

Visual Studio Code

Tips and Tricks

Foreword



Contribute on  GitHub



View Online neave.dev/vscode



Download [PowerPoint](#) and [PDF](#)

What is VS Code?

“ Visual Studio Code is a free code editor that is optimized for building and debugging modern web and cloud applications.”

You can run it on Windows, Linux, Mac or Online
<https://vscode.dev>

Key Features:

- It supports a range of languages,
- Productivity shortcut circuits
 - Keyboard shortcuts
 - Tasks
 - Snippets
- Customizable with
 - Themes
 - Extensions
- Integrated terminal.

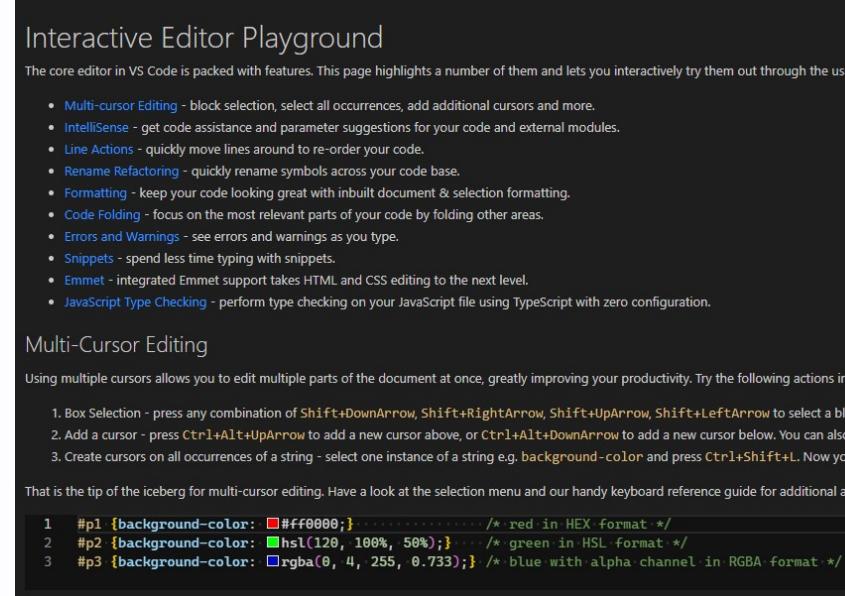
Documentation and Updates

- Official Documentation
 - Intro Videos
 - Tips and Tricks
 - Updates
- VSCode YouTube Channel

Inbuilt Features

Help Menu

- Keyboard shortcut reference
- Walkthroughs
- Interactive Editor Playground



The screenshot shows a dark-themed browser window titled "Interactive Editor Playground". The page content is as follows:

Interactive Editor Playground

The core editor in VS Code is packed with features. This page highlights a number of them and lets you interactively try them out through the use of code snippets.

- Multi-cursor Editing - block selection, select all occurrences, add additional cursors and more.
- IntelliSense - get code assistance and parameter suggestions for your code and external modules.
- Line Actions - quickly move lines around to re-order your code.
- Rename Refactoring - quickly rename symbols across your code base.
- Formatting - keep your code looking great with inbuilt document & selection formatting.
- Code Folding - focus on the most relevant parts of your code by folding other areas.
- Errors and Warnings - see errors and warnings as you type.
- Snippets - spend less time typing with snippets.
- Emmet - integrated Emmet support takes HTML and CSS editing to the next level.
- JavaScript Type Checking - perform type checking on your JavaScript file using TypeScript with zero configuration.

Multi-Cursor Editing

Using multiple cursors allows you to edit multiple parts of the document at once, greatly improving your productivity. Try the following actions in the playground:

1. Box Selection - press any combination of `Shift+DownArrow`, `Shift+RightArrow`, `Shift+UpArrow`, `Shift+LeftArrow` to select a block of text.
2. Add a cursor - press `Ctrl+Alt+UpArrow` to add a new cursor above, or `Ctrl+Alt+DownArrow` to add a new cursor below. You can also use `Ctrl+Shift+UpArrow` and `Ctrl+Shift+DownArrow`.
3. Create cursors on all occurrences of a string - select one instance of a string e.g. `background-color` and press `Ctrl+Shift+L`. Now you can edit all occurrences at once.

That is the tip of the iceberg for multi-cursor editing. Have a look at the selection menu and our handy keyboard reference guide for additional actions.

```
1 #p1 {background-color: #ff0000;} /* red in HEX format */
2 #p2 {background-color: hsl(120, 100%, 50%); /* green in HSL format */
3 #p3 {background-color: rgba(0, 4, 255, 0.733); /* blue with alpha channel in RGBA format */
```



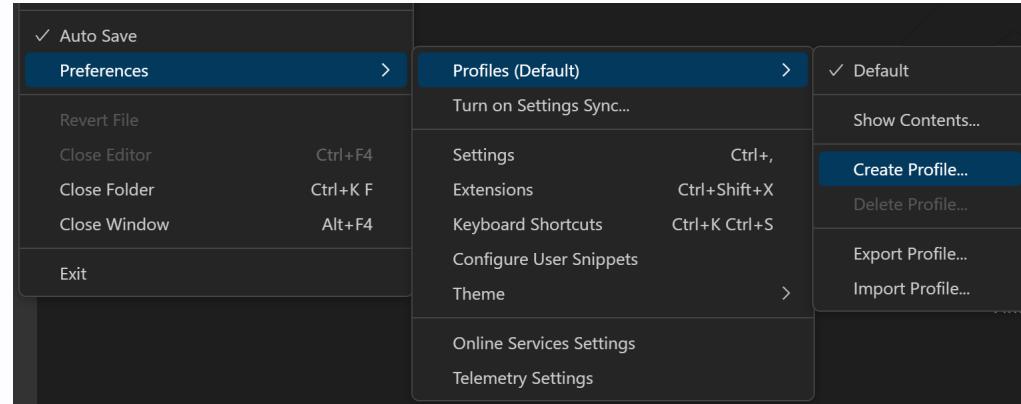
Settings

Inbuilt Features

Profiles

VS Code has a Default Profile. You can have additional **profiles**.

Customize Settings, Keyboard Shortcuts, User Snippets, User Tasks or Extensions



- Have a profile for Frontend, Backend, AWS and Azure or even for a project.
- Customize the profile for a stack ie (Jira, AWS and Python).
- You could install the tools and extensions as part of that project.

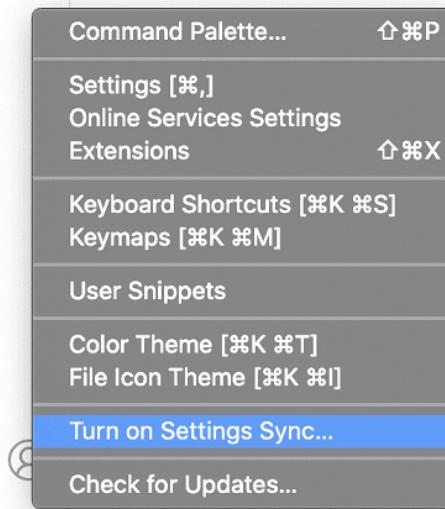
Customisation

- Default settings or open Preferences: Open Default Settings (JSON)
- Customize it with User settings - Ctrl+P and go to Open User Settings (JSON)

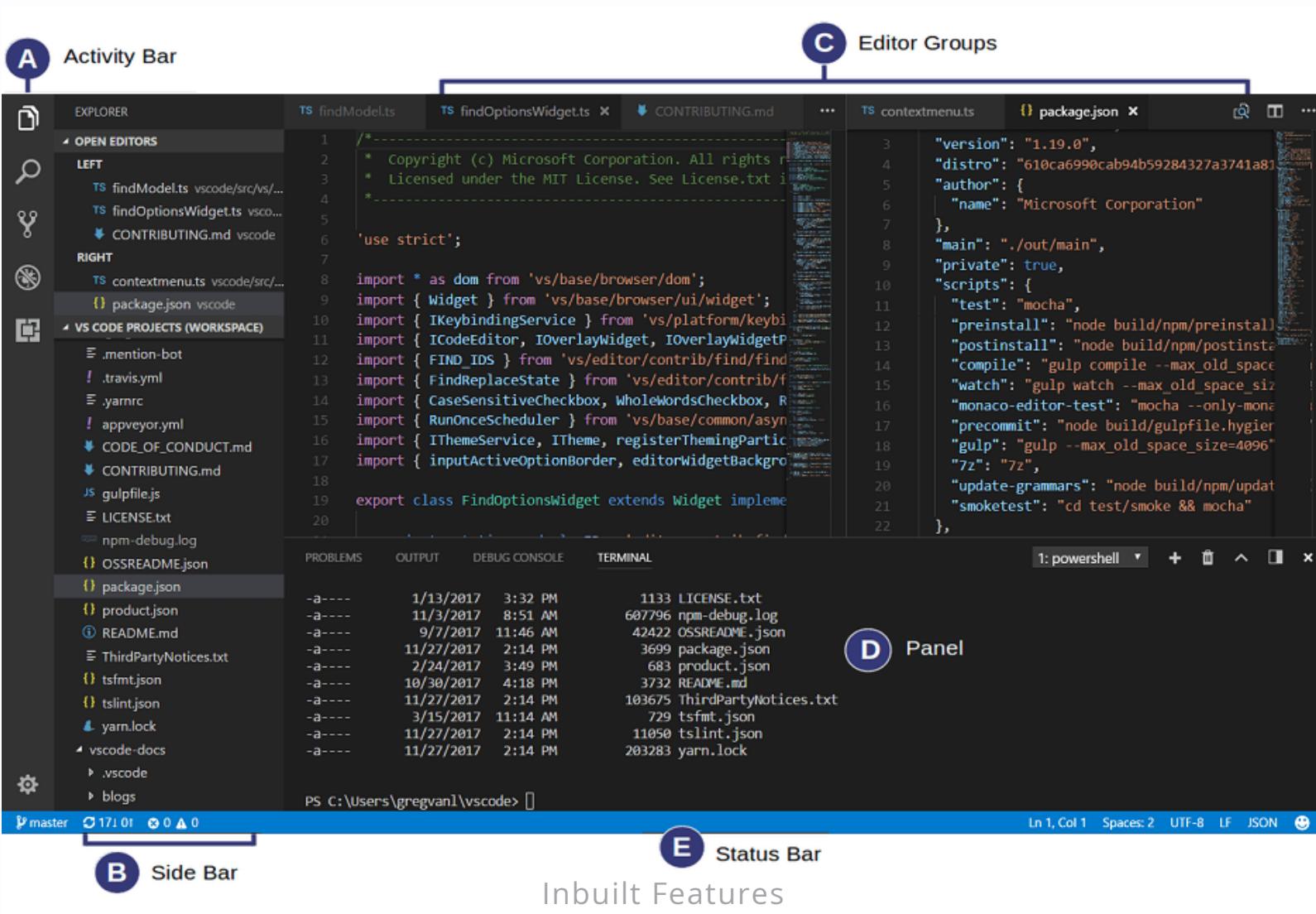
```
{  
  "editor.fontFamily": "'Cascadia Code'",  
  "editor.formatOnSave": true,  
  "editor.rulers": [ 120 ],  
  "editor.stickyScroll.enabled": true,  
  "editor.trimAutoWhitespace": true,  
  "explorer.fileNesting.enabled": true,  
  "files.trimTrailingWhitespace": true,  
  "git.branchPrefix": "yourusername/",  
  "terminal.integrated.fontFamily": "'CaskaydiaCove Nerd Font Mono', 'Cascadia Code', Consolas, 'Courier New', monospace",  
  "workbench.sideBar.location": "right"  
}
```

Synchronized Settings

Settings sync cross
multiple instances of
VS Code



Layout



Customize the layout

- You could move the sidebar to the right then when you `Toggle Sidebar/Ctrl+B` it doesn't make the code jump around.
- Add a secondary sidebar with Outline view for better navigation.

Grouping

Ctrl+\

Inbuilt Features

The screenshot shows a code editor window in VS Code with the file 'ADME.md' open. The file contains JSON configuration for a sidebar and various keyboard shortcuts. A specific section of the code is highlighted with a light gray background, demonstrating the 'Grouping' feature. The highlighted section starts at line 91 and ends at line 99, containing the following code:

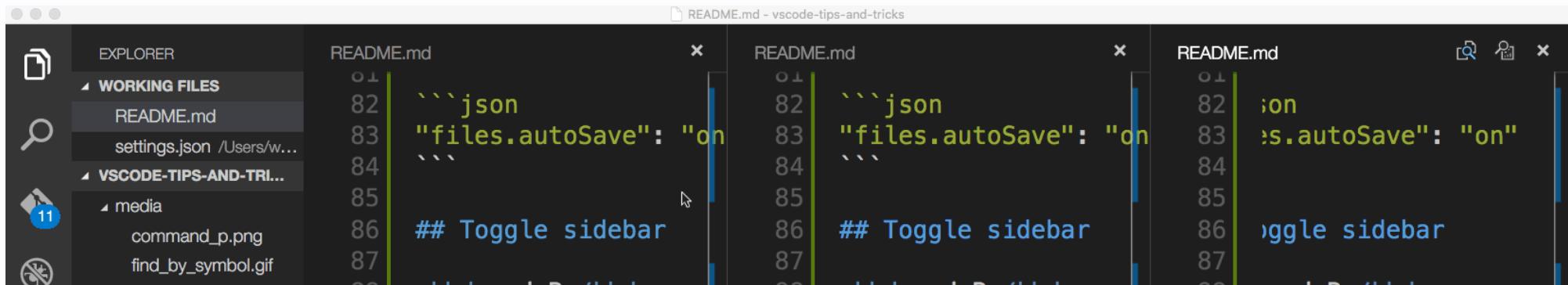
```
## Side by side editing
```

The code editor interface includes a status bar at the top with 'elp' and 'Finish 05:00' and a bottom status bar showing 'Spaces: 4 Ln'.

```
ADME.md
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
00
01
02
03
04
05
06
07
08
```

```
```json
"files.autoSave": "on"
```
## Toggle sidebar
<kbd>cmd+B</kbd>
![toggle side bar](/media/toggle_side_bar.gif)
## Side by side editing
<kbd>cmd+`</kbd> or <kbd>cmd</kbd> then click a file from
## Switch between editors
<kbd>cmd+1</kbd>, <kbd>cmd+2</kbd>, <kbd>cmd+3</kbd>
## History
Navigate entire history with <kbd>ctrl+tab</kbd>
Navigate back with <kbd>ctrl+-</kdbd>.
Navigate forward with <kbd>ctrl+shift+up</kbd>
! [navigate history](/media/navigate_history.gif)
## Navigate to a file
```

Switch to next group with `Ctrl+[1, 2, 3...]`



Alternatively: `Ctrl+P` and View: Move Editor to Next Group

Navigation

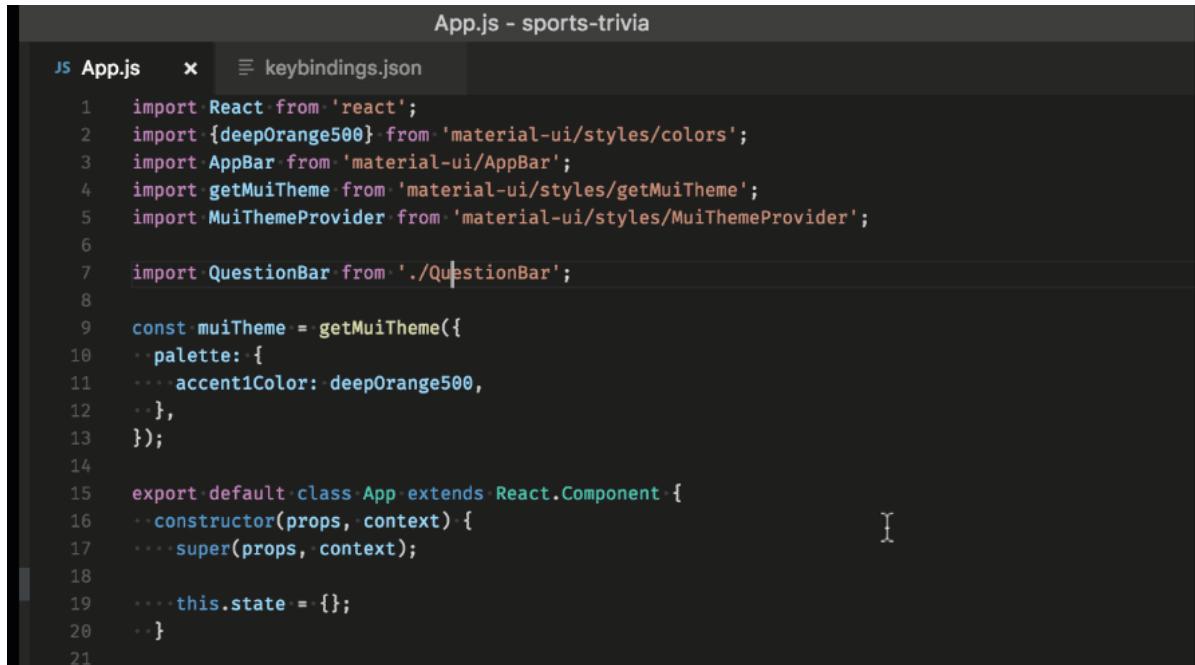
ctrl+R Displays a Quick Pick dropdown with the list from `File > Open Recent` with recently opened folders and workspaces followed by files.

Multi-Root Workspaces

Normally your workspace has a single root/project folder but you can have a workspace with multiple project folders in Visual Studio Code with [multi-root workspaces](#). Uses a `.code-workspace` file

```
{  
  "folders": [  
    {  
      // Source code  
      "name": "Product",  
      "path": "vscode"  
    },  
    {  
      // Docs and release notes  
      "name": "Documentation",  
      "path": "vscode-docs"  
    },  
    {  
      // Yeoman extension generator  
      "name": "Extension generator",  
      "path": "vscode-generator-code"  
    }  
  ]  
}
```

Command Palette



The screenshot shows the VS Code interface with the title bar "App.js - sports-trivia". Below the title bar, there's a tab bar with "App.js" selected, followed by "keybindings.json". The main editor area displays the following code:

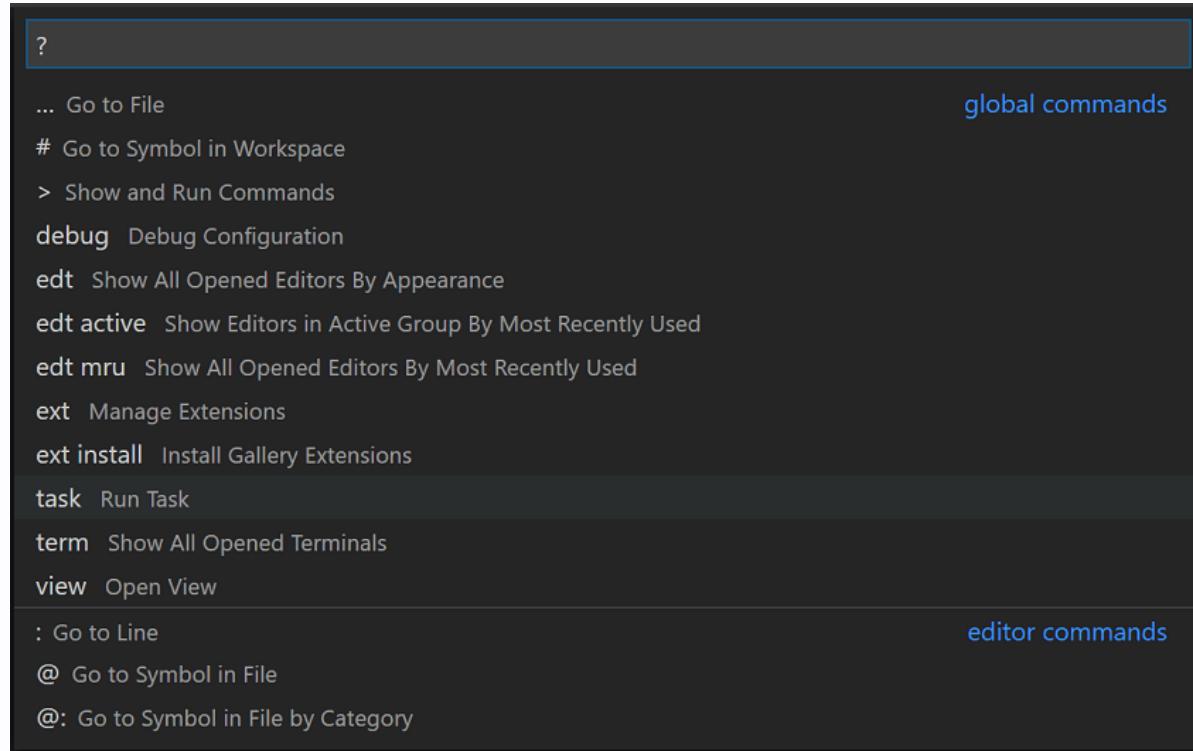
```
1 import React from 'react';
2 import {deepOrange500} from 'material-ui/styles/colors';
3 import AppBar from 'material-ui/AppBar';
4 import getMuiTheme from 'material-ui/styles/getMuiTheme';
5 import MuiThemeProvider from 'material-ui/styles/MuiThemeProvider';
6
7 import QuestionBar from './QuestionBar';
8
9 const muiTheme = getMuiTheme({
10   palette: {
11     accent1Color: deepOrange500,
12   },
13 });
14
15 export default class App extends React.Component {
16   constructor(props, context) {
17     super(props, context);
18
19     this.state = {};
20   }
21 }
```

Ctrl+Shift+P

Quick Open File



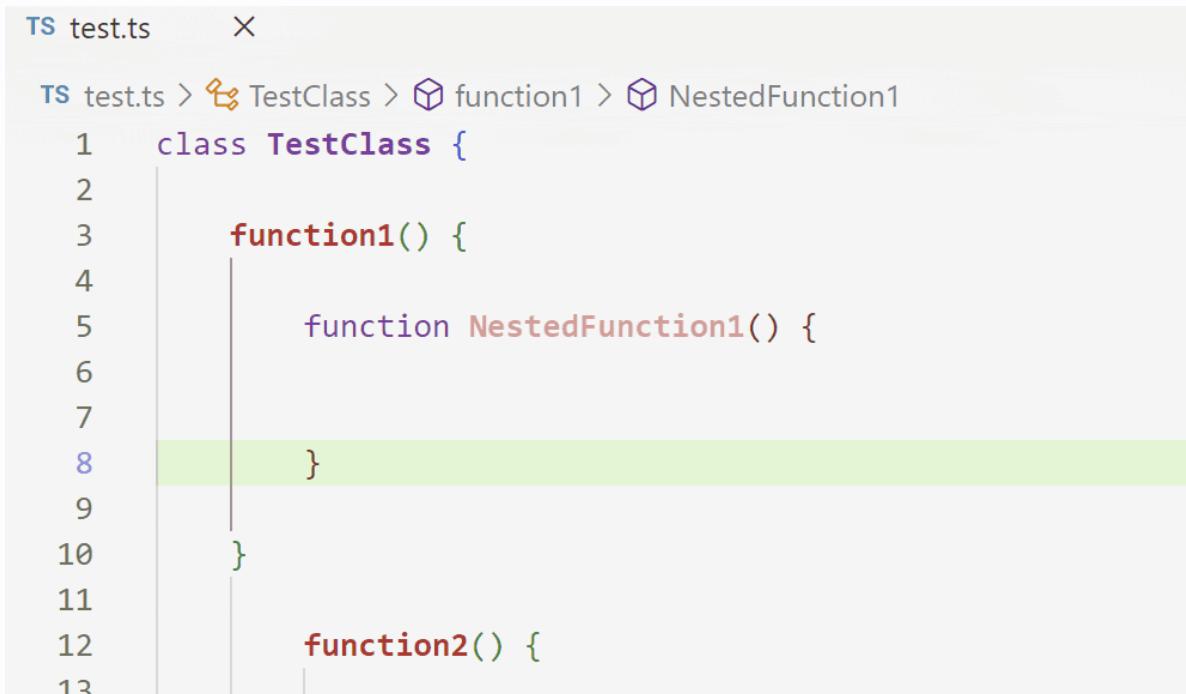
Ctrl+P



Ctrl+P and type ?term to open a new terminal

Sticky Scroll

See the code context with Sticky Scroll - useful for long methods



The screenshot shows a code editor window for a file named 'test.ts'. The file contains the following TypeScript code:

```
TS test.ts      ×  
TS test.ts > 📁 TestClass > 🥑 function1 > 🥑 NestedFunction1  
1  class TestClass {  
2  
3    function1() {  
4  
5      function NestedFunction1() {  
6  
7  
8      }  
9  
10     }  
11  
12     function2() {  
13   }
```

The code is displayed with syntax highlighting. A green horizontal bar highlights the closing brace of the innermost function definition at line 8. The code editor interface includes a status bar at the bottom.

Others

Ctrl+T - Go To Symbol

Ctrl+G - Go to Line

Alt+ ← / → - Jump forward and back

Editing

Inbuilt Features

Text Manipulation

- Sort Lines Ascending/Descending
- Join Lines
- Transform Text

Transform Example

Title Hello World

Upper HELLO WORLD

Lower hello world

Camel helloWorld

Snake helloWorld → hello_world

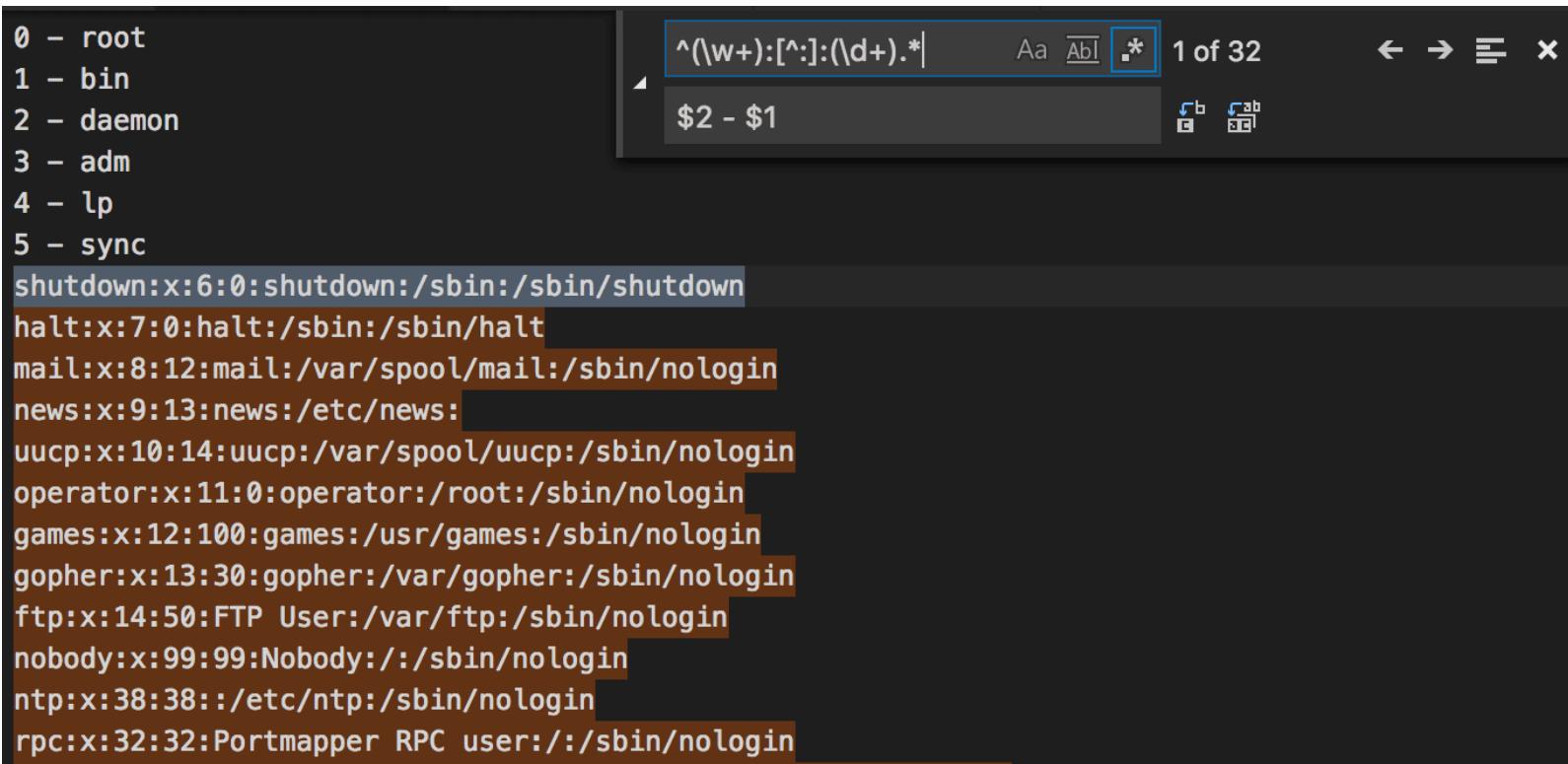
Kebab helloWorld → hello-world

Code Formatting

```
render() {
  return (
    <div className={s.app}>
      <AppBar
        title="Notes"
        iconClassNameRight="muidocs-icon-navigation-expand-right"
      />
      <div style={{
        marginTop: 20,
        marginLeft: 20
      }}>
        <NewNote />
        <Notes items={this.state.notes}/>
      </div>
    </div>
  );
}
```

Inbuilt Features

Regex Search and Replace



A screenshot of a terminal window showing a search and replace operation. The search term is `^(\w+):[^\:](\d+).*`. The replace term is `$2 - $1`. The terminal shows a list of users and their corresponding IDs and names.

| ID | User | Description |
|----|----------|---|
| 0 | root | |
| 1 | bin | |
| 2 | daemon | |
| 3 | adm | |
| 4 | lp | |
| 5 | sync | |
| 6 | shutdown | x:6:0:shutdown:/sbin:/sbin/shutdown |
| 7 | halt | x:7:0:halt:/sbin:/sbin/halt |
| 8 | mail | x:8:12:mail:/var/spool/mail:/sbin/nologin |
| 9 | news | x:9:13:news:/etc/news: |
| 10 | uucp | x:10:14:uucp:/var/spool/uucp:/sbin/nologin |
| 11 | operator | x:11:0:operator:/root:/sbin/nologin |
| 12 | games | x:12:100:games:/usr/games:/sbin/nologin |
| 13 | gopher | x:13:30:gopher:/var/gopher:/sbin/nologin |
| 14 | ftp | x:14:50:FTP User:/var/ftp:/sbin/nologin |
| 99 | nobody | x:99:99:Nobody:/:/sbin/nologin |
| 38 | ntp | x:38:38::/etc/ntp:/sbin/nologin |
| 32 | rpc | x:32:32:Portmapper RPC user:/:/sbin/nologin |

Multi Cursor

Edit Text Vertically with [multi-cursor](#)

Keep selecting text with `Ctrl+D` and update over multiple locations in a file.

```
31 .global-message-list.transition {  
32 →   -webkit-transition: top 200ms linear;  
33 →   -ms-transition:      top 200ms linear;  
34 →   -moz-transition:    top 200ms linear;  
35 →   -khtml-transition:  top 200ms linear;  
36 →   -o-transition:     top 200ms linear;  
37 →   transition:        top 200ms linear;  
38 }
```

Expanding and Shrinking Selection

Shift+Alt+Left and Shift+Alt+Right

Great for finding the start and end of a long <div>

```
7 namespace Hello1
8 {
9     0 references
10    class Program
11    {
12        0 references
13        static void Main(string[] args)
14        {
15            System.Console.WriteLine("Hello World!");
16        }
17    }
18 }
```

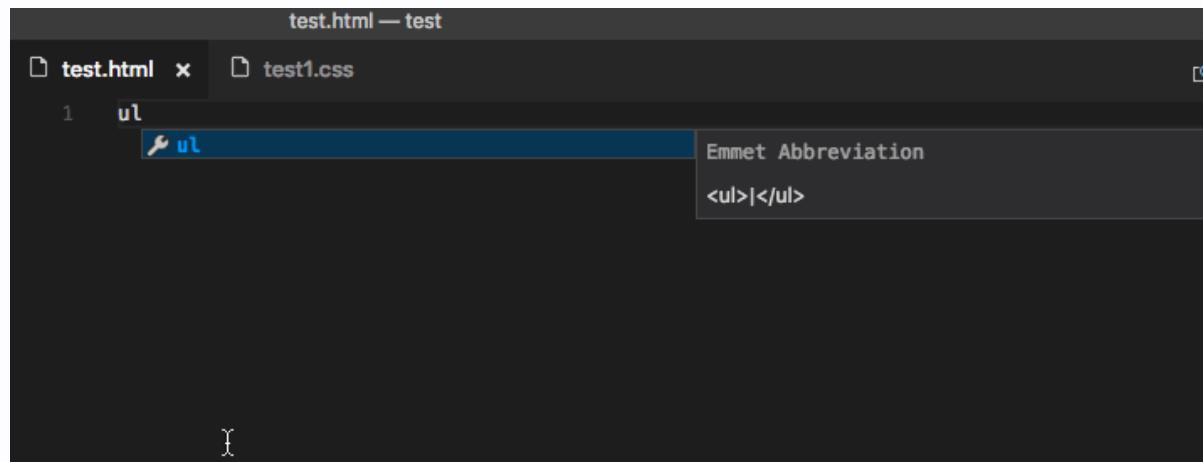
Folding

Hide Text for easier
reading

Fold Level [1 | 2 | 3 | 4],
Fold All, Unfold All

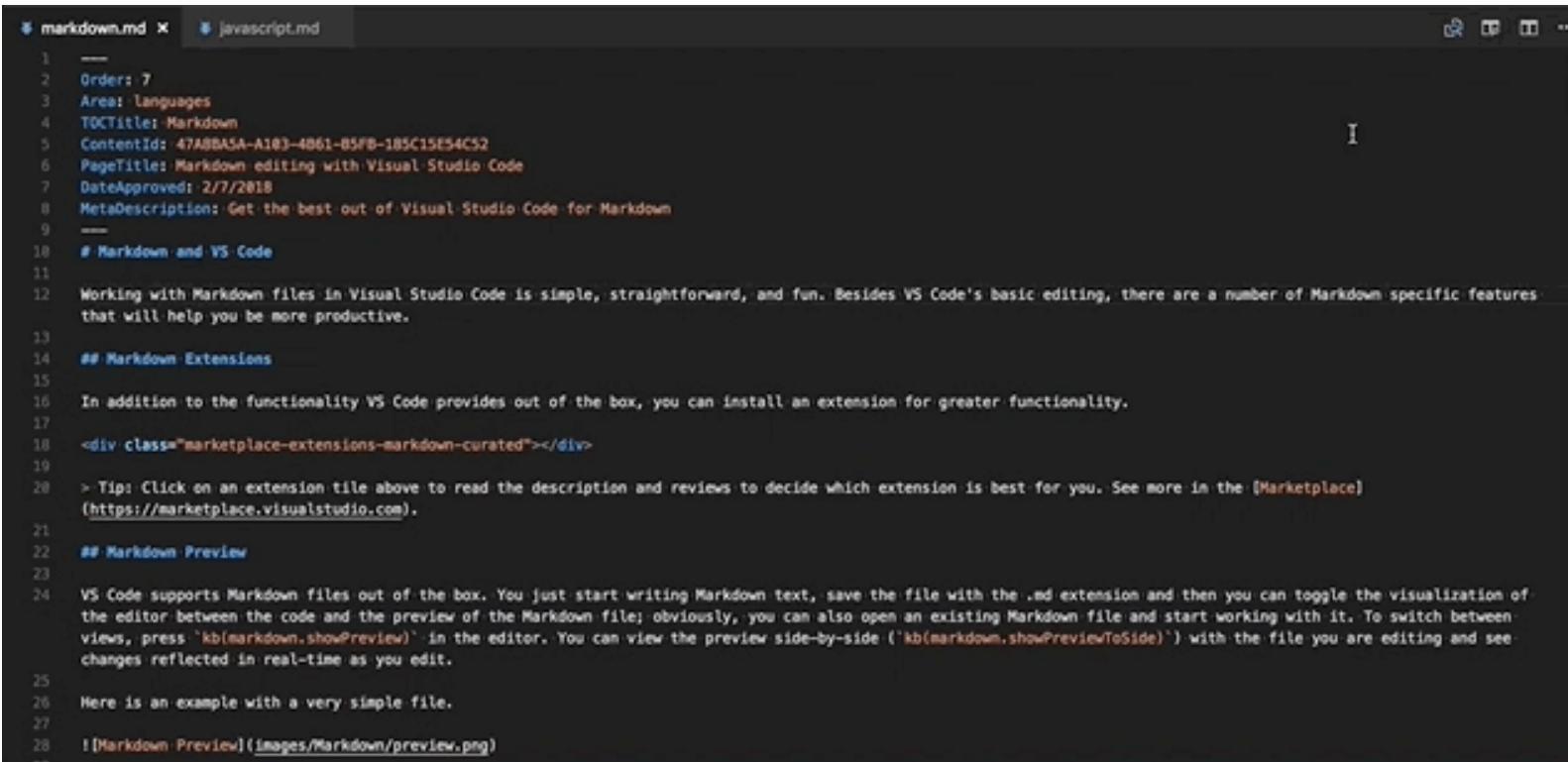
```
class App extends Component {  
  
  constructor() {  
    super();  
    this.state = {  
      notes: [  
        {  
          text: "go to the grocery store"  
        },  
        {  
          text: 'read medium article about engineering'  
        },  
        {  
          text: 'create build session'  
        },  
        {  
          text: 'fix bug #232'  
        }  
      ];  
    };  
  }  
}
```

Emmet



Generate HTML and
CSS from shorthand.
Works with multi-
cursors

Markdown preview



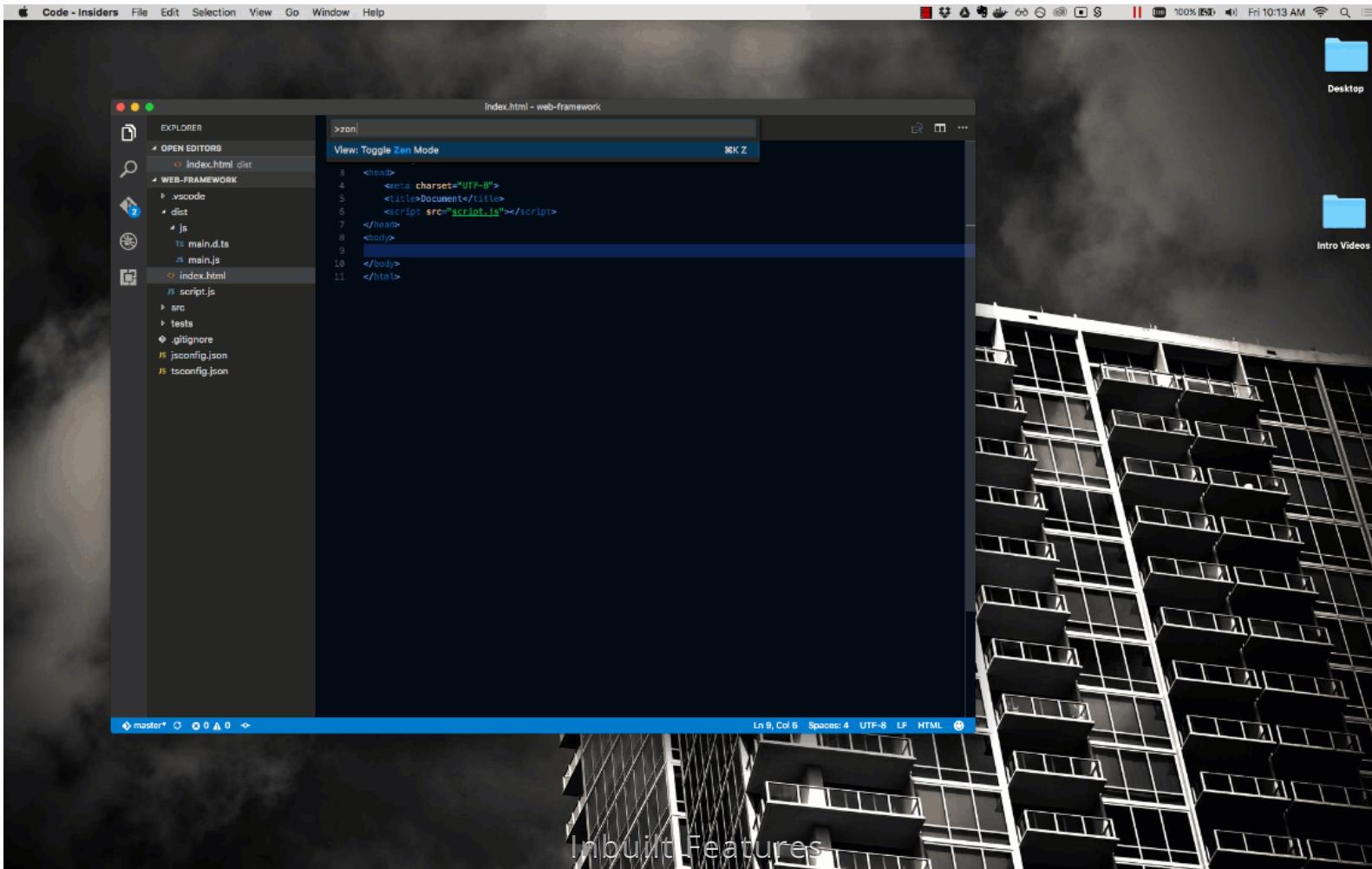
The screenshot shows a dark-themed instance of Visual Studio Code. In the top left, there are two tabs: one for 'markdown.md' and another for 'javascript.md'. The main editor area displays a portion of a Markdown file. The visible text includes:

```
1 —
2 Order: 7
3 Area: Languages
4 TOCTitle: Markdown
5 ContentId: 47A8B85A-A183-4861-85FB-185C15E54C52
6 PageTitle: Markdown editing with Visual Studio Code
7 DateApproved: 2/7/2018
8 MetaDescription: Get the best out of Visual Studio Code for Markdown
9 —
10 # Markdown and VS Code
11
12 Working with Markdown files in Visual Studio Code is simple, straightforward, and fun. Besides VS Code's basic editing, there are a number of Markdown specific features that will help you be more productive.
13
14 ## Markdown Extensions
15
16 In addition to the functionality VS Code provides out of the box, you can install an extension for greater functionality.
17
18 <div class="marketplace-extensions-markdown-curated"></div>
19
20 > Tip: Click on an extension tile above to read the description and reviews to decide which extension is best for you. See more in the [Marketplace] (https://marketplace.visualstudio.com).
21
22 ## Markdown Preview
23
24 VS Code supports Markdown files out of the box. You just start writing Markdown text, save the file with the .md extension and then you can toggle the visualization of the editor between the code and the preview of the Markdown file; obviously, you can also open an existing Markdown file and start working with it. To switch between views, press 'kb(markdown.showPreview)' in the editor. You can view the preview side-by-side ('kb(markdown.showPreviewToSide)') with the file you are editing and see changes reflected in real-time as you edit.
25
26 Here is an example with a very simple file.
27
28 ! [Markdown Preview] (images/Markdown/preview.png)
```

Productivity / Focus

Zen Mode

Focus on the code with Zen Mode



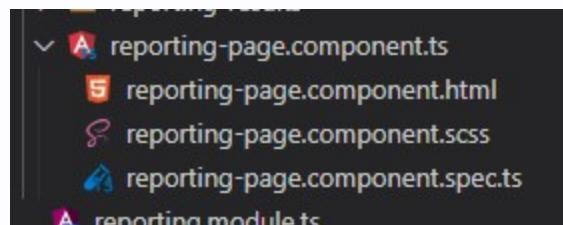
Inbuilt Features

Errors

Jump to the **next error** with **F8**

File Nesting

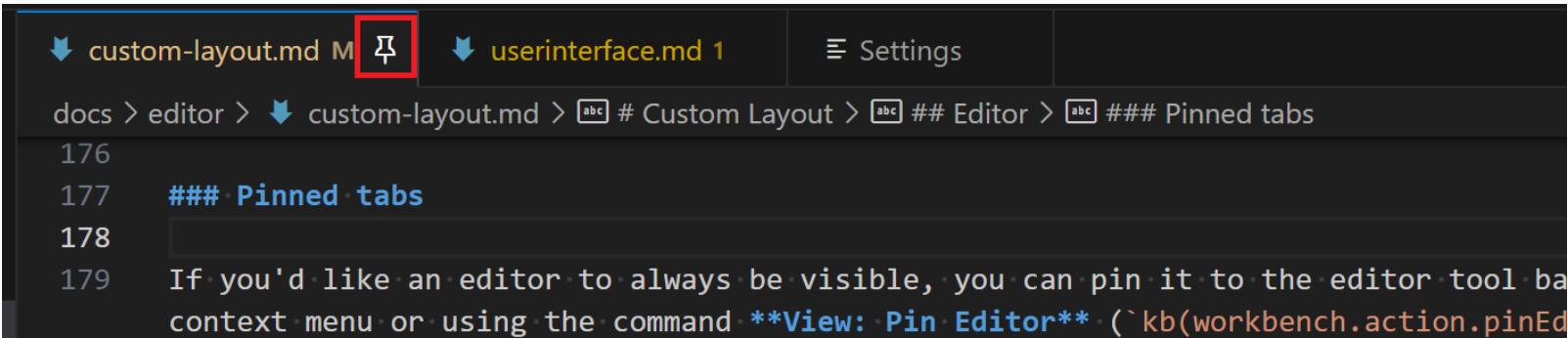
Nest related files under a parent file in the explorer.



```
"explorer.fileNesting.patterns": {  
    "*.component.ts": "$(capture).component.html, $(capture).component.spec.ts, $(capture).component.scss",  
    "README*": "AUTHORS,CHANGELOG*,CODE_OF_CONDUCT*,CONTRIBUTING*,LICENSE*"  
}
```

Pinned Tabs

Pinned Tabs allow you to keep a file open

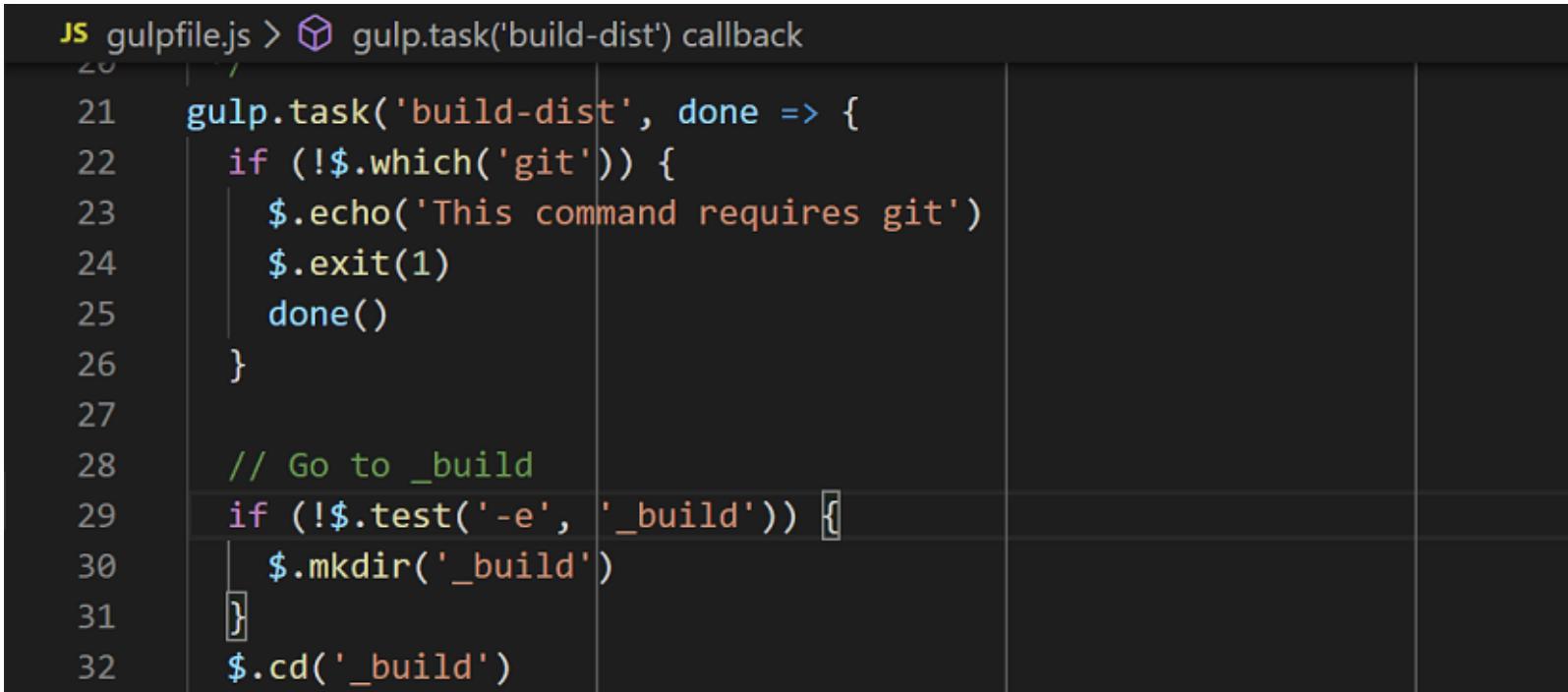


The screenshot shows the VS Code interface with two tabs open: "custom-layout.md" and "userinterface.md 1". The "custom-layout.md" tab has a red box around its pinning icon. The status bar at the bottom displays the navigation path: "docs > editor > custom-layout.md > # Custom Layout > ## Editor > ### Pinned tabs". The code editor shows a section titled "### Pinned tabs" with the following text:

```
176
177 ### Pinned tabs
178
179 If you'd like an editor to always be visible, you can pin it to the editor tool bar context menu or using the command View: Pin Editor (^kb(workbench.action.pinEd
```

Vertical Rulers

Make sure your code isn't getting too long and hard to read.



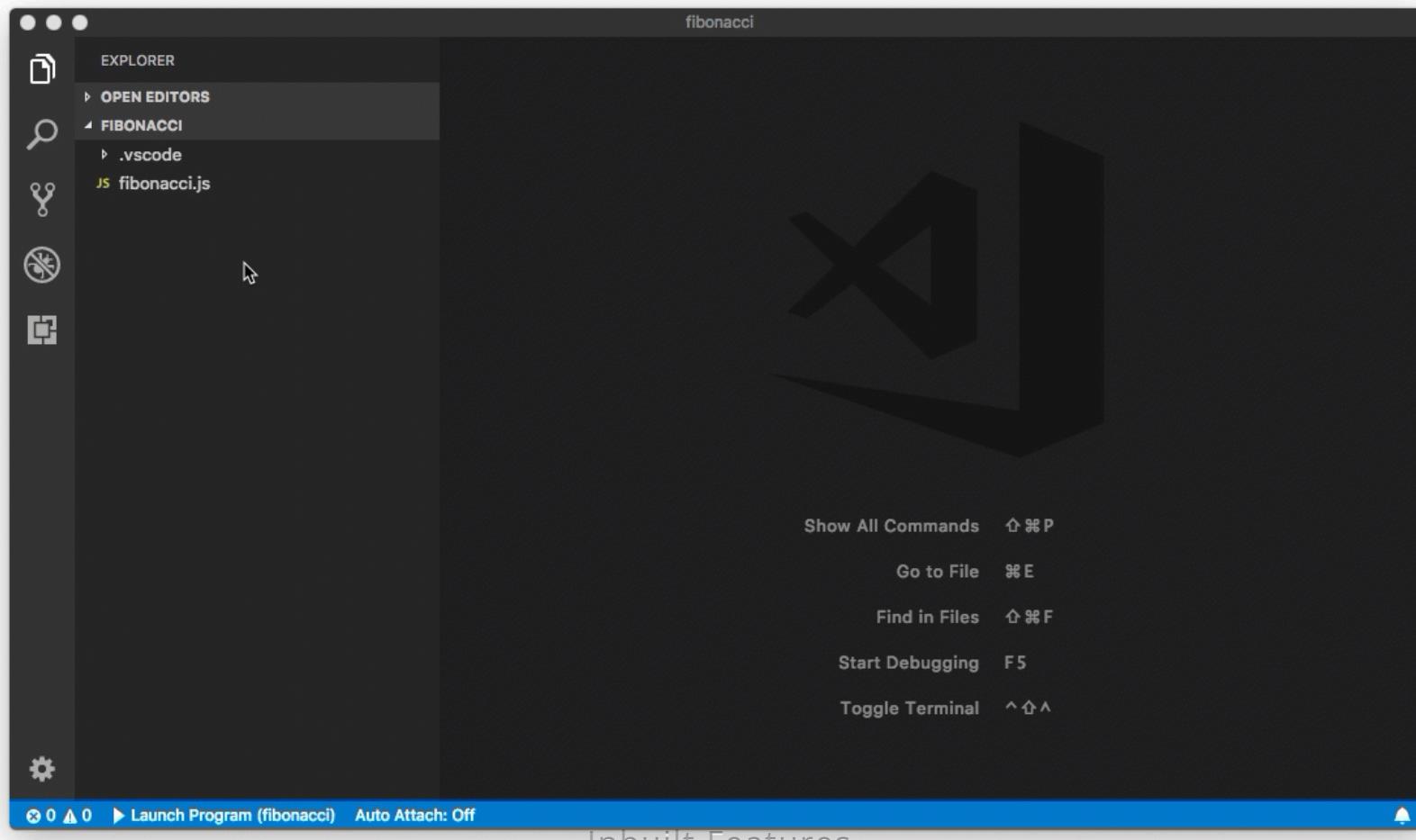
A screenshot of a code editor window titled "JS gulpfile.js > gulp.task('build-dist') callback". The code editor displays the following JavaScript code:

```
JS gulpfile.js > gulp.task('build-dist') callback
21  gulp.task('build-dist', done => {
22    if (!$.which('git')) {
23      $.echo('This command requires git')
24      $.exit(1)
25      done()
26    }
27
28    // Go to _build
29    if (!$.test('-e', '_build')) {
30      $.mkdir('_build')
31    }
32    $.cd('_build')
```

The code editor features vertical ruler lines at the right edge of the code area, indicating specific line lengths or column positions. The first ruler line is positioned at approximately column 75, and the second ruler line is positioned at approximately column 150.

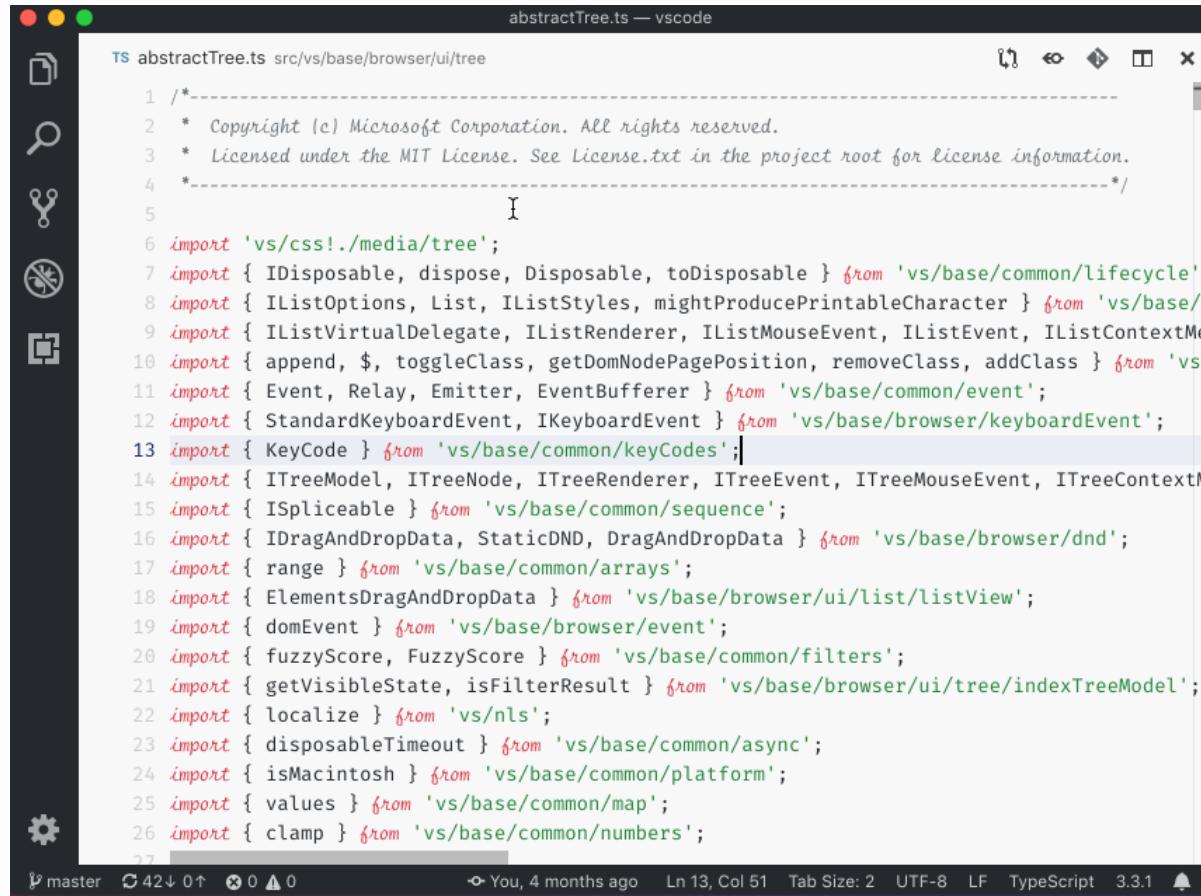
LogPoints

Non-breaking log points to show logging



Inbuilt Features

Screencast Mode



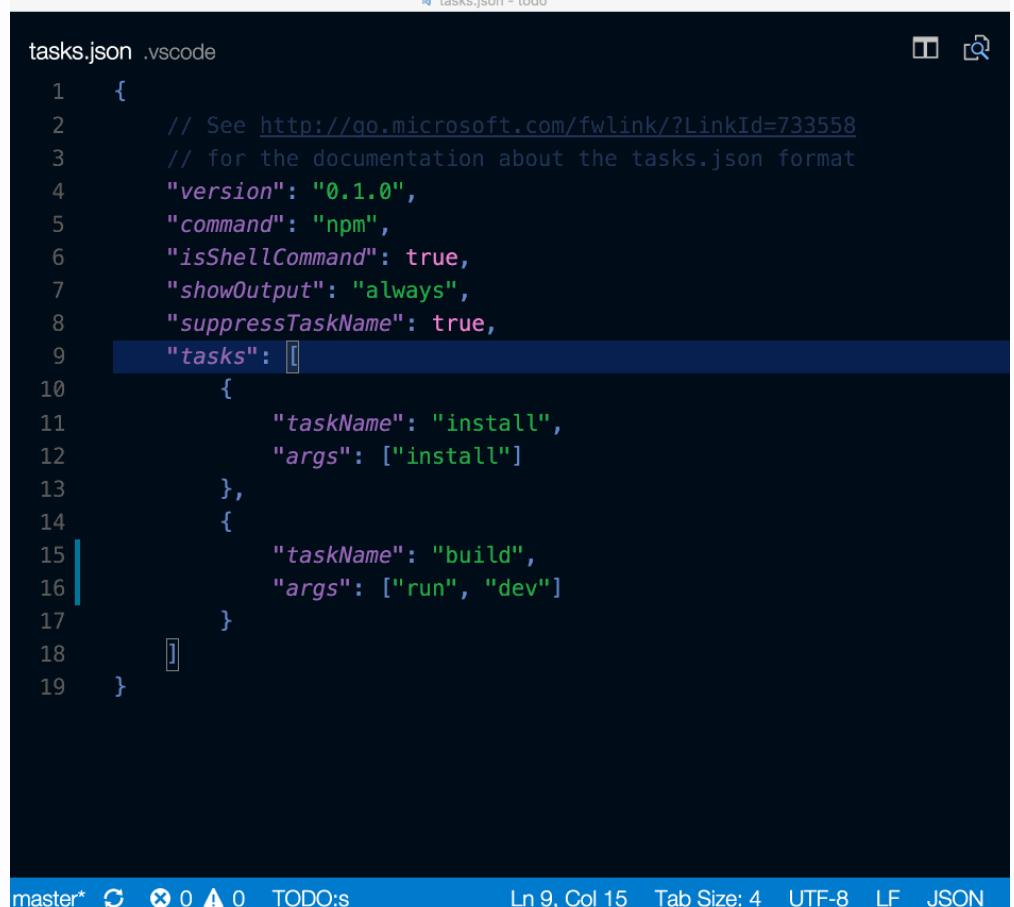
A screenshot of the Visual Studio Code (VS Code) interface in dark mode. The main area shows the code for `abstractTree.ts` located at `src/vs/base/browser/ui/tree`. The code is a TypeScript file containing numerous imports from the `'vs'` namespace, including various interfaces and classes related to tree rendering and events. The interface includes a left sidebar with icons for file operations, a top bar with window controls, and a bottom status bar showing the current file path, commit status, and other details.

```
abstractTree.ts — vscode
TS abstractTree.ts src/vs/base/browser/ui/tree
1 /**
2  * Copyright (c) Microsoft Corporation. All rights reserved.
3  * Licensed under the MIT License. See License.txt in the project root for license information.
4 */
5
6 import 'vs/css!./media/tree';
7 import { IDisposable, dispose, Disposable, toDisposable } from 'vs/base/common/lifecycle';
8 import { IListOptions, List, IListStyles, mightProducePrintableCharacter } from 'vs/base/b';
9 import { IListVirtualDelegate, IListRenderer, IListMouseEvent, IListEvent, IListContextMe
10 import { append, $, toggleClass, getDomNodePagePosition, removeClass, addClass } from 'vs/b
11 import { Event, Relay, Emitter, EventBufferer } from 'vs/base/common/event';
12 import { StandardKeyboardEvent, IKeyboardEvent } from 'vs/base/browser/keyboardEvent';
13 import { KeyCode } from 'vs/base/common/keyCodes';
14 import { ITreeModel, ITreeNode, ITreeRenderer, ITreeEvent, ITreeMouseEvent, ITreeContextM
15 import { ISpliceable } from 'vs/base/common/sequence';
16 import { IDragAndDropData, StaticDND, DragAndDropData } from 'vs/base/browser/dnd';
17 import { range } from 'vs/base/common/arrays';
18 import { ElementsDragAndDropData } from 'vs/base/browser/ui/list/listView';
19 import { domEvent } from 'vs/base/browser/event';
20 import { fuzzyScore, FuzzyScore } from 'vs/base/common/filters';
21 import { getVisibleState, isFilterResult } from 'vs/base/browser/ui/tree/indexTreeModel';
22 import { localize } from 'vs/nls';
23 import { disposableTimeout } from 'vs/base/common/async';
24 import { isMacintosh } from 'vs/base/common/platform';
25 import { values } from 'vs/base/common/map';
26 import { clamp } from 'vs/base/common/numbers';
27
```

master 42 0 0 0 You, 4 months ago Ln 13, Col 51 Tab Size: 2 UTF-8 LF TypeScript 3.3.1

Task Runners

Run tasks like build,
test and other custom
tasks in Task Runners.



The screenshot shows the VS Code interface with a dark theme. A tab labeled "TASKS.JSON - TODO" is open. The code editor displays the contents of a "tasks.json" file:

```
tasks.json .vscode
1  {
2      // See http://go.microsoft.com/fwlink/?LinkId=733558
3      // for the documentation about the tasks.json format
4      "version": "0.1.0",
5      "command": "npm",
6      "isShellCommand": true,
7      "showOutput": "always",
8      "suppressTaskName": true,
9      "tasks": [
10         {
11             "taskName": "install",
12             "args": ["install"]
13         },
14         {
15             "taskName": "build",
16             "args": ["run", "dev"]
17         }
18     ]
19 }
```

The status bar at the bottom shows "master*" and various icons, along with "Ln 9, Col 15" and "Tab Size: 4".

Remote Development

Remote Description

Remote via SSH

Connect to remote machines with Visual Studio Code via SSH.

Work in WSL

Work in Windows Subsystem for Linux.

Develop in Containers

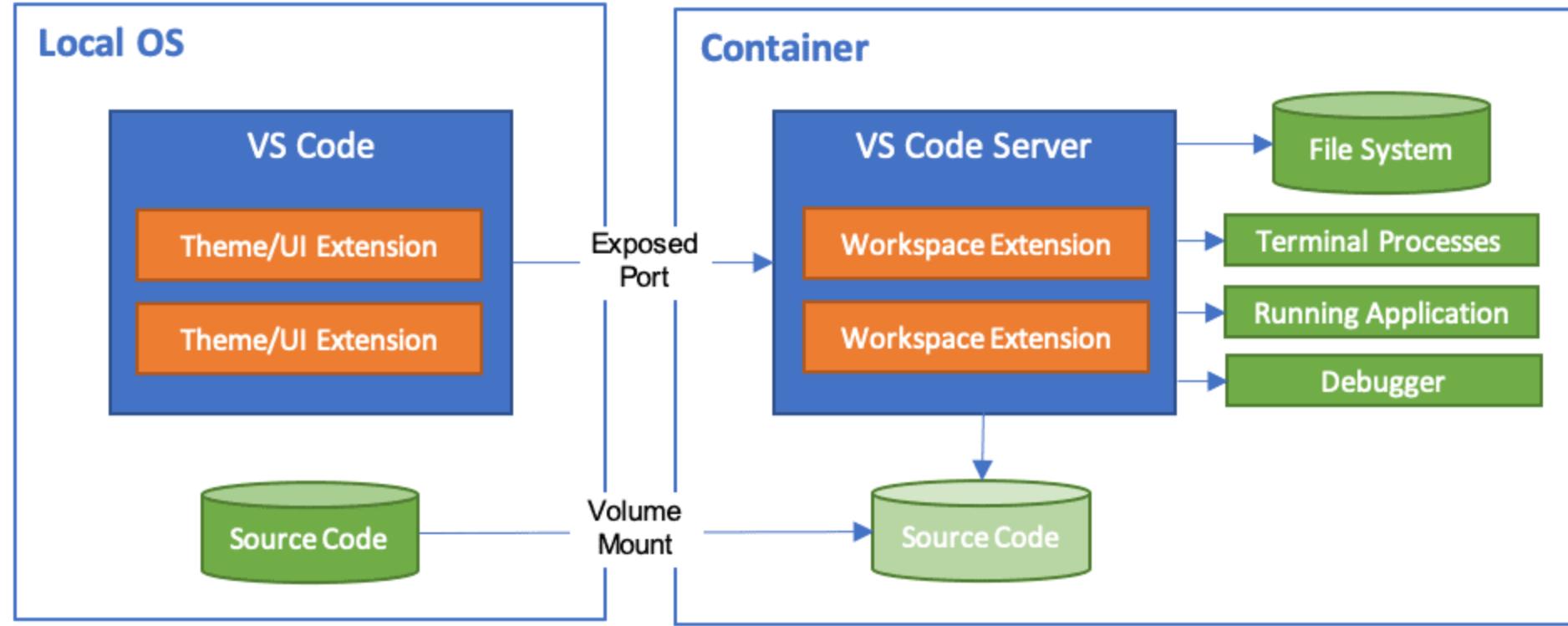
Work in a Docker Container.

GitHub Codespaces

Work in codespace with Visual Studio Code.
Inbuilt Features

Dev Containers

Dev Containers uses a `devcontainer.json` file in your project tells VS Code how to access (or create) a development container with a well-defined tool and runtime stack.



Extensions

Extensions

45

Installation

Install extensions with `Ctrl+Shift+X`. You can add to the workspace `@recommended` extensions for a project.

Most Popular extensions can be found with
`@popular`

| | | | | | |
|--|--|---|---|---|---|
|  Jupyter
Microsoft  microsoft.com  72.6M
Jupyter notebook support, interactive programming and computing that supports...
  |  Extension Pack for Java
Microsoft  microsoft.com  23.4M
Popular extensions for Java development that provides Java IntelliSense, debugging...
  |  C/C++ Extension Pack
Microsoft  microsoft.com  23M
Popular extensions for C++ development in Visual Studio Code.
  |  Python Extension Pack
Don Jayamanne   7M
Popular Visual Studio Code extensions for Python
  |  Remote Development
Microsoft  microsoft.com  4.8M
An extension pack that lets you open any folder in a container, on a remote...
  |  PHP Extension Pack
Xdebug  xdebug.org  4.5M
Everything you need for PHP development
  |
| 
Live Sass Compiler
Ritwick Dey   2.5M
Compile Sass or Scss to CSS at realtime with live browser reload.
  | 
json
ZainChen   2.1M
Json for Visual Studio Code
  | 
Auto Complete Tag
Jun Han   2M
Extension adds autocomplete tag and paired tag automatically for HTML/XML
  | 
Spring Boot Extension
VMware  vmware.com  1.9M
Collection of extensions for developing Spring Boot applications
  | 
Git Extension Pack
Don Jayamanne   1.6M
Popular Visual Studio Code extensions for Git
  | 
Live Sass Compiler
Glenn Marks   1.4M
Compile Sass or Scss to CSS at realtime.
  |
| 
Salesforce Extension
Salesforce  salesforce.com  1.2M
Extensions for developing on the Salesforce Platform
 | 
Angular Essentials (Vue)
John Papa  johnpapa.net  1.2M
Essential extensions for Angular developers
 | 
Unity Tools
Avin Zarlez   1.2M
Various tools to help with Unity development
 | 
Azure Tools
Microsoft  microsoft.com  1M
Get web site hosting, SQL and MongoDB data, Docker
 | 
Node.js Extension Pack
Wade Anderson   980K
Popular VS Code extensions for Node.js development.
 | 
Python Path
Mathias Gesbert   868K
Python imports utils.
 47 |

Marp

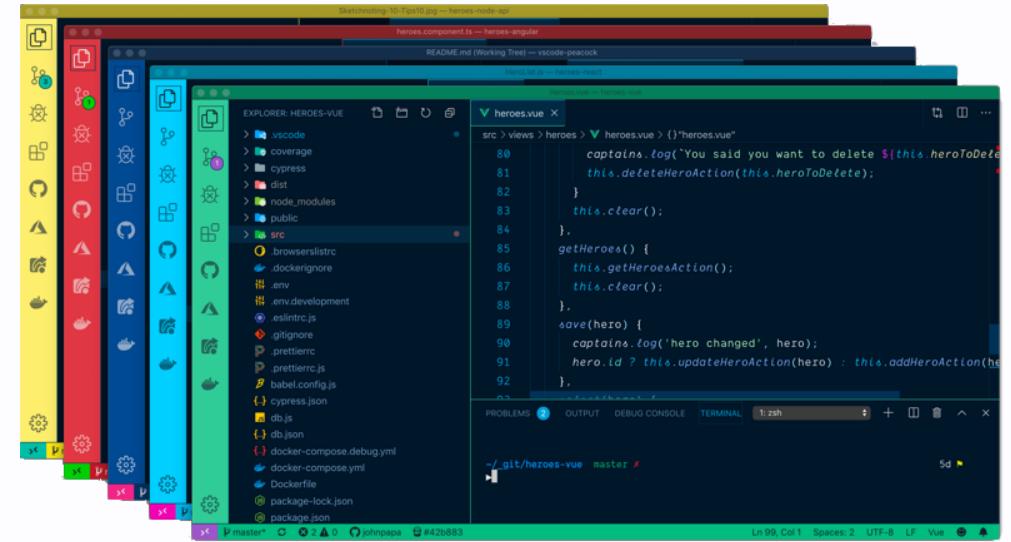
This slide deck written in Markdown within VSCode. I used the Marp extension preview it.

<https://marp.app/>

Marp has a CLI - so this slide has a CICD pipeline 😊

Peacock

Peacock allows you to colour code VS Code.



IntelliCode

AI-assisted
development features
for Python,
TypeScript/JavaScript
and Java developers

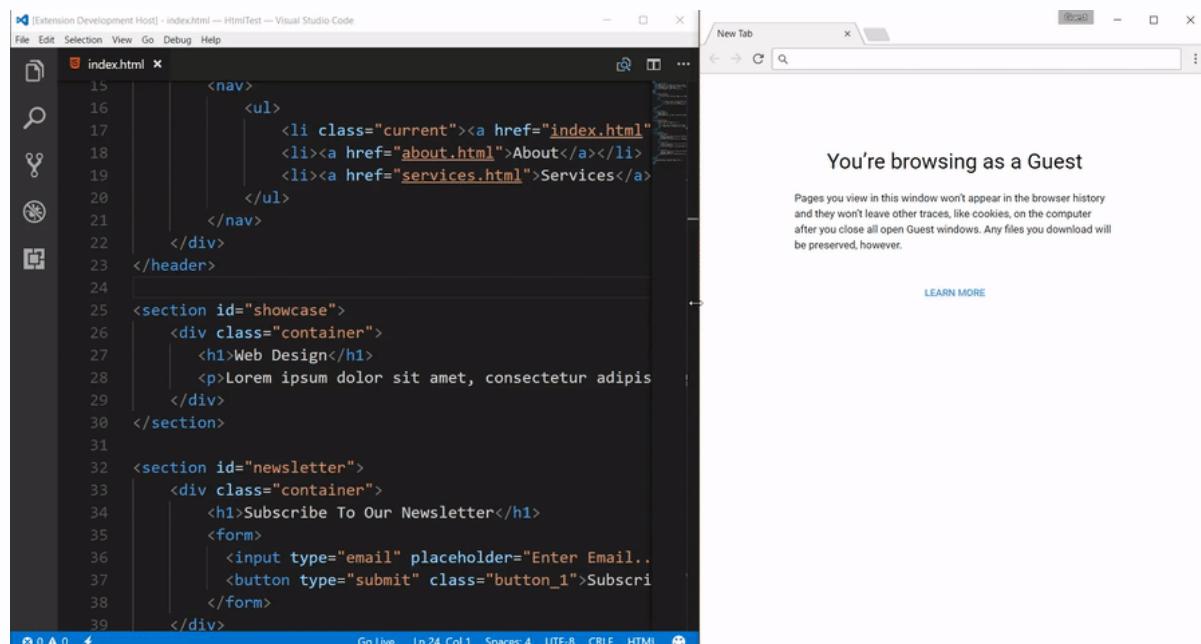
IntelliCode

```
loss = tf.reduce_sum(tf.square(linear_model - y))  
optimizer = tf.train.GradientDescentOptimizer(0.01)  
  
train = optimizer
```

LiveServer

Launch a development local Server with live reload feature for static & dynamic pages

LiveServer



The screenshot shows the Visual Studio Code interface with the 'LiveServer' extension installed. On the left, the 'index.html' file is open in the editor, displaying HTML code for a navigation bar and two sections. On the right, a browser window titled 'New Tab' shows a guest message: 'You're browsing as a Guest'. Below this message is a note about guest browsing: 'Pages you view in this window won't appear in the browser history and they won't leave other traces, like cookies, on the computer after you close all open Guest windows. Any files you download will be preserved, however.' A 'LEARN MORE' link is also present. The bottom status bar of the VS Code window shows 'Go Live' and other settings.

Indent Rainbow

Indent Rainbow

```
1 import nimbench          1 import nimbench
2
3 bench(str, m):
4     for n in 0..<m:
5         var dest ="apple"
6         while(dest.len<100):
7             var
8                 offs=dest.len-5
9                 i=offs
10                while i<offs+10:
11                    dest.add(dest[i])
12                    i.inc
13                doNotOptimizeAway(dest)
14
15 runBenchmarks()          1 import nimbench
2
3 bench(str, m):
4     for n in 0..<m:
5         var dest ="apple"
6         while(dest.len<100):
7             var
8                 offs=dest.len-5
9                 i=offs
10                while i<offs+10:
11                    dest.add(dest[i])
12                    i.inc
13                doNotOptimizeAway(dest)
14
15 runBenchmarks()
```

Path Intellisense

Path Intellisense
autocompletes
filenames

```
EXPLORER
WORKING FILES 1 UNSAVED
● index.html

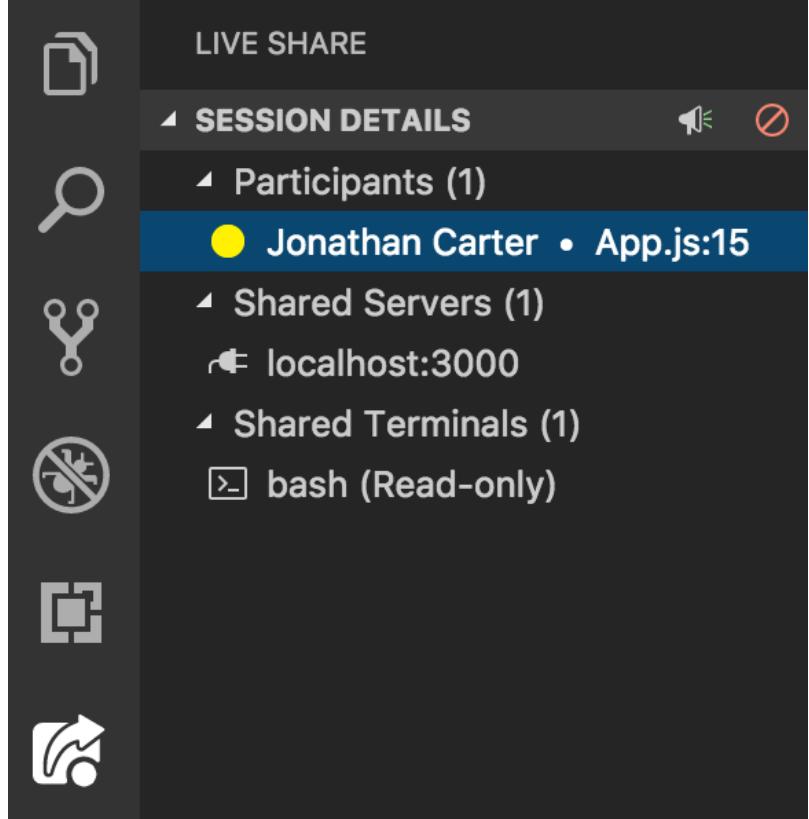
MY-PROJECT
CSS
  styles.css
SRC
  foo.js
  app.js
index.html

• index.html
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <title>Document</title>
6    ...
7  </head>
8  <body>
9    ...
10 </body>
11 </html>
```

Code Spell Checker

Code Spell Checker for source code

```
• validator.ts src
  1 import {
  2   | TextDocument, Diagnostic, DiagnosticSeverity,
  3 } from 'vscode-languageserver';
  4 import { isWordInDictionary } from './spellChecker';
  5 import * as Text from './util/text';
  6
  7 import * as Rx from 'rx';
  8 import { merge } from 'tsmerge';
  9
 10 /**
 11  * defalt costants
 12 */
 13
 14 const defaultMaxNumberOfProblems = 200;
 15 const defaultMinWordLength      = 4;
 16
 17 export interface ValidationOptions {
 18   maxNumberOfProblems?: number;
 19   minWordLength?: number;
 20   // words to always flag as an error
 21   flagWords?: string[];
 22 }
 23
 24 export function validateTextDocument(textDocument: TextDocument, options: ValidationOpti
 25   | return validateTextDocumentAsync(textDocument, options)
 26   |   .toArray()
```



LiveShare

Real-time collaborative development from the comfort of your favourite tools.

[LiveShare / Docs](#)

Jira and Bitbucket

Bringing the power of Jira and Bitbucket to VS Code

- You can create and view issues
- Start work on issues
- Create pull requests
- Do code reviews
- Start builds
- Get build statuses and more!

Jira and Bitbucket

Error Lens

Improve highlighting of errors, warnings and other language diagnostics.

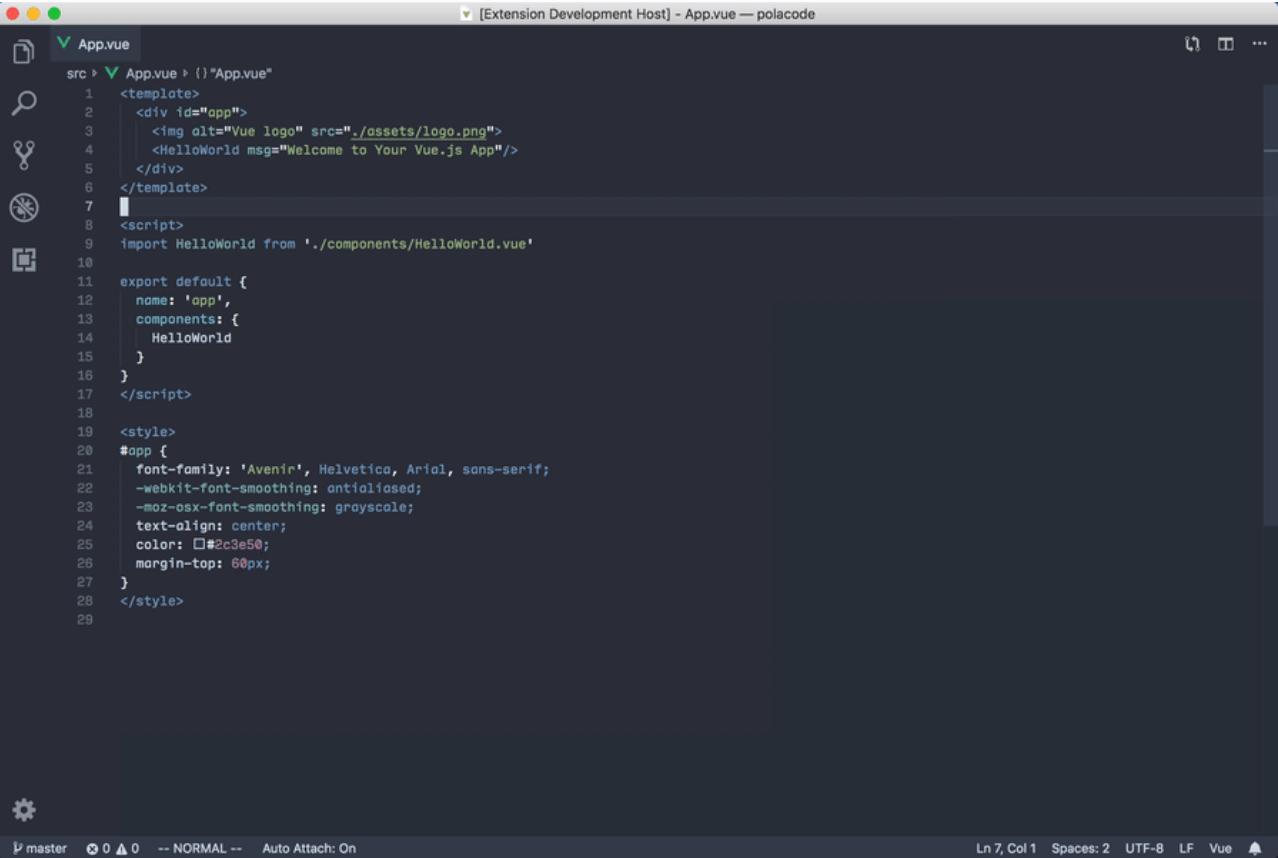
Error Lens

```
vscode.workspace.onDidOpenTextDocument(textDocument => {
  updateDecorationsForUri(textDocument.uri) Missing semicolon (semicolon)
}, undefined, context.subscriptions; ')' expected.
```

Polacode

 Polaroid for your code. Take screenshots of your code

Polacode



```
src > App.vue > () "App.vue"
1  <template>
2    <div id="app">
3      
4      <HelloWorld msg="Welcome to Your Vue.js App"/>
5    </div>
6  </template>
7
8  <script>
9  import HelloWorld from './components/HelloWorld.vue'
10
11 export default {
12   name: 'app',
13   components: {
14     HelloWorld
15   }
16 }
17 </script>
18
19 <style>
20 #app {
21   font-family: 'Avenir', Helvetica, Arial, sans-serif;
22   -webkit-font-smoothing: antialiased;
23   -moz-osx-font-smoothing: grayscale;
24   text-align: center;
25   color: #2c3e50;
26   margin-top: 60px;
27 }
28 </style>
29
```

GitHub Copilot

GitHub Copilot Your AI
pair programmer

```
1 def common_prefix(a, b):
2     """Return the common prefix of two lists."""
3     if len(a) < len(b):
4         return common_prefix(b, a)
5     for i in range(len(a)):
6         if a[i] != b[i]:
7             return a[:i]
8     return a
9
10
11 | |
```

C# Dev Kit

Official C# extension from Microsoft

- Project System / Solution Explorer
- Code Editing (Uses the C# extension)
- Package Management
- Debugging
- Testing

C# Dev Kit

Markdown Example

```
Console.WriteLine("C# Example");
✓ 0.0s
```

csharp - C# Script Code

C# Example

```
gantt
title A Gantt Diagram
dateFormat YYYY-MM-DD
section Section
A task :a1, 2024-01-01, 1d
Another task :after a1, 2d
section Another
Task in sec :2024-01-04, 3d
another task : 2d
✓ 0.0s
```

mermaid - Mermaid Code

```
<h1>HTML Example</h1>
✓ 0.0s
```

html - HTML Code

HTML Example

Polygot Notebooks

Polygot Notebooks allows C#, F#, PowerShell, JavaScript, SQL, KQL, Python, R, HTML, HTTP, Mermaid.

API Clients

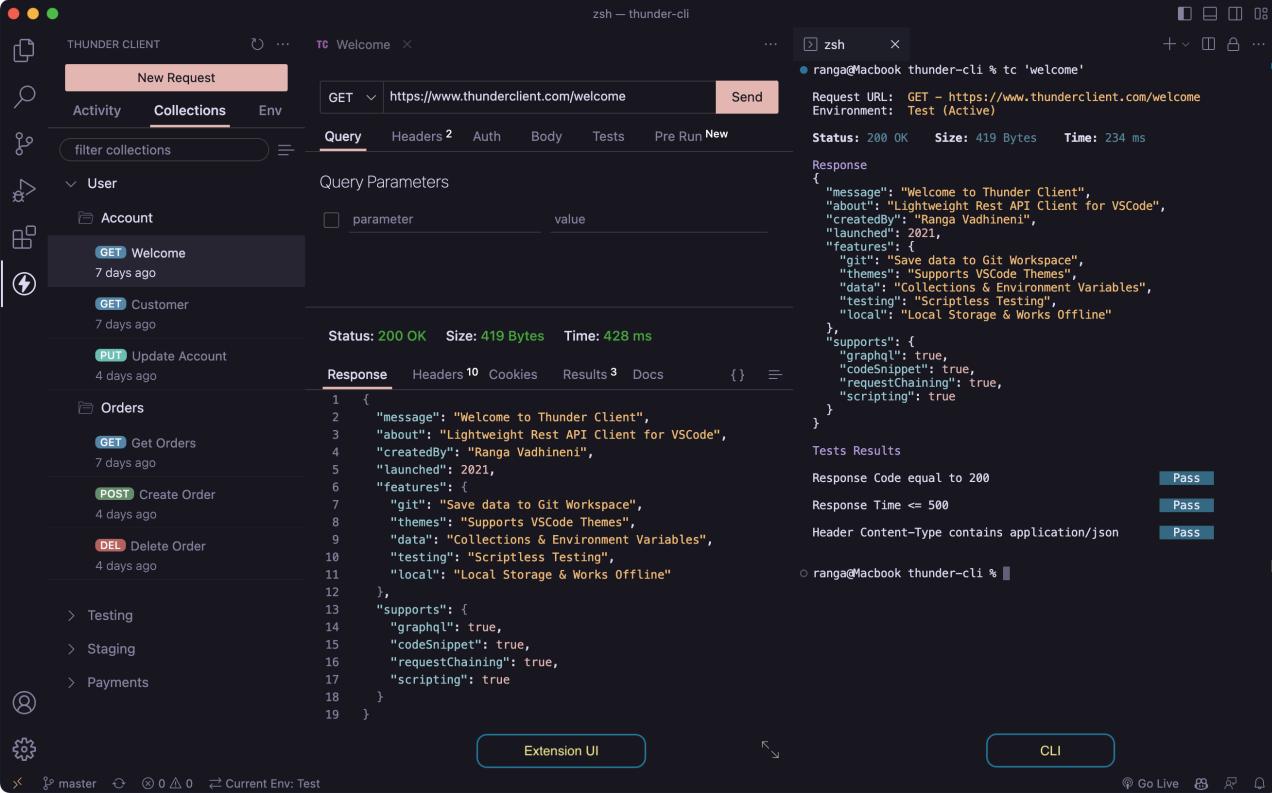
REST Client

```
GET https://example.com/topics/1 HTTP/1.1  
  
POST https://example.com/comments HTTP/1.1  
content-type: application/json  
  
{  
    "name": "sample",  
    "time": "Wed, 21 Oct 2015 18:27:50 GMT"  
}
```

```
curl https://www.google.com
```

Thunder Client

Lightweight Rest API Client for VS Code



A screenshot of the Visual Studio Code interface demonstrating the Todo Tree extension. The left sidebar displays a tree view of files, with the 'todo-tree' folder expanded to show its contents. A specific todo item, 'TODO Fix this!', is selected and highlighted in blue. The main editor area shows a file named 'extension.js' with several todo items marked with purple circles. The bottom right corner of the editor shows a terminal window with the output of a todo search command, indicating 56 results found.

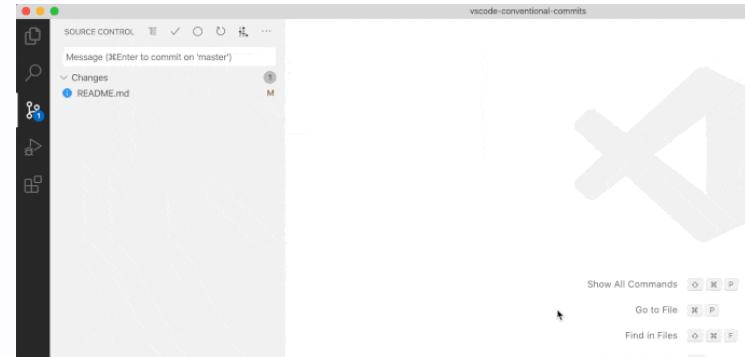
TODO Tree

TODO Tree quickly find all your todo items.

Conventional Commits

```
<type>[optional scope]: <description>  
[optional body]  
[optional footer(s)]
```

Conventional Commits
conventionalcommits.org
which then dovetails to
semver.org
Extensions



```
2 export class MyClass {  
3  
4     /**  
5      * MyMethod  
6      * * Important information is highlighted  
7      * ! Deprecated method, do not use  
8      * ? Should this method be exposed in the public API?  
9      * TODO: refactor this method so that it conforms to the API  
10     * @param myParam The parameter for this method  
11     */  
12     public MyMethod(myParam: any): void {  
13         let myVar: number = 123;  
14  
15         /* This is highlighted  
16         if (myVar > 0) {  
17             throw new TypeError(); //! This is an alert  
18         }  
19  
20         //? This is a query  
21         let x = 1;  
22  
23         //!!! this.lineOfCode() == commentedOut;  
24  
25         //TODO: Create some test cases  
26     }  
27 }
```

Better Comments

Better Comments
colour codes your
commit messages

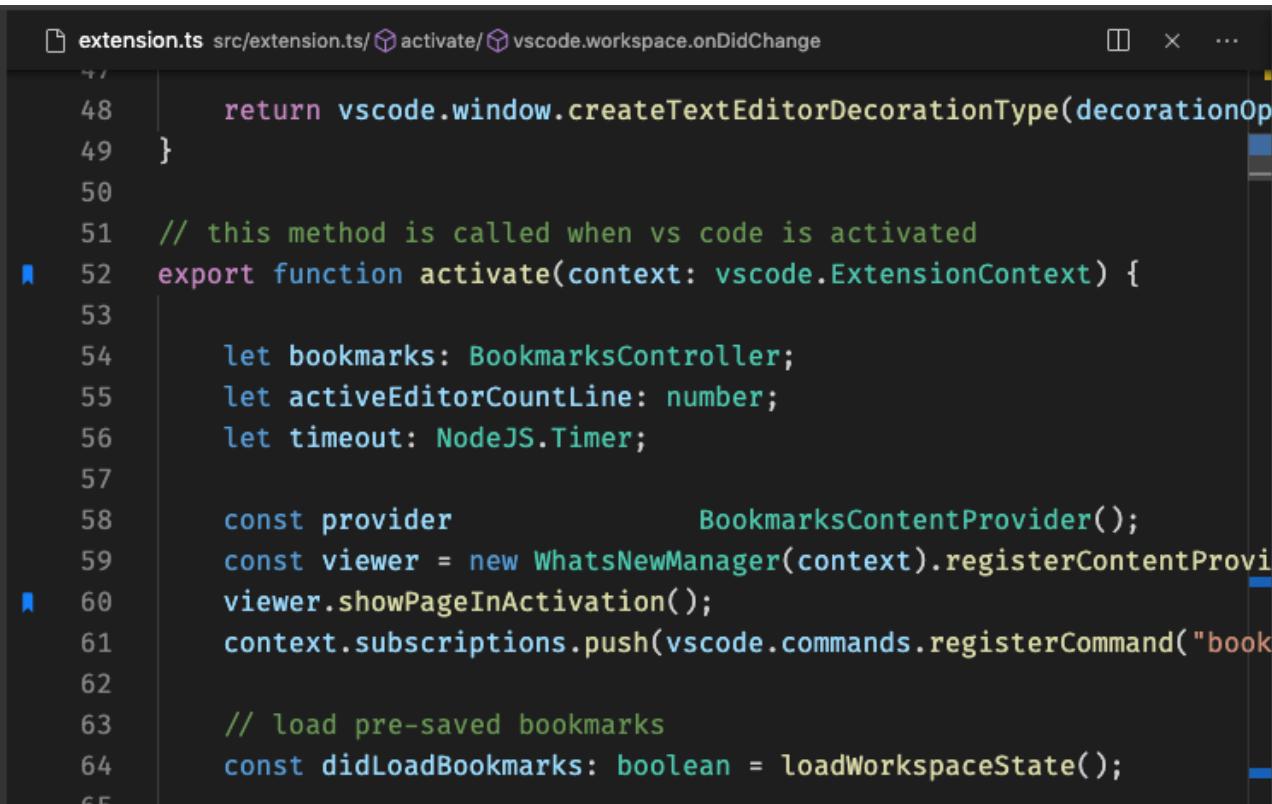
Unique Lines

Keep **unique lines** of text and remove duplicates from current selection. Also includes a command to shuffle currently selected lines.

Encode Decode

Encode Decode

- Convert String **to/from** Base64, HTML Entities, JSON Byte Array, JSON String, Unicode, XML Entities
- Convert String to MD5/SHA1/SHA256/SHA512 (as Base64 or Hex)



A screenshot of a VS Code editor window showing the file `extension.ts`. The code is written in TypeScript and handles the activation of an extension. It includes logic for creating text editor decorations, initializing a `BookmarksController`, and registering a command to show a viewer. It also loads pre-saved bookmarks from workspace state.

```
extension.ts src/extension.ts/ activate/ vscode.workspace.onDidChange
48     return vscode.window.createTextEditorDecorationType(decorationOp
49 }
50
51 // this method is called when vs code is activated
52 export function activate(context: vscode.ExtensionContext) {
53
54     let bookmarks: BookmarksController;
55     let activeEditorCountLine: number;
56     let timeout: NodeJS.Timer;
57
58     const provider          BookmarksContentProvider();
59     const viewer = new WhatsNewManager(context).registerContentProv
60     viewer.showPageInActivation();
61     context.subscriptions.push(vscode.commands.registerCommand("book
62
63     // load pre-saved bookmarks
64     const didLoadBookmarks: boolean = loadWorkspaceState();
```

Bookmarks

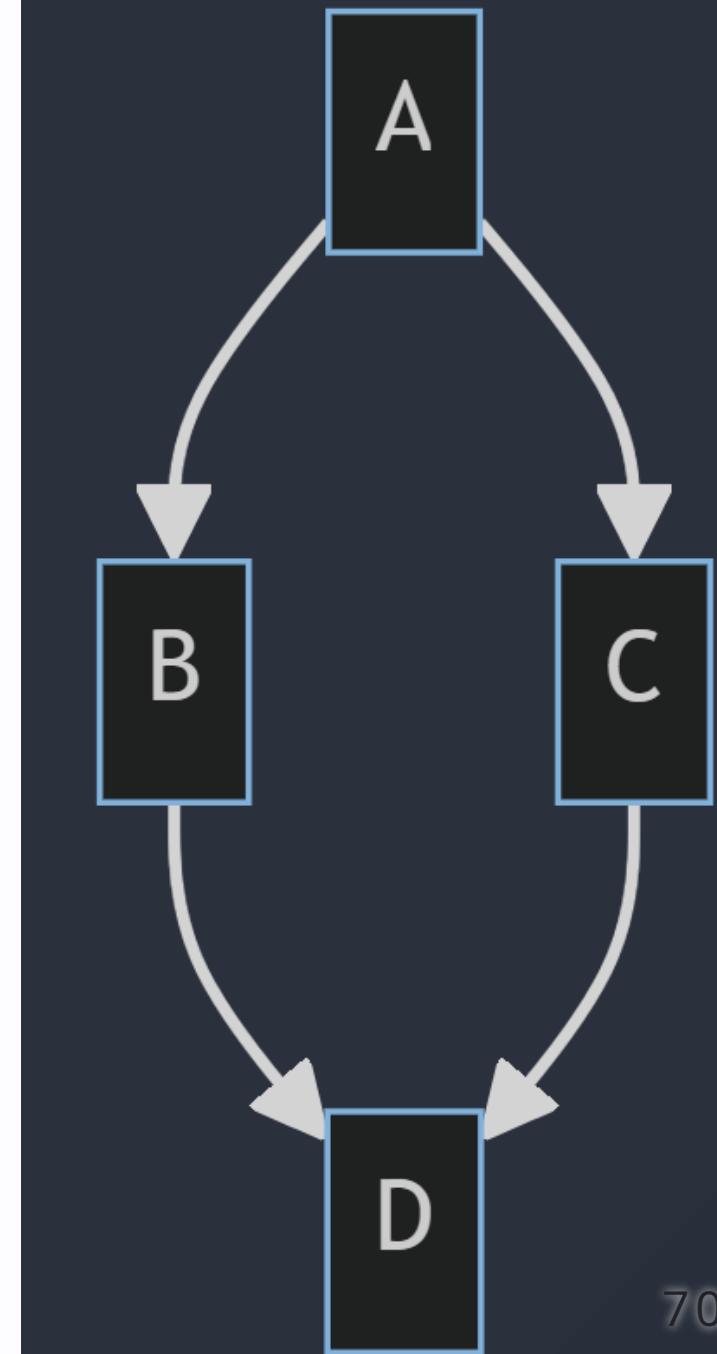
Bookmark your code
and quickly jump to
bookmarks.

Markdown Preview

Markdown Preview allows you to create Mermaid JS documents that can be included in Markdown documents.

```
graph TD;
    A-->B;
    A-->C;
    B-->D;
    C-->D;
```

Extensions



Aside: Mermaid JS can generate

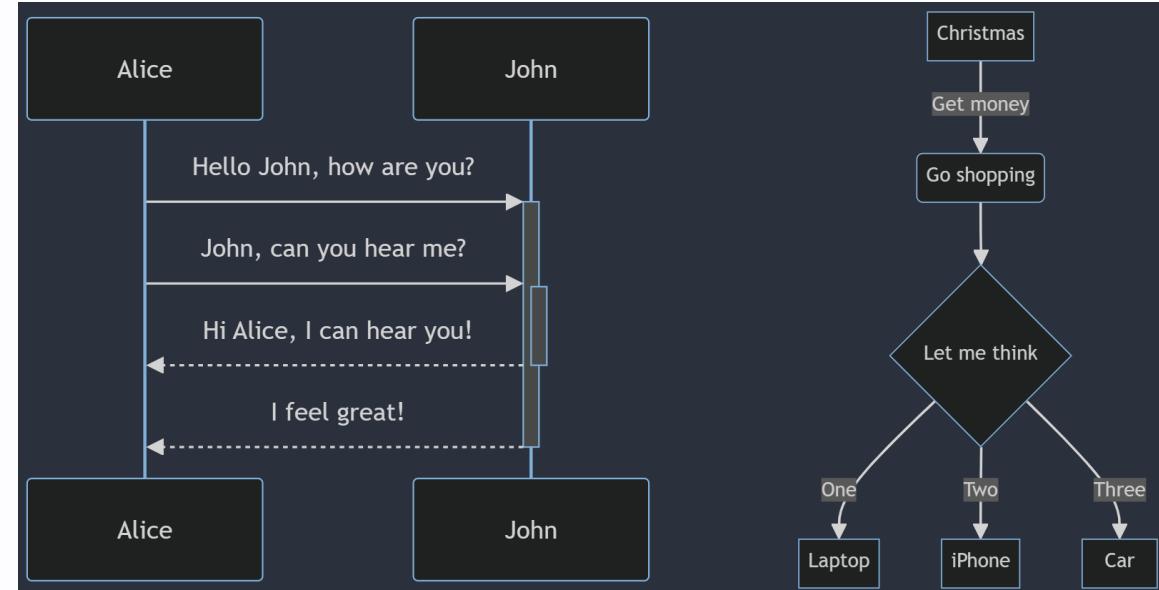
- Generate Flowchart
- Sequence Diagram
- Class Diagram
- State Diagram
- Entity Relationship Diagram
- User Journey
- Gantt
- Pie Chart
- Quadrant Chart
- Requirement Diagram

```
sequenceDiagram
```

```
    Alice->>+John: Hello John, how are you?  
    Alice->>+John: John, can you hear me?  
    John-->-Alice: Hi Alice, I can hear you!  
    John-->-Alice: I feel great!
```

```
flowchart TD
```

```
    A[Christmas] -->|Get money| B((Go shopping))  
    B --> C{Let me think}  
    C -->|One| D[Laptop]  
    C -->|Two| E[iPhone]  
    C -->|Three| F[fa:fa-car Car]
```



```
{} heroes.json ✘  
{} heroes.json > [ ]members > {} 0 > [ ]powers  
1 {  
2   "squadName": "Super hero squad",  
3   "homeTown": "Metro City",  
4   "formed": 2016,  
5   "secretBase": "Super tower",  
6   "active": true,  
7   "members": [  
8     {  
9       "name": "Molecule Man",  
10      "age": 29,  
11      "secretIdentity": "Dan Jukes",  
12      "powers": [  
13        "Radiation resistance",  
14        "Turning tiny",  
15        "Radiation blast"  
16      ]  
17    },  
18    {  
19      "name": "Aykut Sarac",  
20      "age": 21,  
21      "secretIdentity": "John Doe",  
22      "powers": [  
23        "React",  
24        "TypeScript",  
25        "JavaScript"  
26      ]  
27    },  
28  ]
```

JSON Crack

JSON Crack
seamlessly
visualize your
JSON data
instantly into
graphs.

SemanticDiff

SemanticDiff is a
programming
language aware
diffs.

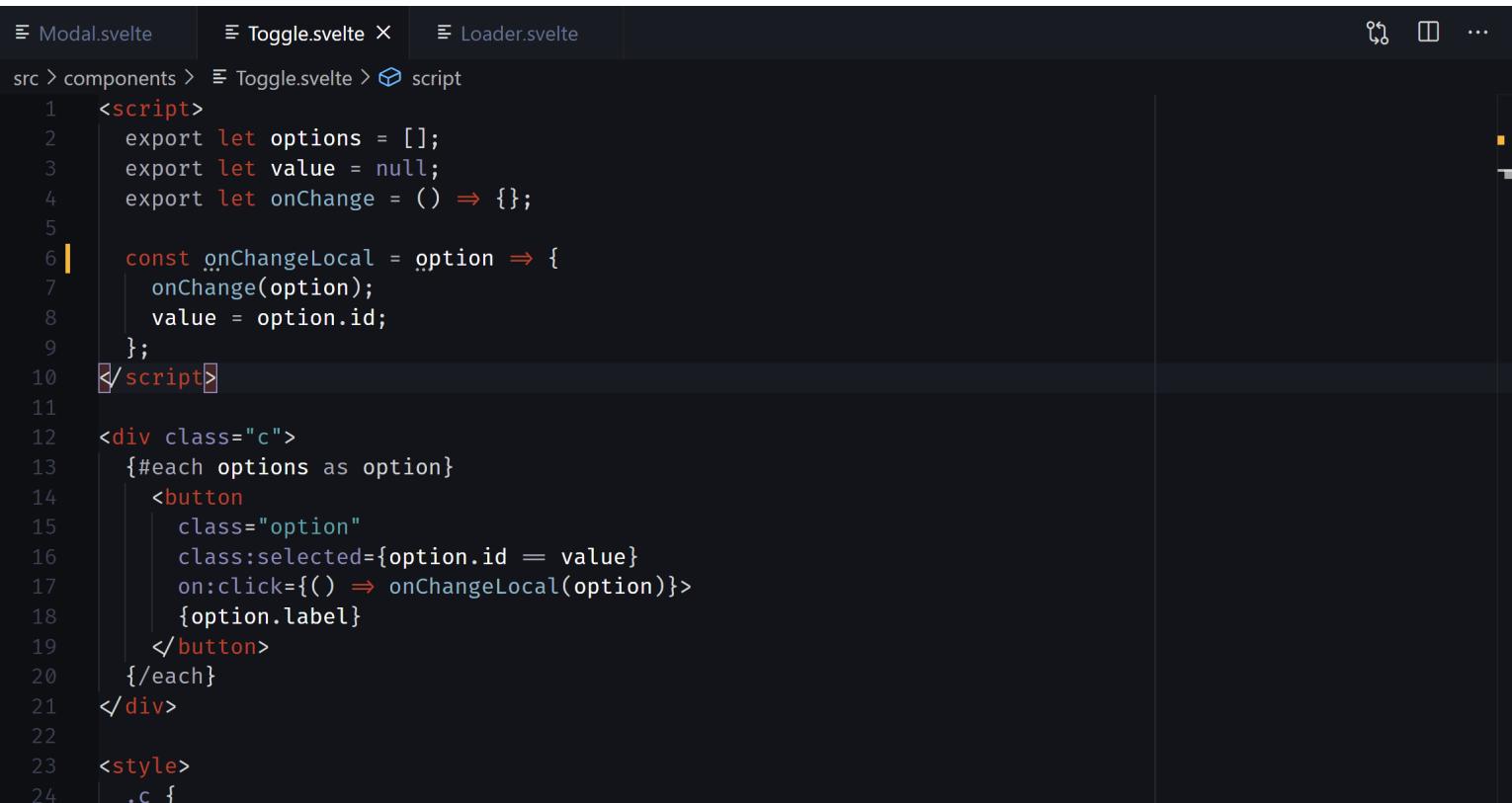
Can show
differences when
you move code.

```
1 import os
2 import hashlib
3
4 def print_hashes(path):
5     for file in os.listdir(path):
6         file_path = os.path.join(path, file)
7         if os.path.islink(file_path):
8             continue
9         if not os.path.isfile(file_path):
10            continue
11
12     def calc_hash(file_path):
13         h = hashlib.md5()
14         with open(file_path, "rb") as fp:
15             while True:
16                 data = fp.read(4096)
17                 if not data:
18                     break
19                 h.update(data)
20
21         return h
22
23     h = calc_hash(file_path)
24     print(f"{h.hexdigest()} {file}")
25
26 print_hashes(".")

1 import os
2 import hashlib
3
4 def calc_hash(file_path):
5+     h = hashlib.sha256()
6     with open(file_path, "rb") as fp:
7         while True:
8             data = fp.read(4096)
9+             if data == b"":
10                break
11             h.update(data)
12
13     return h
14
15+ def print_hashes(path):
16+     for file in os.listdir(path):
17+         file_path = os.path.join(path, file)
18+         if os.path.islink(file_path):
19+             continue
20+         if not os.path.isfile(file_path):
21+             continue
22+
23     h = calc_hash(file_path)
24     print(f"{h.hexdigest()} {file}")
25
26 print_hashes(".")
```

FootSteps

FootSteps
Highlight and
navigate
between your
most recently
edited chunks of
code

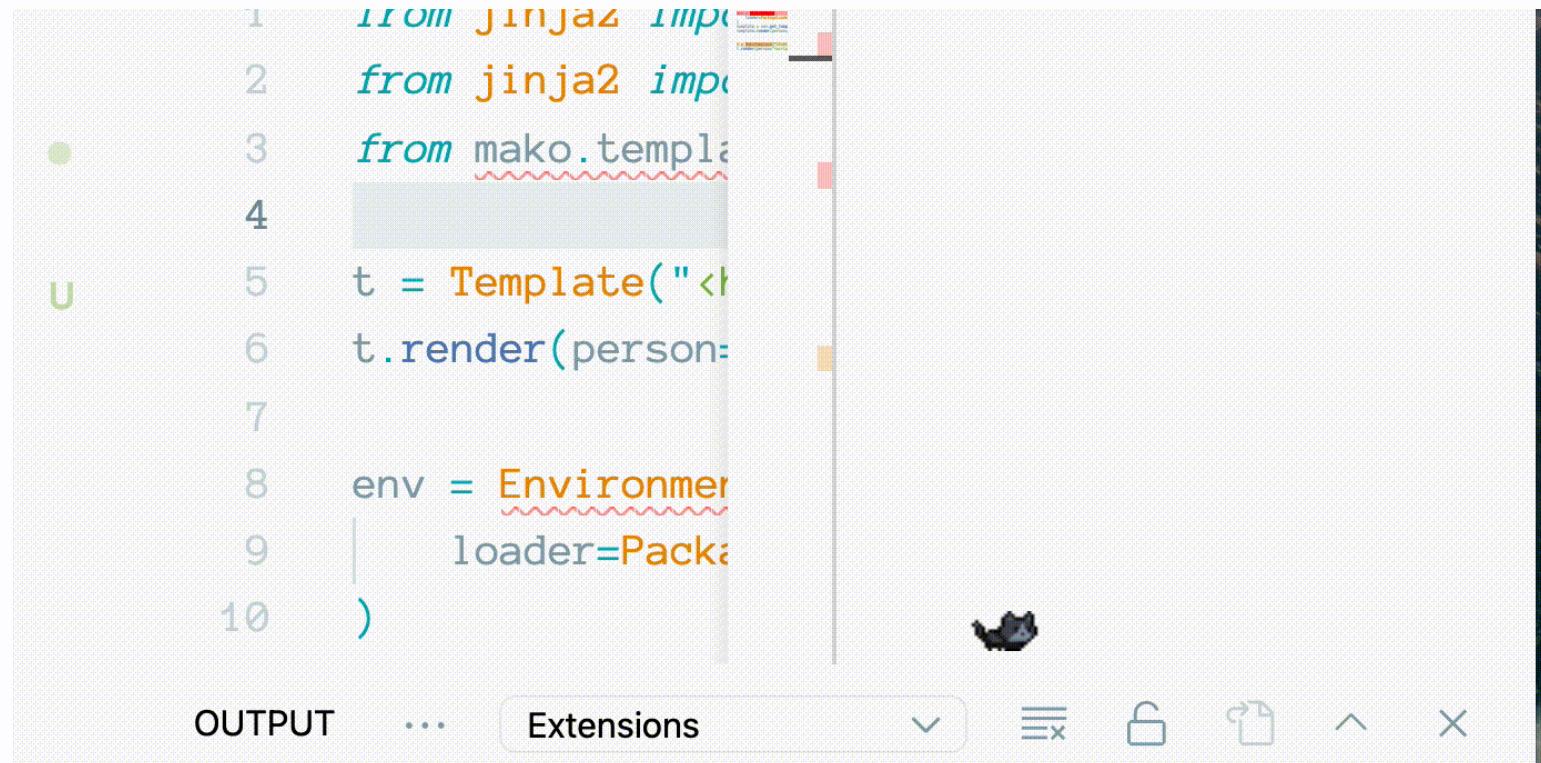


A screenshot of a code editor interface. At the top, there are three tabs: "Modal.svelte", "Toggle.svelte X", and "Loader.svelte". Below the tabs, the file "Toggle.svelte" is open, showing Svelte code. A vertical yellow highlight bar is positioned on the left side of the editor, spanning from line 6 down to line 24. The code in the editor includes:

```
src > components > Toggle.svelte > script
1  <script>
2    export let options = [];
3    export let value = null;
4    export let onChange = () => {};
5
6    const onChangeLocal = option => {
7      onChange(option);
8      value = option.id;
9    };
10   </script>
11
12  <div class="c">
13    {#each options as option}
14      <button
15        class="option"
16        class:selected={option.id == value}
17        on:click={() => onChangeLocal(option)}>
18        {option.label}
19      </button>
20    {/each}
21  </div>
22
23  <style>
24    .c {
```

Pets for your VS Code

Pets gives you a virtual pet to play with while you are coding.



A screenshot of the Visual Studio Code interface. The code editor shows a snippet of Python code:

```
1 from jinja2 import
2 from jinja2 import
3 from mako.template import
4
5 t = Template("{{ person.name }}")
6 t.render(person=person)
7
8 env = Environment(
9     loader=PackageLoader('myapp', 'templates'))
10 )
```

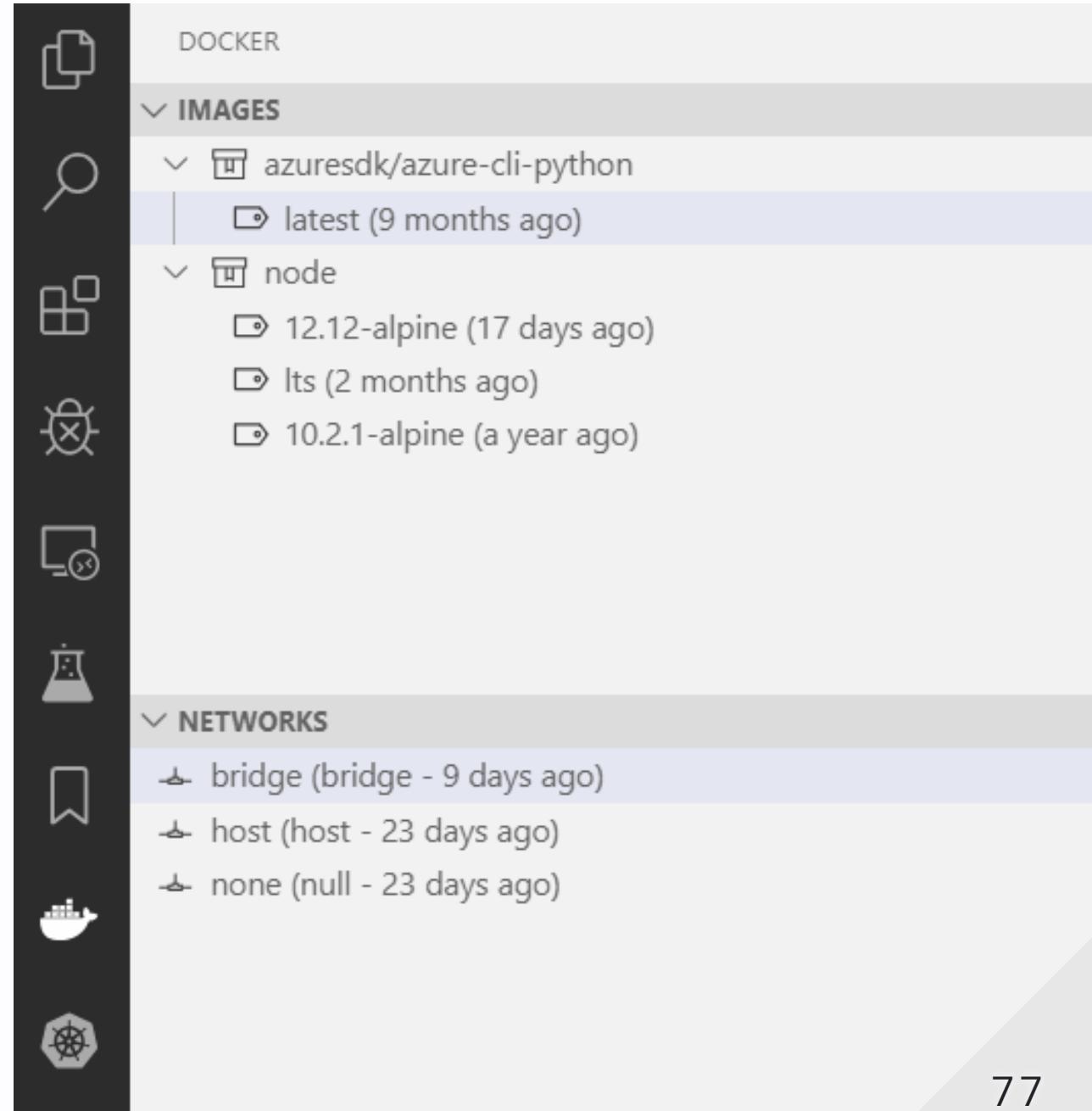
The word "Template" is highlighted in orange. Below the code editor is a status bar with tabs for "OUTPUT", "...", and "Extensions". To the right of the status bar are several icons: a dropdown arrow, a refresh symbol, a lock, a file, a double arrow, and an "X". A small black cat icon is positioned to the right of the status bar.

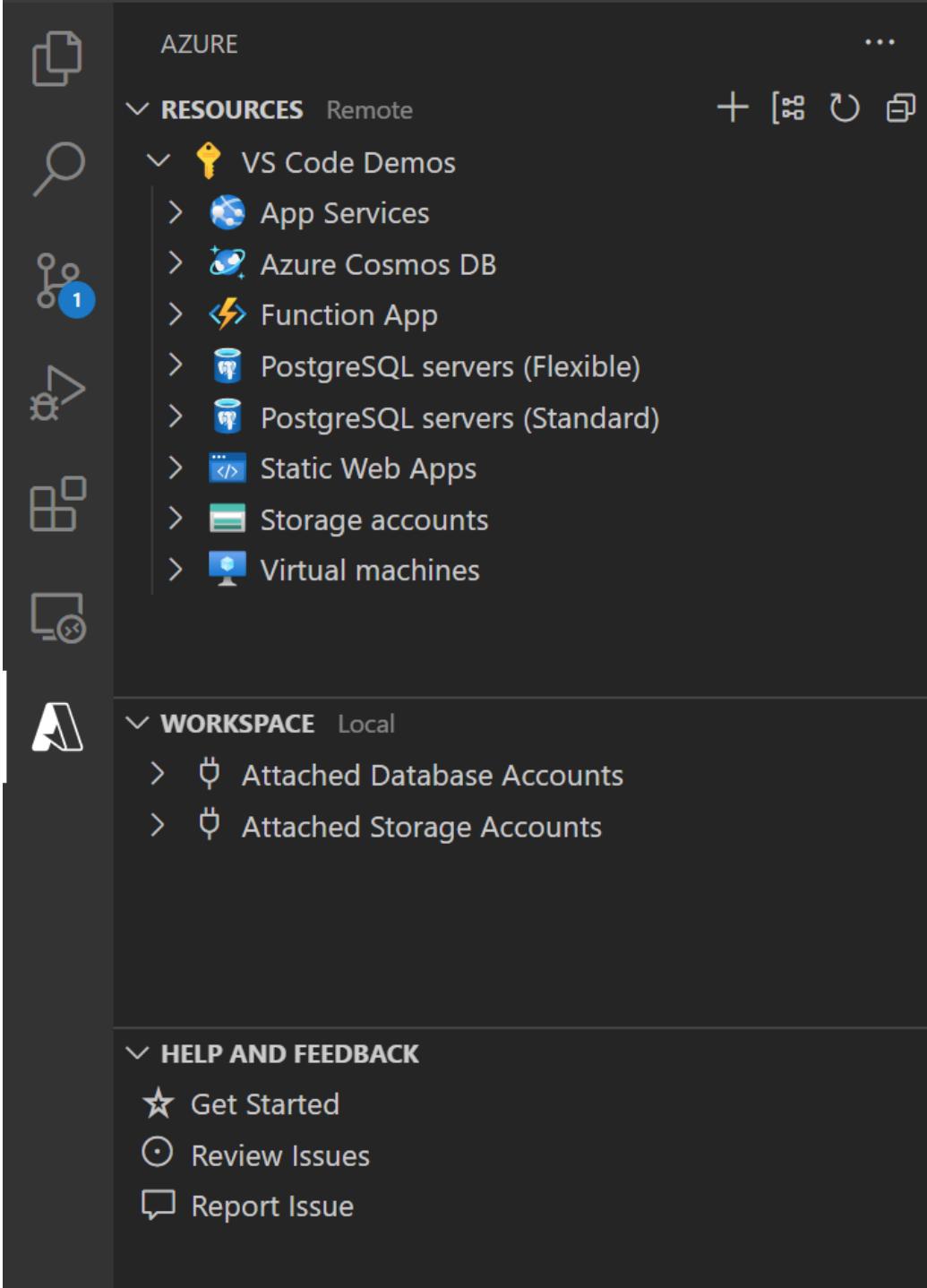
Docker

Makes it easy to
create, manage, and
debug containerized
applications.

Docker

Extensions





Azure Tools

Get web site hosting,
SQL and MongoDB
data, Docker
Containers, Serverless
Functions and more,
all on Azure.

Azure Tools

AWS Toolkit

Including CodeWhisperer, CodeCatalyst, and support for Lambda, S3, CloudWatch Logs, and many other services

AWS Toolkit

Extensions

