



An Introduction to VS Code

Peng Lian

2023-06-28

Table of contents

- Overview
- User Interface
- The Editor
- Source control
- Work with Python
- Acknowledgement
- Questions?
- Thanks for your attention!

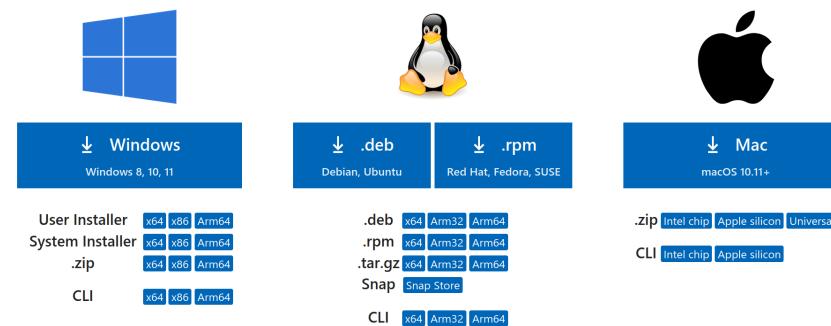
Overview

What is VS Code

Visual Studio Code (VS Code) is a free and open-source code editor developed by Microsoft. It is available for Windows, macOS, and Linux. VS Code is a powerful code editor that can be used for a variety of tasks, including:

- Programming
- Editing text
- Debugging
- Building and running applications
- Extending with extensions

VS Code is a great choice for both beginners and experienced developers. In the Stack Overflow 2022 Developer Survey, VS Code was ranked **the most popular** developer environment tool among 71,010 respondents, with 74.48% reporting that they use it.



Note: [Visual Studio Code](#) is a totally separate product from [Visual Studio](#) the IDE.

Source code: <https://github.com/Microsoft/vscode/>

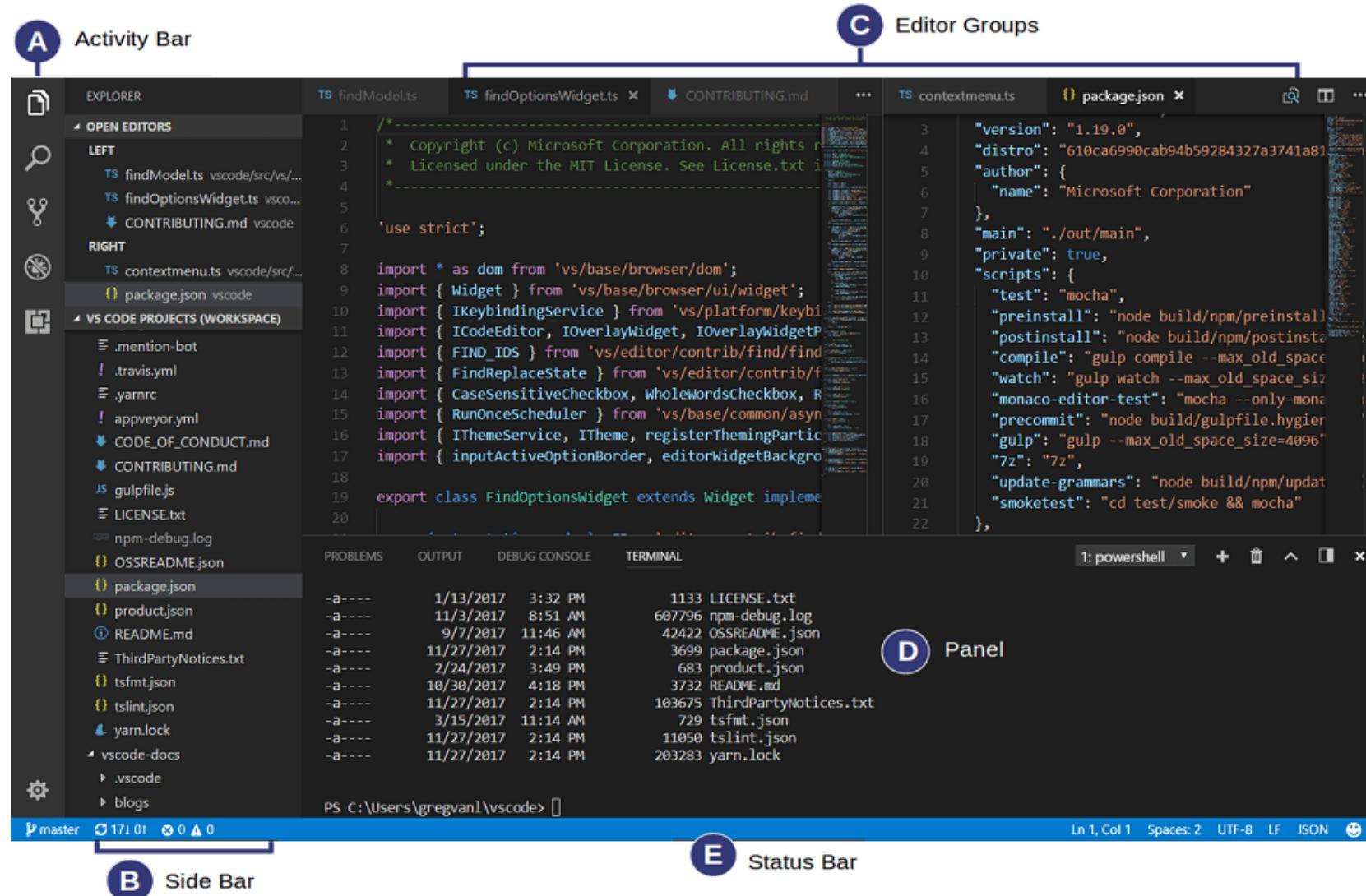
Documentations: <https://code.visualstudio.com/docs>

VS Code on BioHPC

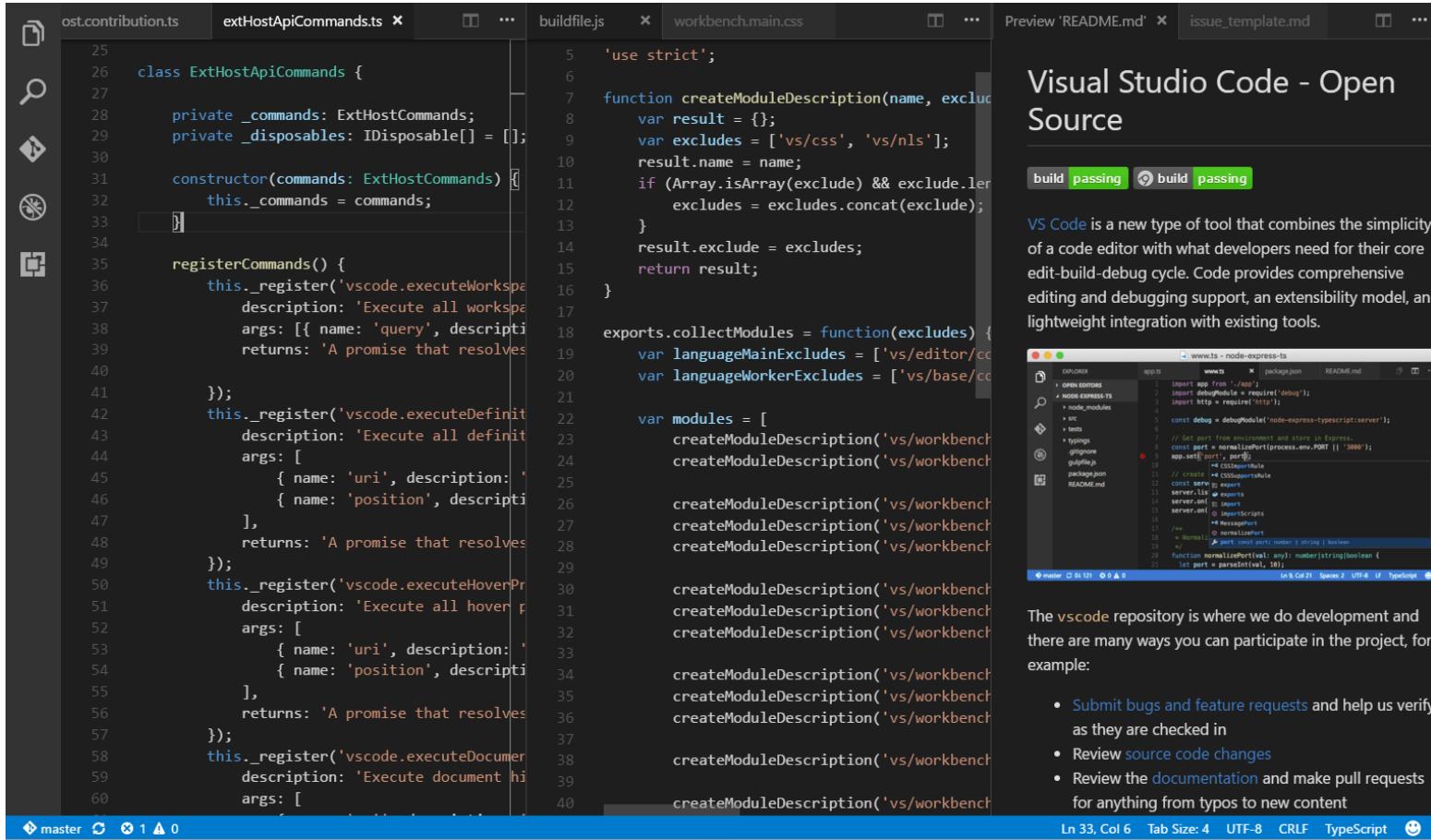
1. Go to portal.biohpc.swmed.edu -> Cloud Services -> [Web Visualization](#)
2. Open a terminal
3. module load vscode 1.66.1
4. code

User Interface

Basic layout



Side by side editing



The screenshot shows the Visual Studio Code interface with multiple tabs open in a split editor view. The tabs include:

- `ost.contribution.ts`
- `extHostApiCommands.ts`
- `buildfile.js`
- `workbench.main.css`
- `Preview 'README.md'`
- `issue_template.md`

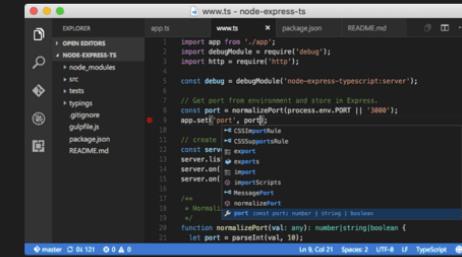
The code editor displays several files, including `extHostApiCommands.ts` and `buildfile.js`, which contains a function for creating module descriptions. The status bar at the bottom indicates the current file is `www.ts - node-express-ts`, with line 33, column 6.

- Click the **Split Editor** button
- Or **Alt+Click** on a file in the Explorer
- Or drag and drop a tab

Visual Studio Code - Open Source

[build](#) passing [build](#) passing

VS Code is a new type of tool that combines the simplicity of a code editor with what developers need for their core edit-build-debug cycle. Code provides comprehensive editing and debugging support, an extensibility model, and lightweight integration with existing tools.

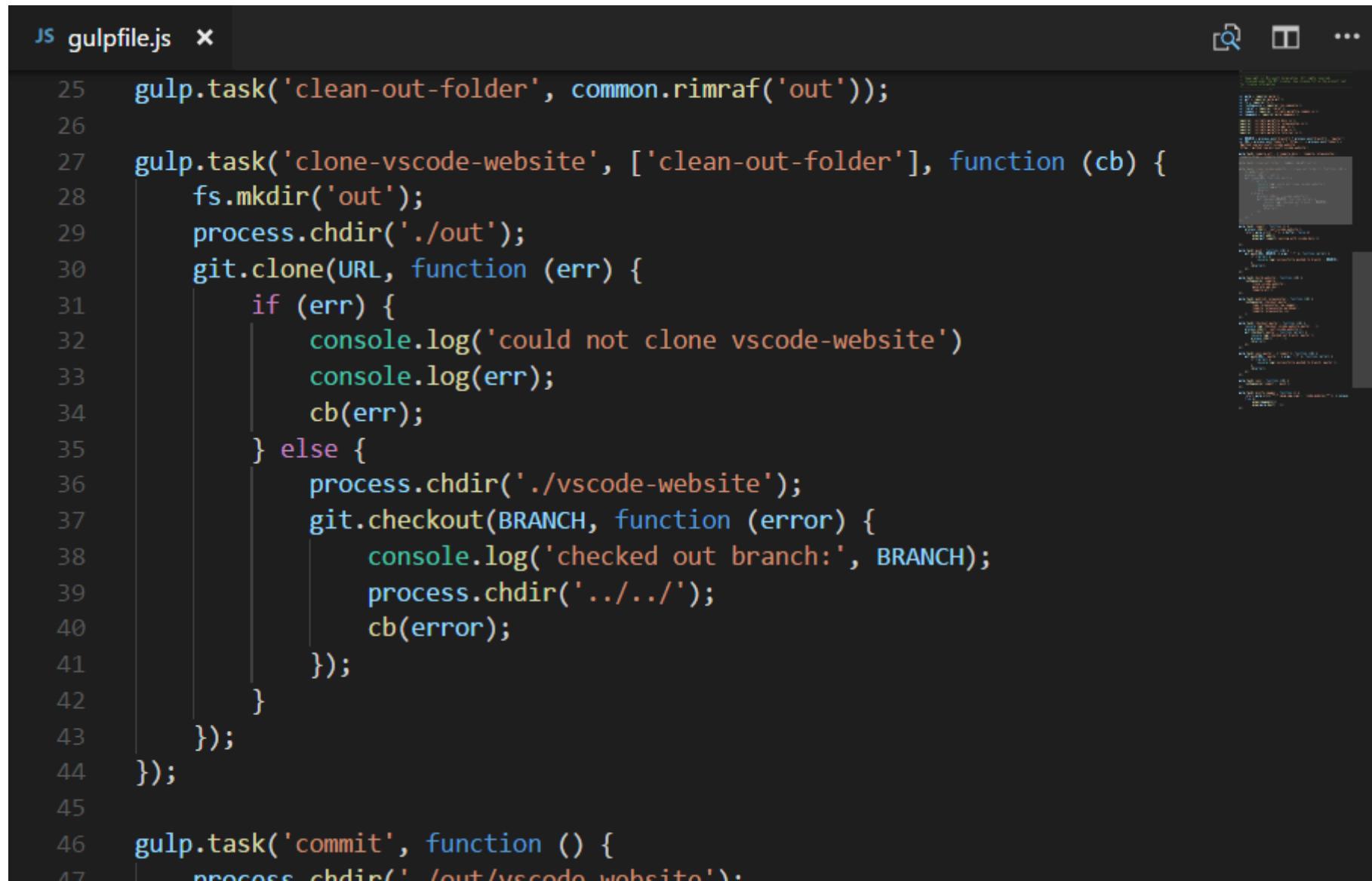


The screenshot shows the Visual Studio Code GitHub repository page. It includes sections for `EXPLORER`, `app.ts`, `www`, and `package.json`. The `app.ts` file is shown with code related to port normalization and server configuration.

The `vscode` repository is where we do development and there are many ways you can participate in the project, for example:

- Submit bugs and feature requests and help us verify as they are checked in
- Review source code changes
- Review the documentation and make pull requests for anything from typos to new content

Minimap



A screenshot of the Visual Studio Code (VS Code) interface. The main editor area shows a Gulpfile.js file with the following code:

```
JS gulpfile.js x

25  gulp.task('clean-out-folder', common.rimraf('out'));

26

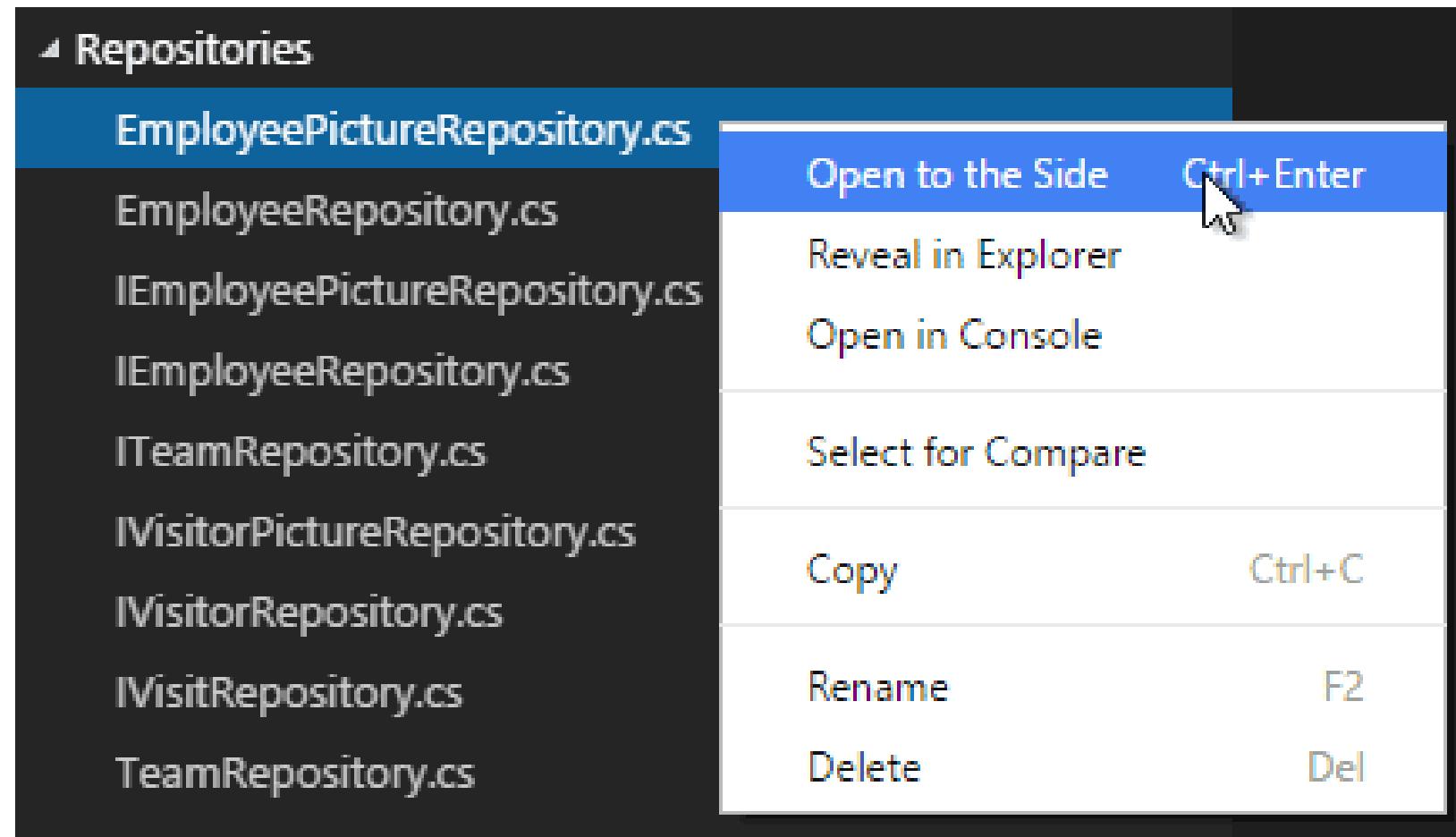
27  gulp.task('clone-vscode-website', ['clean-out-folder'], function (cb) {
28      fs.mkdir('out');
29      process.chdir('./out');
30      git.clone(URL, function (err) {
31          if (err) {
32              console.log('could not clone vscode-website')
33              console.log(err);
34              cb(err);
35          } else {
36              process.chdir('./vscode-website');
37              git.checkout(BRANCH, function (error) {
38                  console.log('checked out branch:', BRANCH);
39                  process.chdir('../..');
40                  cb(error);
41              });
42          }
43      });
44  });

45

46  gulp.task('commit', function () {
47      process.chdir('./out/vscode-website');

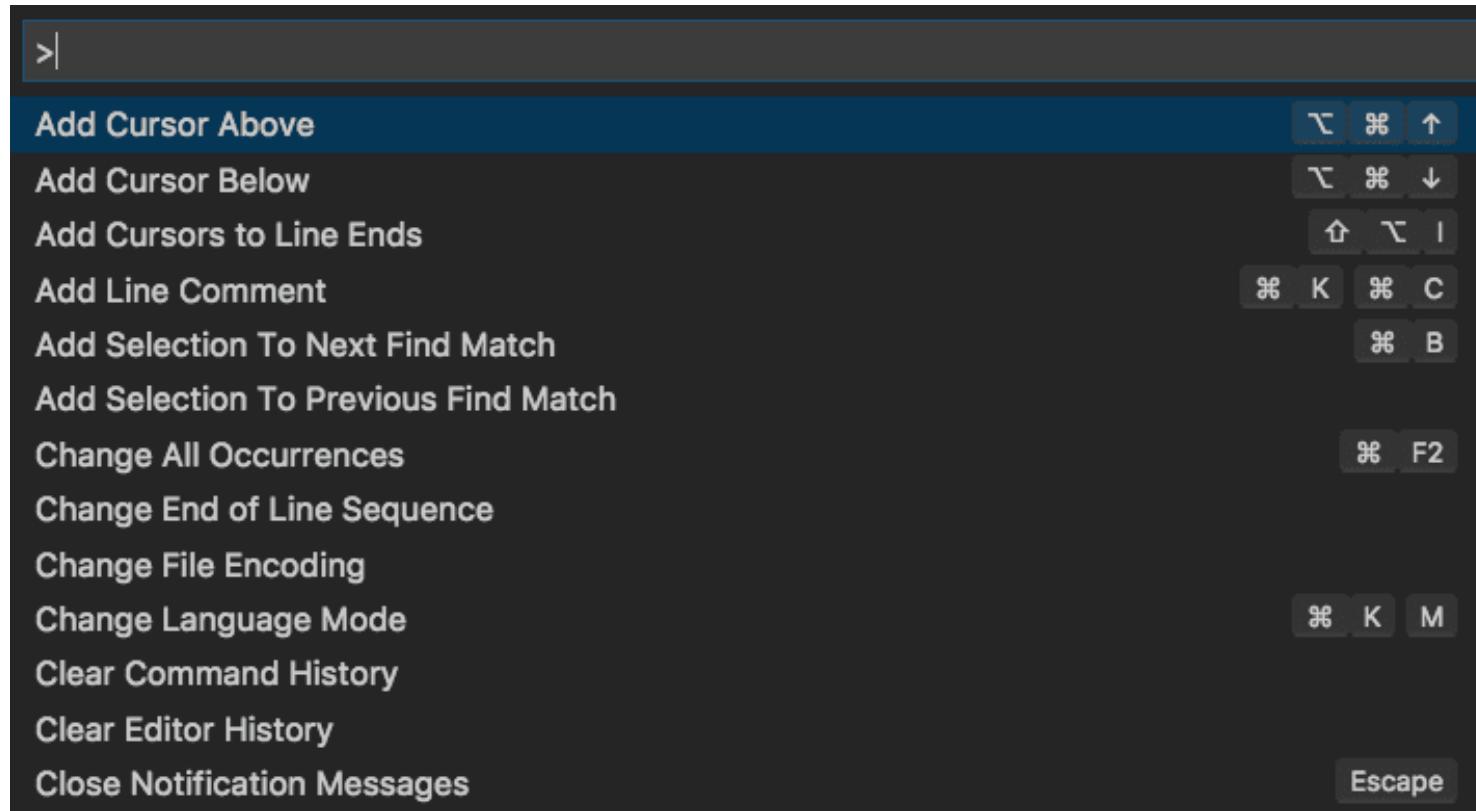
The code defines two Gulp tasks: 'clean-out-folder' and 'clone-vscode-website'. The 'clone-vscode-website' task runs after 'clean-out-folder'. It creates a 'out' directory, changes the working directory to 'out', clones a repository from 'URL' into 'out/vscode-website', and then checks out a specific branch 'BRANCH'. Finally, it changes back to the parent directory. The 'commit' task is also defined but has no implementation shown.
```

Explorer



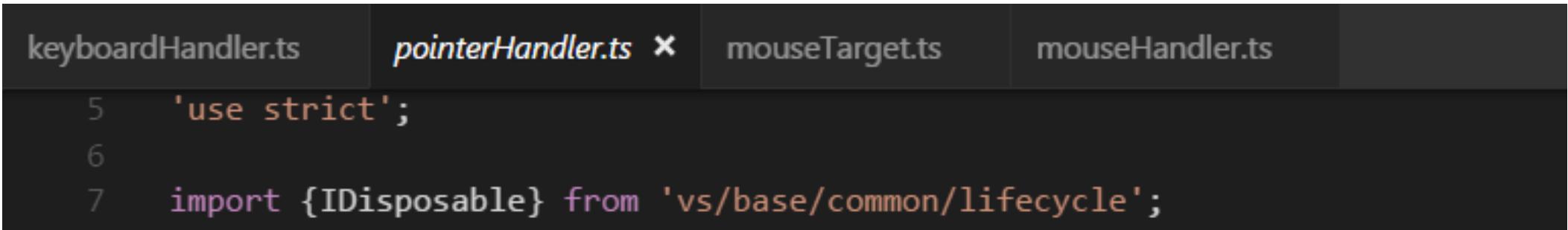
A high-level overview of the source code, which is useful for quick navigation and code understanding.

Command Palette



The Command Palette provides access to many commands. You can execute editor commands, open files, search for symbols, and see a quick outline of a file.

Preview mode

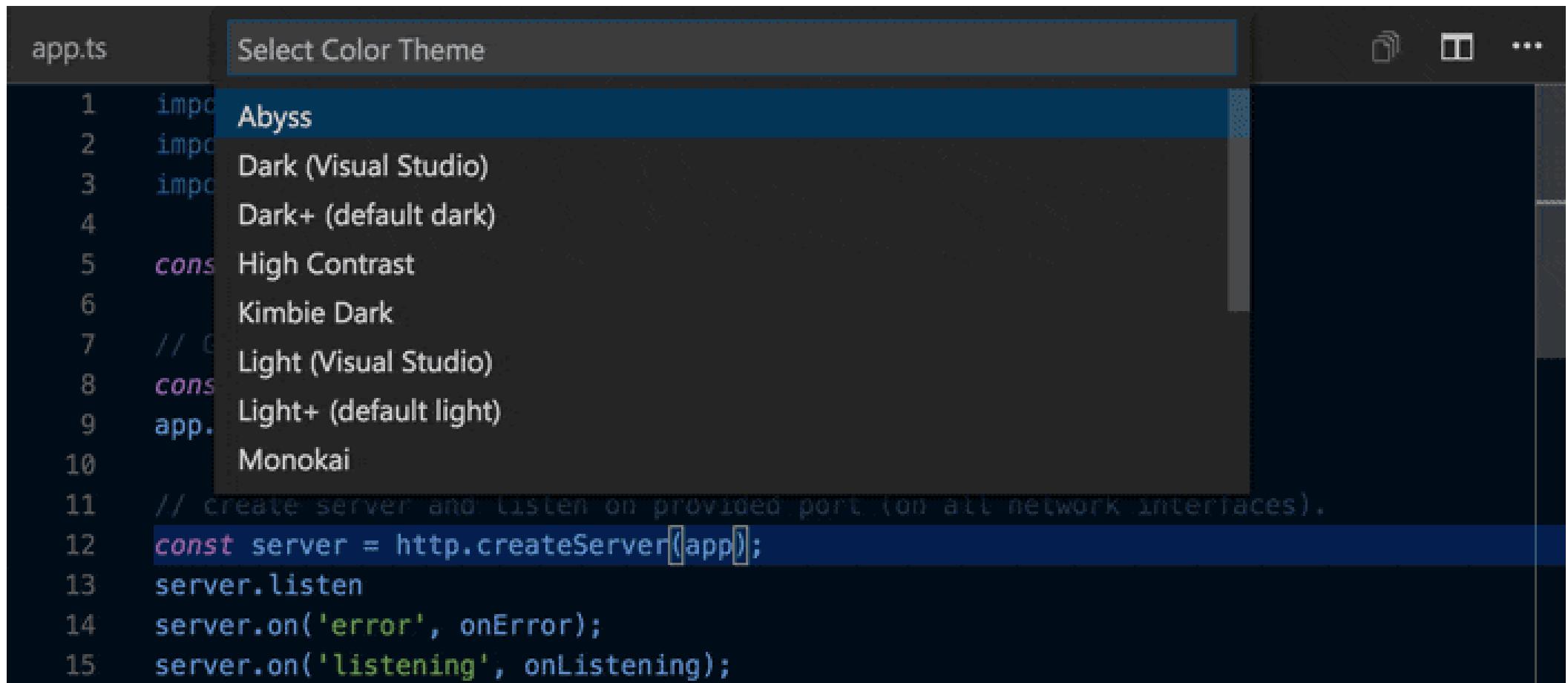


```
pointerHandler.ts ×
5  'use strict';
6
7  import {IDisposable} from 'vs/base/common/lifecycle';
```

When you single-click or select a file in the Explorer, it is shown in a preview mode and reuses an existing Tab. This is useful if you are quickly browsing files and don't want every visited file to have its own Tab. When you start editing the file or use double-click to open the file from the Explorer, a new Tab is dedicated to that file.

Preview mode is indicated by *italics* in the Tab heading.

Selecting the Color Theme



The screenshot shows a code editor window with a dark theme. A dropdown menu titled "Select Color Theme" is open, listing several color schemes:

- Abyss
- Dark (Visual Studio)
- Dark+ (default dark)
- High Contrast
- Kimbie Dark
- Light (Visual Studio)
- Light+ (default light)
- Monokai

The code editor's interface includes a status bar at the bottom with file information: "app.ts 15 lines 100%". The main code area shows a snippet of TypeScript code:

```
1 import 'express';
2 import 'path';
3 import 'fs';
4
5 const app = express();
6
7 // Create a new express application.
8
9 app.get('/', (req, res) => {
10   res.send('Hello World!');
11 })
12
13 // Create a server and listen on provided port (on all network interfaces).
14 const server = http.createServer(app);
15
16 server.listen(3001, () => {
17   console.log(`Server listening on port 3001`);
18 });
19
20 // Error handler
21 const onServerError = (err) => {
22   console.error(err);
23   if (err.status === undefined) {
24     err.status = 500;
25   }
26   res.status(err.status).send(`Error: ${err.message}`);
27 };
28
29 // Listening event handler
30 const onListening = () => {
31   const address = server.address();
32   const port = typeof address === 'number' ? address : address.port;
33   console.log(`Listening on ${port}`);
34 };
35
36 // Error event handler
37 const onError = (err) => {
38   console.error(`Error: ${err.message}`);
39 };
40
41 // Start the server
42 server.on('error', onServerError);
43 server.on('listening', onListening);
```

Open the Color Theme picker:

- File -> Preferences -> Theme -> Color Theme
- Or **Ctrl+K Ctrl+T**

The Editor

Settings

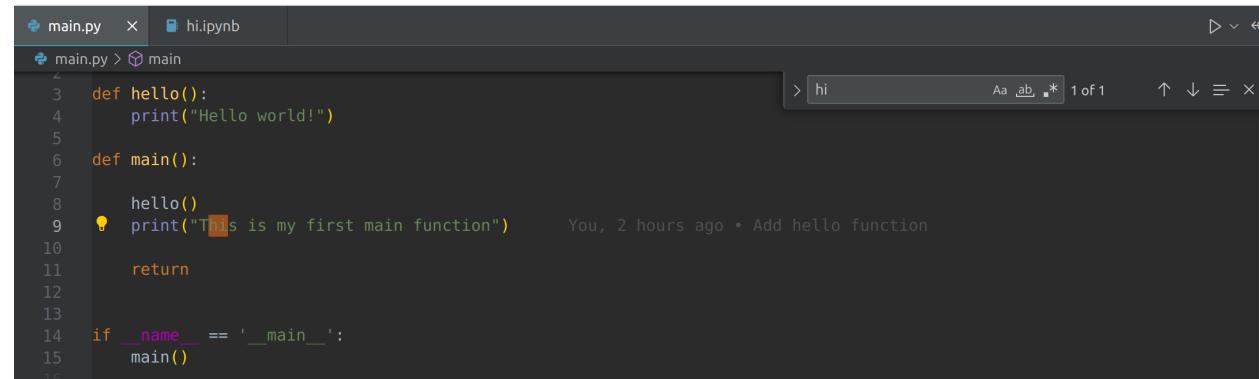
- Settings
 - File → Preferences → Settings
 - Or **Ctrl+,**
 - Global settings: `~/.config/Code/User/settings.json`
 - Project settings: `.vscode/settings.json`
- Settings Sync
 - File → Preferences → Settings Sync
- Zen mode Zen Mode lets you focus on your code by hiding all UI except the editor, going to full screen and centering the editor layout.
 - View → Appearance → Zen mode
 - **Ctrl+K Z**
 - **Double Esc** exits Zen Mode

Multiple selections

```
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 }
```

Find and replace

- **Ctrl+F** ~ Find string in editor or find file in explorer
- Enter or Shift+Enter to navigate



A screenshot of a code editor window titled "main.py". The code contains a function "hello" that prints "Hello world!" and a function "main" that calls "hello" and prints a message. A search bar at the top right shows the text "hi". Below the code, a status bar indicates "Aa ab.* 1 of 1".

```
main.py
3 def hello():
4     print("Hello world!")
5
6 def main():
7
8     hello()
9     print("This is my first main function")    You, 2 hours ago • Add hello function
10
11     return
12
13
14 if __name__ == '__main__':
15     main()
```

- **Ctrl+Shift+F** ~ Find string across files

Folding and unfolding

```
364 ┏    private onCursorSelectionChanged(): void {
365 ┏        if (this.state === State.Hidden) {
366            return;
367        }
368
369        this.editor.layoutContentWidget(this);
370    }
371
372 ┏    private onEditorBlur(): void { ...
373    }
374
375
376 ┏    private onListSelection(e: ISelectionChangeEvent<CompletionItem>): void {
377        if (!e.elements.length) {
378            return;
379        }
380    }
381
382
383
```

Indentation & file encoding

VS Code analyzes your open file and determines the indentation used in the document. The auto-detected indentation overrides your default indentation settings. The detected setting is displayed on the right side of the Status Bar.

Ln 1, Col 1

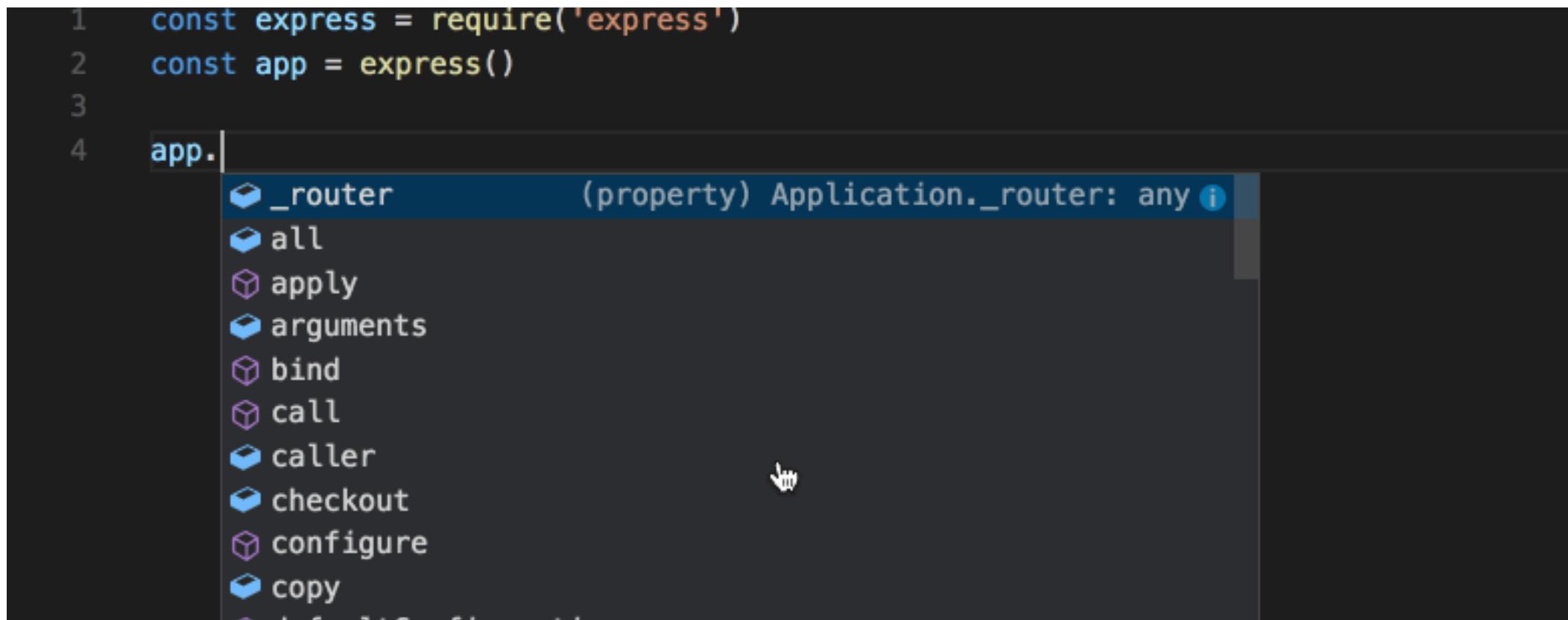
Tab Size: 4

UTF-8

CRLF

IntelliSense

IntelliSense is a general term for various code editing features including: code completion, parameter info, quick info, and member lists. IntelliSense features are sometimes called by other names such as “code completion”, “content assist”, and “code hinting.”



The screenshot shows a code editor window with the following code:

```
1 const express = require('express')
2 const app = express()
3
4 app.
```

An IntelliSense dropdown menu is open at the end of the line 'app.'. The menu lists several properties of the 'app' object, each with a small icon and a tooltip:

- _router (property) Application._router: any
- all
- apply
- arguments
- bind
- call
- caller
- checkout
- configure
- copy

A cursor arrow points to the 'configure' option in the list.

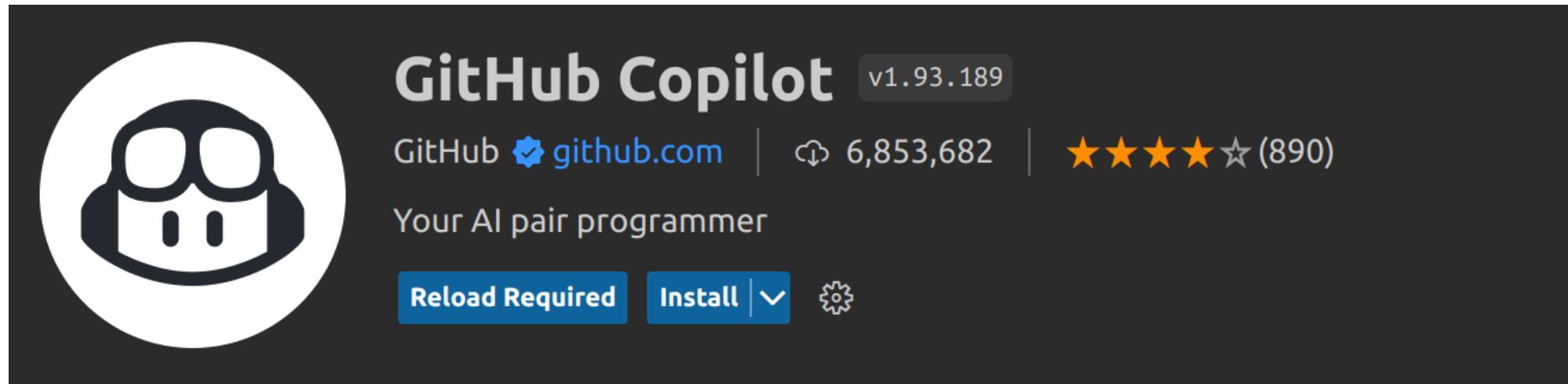
Snippets

Code snippets are templates that make it easier to enter repeating code patterns, such as loops or conditional-statements.

In Visual Studio Code, snippets appear in IntelliSense **Ctrl+Space** mixed with other suggestions, as well as in a dedicated snippet picker (Insert Snippet in the Command Palette).

AI tools in VS Code

The GitHub Copilot extension is an AI pair programmer tool that helps you write code faster and smarter. You can use the Copilot extension in VS Code to generate code, learn from the code it generates, and even configure your editor.



Source control

Using Git in VS Code

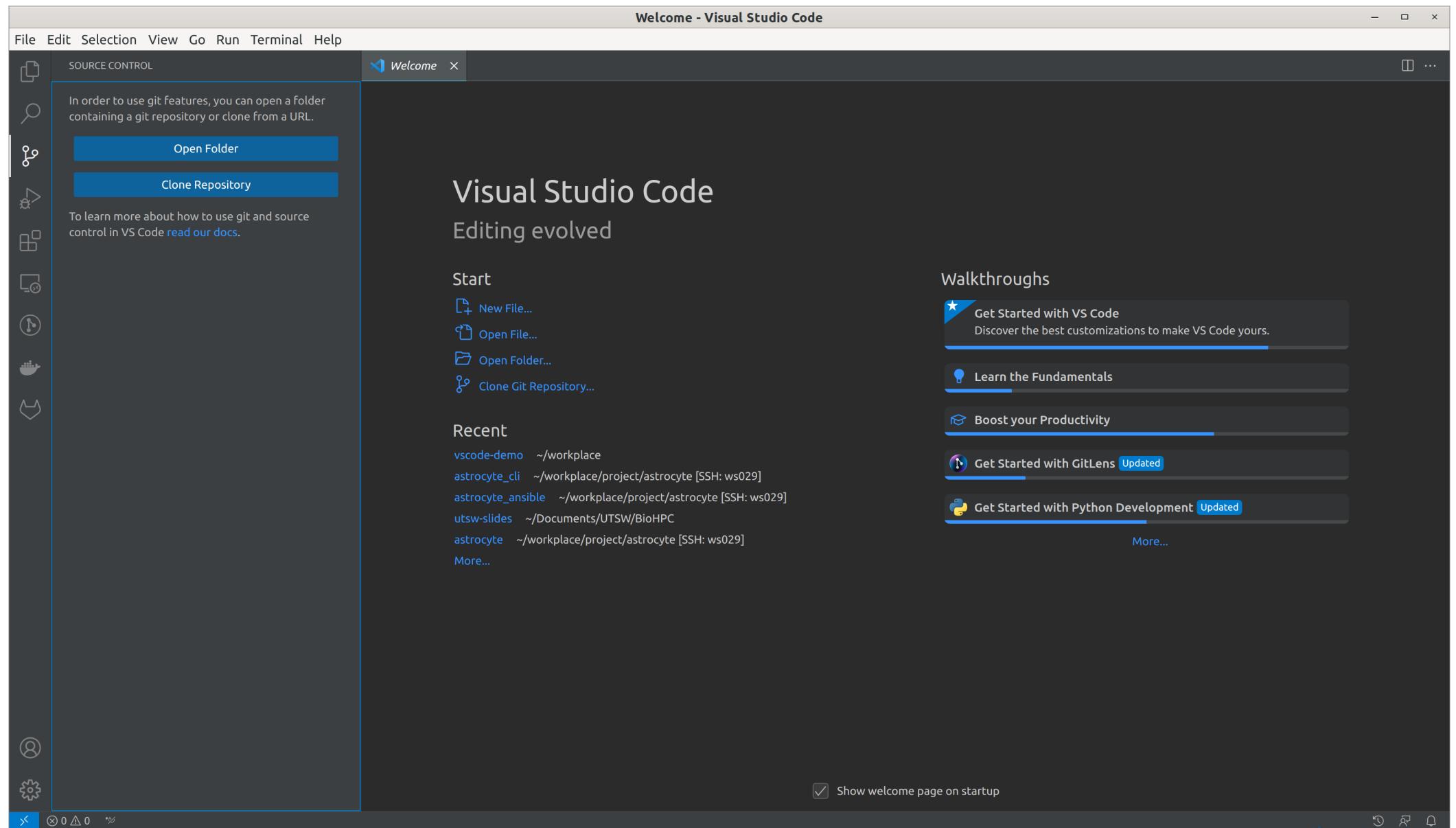
Visual Studio Code has integrated source control management (SCM) and includes Git support out-of-the-box. Many other source control providers are available through extensions.

Git documentations:

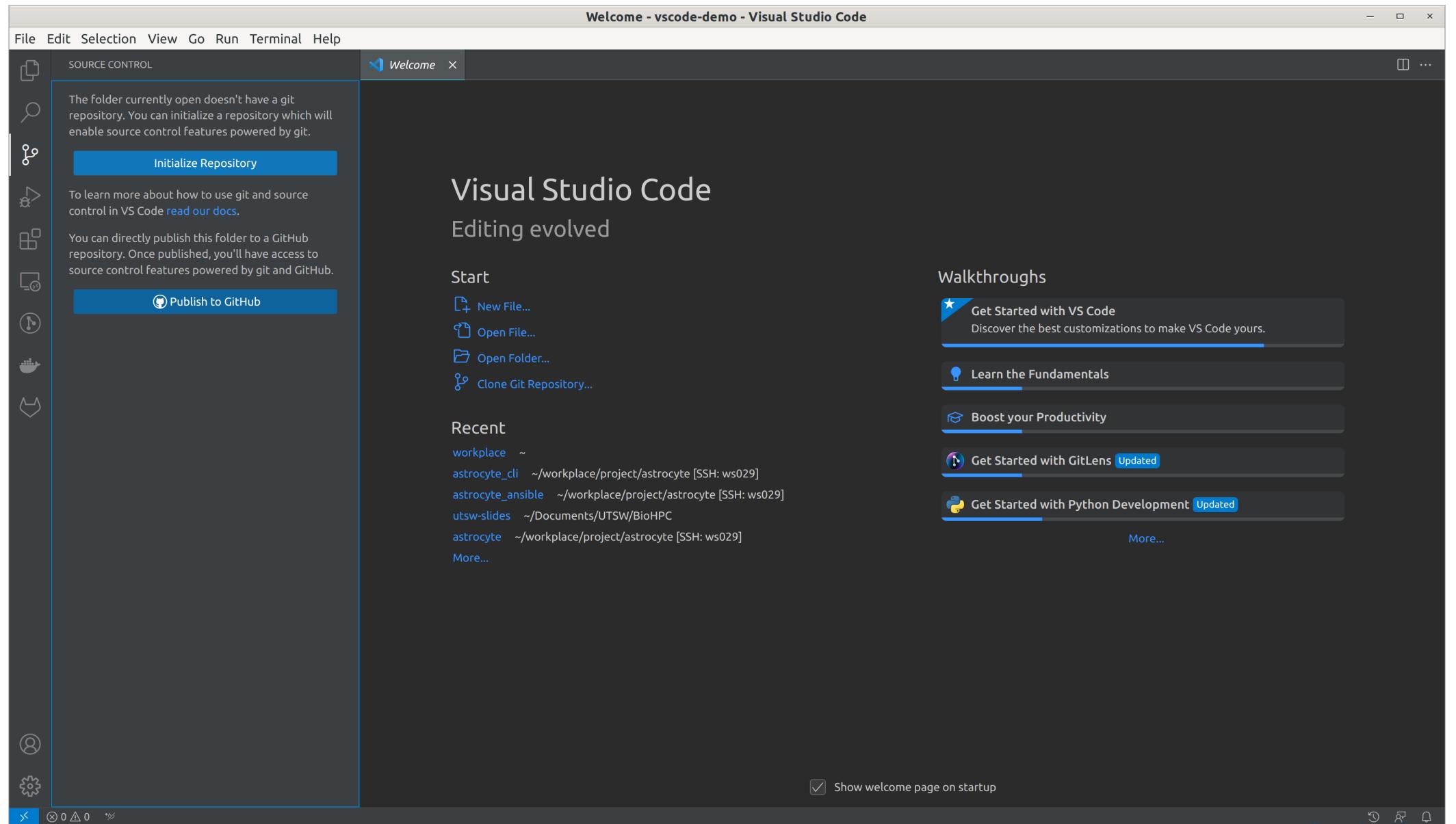
- [Cheat Sheet](#)
- [Git-SCM website](#)
- [Manual](#)
- [Book](#)
- [Videos](#)

Note: make sure you have Git installed on your computer.

Open folder or clone repo locally



Initialize the local repo



Create new files

The screenshot shows the Visual Studio Code interface with a dark theme. The title bar reads "main.py - vscode-demo - Visual Studio Code". The menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. On the left is the Source Control sidebar, which shows a repository named "vscode-demo" with a commit message "Message (Ctrl+Enter to commit on "main")" and a "Commit" button. Below the commit message, there are sections for Changes, .gitignore, and main.py. The main editor area displays the following code:

```
import os
def main():
    print("This is the main function")
    return
if __name__ == '__main__':
    main()
```

To the right of the editor are two other tabs: ".gitignore" and ".history". The bottom status bar shows file information: Ln 6, Col 11, Spaces: 4, UTF-8, LF, MagicPython, 3.10.6 64-bit, Spell, and a search icon.

Stage changes & commit

The screenshot shows the Visual Studio Code interface with the following details:

- Title Bar:** main.py - vscode-demo - Visual Studio Code
- File Menu:** File Edit Selection View Go Run Terminal Help
- SOURCE CONTROL Panel:** Shows a repository named "vscode-demo" with a commit message "Message (Ctrl+Enter to commit on "main")". A "Commit" button is highlighted.
- Staged Changes:** main.py (1 staged change)
- Changes:** .gitignore (1 staged change)
- Code Editor:** The main editor window displays the content of `main.py`:

```
import os

def main():

    print("This is the main function")
    return

if __name__ == '__main__':
    main()
```
- .gitignore:** The right-hand panel shows the content of `.gitignore`:

```
.history
```
- Bottom Status Bar:** Ln 11, Col 1 Spaces: 4 UTF-8 LF MagicPython 3.10.6 64-bit ✓ Spell Git Graph

Stage changes & commit

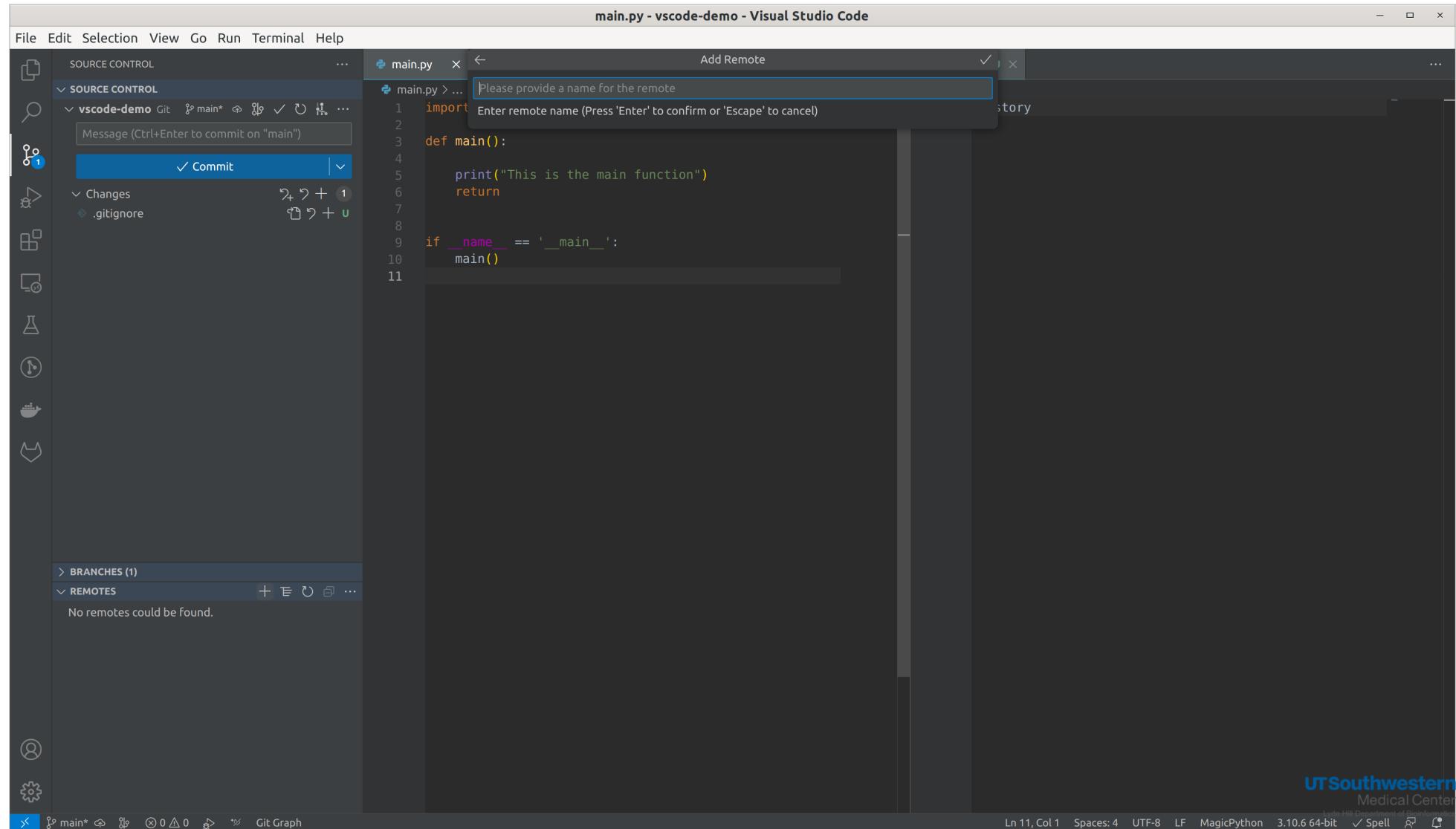
The screenshot shows the Visual Studio Code interface with the title bar "main.py - vscode-demo - Visual Studio Code". The menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The left sidebar is the Source Control view, showing a tree structure for the "vscode-demo" repository. It has a "Staged Changes" section where "main.py" is listed with a status of "A" (added). A prominent blue button labeled "✓ Commit" is visible. Below it, there's a message box with "Message (Ctrl+Enter to commit on "main")". The main editor area displays the Python code for "main.py":

```
1 import os
2
3 def main():
4     print("This is the main function")
5     return
6
7
8 if __name__ == '__main__':
9     main()
```

To the right of the editor is the ".gitignore" file, which contains ".history". The bottom status bar shows "Ln 11, Col 1" and other settings like "Spaces: 4", "UTF-8", "LF", "MagicPython", "3.10.6 64-bit", "Spell", and "Git Graph".

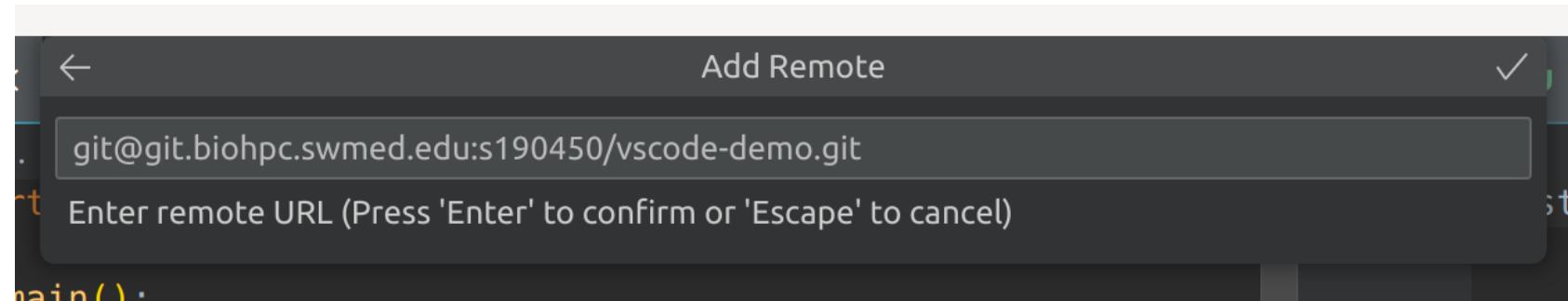
Add remote

- Add remote name



Add remote URL and confirm

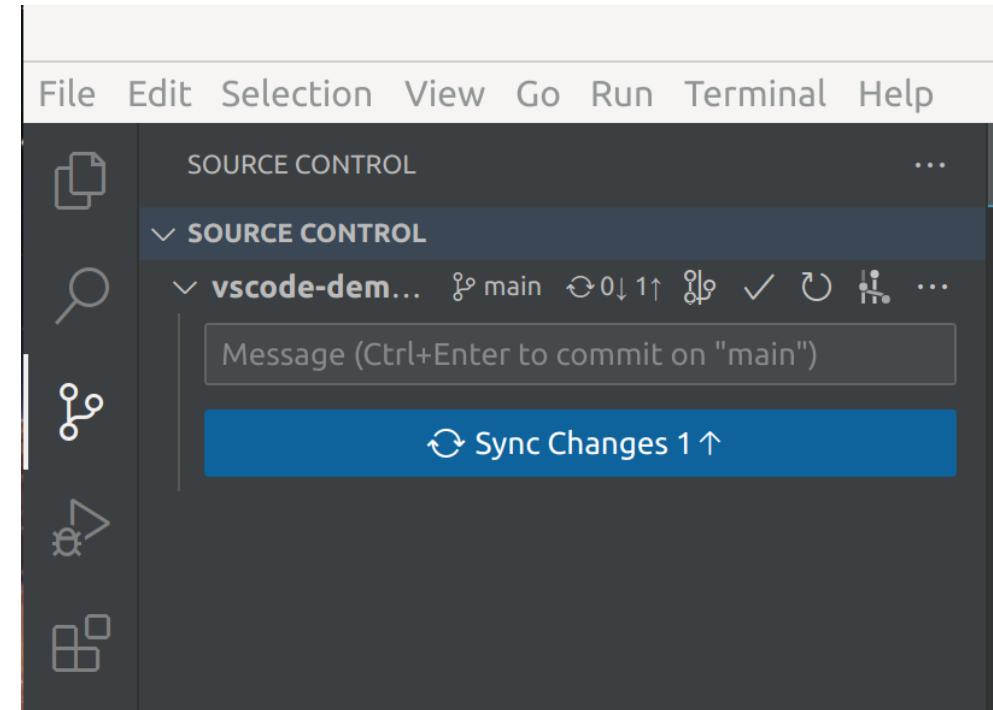
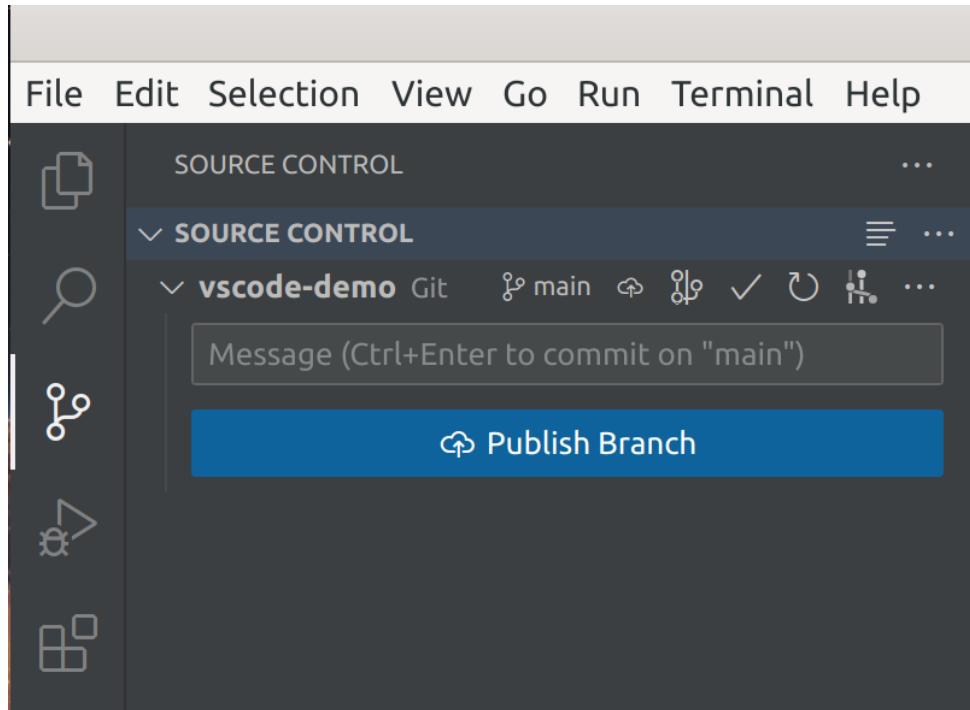
- Add remote URL



- Confirm add remote

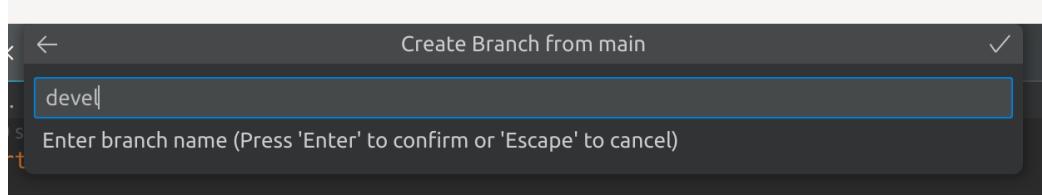
Push/Sync changes

- Push branch to remote
- Sync changes

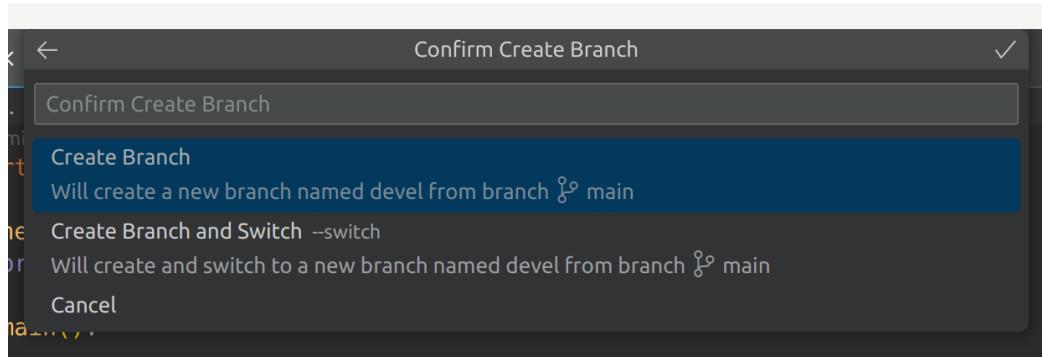


Working with branch

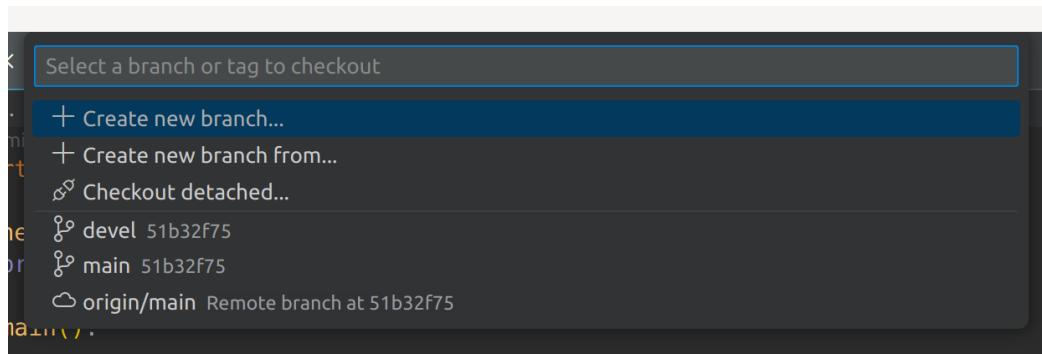
- Add a branch name



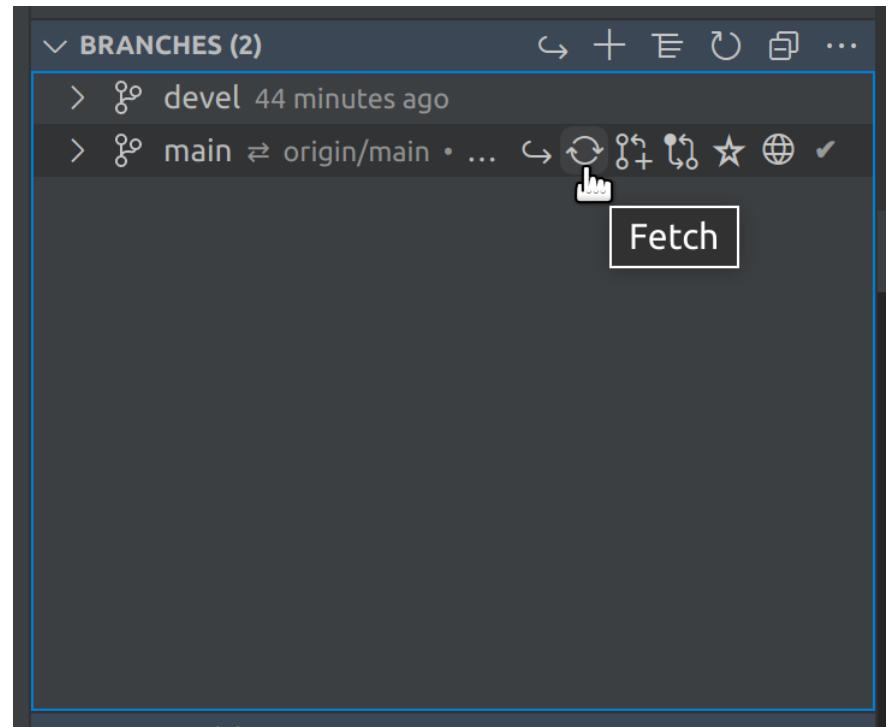
- Create branch



- Switch branch



- Fetch branch



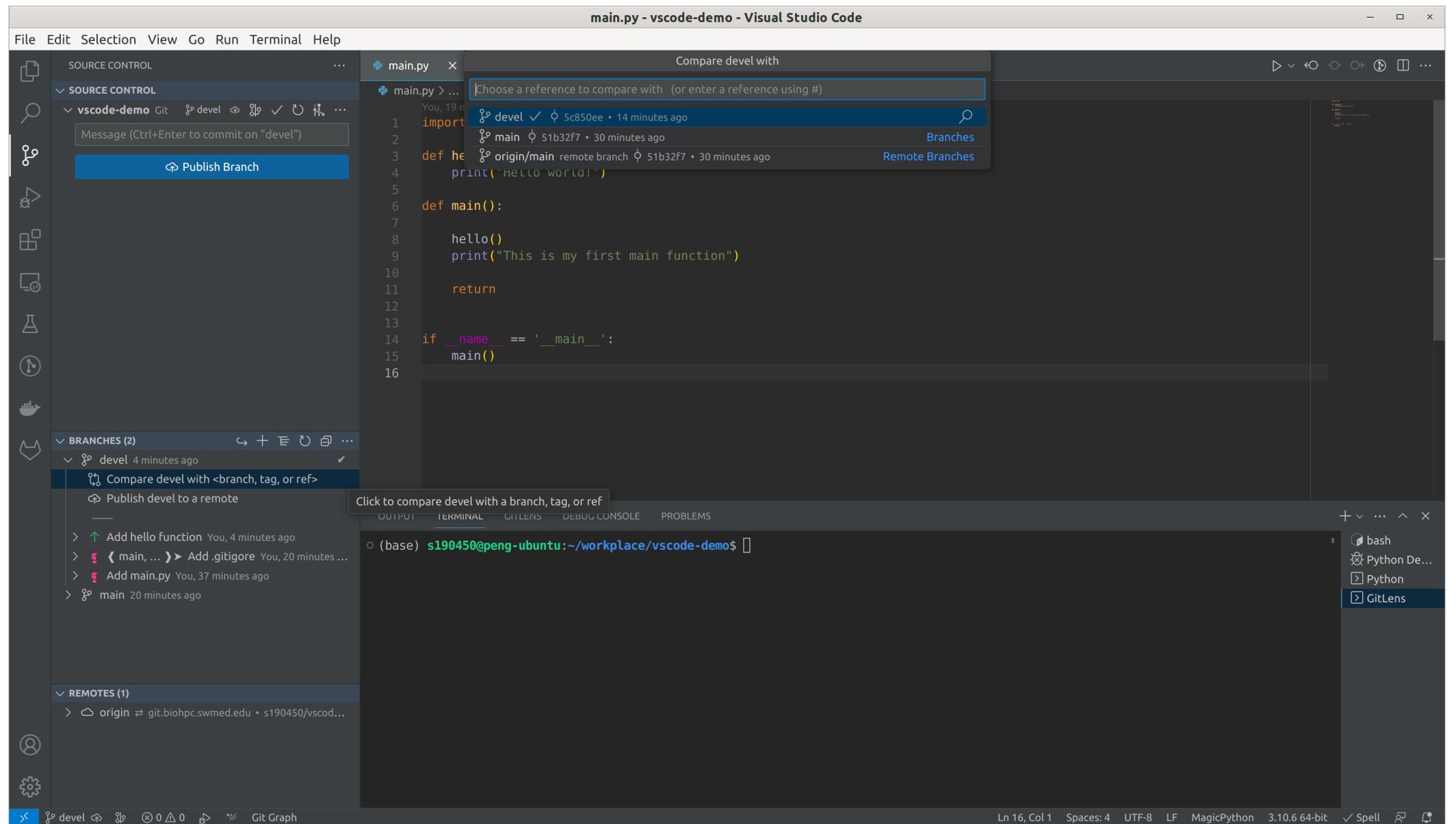
Add changes and commit to dev

The screenshot shows a Visual Studio Code interface with the following details:

- Title Bar:** main.py - vscode-demo - Visual Studio Code
- File Menu:** File Edit Selection View Go Run Terminal Help
- SOURCE CONTROL Panel:** Shows a repository named "vscode-demo" with a single file "main.py". A message box says "Message (Ctrl+Enter to commit on "main")". A "Commit" button is highlighted.
- Changes List:** Shows "main.py" with a green plus sign and a red minus sign, indicating changes.
- Code Editor:** Displays the content of main.py:

```
1 import os
2
3 def hello():
4     print("Hello world!")
5
6 def main():
7
8     hello()
9     print("This is my first main function")
10
11     return
12
13
14 if __name__ == '__main__':
15     main()
```
- Terminal:** Shows the command `(base) s190450@peng-ubuntu:~/workplace/vscode-demo$` and a dropdown menu with options: bash, Python De..., and Python.
- Bottom Status Bar:** Shows file path main*, line 16, column 1, spaces: 4, encoding UTF-8, LF, MagicPython, version 3.10.6 64-bit, and spell check status.

Compare branches



File differences

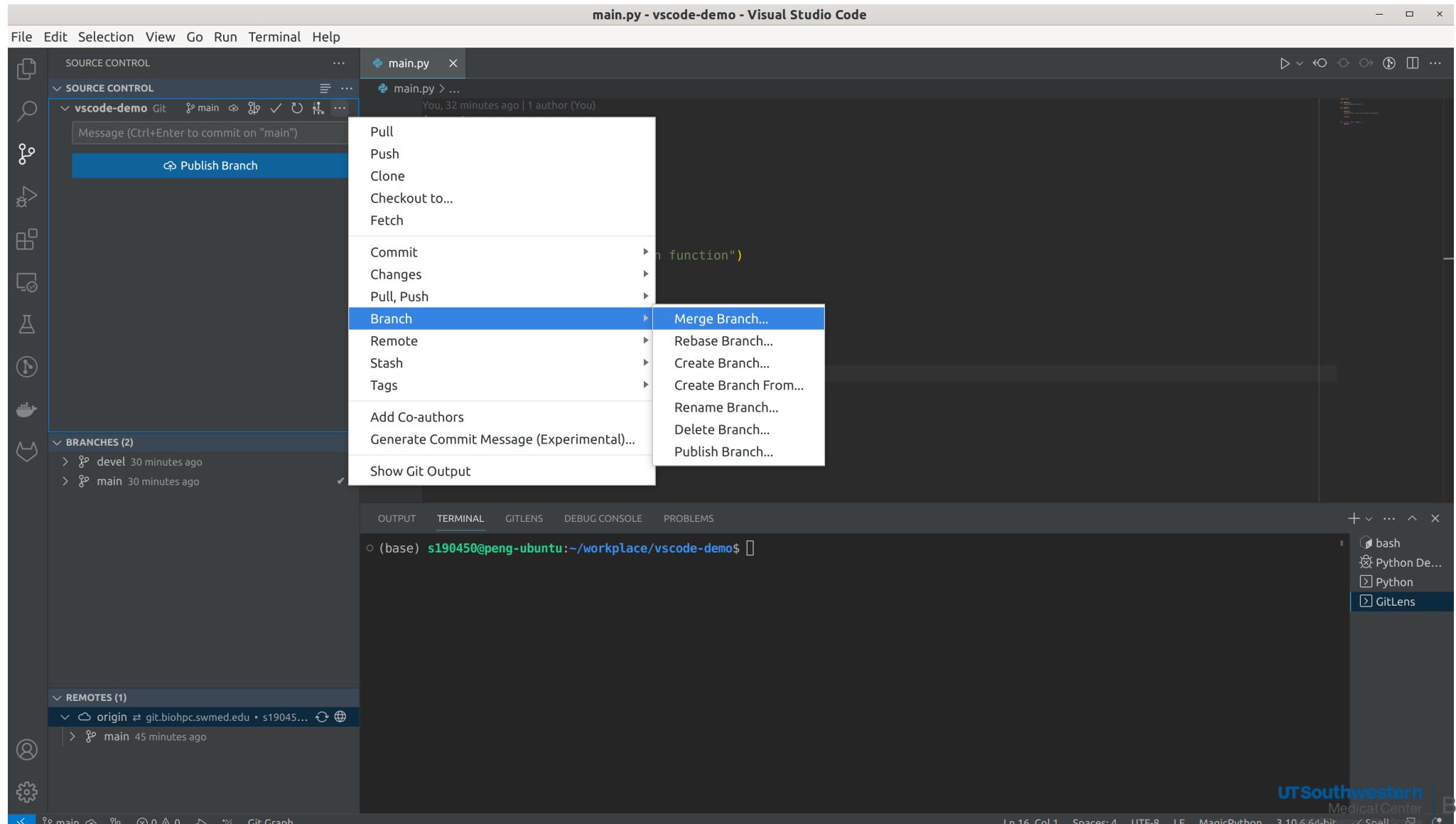
The screenshot shows a Visual Studio Code interface with the following details:

- Title Bar:** main.py (c4bc725) ↔ main.py (5c850ee) - vscode-demo - Visual Studio Code
- File Menu:** File Edit Selection View Go Run Terminal Help
- SOURCE CONTROL Panel:** A tree view showing the workspace structure:
 - vscode-demo Git (devel)
 - Message (Ctrl+Enter to commit on "devel")
 - Publish Branch (button)
- BRANCHES Panel:** A list of branches:
 - devel (4 minutes ago)
 - Compare devel with main
 - Behind 0 commits
 - Ahead 1 commit
 - 1 file changed: 6 additions (+), 1 deletion (-)
- Code Editor:** Two files are compared:
 - main.py (c4bc725):** Contains code to print the main function.
 - main.py (5c850ee):** Contains code to print a hello function and the main function.

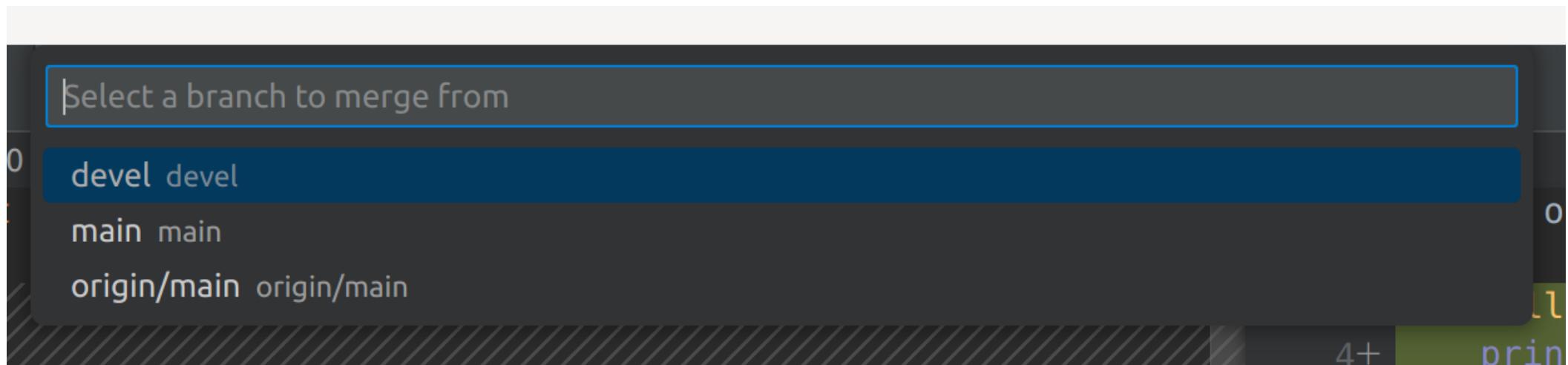
The diff highlights changes in line 3 and line 5. Line 3 has been deleted, and line 5 has been added.
- Terminal:** (base) s190450@peng-ubuntu:~/workplace/vscode-demo\$
- GitLens Sidebar:** Shows icons for bash, Python Development, Python, and GitLens.
- Bottom Status Bar:** dev (devel) 0 0 ▲ 0 ⌂ ⌂ Git Graph You, 15 minutes ago Ln 3, Col 1 Spaces: 4 UTF-8 MagicPython 3.10.6 64-bit Spell

Merge branches

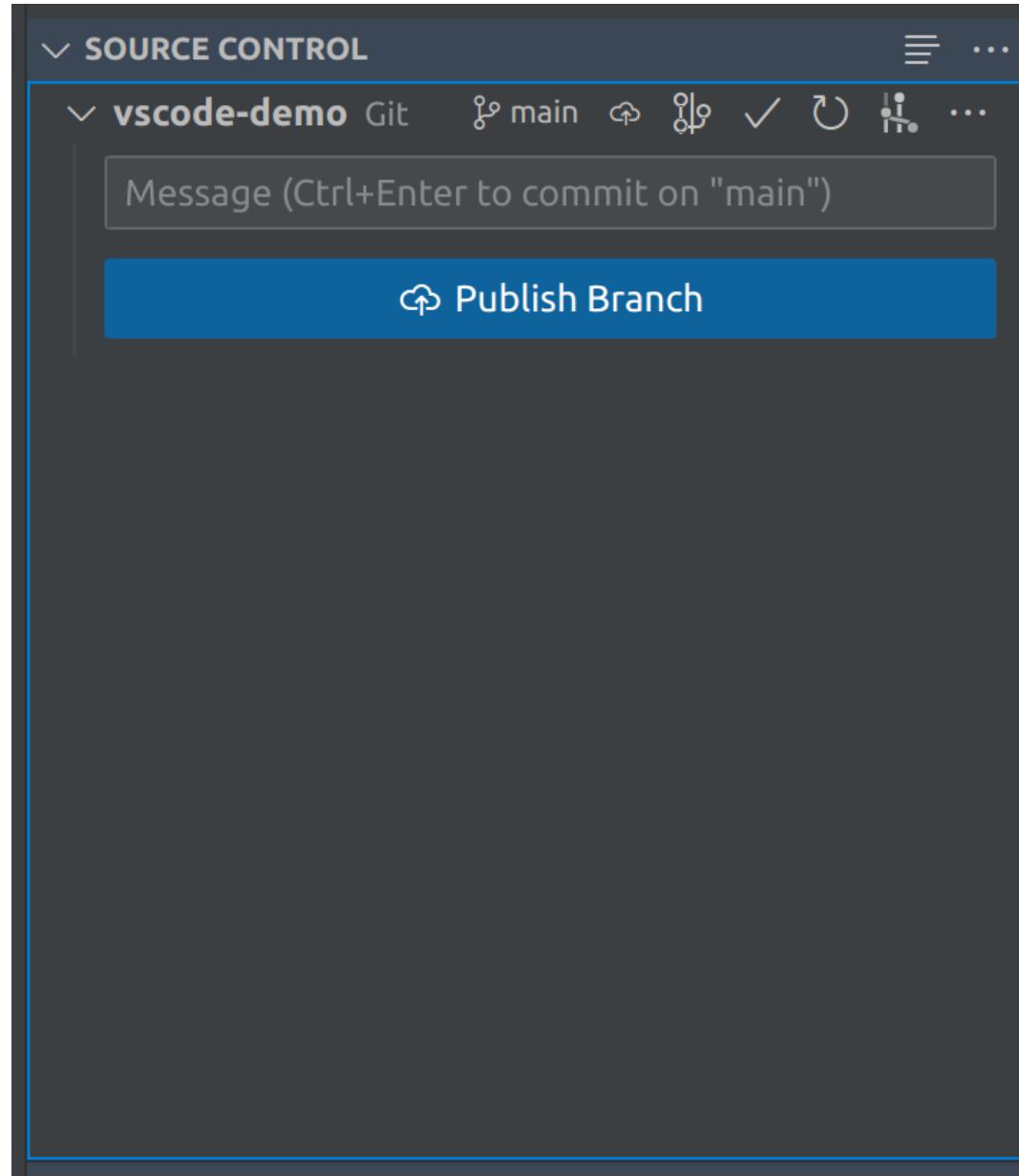
Note: Switch to the target branch before starting to merge



Select a branch to merge from



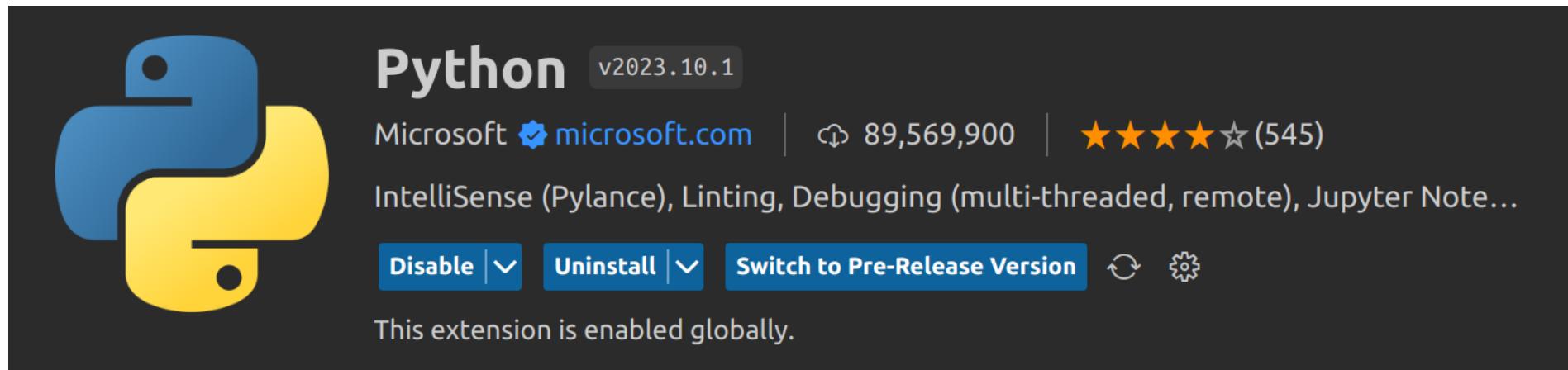
Publish merged branch



Work with Python

Prerequisites

- Python 3
- VS Code Python extension



- Jupyter extension (optional)



Jupyter

v2023.5.1101742258

Microsoft microsoft.com

67,741,555

(288)

Jupyter notebook support, interactive programming and computing that supports...

[Disable](#)

[Uninstall](#)

[Switch to Pre-Release Version](#)



This extension is enabled globally.

Create a virtual environment

- Command Palette (**Ctrl+Shift+P**) -> Python: Create Environment

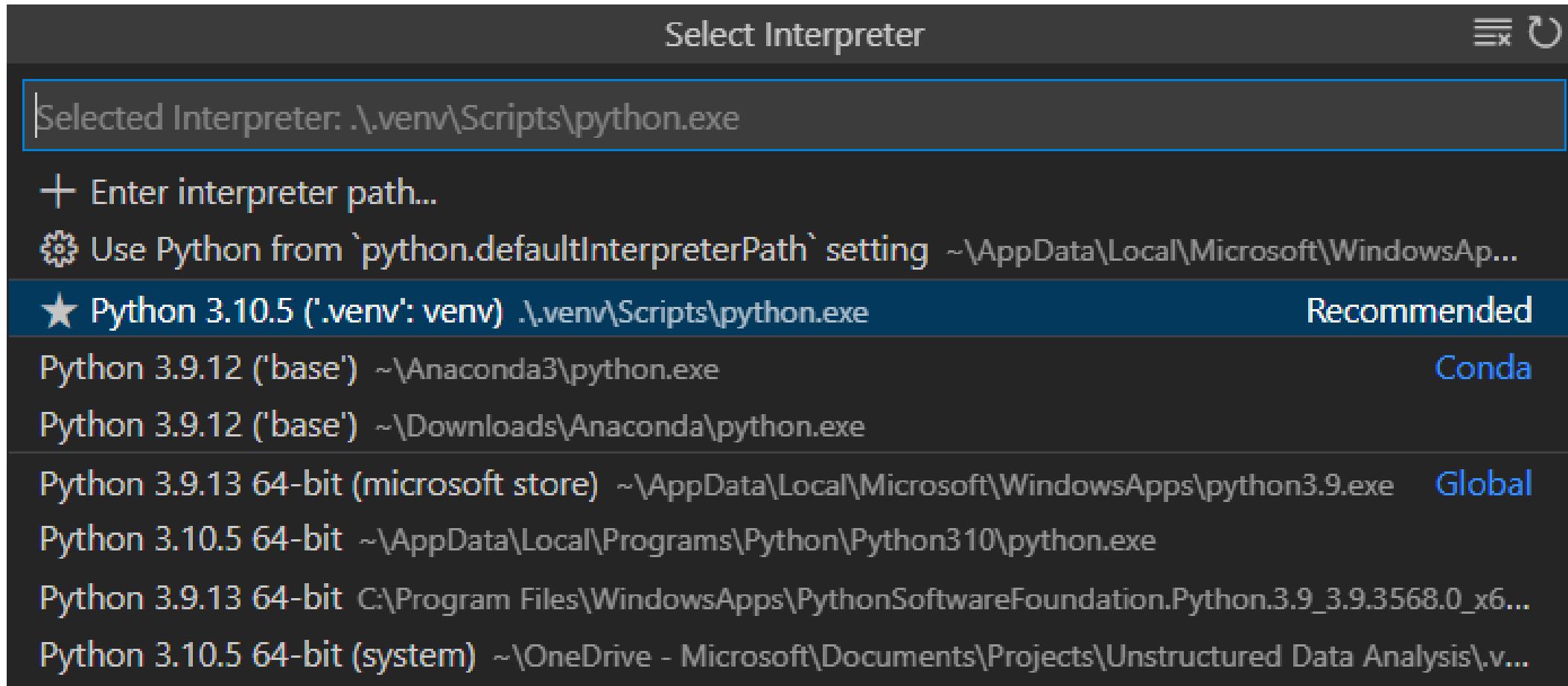
Select an environment type

Venv Creates a `venv` virtual environment in the current workspace

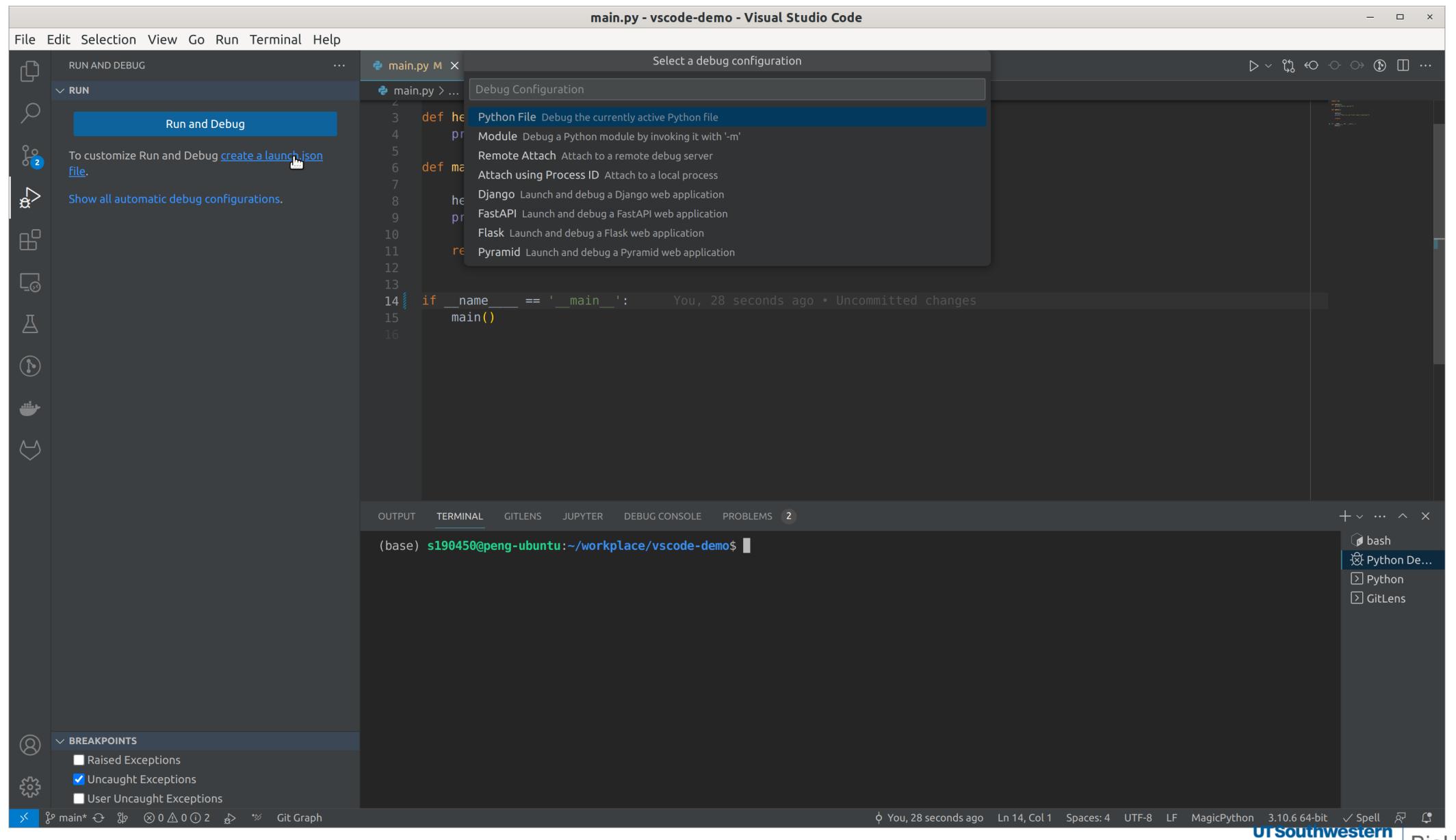
Conda Creates a `conda` Conda environment in the current workspace

Select a interpreter

- Command Palette (**Ctrl+Shift+P**) -> Python: Select Interpreter



Create debugging configuration



Debugging

main.py - vscode-demo - Visual Studio Code

File Edit Selection View Go Run Terminal Help

RUN AND DEBUG No Configurations ...

VARIABLES

- Locals
 - > special variables
 - > function variables
 - > os: <module 'os' from '/usr/lib/python3.10/os.p...
- Globals

WATCH

CALL STACK

name '__name__' is not defined

OUTPUT TERMINAL GITLENS JUPYTER DEBUG CONSOLE PROBLEMS 2

```

./debugpy/../debugpy/server/cli.py", line 284, in run_file
    runpy.run_path(target, run_name="__main__")
  File "/home/s190450/.vscode/extensions/ms-python.python-2023.10.1/pythonFiles/lib/python/debugpy/_vendored/pydevd/_pydevd_bundle/pyd
evd_runpy.py", line 321, in run_path
    return _run_module_code(code, init_globals, run_name,
  File "/home/s190450/.vscode/extensions/ms-python.python-2023.10.1/pythonFiles/lib/python/debugpy/_vendored/pydevd/_pydevd_bundle/pyd
evd_runpy.py", line 135, in _run_module_code
    run_code(code, mod_globals, init_globals,
  File "/home/s190450/.vscode/extensions/ms-python.python-2023.10.1/pythonFiles/lib/python/debugpy/_vendored/pydevd/_pydevd_bundle/pyd
evd_runpy.py", line 124, in _run_code
    exec(code, run_globals)
  File "/home/s190450/workplace/vscode-demo/main.py", line 14, in <module>
    if __name__ == '__main__':
NameError: name '__name__' is not defined. Did you mean: '__name__'?
(base) s190450@peng-ubuntu:~/workplace/vscode-demo$ cd /home/s190450/workplace/vscode-demo ; /usr/bin/env /bin/python3 /home/s190450/
.vscode/extensions/ms-python.python-2023.10.1/pythonFiles/lib/python/debugpy/adapter///../../../debugpy/launcher 52789 -- /home/s190450/wor
kplace/vscode-demo/main.py

```

BREAKPOINTS

- Raised Exceptions
- Uncaught Exceptions
- User Uncaught Exceptions

main* ↻ ⌂ 0 ▲ 0 ⌂ 2 ↺ ⌂ Git Graph

You, 28 seconds ago Ln 14, Col 1 Spaces: 4 UTF-8 LF MagicPython 3.10.6 64-bit ✓ Spell ⌂

UTSouthwestern Medical Center Lyda Hill Department of Bioinformatics BioHPC

Jupyter in VS Code

hi.ipynb - vscode-demo - Visual Studio Code

File Edit Selection View Go Run Terminal Help

RUN AND DEBUG ... RUN Run and Debug

To customize Run and Debug [create a launch.json file.](#)

Show all automatic debug configurations.

main.py M hi.ipynb U

hi.ipynb > Jupyter Notebook Editor > x = np.linspace(0, 20, 100)

+ Code + Markdown | Run All Clear All Outputs Restart | Variables Outline ...

base (Python 3.7.4)

import os
[1] ✓ 0.0s

os.listdir('.')
[2] ✓ 0.0s

... ['hi.ipynb', '.gitignore', '.history', 'main.py', '.git']

import matplotlib.pyplot as plt
import numpy as np
[3] ✓ 0.5s

x = np.linspace(0, 20, 100)
plt.plot(x, np.sin(x))
plt.show()
[4] ✓ 0.3s

1.00
0.75
0.50
0.25
0.00
-0.25
-0.50
-0.75
-1.00
0.0 2.5 5.0 7.5 10.0 12.5 15.0 17.5 20.0

BREAKPOINTS

- Raised Exceptions
- Uncaught Exceptions
- User Uncaught Exceptions

main* Git Graph

Ln 3, Col 11 Spaces: 4 LF Cell 5 of 6 ✓ Spell BioHPC

UT Southwestern Medical Center Lyda Hill Department of Bioinformatics

Acknowledgement

- Thank all BioHPC team members for their support.
- Please acknowledge our contribution by adding the following sentence to your paper:

This research was supported in part by the computational resources provided by the BioHPC supercomputing facility located in the Lyda Hill Department of Bioinformatics, UT Southwestern Medical Center.

Questions?

Thanks for your
attention!