an attempt to create a similar UI and an experience that doesn't feel like typing code in Word. There should also be a way to create extensions, and as already mentioned the UI should be at least bearable to look at. There should also be themes, because colors are good.

2. Software

Java 22 (openjdk-22), Intellij, there are external libraries outside of the ones written by me included in the project.

3. Description

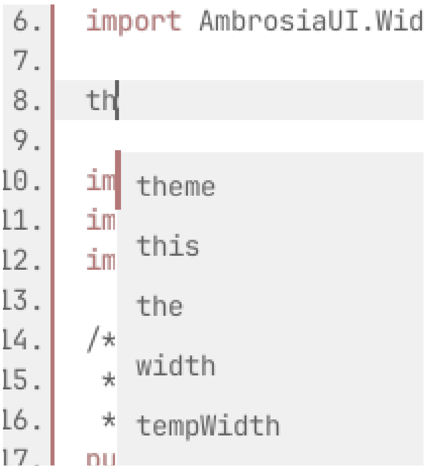
The text editor consists of a few main parts, the file view on the side, the dropdown menus in the top bar and the space reserved for open editors, there can be more editors open at one time. There are a total of three distinct editors. The whole text editor including the icons have colors defined in a theme configuration file.



```
color primary 255 255 255
color secondary 240 240 240

color accent 180 120 120
color accentText 255 255 255
color accent2 120 80 120

color text2 100 100 100
color text1 100 100 100
color text1_disabled 210 210 210

color selection 200 200 255

color keywords 200 200 100
color strings 100 100 100
color comment 50 50 50
color numbers 150 150 200

load_font "resources/fonts/JetBrainsMono[wght].ttf" roboto

font normal Monospaced 14
font small Monospaced 12
font big Monospaced 24

font normal roboto 14
font small roboto 12
font big roboto 24
```

3.1. Text editor

This part is the driving horse of the entire project. It's a text editor with most of the essential working features. It also supports syntax highlighting based on configuration files that can be changed. New language support can be added easily. The text editor also provides a hinting system (though quite sketchy).



```
import AmbrosiaUI.Wid

th

im
im
im

/*
 *
 *
nu
```

3.2. Icon editor

Is a very simple editor for editing a custom icon format designed to be compatible with themes. The icons are similar to SVGs. stored in readable text format.

3.3. Hex editor

Formally called an editor even though it really is not one. It only serves for viewing the contents of any file in hexadecimal form

3.4. File viewer

If a file viewer is almost identical functionally to the one Visual Studio code has. It displays files and folders, there are even some predefined icons for certain file types, though never fully implemented.

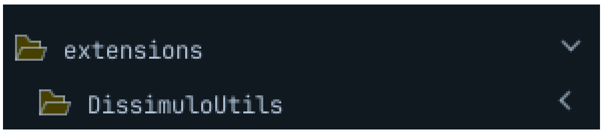
3.5. Selection prompts & others

All the prompts along with the editor are custom made, the most significant one used for selecting files in the filesystem is based around the same system the fileviewer is, it has a few variations for selecting for example folders, or creating files.

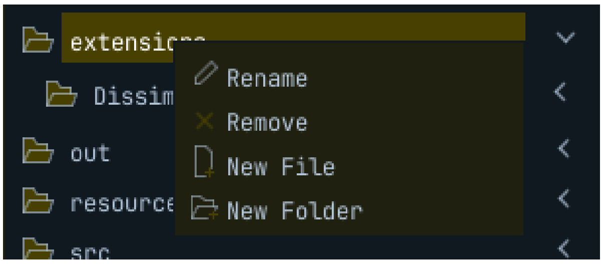
4. Manual

An essential key bind is `Ctrl + X` because it reloads all configuration files and refreshes the editor. Most controls are identical to ones in other software so I won't mention them all. But there are some features that are not apparent immediately, for example if you right click a tab in the active editors it will close.

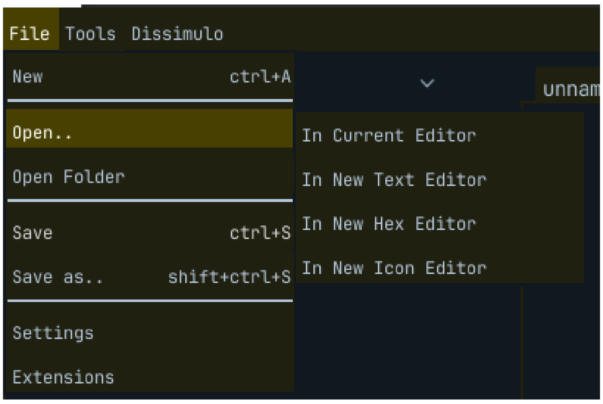
In the file viewer on the left side of the editor you can expand folders with the little arrows on their right.

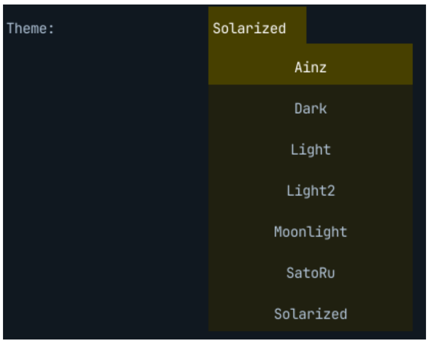


You can also open a context menu that shows you options of what you can do with files (though context menus are not adapted very widely anywhere else)



You can open files in different editors using the open option in a dropdown menu in the upper bar. You can also find the settings there. That only have a theme selection.




Theme: Solarized

5. Conclusion

To assess how I had done making the editor. I had tried to use the inbuilt library for making GUIs, which I hated. So I chose to build on top of its most fundamental features to make a library that looks better and is easier to work with, which I think I achieved at least partially, though I wouldn't recommend anyone to use it because of some of its questionable design choices I made when making it. I also made a custom format for icons, which looks good when paired with the editors themes but was also a nightmare to get going. All of this UI put together is drawn from almost nothing, based around some basic functions provided by Graphics from the native library. To actually conclude this, making a text editor? Yeah, that was fun. Making an entire GUI system with all the widgets and custom icons? Yeah, not so much. But I did learn a lot.