



Documentation

CSCI 5210 Software Engineering I, Fall 2024

Pod 8:

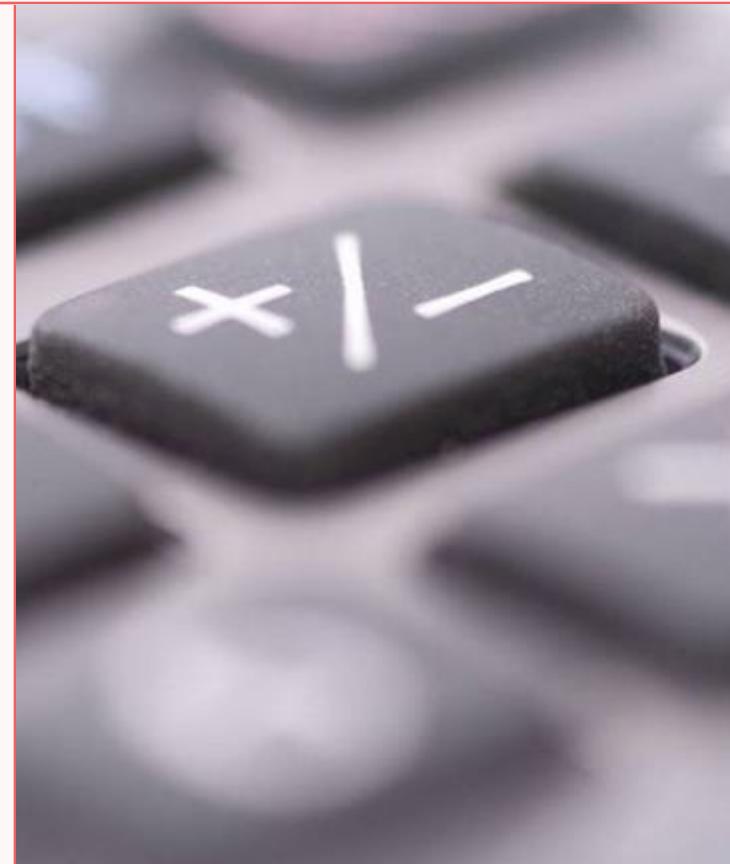
Nhi Pham

Victoria Pham

Jenny Spicer

Agenda

Introduction
README Files
Commit Messages
GitHub Pages & Wiki
Auto-Documentation



Introduction

- What is documentation?
- Why is documentation important?
- Types of documentation



<https://blog.codacy.com/code-documentation#:~:text=Share:,better%20documentation%20for%20your%20code>

README Files

- Introduction to project
- Communicates important information
- Especially crucial for open-source

The screenshot shows a code editor window displaying a README file template. The file is titled "README-Template.md". The content includes sections for "Project Title", "Description", "Getting Started", "Dependencies", "Installing", and "Executing program". A note at the bottom indicates "code blocks for commands".

```
README-Template.md
Project Title
Simple overview of use/purpose.

Description
An in-depth paragraph about your project and overview of use.

Getting Started

Dependencies
• Describe any prerequisites, libraries, OS version, etc., needed before installing program.
• ex. Windows 10

Installing
• How/where to download your program
• Any modifications needed to be made to files/folders

Executing program
• How to run the program
• Step-by-step bullets

code blocks for commands
```

README Components

- At a minimum:
 - Title
 - Description
 - Installation & Usage
 - How to Contribute
 - Licensing
 - Contact Information

```
# Project Title  
  
**Description:**  
[Brief overview of the project]  
  
**Installation:**  
[Prerequisites]  
[Installation steps]  
  
**Usage:**  
[Basic usage examples]  
[Advanced usage examples]  
  
**Contributing:**  
[Code of conduct]  
[Development process]  
[Issue tracker]  
  
**License:**  
[License type]  
[Link to license text]  
  
**Contact:**  
[Author/maintainer]  
[Contact information]  
  
**Additional Sections (optional):**  
[Changelog]  
[Acknowledgements]  
[FAQs]
```

GitHub-Flavored Markdown

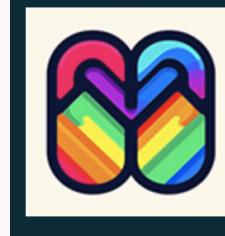
- Markdown is...
 - Intuitive
 - Portable
 - Efficient
 - Versatile

<https://www.markdownguide.org/basic-syntax/>

Hyperlink	This is an [example link](https://www.makeuseof.com)
Image	![Alt Text](http://example.com/image/path.png)
Ignore Markdown	Prefix Markdown characters with *backslashes* to ignore formatting.

Basic Elements		Extended Elements	
Format Type	Markdown Syntax	Format Type	Markdown Syntax
H1 to H6 Headings	# Heading Text ## Heading Text ### Heading Text #### Heading Text ##### Heading Text ###### Heading Text	Code (Inline)	`This is inline code` ~~
Italics	*This text is italicized*	Code (Block)	This is a block of code It supports multiple lines ~~
Bold	**This text is bold**	Strikethrough	~~This text is crossed out~~
Blockquote	> Blockquote paragraphs must have > a right-arrow bracket at the start > of every single line. > > Use a blank line for multiple paragraphs.	Hard Line Break	This is some text! This text is a new line, not a new paragraph
Unordered List	- Bullet list item - Bullet list item - Bullet list item - Use a two-space indent for nested lists	Table	First Header Second Header ----- ----- Content cell 1 Content cell 2 Content column 1 Content column 2
Ordered List	1. Bullet list item 2. Bullet list item 3. Bullet list item 1. Ordered lists can also be nested	Task Lists	Note: Preceding blank line is necessary. - [x] Completed task item - [] Unfinished task item - [] \{Optional\} Mark parentheses to be ignored
Mixed List	1. Can you mix list types? - Yes, you can!	Mention	You can mention @users and @teams on GitHub. Mainly useful when submitting or commenting on bugs and issues. :emojicode:
Horizontal Line	---	Emoji	

Easier README:



Markdown Preview Enhanced v0.8.14

Yiyi Wang | ⚡ 6,070,085 | ★★★★★ (116)

Markdown Preview Enhanced ported to vscode

[Disable](#) | [Uninstall](#) | Auto Update

The screenshot displays the Markdown Preview Enhanced extension within the Visual Studio Code interface. On the left, a preview window shows a blank README.md file with placeholder data for repository metrics like contributors, forks, stars, issues, and license, all labeled "REPO NOT FOUND". It also features a placeholder logo and links to "project_title", "project_description", and "Explore the docs". Below this, there are links to "View Demo", "Report Bug", and "Request Feature". A "Table of Contents" button is visible. On the right, the actual Markdown code is shown in a code editor pane, which is syntax-highlighted and includes code snippets for generating shields, logos, and structured data. The status bar at the bottom indicates "4 min read" and shows the current file path: "Users > jenspi > src > DocumentationWorkshop > docs > BLANK_README.md".

<https://marketplace.visualstudio.com/items?itemName=shd101wyy.markdown-preview-enhanced>

Easier README: readme.so

The screenshot shows the README editor interface. At the top, there's a dark header bar with a logo (# icon), a moon icon for dark mode, and a green "Download" button. Below the header, the left sidebar is titled "Sections" and contains a tree view of sections: Documentation (selected), Installation, Roadmap, and License. It also includes a search bar and buttons for "+ Custom Section", "Acknowledgements", "API Reference", "Appendix", and "Authors". The main editor area shows the following Markdown code:

```
## Documentation  
[Documentation] (https://linktodocumentation)
```

The right side shows the "Preview" of the generated README. The preview content includes:

- Documentation**
- Installation**

Install my-project with npm

```
npm install my-project  
cd my-project
```
- Roadmap**
 - Additional browser support
 - Add more integrations
- License**

MIT

A yellow circular icon with a coffee cup symbol is located at the bottom right of the preview area.

<https://readme.so/>

Keep Your README Up to Date!

- First point of contact
- Saves time
- Encourages collaboration
- Shows organization and professionalism
- Discoverability

README Examples

- **Ours:**

<https://github.com/bunnhimaybe/DocumentationWorkshop>

- **Explore in your free time:**

<https://github.com/matiassingers/awesome-readme>

Introduction to Commit Messages

- **What** is a commit message?

- Brief description(s)

- **Why** it is important

- Tracking changes
 - Collaboration
 - Debugging
 - Code Review

docs	Update _config.yml
images	update organization, wip (#3)
samples	update organization, wip (#3)
sphinx	fix Update README.rst
.gitmodules	update organization, wip (#3)
LICENSE.txt	feat Add MIT License
README.md	Edit README.md

Edit README.md

 actuallyvee committed 4 hours ago · ✓ 3 / 3

fix Update README.rst

 actuallyvee committed 4 hours ago · ✓ 3 / 3

Add Sphinx screenshots

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Add in Sphinx directory documentation

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Create README.rst

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Delete Sphinx directory

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Delete Sphinx HTML

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Delete Sphinx build

 actuallyvee committed 6 hours ago · ✘ 0 / 3

Update README.rst

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Update README.rst

 actuallyvee committed 6 hours ago · ✘ 0 / 3

Merge pull request #6 from bunnhimaybe/actuallyvee-patch-5

 actuallyvee committed 6 hours ago · ✓ 3 / 3

Add Sphinx README

 actuallyvee committed 6 hours ago

Best Practices

- Capitalization and Punctuation
- Imperative Action
- Commit Type
 - feat, fix, refactor, docs, style
- Length
- Content

COMMIT MESSAGES

12



97/05/20 21:31

Hussain Moradi authored 6 months ago

03 Aug, 2018 2 commits



97/05/12 23:29

Hussain Moradi authored 6 months ago



97/05/12 14:04

Hussain Moradi authored 6 months ago

02 Aug, 2018 3 commits



finish

Hussain Moradi authored 6 months ago



Saeed's requirements

Hussain Moradi authored 6 months ago



97/05/11 10:04

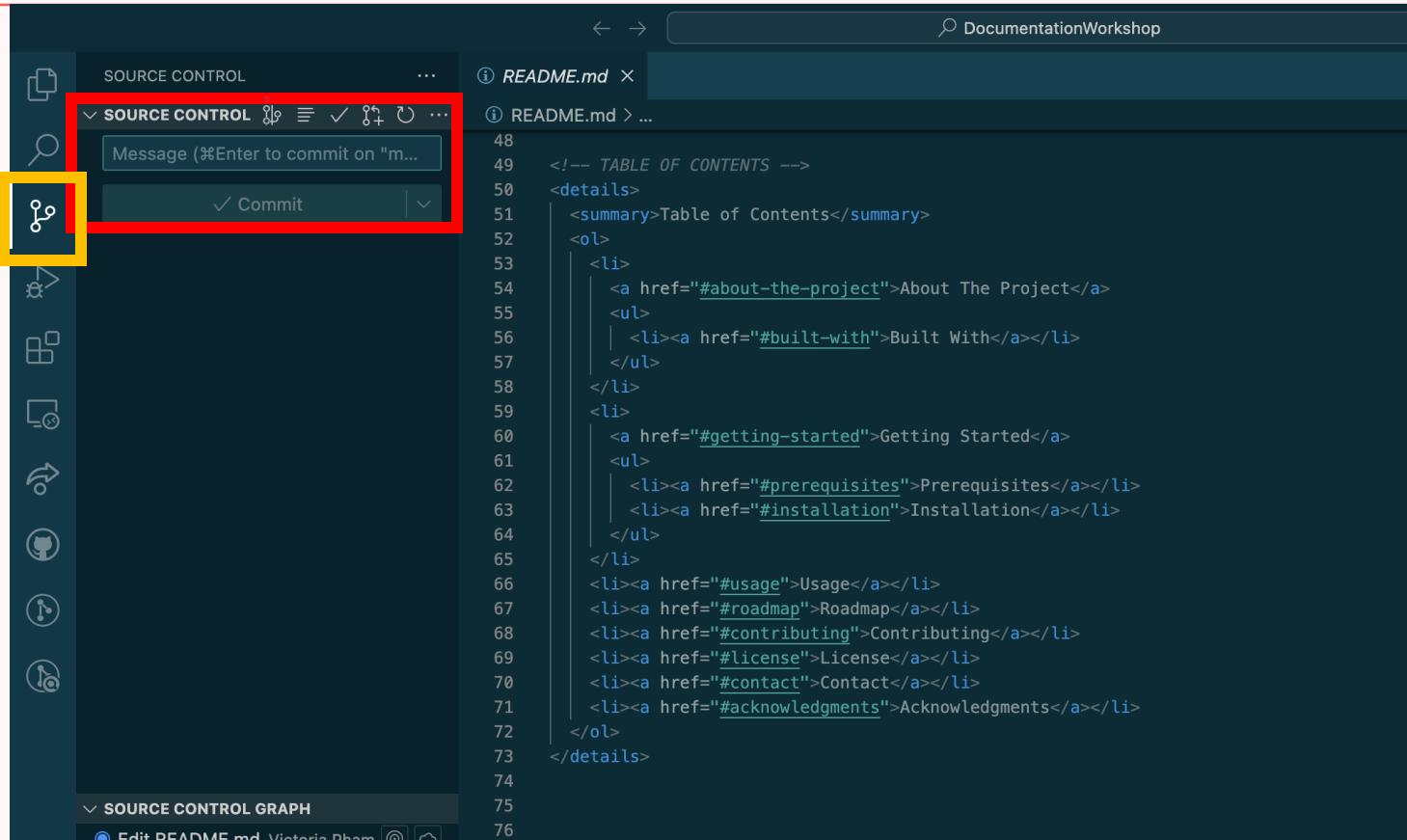
Hussain Moradi authored 6 months ago

Common Mistakes

COMMIT MESSAGES

- Rambling
- Doesn't contain objective

Commit How-to: VSCode



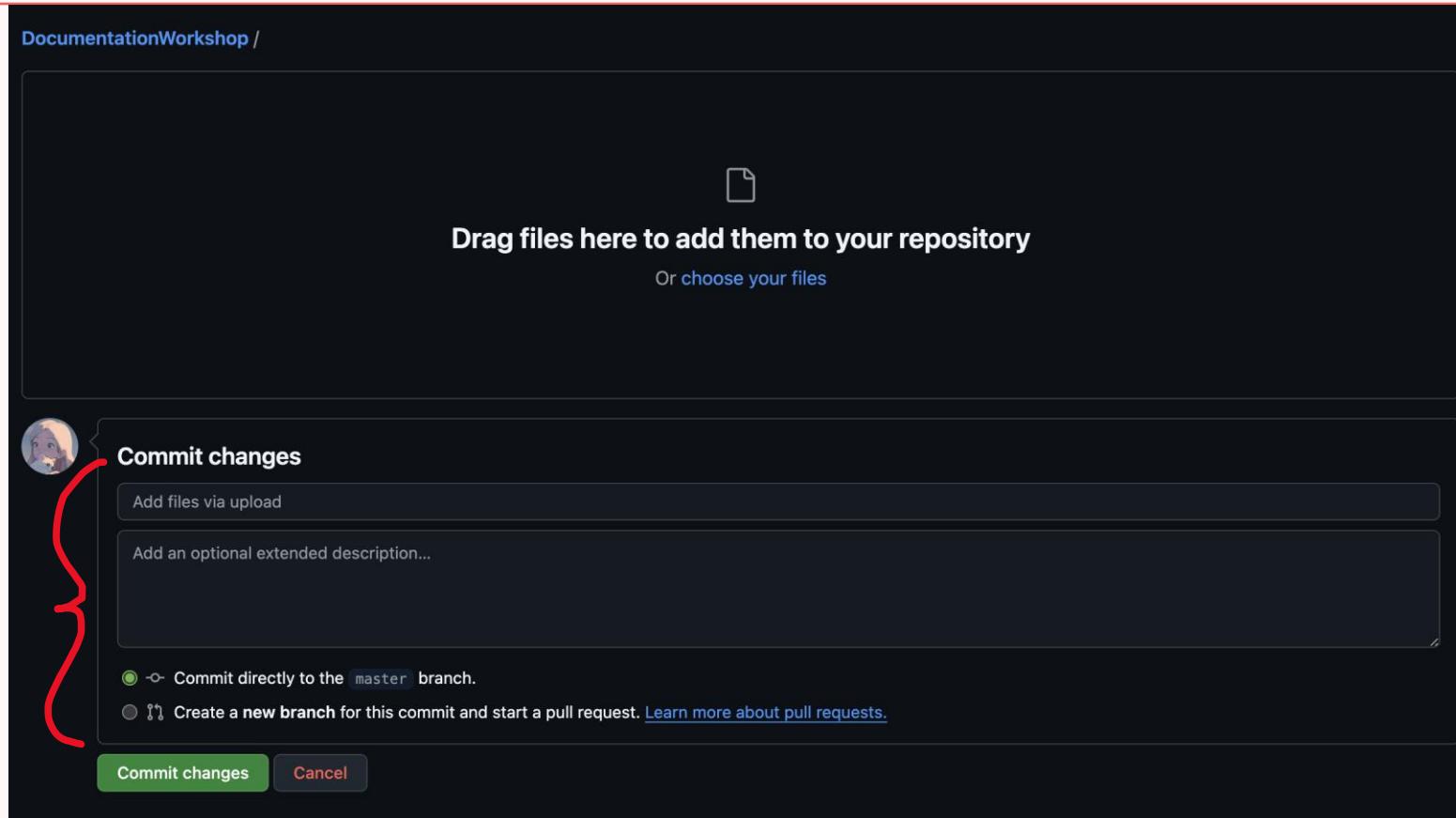
Commit How-to: Terminal

```
[user]@LAPTOP-U24S3V3Q MINGW64 /d/git_repo/Demo_folder/Demo_repository (main)
$ git commit -m "Making my first commit"
[main 110f2/t] Making my first commit
 4 files changed, 0 insertions(+), 0 deletions(-)
  create mode 100644 image_1.jpg
  create mode 100644 image_2.jpg
  create mode 100644 image_3.jpg
  create mode 100644 image_4.jpg

[user]@LAPTOP-U24S3V3Q MINGW64 /d/git_repo/Demo_folder/Demo_repository (main)
$ git status
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
```

Commit How-to: In-browser



GitHub Education

Student Developer Pack

GITHUB



Learning Paths
Experiences
Community Exchange
Events
Custom Domains

GitHub Pro –
Unlimited repositories
Packages
Codespaces
Copilot

<https://education.github.com/pack>

GitHub Pages

Documentation Workshop

SWE Pod 8

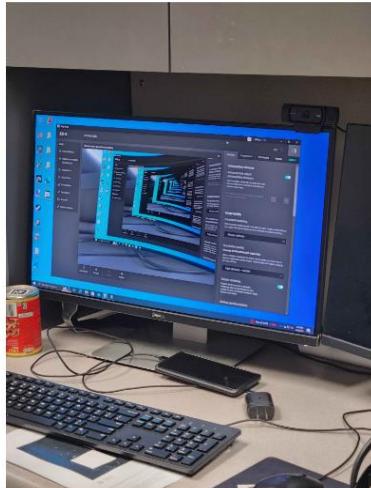
[View the Project on GitHub](#)
bunnhimaybe/DocumentationWorkshop

This project is maintained by
[bunnhimaybe](#)

Hosted on GitHub Pages — Theme by [orderedlist](#)

Welcome to the Documentation Workshop Page!

▶ Table of Contents

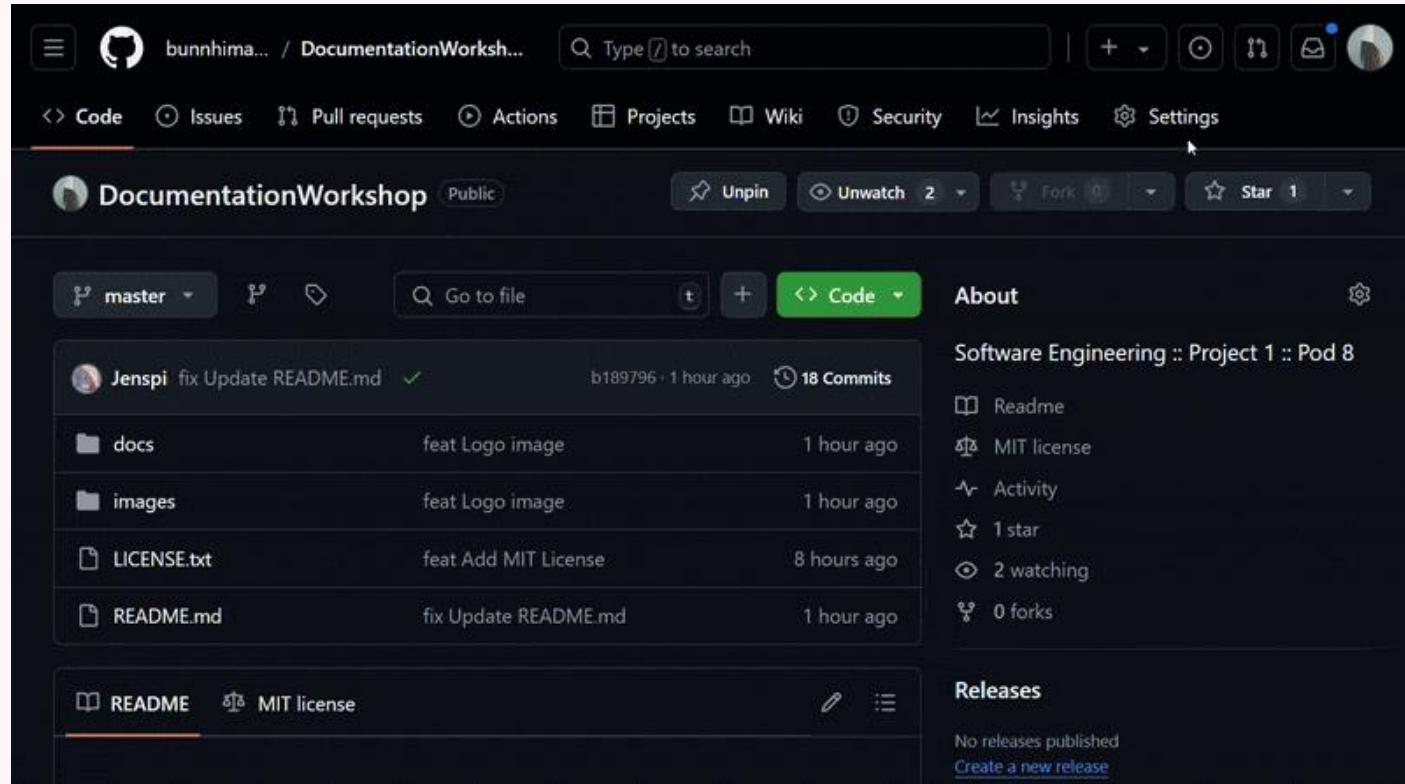


<https://bunnhimaybe.github.io/DocumentationWorkshop/>

- Free static web hosting service
- Accounts can have one user site and unlimited project sites

<https://docs.github.com/en/pages>

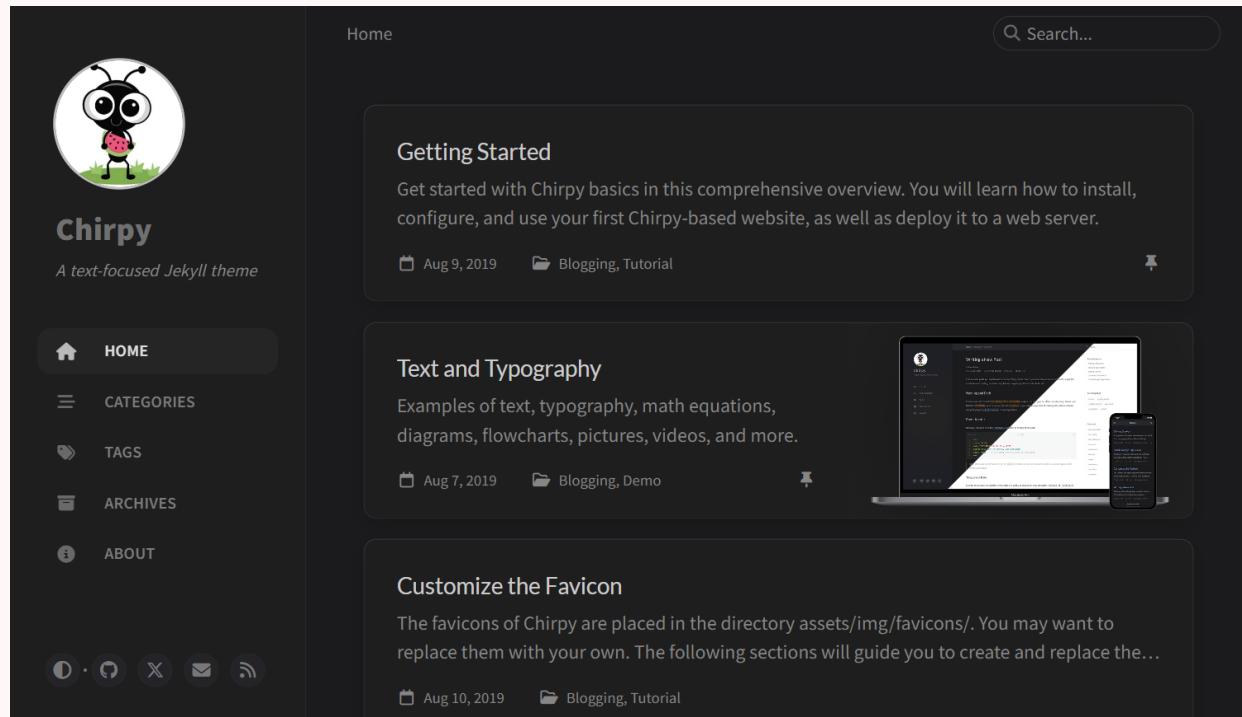
Set Up GitHub Pages



<https://pages.github.com/>

1. Create repository
 - * User default:
username.github.io
2. Clone repository
3. Create entry file
(***index.html*** or
README.md)
4. Publish
5. Enable GitHub Pages

<https://github.com/cotes2020/jekyll-theme-chirpy>

A screenshot of a Jekyll website titled "Chirpy" using the "chirpy" theme. The site has a dark background with white text. On the left is a sidebar with navigation links: HOME (selected), CATEGORIES, TAGS, ARCHIVES, and ABOUT. Below the sidebar are social media sharing icons. The main content area features three blog posts: "Getting Started" (published on Aug 9, 2019, under "Blogging, Tutorial"), "Text and Typography" (published on Aug 7, 2019, under "Blogging, Demo"), and "Customize the Favicon" (published on Aug 10, 2019, under "Blogging, Tutorial"). Each post includes a small thumbnail image showing the website's layout on a laptop and a smartphone.

Deployment

<https://docs.github.com/en/pages/getting-started-with-github-pages/configuring-a-publishing-source-for-your-github-pages-site>

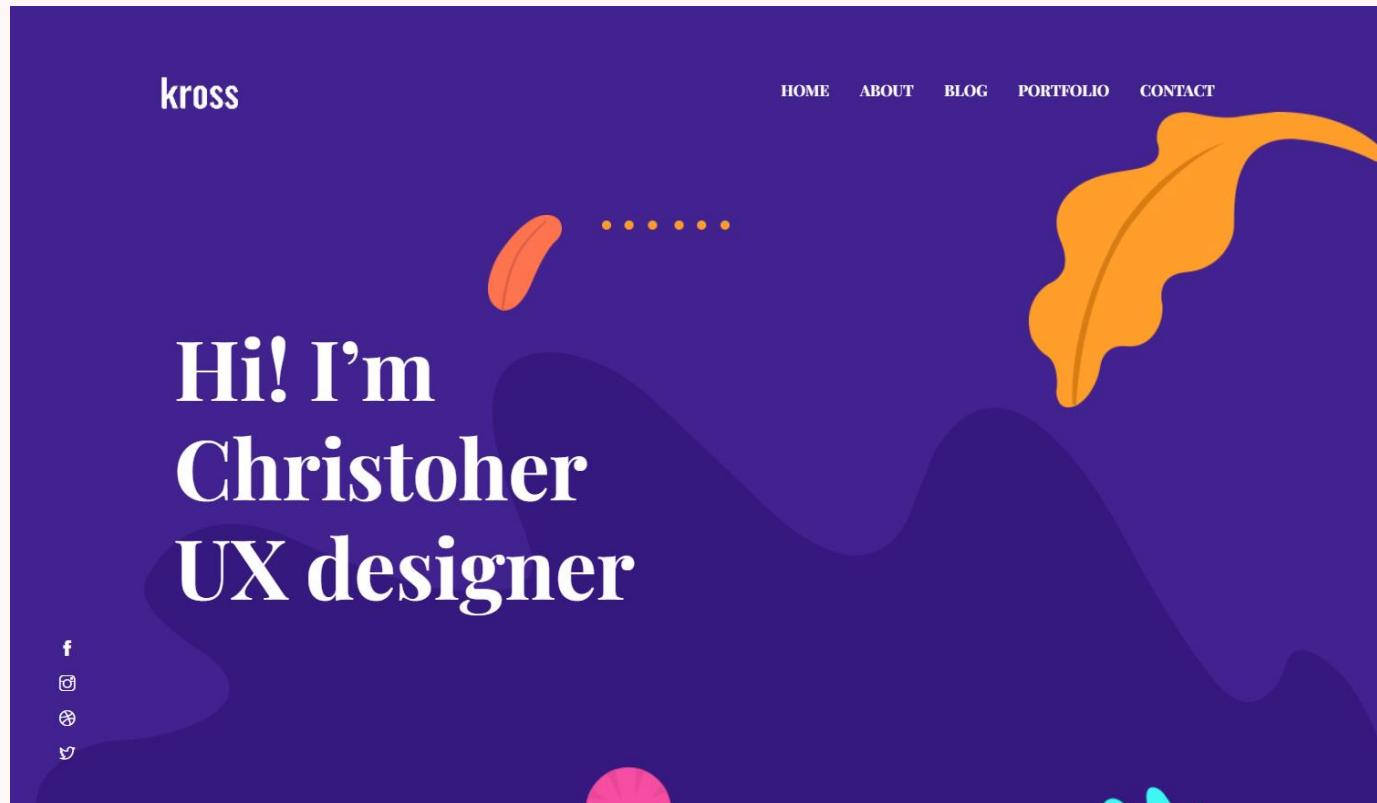
GitHub Actions

- enabled to use built-in Jekyll

From a branch

- Other builds
- Disable Github Actions and add `.nojekyll` file to root to deploy the content directly

Customization



<https://themefisher.com/products/kross-bootstrap>

- Jekyll Themes
- Supported Themes
<https://pages.github.com/themes/>
- Templates
 - HTML
 - Repositories
- Custom Domains

Best Practices



Square Open Source

github.com/square

As a company built on open source, here are some of the internally-developed libraries we have contributed back to the community.

[Android](#) [C](#) [Go](#) [iOS](#) [Java](#) [JavaScript](#) [Kotlin](#) [Ruby](#) [Other](#)

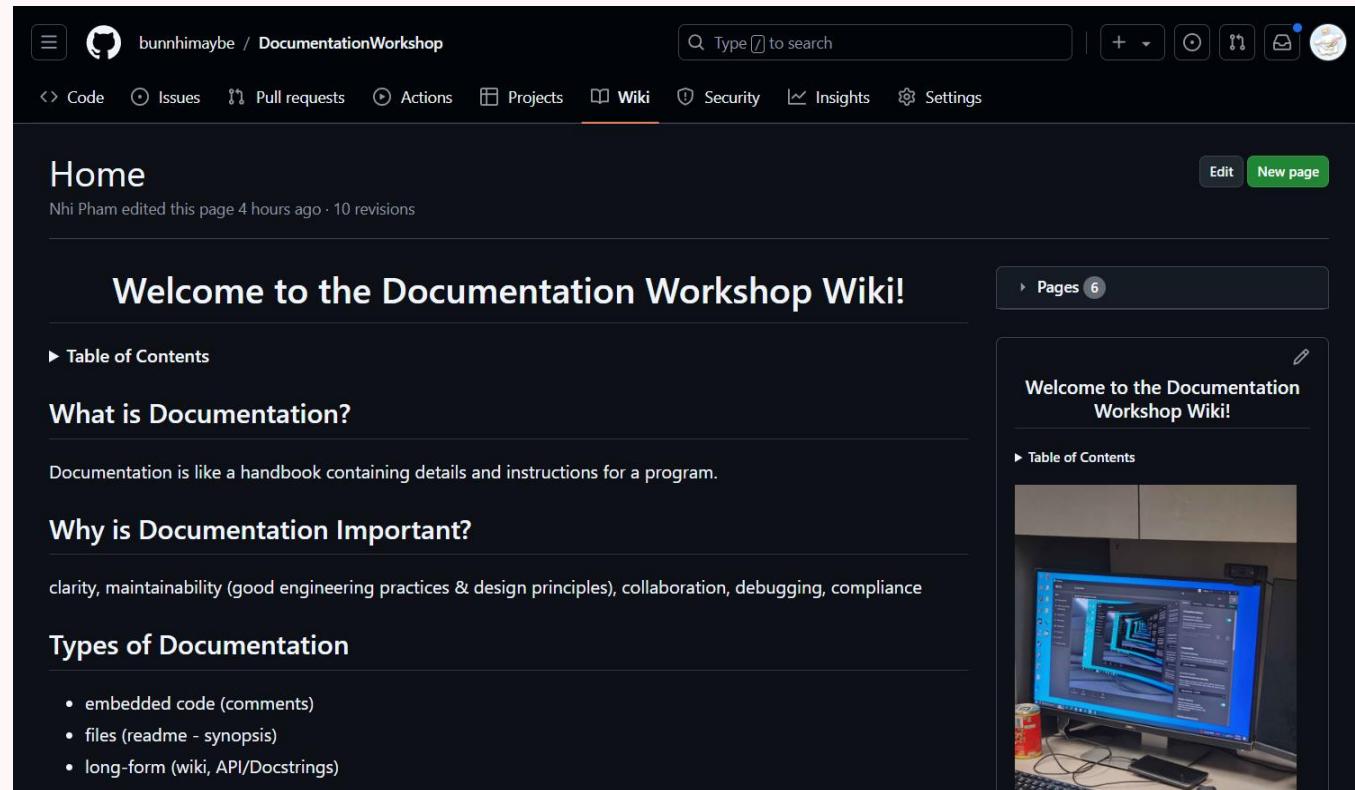
Android



<https://square.github.io>

- detailed
- up-to-date
- well-structured and easy to navigate
- uses Markdown formatting

GitHub Wikis

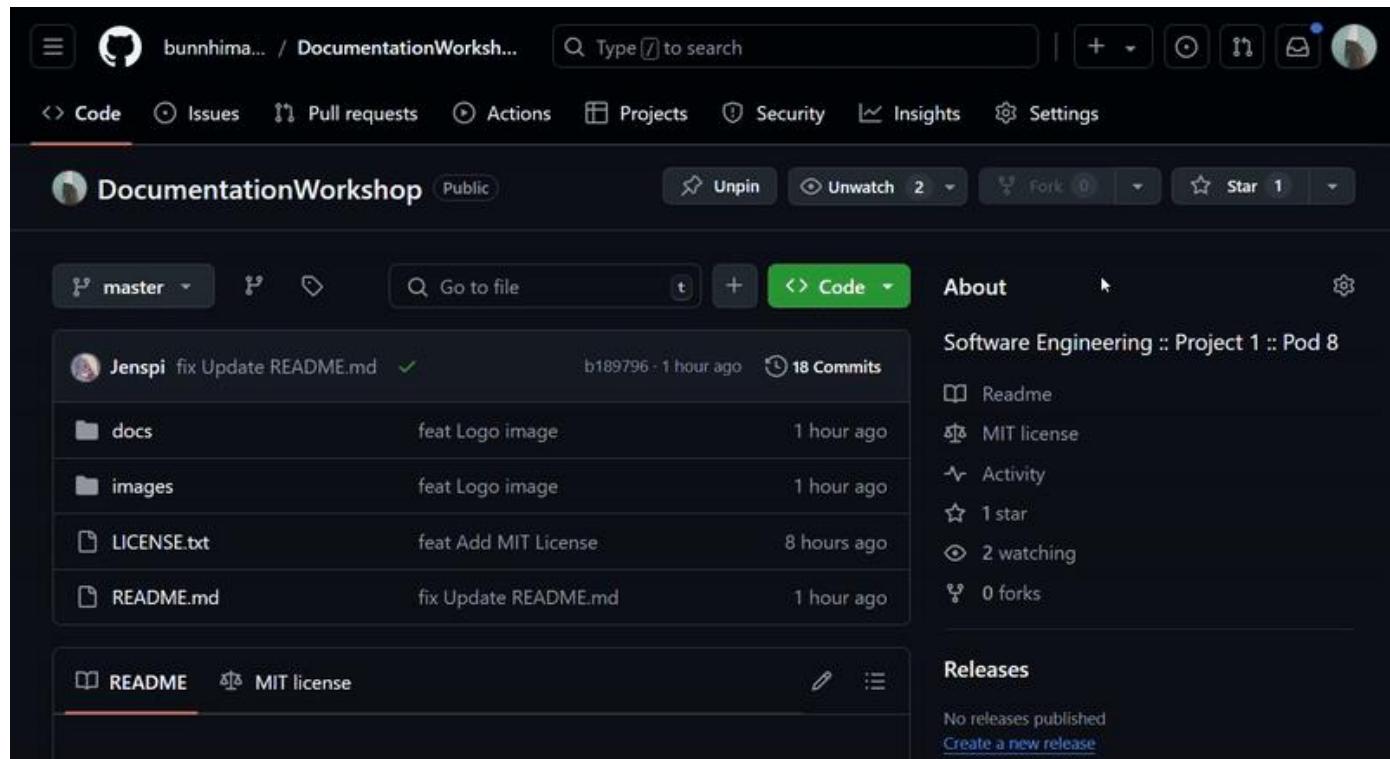


The screenshot shows a GitHub repository named 'DocumentationWorkshop' with a dark theme. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki (which is underlined), Security, Insights, and Settings. A search bar at the top right contains the placeholder 'Type ⌘ to search'. Below the navigation, there's a 'Home' section with a message from 'Nhi Pham' edited 4 hours ago. The main content area features a heading 'Welcome to the Documentation Workshop Wiki!' followed by several sections: 'Table of Contents', 'What is Documentation?', 'Why is Documentation Important?', and 'Types of Documentation'. The 'Types of Documentation' section lists three items: 'embedded code (comments)', 'files (readme - synopsis)', and 'long-form (wiki, API/Docstrings)'. To the right of the main content, there's a sidebar titled 'Pages 6' which also displays the 'Welcome to the Documentation Workshop Wiki!' page.

- Host long-form documentation within repositories
- Edit in-browser or clone locally

<https://github.com/bunnhimaybe/DocumentationWorkshop/wiki>

Set Up GitHub Wikis



A section to host long-form documentation within your repository

- Cloning
- Header
- Footer
- Sidebar

<https://docs.github.com/communities/about-wikis>

Best Practices

The screenshot shows a GitHub Wiki page for the 'openlibrary' repository. The page title is 'Home'. It features a dark theme with white text. At the top, there's a navigation bar with links for Code, Issues (798), Pull requests (142), Actions, Projects (4), Wiki (which is underlined), Security, and Insights. Below the navigation, there's a search bar with placeholder text 'Type / to search'. On the right side of the header, there are several icons: a plus sign, a dropdown arrow, a circular arrow, a gear, an envelope, and a user profile icon. The main content area starts with a heading 'Welcome to the Open Library Handbook!'. Below it, a paragraph explains the purpose of the document: 'For a top-level executive summary of the Open Library project, please see the [Main Open Library Index](#). This document contains a year-by-year breakdown of board reports, roadmaps, community call documents, a project index, and top-level team documents spanning engineering, design, communications, and more.' Underneath this, there's a section titled 'About Open Library' with a detailed paragraph about the project's history and mission. To the right of the main content, there's a sidebar with a heading 'Pages 65'. It lists several sections under 'Getting Started & Contributing': 1. Setting up your developer environment, 2. Using git in Open Library, 3. Finding good First Issues, 4. Code Recipes, 5. Testing Your Code, Debugging & Performance Profiling, 6. Loading Production Site Data → Dev Instance, 7. Submitting good Pull Requests, 8. Asking Questions on Gitter Chat, 9. Joining the Community Slack, 10. Attending Weekly Community Calls @ 9a PT, and 11. Applying to Google Summer of Code & Fellowship Opportunities. Below this, there's another section titled 'Developer Resources' with a single item: 1. FAQ: Frequently Asked Questions.

<https://github.com/bunnhimaybe/DocumentationWorkshop/wiki>

- detailed and up-to-date
- well-structured and easy to navigate
- uses Markdown formatting
- doesn't repeat README information
- **Guidelines:**
<https://github.com/MyHoneyBadger/awesome-github-wiki>
- **Example:**
<https://github.com/internetarchive/openlibrary/wiki>

25*

Your Sphinx Project

Navigation

Version 2.x

Version 1.x

Quick search

Go

Your Sphinx Project's documentation!

You have successfully built your first Sphinx project. Now you can continue to write your project's documentation so that others can use it.

Now you can:

- Add more documentation, additional `*.rst` files and code samples
- Link your documentation directly to your source code
- Customize your Sphinx output with themes

Indices and tables

- Index
- Module Index
- Search Page

©2018, Your Sphinx Project. | Powered by Sphinx 1.7.8 & Alabaster 0.7.11 | [Page source](#)

Sphinx

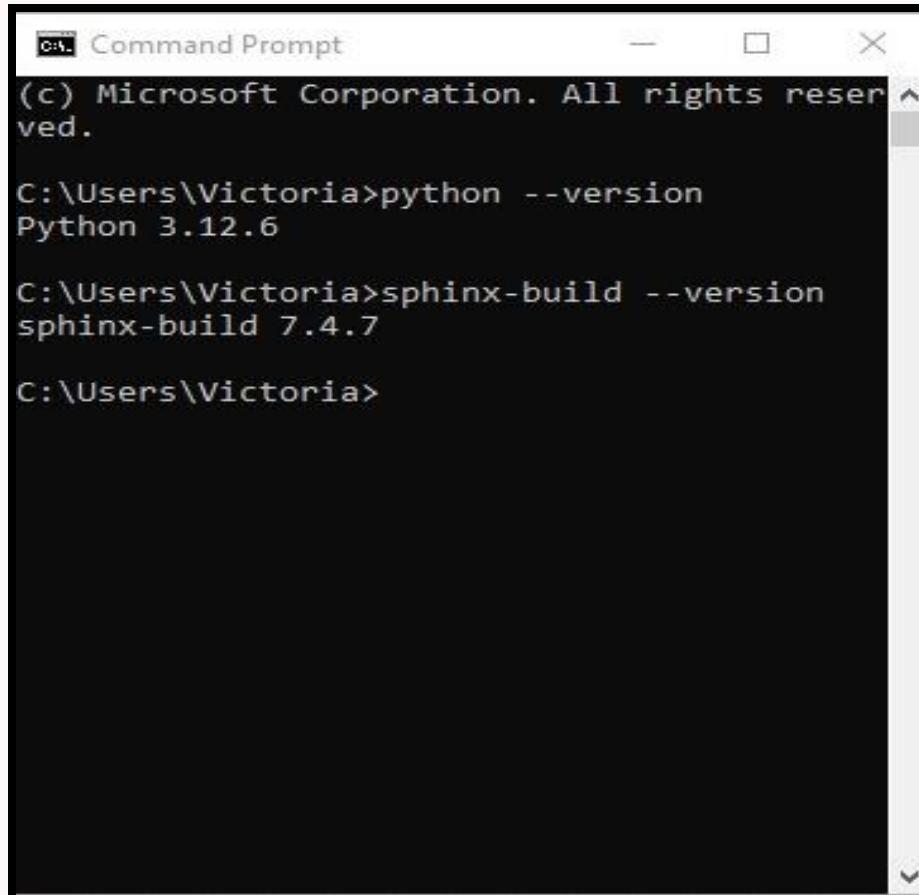
- Auto-documentation generator tool for various documentation files that can create webpages, PDF, and ePub
- Uses reStructured Text and Markdowns

Installation of Sphinx

- **Prerequisite:** Must have Python V.3+ installed.
- **Windows:** Make sure to check-mark 'pip' and 'PATH' during download set-up.
- **MacOS:** Check version via 'python3' command

TERMINAL COMMAND:

- **Windows:** \$ pip install -U sphinx
- **Linux:** \$ apt-get install python3-sphinx
- **MacOS:** \$ brew install sphinx-doc



```
Command Prompt
(c) Microsoft Corporation. All rights reserved.

C:\Users\Victoria>python --version
Python 3.12.6

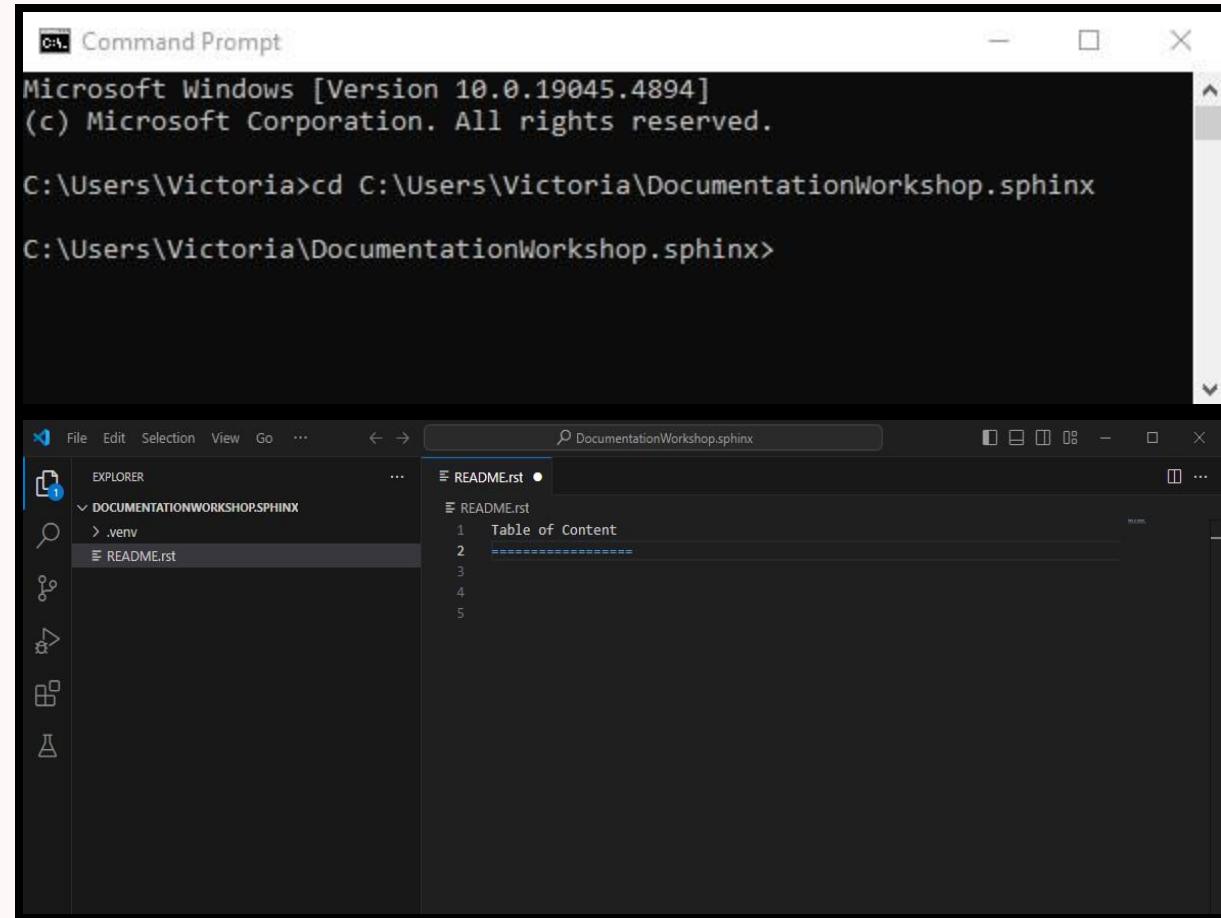
C:\Users\Victoria>sphinx-build --version
sphinx-build 7.4.7

C:\Users\Victoria>
```

Environment Set-Up

FOLDER AND PATHWAY:

- Create a folder in a directory
- Pathway: C Drive > Users > [YourFolder] > [CreateFolder]
- Open [CreateFolder] in Visual Code Studio (VS)
- Create a new file and name it README.rst



Command Prompt

Microsoft Windows [Version 10.0.19045.4894]
(c) Microsoft Corporation. All rights reserved.

```
C:\Users\Victoria>cd C:\Users\Victoria\DocumentationWorkshop.sphinx
C:\Users\Victoria\DocumentationWorkshop.sphinx>
```

File Edit Selection View Go ...

EXPLORER DOCUMENTATIONWORKSHOPSPHINX .venv README.rst

README.rst

Table of Content

====

1

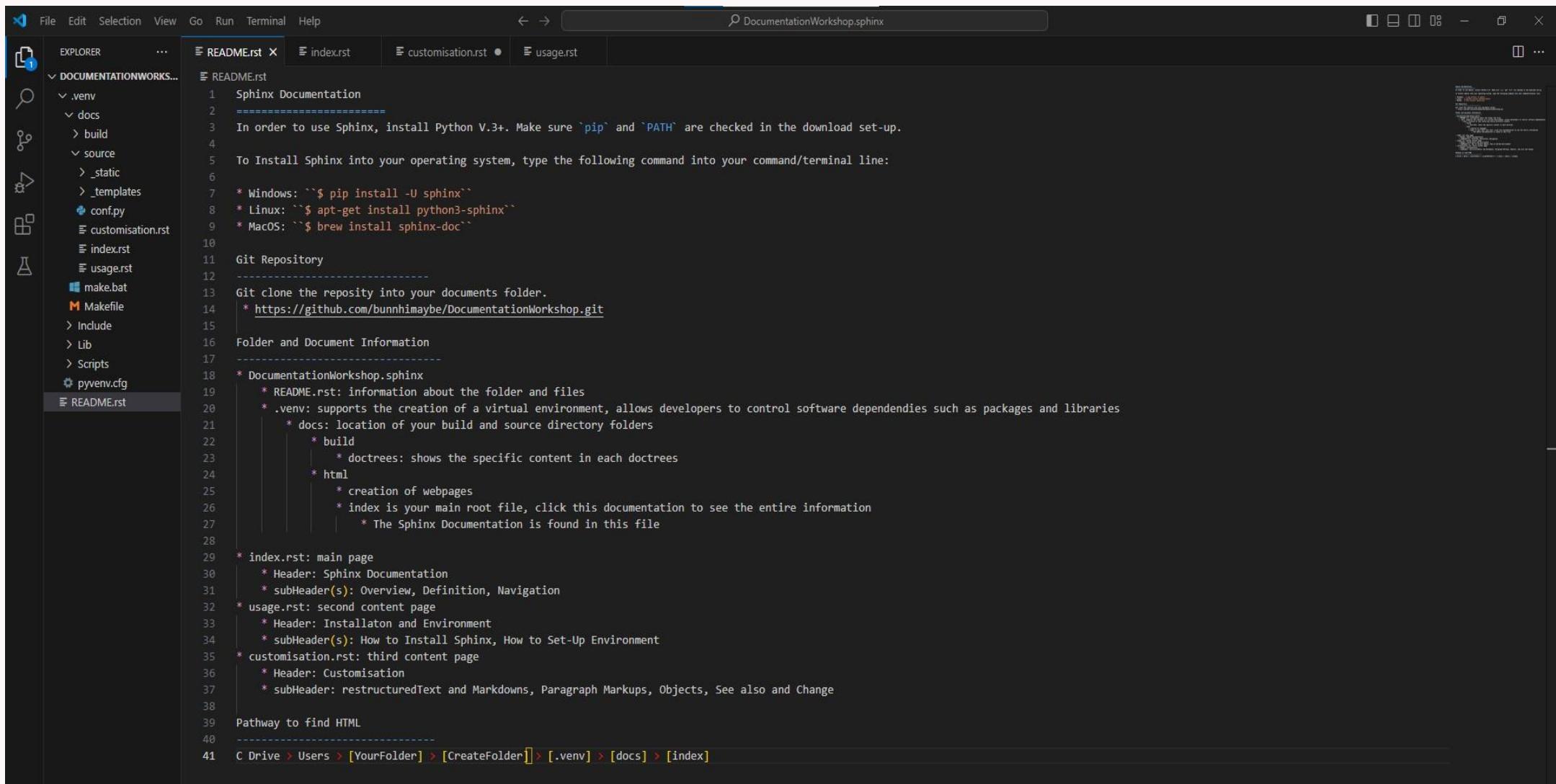
2

3

4

5

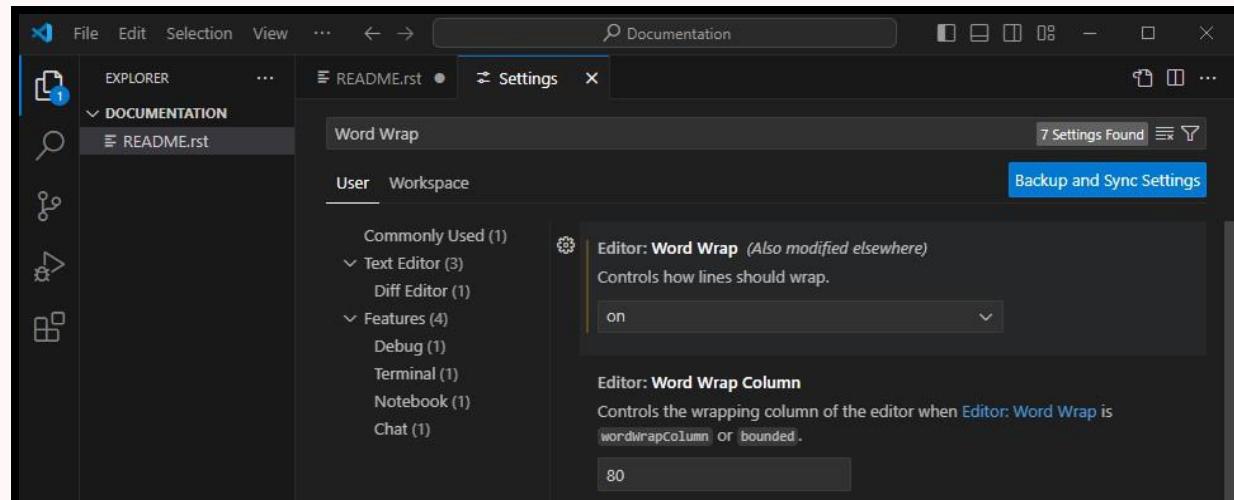
README.rst



```
File Edit Selection View Go Run Terminal Help
EXPLORER DOCUMENTATIONWORKSHOP...
 README.rst index.rst customisation.rst usage.rst

 README.rst
 1 Sphinx Documentation
 2 -----
 3 In order to use Sphinx, install Python V.3+. Make sure `pip` and `PATH` are checked in the download set-up.
 4
 5 To Install Sphinx into your operating system, type the following command into your command/terminal line:
 6
 7 * Windows: ``$ pip install -U sphinx``
 8 * Linux: ``$ apt-get install python3-sphinx``
 9 * MacOS: ``$ brew install sphinx-doc``
10
11 Git Repository
12 -----
13 Git clone the repository into your documents folder.
14 * https://github.com/bunnhimaybe/DocumentationWorkshop.git
15
16 Folder and Document Information
17 -----
18 * DocumentationWorkshop.sphinx
19     * README.rst: information about the folder and files
20     * .venv: supports the creation of a virtual environment, allows developers to control software dependendies such as packages and libraries
21         * docs: location of your build and source directory folders
22             * build
23                 * doctrees: shows the specific content in each doctrees
24             * html
25                 * creation of webpages
26                 * index is your main root file, click this documentation to see the entire information
27                     * The Sphinx Documentation is found in this file
28
29 * index.rst: main page
30     * Header: Sphinx Documentation
31     * subHeader(s): Overview, Definition, Navigation
32 * usage.rst: second content page
33     * Header: Installation and Environment
34     * subHeader(s): How to Install Sphinx, How to Set-Up Environment
35 * customisation.rst: third content page
36     * Header: Customisation
37     * subHeader: restructuredText and Markdowns, Paragraph Markups, Objects, See also and Change
38
39 Pathway to find HTML
40 -----
41 C Drive > Users > [YourFolder] > [CreateFolder] > [.venv] > [docs] > [index]
```

VS SETTINGS & EXTENSIONS



Wordwrap allows a string to be broken into new lines when it reaches a specific length

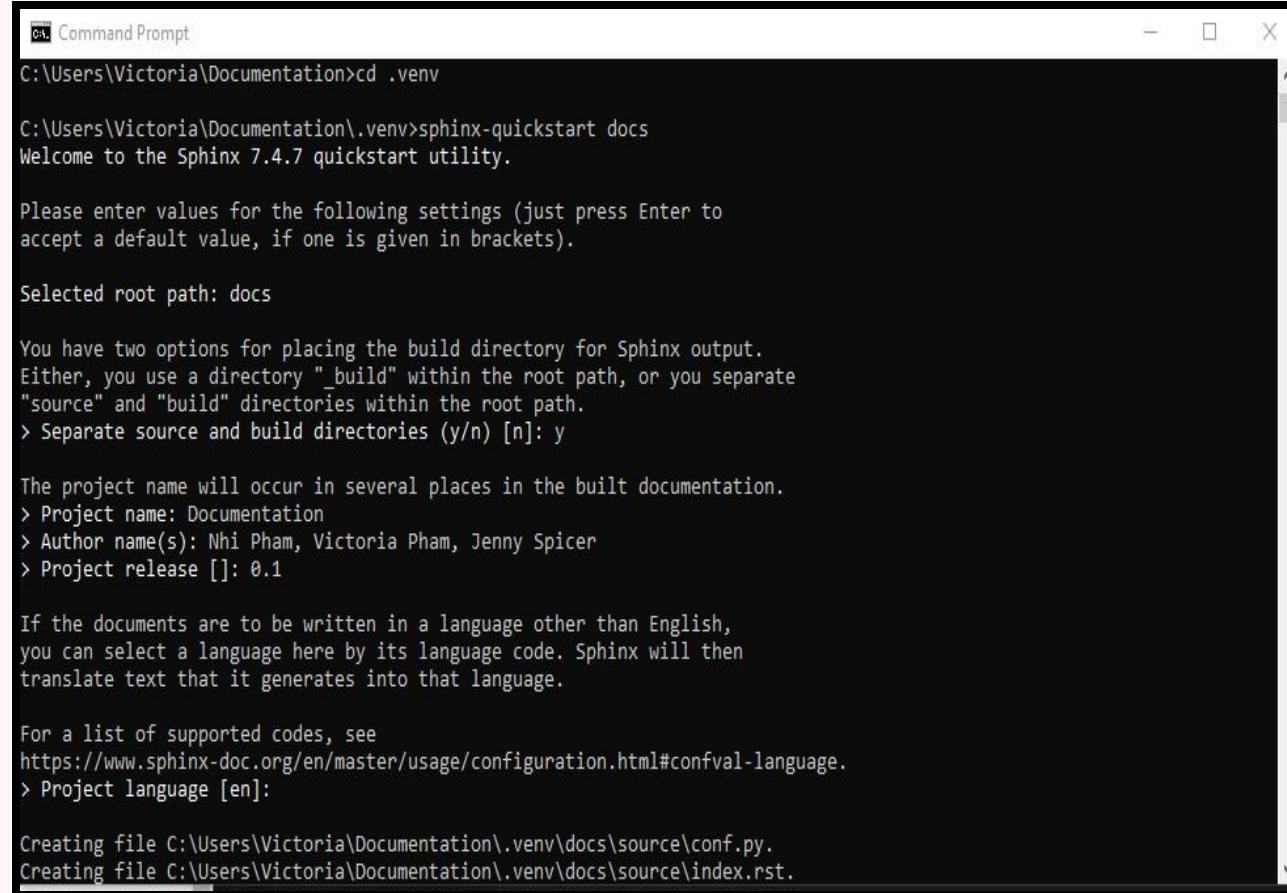


reStructured Text language support,
note the extension should be customised
to allow 'enter' and 'backspacing'

Environment Set-Up

TERMINAL COMMAND AND VS:

- Change directory to the [CreateFolder] pathway and type the following:
 - python -m venv .venv
- Change directory to the \CreateFolder\.venv pathway and type the following:
 - sphinx-quickstart docs
- Separate source and build directory, choose 'yes'
- Insert project name, author, release, and language information



```
C:\Users\Victoria\Documentation>cd .venv
C:\Users\Victoria\Documentation\.venv>sphinx-quickstart docs
Welcome to the Sphinx 7.4.7 quickstart utility.

Please enter values for the following settings (just press Enter to
accept a default value, if one is given in brackets).

Selected root path: docs

You have two options for placing the build directory for Sphinx output.
Either, you use a directory "_build" within the root path, or you separate
"source" and "build" directories within the root path.
> Separate source and build directories (y/n) [n]: y

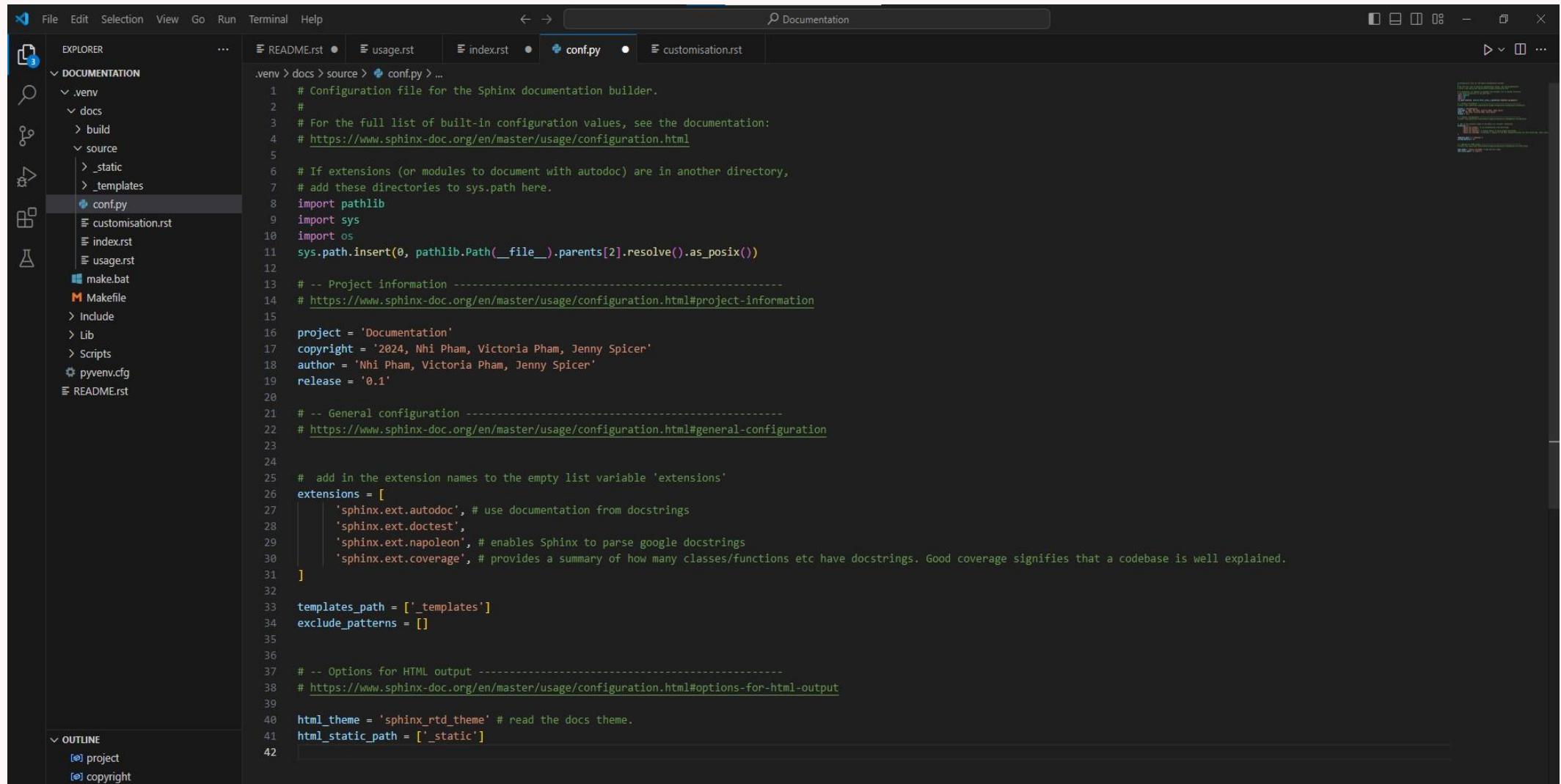
The project name will occur in several places in the built documentation.
> Project name: Documentation
> Author name(s): Nhi Pham, Victoria Pham, Jenny Spicer
> Project release []: 0.1

If the documents are to be written in a language other than English,
you can select a language here by its language code. Sphinx will then
translate text that it generates into that language.

For a list of supported codes, see
https://www.sphinx-doc.org/en/master/usage/configuration.html#confval-language.
> Project language [en]: en

Creating file C:\Users\Victoria\Documentation\.venv\docs\source\conf.py.
Creating file C:\Users\Victoria\Documentation\.venv\docs\source\index.rst.
```

CONFIG.PY

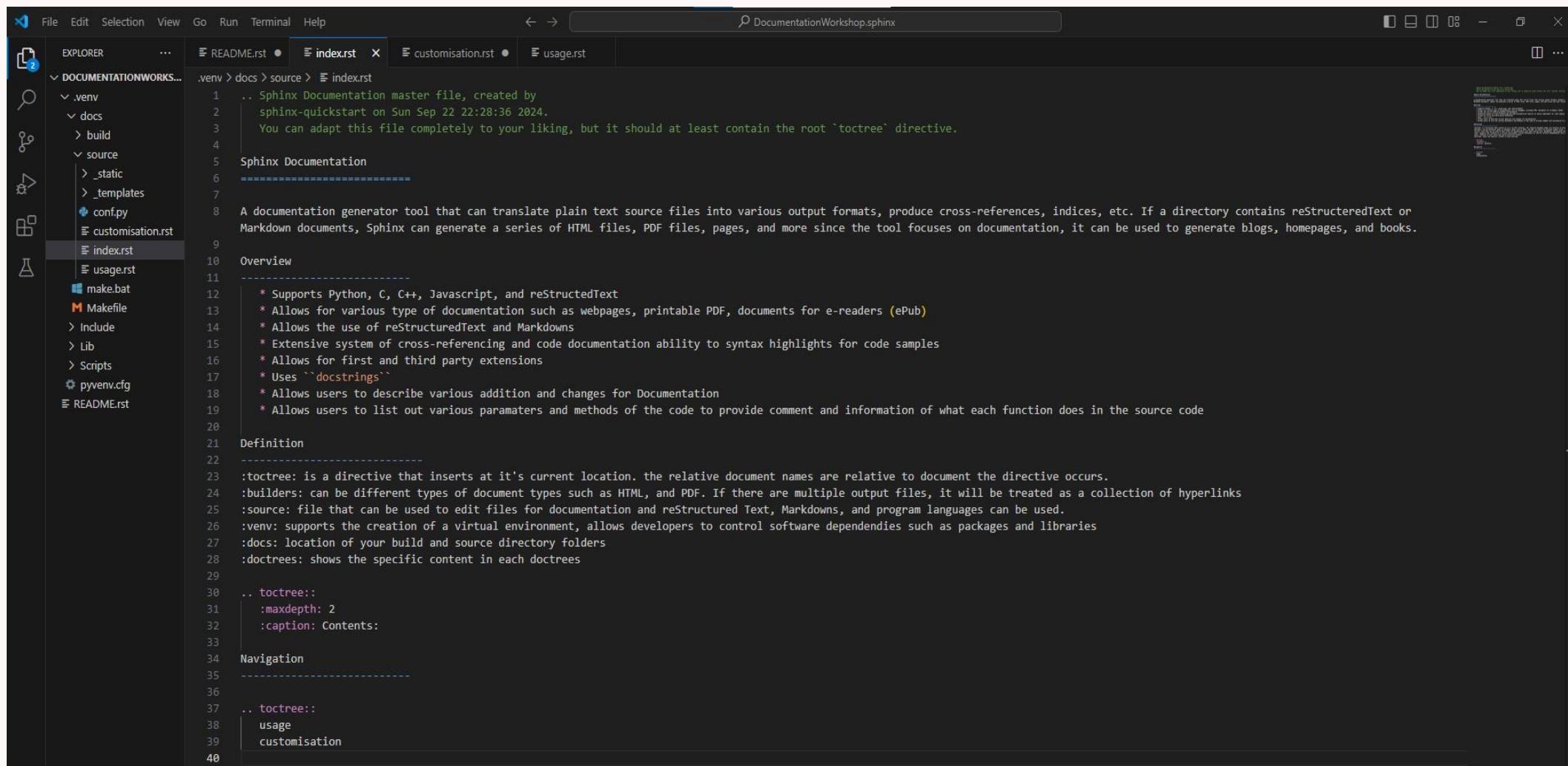


The screenshot shows a code editor interface with a dark theme. The title bar reads "Documentation". The left sidebar has sections for "EXPLORER" and "OUTLINE". In the "EXPLORER" section, "conf.py" is selected. The main area displays the contents of the "conf.py" file:

```
.venv > docs > source > conf.py > ...
1 # Configuration file for the Sphinx documentation builder.
2 #
3 # For the full list of built-in configuration values, see the documentation:
4 # https://www.sphinx-doc.org/en/master/usage/configuration.html
5
6 # If extensions (or modules to document with autodoc) are in another directory,
7 # add these directories to sys.path here.
8 import pathlib
9 import sys
10 import os
11 sys.path.insert(0, pathlib.Path(__file__).parents[2].resolve().as_posix())
12
13 # -- Project information --
14 # https://www.sphinx-doc.org/en/master/usage/configuration.html#project-information
15
16 project = 'Documentation'
17 copyright = '2024, Nhi Pham, Victoria Pham, Jenny Spicer'
18 author = 'Nhi Pham, Victoria Pham, Jenny Spicer'
19 release = '0.1'
20
21 # -- General configuration --
22 # https://www.sphinx-doc.org/en/master/usage/configuration.html#general-configuration
23
24
25 # add in the extension names to the empty list variable 'extensions'
26 extensions = [
27     'sphinx.ext.autodoc', # use documentation from docstrings
28     'sphinx.ext.doctest',
29     'sphinx.ext.napoleon', # enables Sphinx to parse google docstrings
30     'sphinx.ext.coverage', # provides a summary of how many classes/functions etc have docstrings. Good coverage signifies that a codebase is well explained.
31 ]
32
33 templates_path = ['_templates']
34 exclude_patterns = []
35
36
37 # -- Options for HTML output --
38 # https://www.sphinx-doc.org/en/master/usage/configuration.html#options-for-html-output
39
40 html_theme = 'sphinx_rtd_theme' # read the docs theme.
41 html_static_path = ['_static']
42
```

system libraries, source codes (py), project information, extensions, themes

INDEX RST FILE



The screenshot shows a dark-themed interface of the Documentation Workshop application. The title bar reads "DocumentationWorkshop.sphinx". The left sidebar is titled "EXPLORER" and lists the project structure: ".venv", "docs" (with "source" and "build" subfolders), ".static", ".templates", "conf.py", "customisation.rst", "index.rst" (which is currently selected and highlighted in grey), "usage.rst", "make.bat", "Makefile", "Include", "Lib", "Scripts", and "pyenv.cfg". The main pane displays the content of "index.rst". The code is as follows:

```
.venv > docs > source > index.rst
1 .. Sphinx Documentation master file, created by
2 sphinx-quickstart on Sun Sep 22 22:28:36 2024.
3 You can adapt this file completely to your liking, but it should at least contain the root `toctree` directive.
4
5 Sphinx Documentation
6 =====
7
8 A documentation generator tool that can translate plain text source files into various output formats, produce cross-references, indices, etc. If a directory contains reStructuredText or
9 Markdown documents, Sphinx can generate a series of HTML files, PDF files, pages, and more since the tool focuses on documentation, it can be used to generate blogs, homepages, and books.
10
11 Overview
12 -----
13 * Supports Python, C, C++, Javascript, and reStructuredText
14 * Allows for various type of documentation such as webpages, printable PDF, documents for e-readers (ePub)
15 * Allows the use of reStructuredText and Markdowns
16 * Extensive system of cross-referencing and code documentation ability to syntax highlights for code samples
17 * Allows for first and third party extensions
18 * Uses ``docstrings``
19 * Allows users to describe various addition and changes for Documentation
20 * Allows users to list out various parameters and methods of the code to provide comment and information of what each function does in the source code
21
22 Definition
23 -----
24 :toctree: is a directive that inserts at its current location. the relative document names are relative to document the directive occurs.
25 :builders: can be different types of document types such as HTML, and PDF. If there are multiple output files, it will be treated as a collection of hyperlinks
26 :source: file that can be used to edit files for documentation and reStructured Text, Markdowns, and program languages can be used.
27 :venv: supports the creation of a virtual environment, allows developers to control software dependencies such as packages and libraries
28 :docs: location of your build and source directory folders
29 :doctrees: shows the specific content in each doctrees
30
31 .. toctree::
32   :maxdepth: 2
33   :caption: Contents:
34
35 Navigation
36 -----
37 .. toctree::
38   usage
39   customisation
```

main root file, first page seen in the document, includes toctree and navigation

USAGE RST FILE

File Edit Selection View Go Run Terminal Help

EXPLORER README.rst index.rst customisation.rst usage.rst

.venv docs source usage.rst

6
7 Python must be installed and the recommended download is Python 3+ but Python 2.7 is fine. Make sure the 'pip' and 'PATH' boxes during download installation is checked.
8
9 To install Sphinx, run the following command based on the operating system (OS) used in your command or terminal line.
10
11 .. py:function:: pip install -U sphinx
12
13 initialises the package installation for Python and other indexes
14
15 :sphinx: an auto-document extension for python
16
17 For Windows users:
18 ``\$ pip install -U sphinx``
19 For Linux users:
20 ``\$ apt-get install python3-sphinx``
21 For MacOS users:
22 ``\$ brew install sphinx-doc``
23
24 How to Set-Up Environment
25
26
27 * Create a folder in a directory
28 * Pathway: C Drive > Users > [YourFolder] > [CreateFolder]
29 * Open [CreateFolder] in Visual Code Studio
30 * Create a new file and name it README.rst
31 * Open command line and change directory to the [CreateFolder] pathway and type the following:
32 * ``python -m venv .venv``
33 * Change directory to the [CreateFolder] > [.venv] pathway and type the following:
34 * ``sphinx-quickstart docs``
35 * Separate source and build directory, choose 'yes'
36 * Insert project name, author, release, and language information
37 * Make HTML document using the following command:
38 * ``make html``
39 * Add additional ``doctree`` by creating new file.rst and place the file name ``index.rst`` file under ``toctree``
40
41 .. note:
42 * config.py: includes imports, extensions, project information,
43 * index.rst: main root page of the documentation
44 * usage.rst: second page of the documentation
45 * customisation.rst: third page of the documentation

second navigation file, file name '**usage**' must be included in index file in the toctree

CUSTOMISATION RST FILE

The screenshot shows a code editor interface with a dark theme. The title bar reads "DocumentationWorkshop.sphinx". The left sidebar is an "EXPLORER" view showing a project structure:

- .venv
- docs
 - build
 - source
 - _static
 - _templates
 - conf.py
- customisation.rst
- index.rst
- usage.rst
- make.bat
- Makefile
- Include
- Lib
- Scripts
- pyvenv.cfg
- README.rst

The main editor area displays the content of the "customisation.rst" file:

```
Paragraph Markups
-----
10 These markup allows the creation of short paragraphs inside a blocked unit that brings reader attention. Some of those are notes, important, attention, tips, error, important, and more.
11
12 .. warning::
13     This is an example of a warning block unit!
14
15 A best practice to use Markdown for objects is using the code ``.. function::``
16
17 In reStructuredText,
18     * Create Headers, using ``==`````
19     * Create sub-headers using ``---`````
20         * The assignment and hyphen symbols must be the same length as the Text
21
22 .. important::
23     This is an example of an important block unit!
24
25 Please be aware that indentations are important! Indentations can create bullet point list using the asterisks (*) symbol and/or assist in ordered listings.
26
27 .. tip::
28     This is an example of a tip block unit!
29
30 Remember to save your work and commit any changes!
31
32 Objects
33 -----
34 Sphinx's objective is to make easy documentation of objects in any domain. A domain is a collection of object types that belong together, complete with mark-up to create and reference description of the objects
35
36 To document objects, the ``.. py:function::`` directive to the function and list summaries, descriptions, etc. of the objects, paramaters, and so forth. While users must still write every object, Sphinx will generate the formatted document.
37
38 See also and Change
39 -----
40 See also and Change directives can be included in your documentation with the following tags:
41     * ``..seealso::``
42     * ``..versionadded::``
43     * ``..versionchanged::``
```

third navigation file, file name '**customisation**' must be included in index file in the toctree

SPHINX BUILD: HTML FILE

```
Command Prompt
where "builder" is one of the supported builders, e.g. html, latex or linkcheck.

C:\Users\Victoria\Documentation\.venv>sphinx-build -M html docs/source/ docs/build/
Running Sphinx v7.4.7
loading translations [en]... done
making output directory... done
building [mo]: targets for 0 po files that are out of date
writing output...
building [html]: targets for 1 source files that are out of date
updating environment: [new config] 1 added, 0 changed, 0 removed
reading sources... [100%] index
looking for now-outdated files... none found
pickling environment... done
checking consistency... done
preparing documents... done
copying assets...
copying static files... done
copying extra files... done
copying assets: done
writing output... [100%] index
generating indices... genindex done
writing additional pages... search done
dumping search index in English (code: en)... done
dumping object inventory... done
build succeeded.

The HTML pages are in docs\build\html.

C:\Users\Victoria\Documentation\.venv>cd docs
C:\Users\Victoria\Documentation\.venv\docs>make html
```

sphinx build M html docs/source docs/build , build and source folders

SPHINX DOCUMENTATION RESULTS

The screenshot displays two pages from a Sphinx documentation site. The left page, titled 'Overview', lists various features of Sphinx, such as supporting multiple programming languages and generating different output formats. The right page, titled 'Definition', provides detailed explanations for Sphinx's configuration terms like 'toctree', 'builders', and 'source'. Both pages include a sidebar with navigation links like 'Installation and Environment' and 'Customisation'.

Overview

- Supports Python, C, C++, Javascript, and reStructuredText
- Allows for various type of documentation such as webpages, printable PDF, documents for e-readers (ePub)
- Allows the use of reStructuredText and Markdowns
- Extensive system of cross-referencing and code documentation ability to syntax highlights for code samples
- Allows for first and third party extensions
- Uses `docstrings`
- Allows users to describe various addition and changes for Documentation
- Allows users to list out various parameters and methods of the code to provide comment and information of what each function does in the source code

Definition

toctree: is a directive that inserts at it's current location. the relative document names are relative to document the directive occurs.

builders: can be different types of document types such as HTML, and PDF. If there are multiple output files, it will be treated as a collection of hyperlinks

source: file that can be used to edit files for documentation and reStructured Text, Markdowns, and program languages can be used.

venv: supports the creation of a virtual environment, allows developers to control software dependencies such as packages and libraries

docs: location of your build and source directory folders

doctrees: shows the specific content in each doctrees

for code samples

- Allows for first and third party extensions
- Uses `docstrings`
- Allows users to describe various addition and changes for Documentation
- Allows users to list out various parameters and methods of the code to provide comment and information of what each function does in the source code

Navigation

- Installation and Environment
 - How to Install Sphinx
 - How to Set-Up Environment
- Customisation
 - reStructured Text and Markdowns
 - Paragraph Markups
 - Objects
 - See also and Change

Next ➔

© Copyright 2024, Nhi Pham, Victoria Pham, Jenny Spicer.
Built with Sphinx using a theme provided by Read the Docs.

GitHub Page: <https://actuallyvee.github.io/sphinx/>

SPHINX DOCUMENTATION RESULTS

The screenshot shows the Sphinx Documentation Results page. The header includes the title 'SPHINX DOCUMENTATION RESULTS' and a search bar. The main content area has a blue header bar with the title 'Documentation' and a search bar. Below this, the page title is 'Installation and Environment'. On the right, there is a link 'View page source'. The left sidebar contains navigation links: 'Installation and Environment' (selected), 'How to Install Sphinx', 'How to Set-Up Environment', and 'Customisation'. The main content area contains a heading 'How to Install Sphinx' with a note about Python version requirements and command-line installation instructions for Windows, Linux, and Mac OS. It also includes a section 'How to Set-Up Environment' with a note about creating a folder and command-line setup.

The screenshot shows the Sphinx Documentation Results page. The header includes the title 'SPHINX DOCUMENTATION RESULTS' and a search bar. The main content area has a blue header bar with the title 'Documentation' and a search bar. Below this, the page title is 'How to Set-Up Environment'. On the right, there is a link 'View page source'. The left sidebar contains navigation links: 'Installation and Environment' (selected), 'How to Install Sphinx', 'How to Set-Up Environment', and 'Customisation'. The main content area contains a heading 'How to Set-Up Environment' with a note about creating a folder and command-line setup. It includes a 'Note' section with a list of file types: config.py, index.rst, usage.rst, and customisation.rst. At the bottom, there are 'Previous' and 'Next' navigation buttons.

header: installation and environment, subHeader: how-to install, set-up, and folder directory

SPHINX DOCUMENTATION RESULTS

The screenshot shows the Sphinx Documentation Results page with the title "SPHINX DOCUMENTATION RESULTS". The main content area displays the "Customisation" section. The sidebar on the left includes a "Documentation" header, a search bar, and navigation links for "Installation and Environment", "Customisation", "reStructured Text and Markdowns", "Paragraph Markups", "Objects", and "See also and Change". The main content area has a breadcrumb trail "Documentation / Customisation" and a "View page source" link. The "Customisation" section contains a heading "Customisation" and a sub-section "reStructured Text and Markdowns". It includes a note that Sphinx documentation can be customised using various reStructuredText and Markdowns. Below this, there are three callout boxes: a "Warning" box with the text "This is an example of a warning block unit!", a "Important" box with the text "This is an example of an important block unit!", and a "Tip" box with the text "This is an example of a tip block unit!".

The screenshot shows the Sphinx Documentation Results page with the title "SPHINX DOCUMENTATION RESULTS". The main content area displays the "Customisation" section. The sidebar on the left includes a "Documentation" header, a search bar, and navigation links for "Installation and Environment", "Customisation", "reStructured Text and Markdowns", "Paragraph Markups", "Objects", and "See also and Change". The main content area has a breadcrumb trail "Documentation / Customisation" and a "View page source" link. The "Customisation" section contains a heading "Customisation" and a sub-section "reStructured Text and Markdowns". It includes a note that Sphinx documentation can be customised using various reStructuredText and Markdowns. Below this, there are three callout boxes: a "Tip" box with the text "This is an example of a tip block unit!", a "Important" box with the text "This is an example of an important block unit!", and a "Tip" box with the text "Remember to save your work and commit any changes!".

The screenshot shows the Sphinx Documentation Results page with the title "SPHINX DOCUMENTATION RESULTS". The main content area displays the "Objects" section. The sidebar on the left includes a "Documentation" header, a search bar, and navigation links for "Installation and Environment", "Customisation", "reStructured Text and Markdowns", "Paragraph Markups", "Objects", and "See also and Change". The main content area has a breadcrumb trail "Documentation / Objects" and a "View page source" link. The "Objects" section contains a heading "Objects" and a note that Sphinx's objective is to make easy documentation of objects in any domain. It describes how a domain is a collection of object types that belong together, complete with mark-up to create and reference description of the objects. It also notes that to document objects, the `... py:function::` directive is used for functions and lists summaries, descriptions, etc. of the objects, parameters, and so forth. While users must still write every object, Sphinx will generate the formatted document. Below this, there is a "See also and Change" section with a note that "See also and Change" directives can be included in your documentation with the following tags: `...seealso:`, `...versionadded:`, and `...versionchanged:`. A "Previous" button is also visible.

header: customisation , subheader: reStructuredtext, objects, see also and change

OTHER EXAMPLES (MORE OBJECT)

Documentation_demo 0.1 documentation » data » data package

Table of Contents

- data package
 - Submodules
 - data.demo module
 - DemoClass
 - no_docstring()
 - speed()
 - word_add()
 - Module contents

data package

Submodules

data.demo module

Classes:

`DemoClass(varA, varB)` A class containing demo methods for the purpose of creating auto-documentation

Functions:

`no_docstring(a, b)`

`speed(distance, time_seconds)` This function calculates speed given distance and time.

`word_add(word1, word2)` Combines both inputs to make a concatenated word with a full stop as a delimiter.

`class data.demo.DemoClass(varA, varB)`

Bases: `object`

A class containing demo methods for the purpose of creating auto-documentation

Methods:

`class_method1()` This method adds two variables together and returns the sum.

`class_method1()`

This method adds two variables together and returns the sum.

Returns: Sum of two variables
Return type: int

`data.demo.no_docstring(a, b)`

`data.demo.speed(distance: int, time_seconds: int) → int`

This function calculates speed given distance and time.

Parameters: • `distance (int)` – Distance variable
 • `time_seconds (int)` – Time (in seconds) variable

Raises: `ValueError` – Specifies if input distance is less than zero.

Returns: Speed variable, which is calculated using distance and time.
Return type: int

`data.demo.word_add(word1: str, word2: str) → str`

Combines both inputs to make a concatenated word with a full stop as a delimiter.

Parameters: • `word1 (str)` – First word input
 • `word2 (str)` – Second word input

Raises: `ValueError` – Suggests a new word if word entered contains more than 10 characters

Returns: Combined word made from word1 and word2
Return type: str

Module contents

Documentation_demo 0.1 documentation

Search docs

Getting started

Commands

data

- data package
- Submodules
- data.demo module
- Module contents

/ data / data package

View page source

data package

Submodules

data.demo module

Classes:

`DemoClass (varA, varB)` A class containing demo methods for the purpose of creating auto-documentation

Functions:

`no_docstring (a, b)`

`speed (distance, time_seconds)` This function calculates speed given distance and time.

`word_add (word1, word2)` Combines both inputs to make a concatenated word with a full stop as a delimiter.

`class data.demo.DemoClass(varA, varB)`

Bases: `object`

A class containing demo methods for the purpose of creating auto-documentation

Methods:

`class_method1 ()` This method adds two variables together and returns the sum.

`class_method1()`

This method adds two variables together and returns the sum.

Returns: Sum of two variables
Return type: int

`data.demo.no_docstring(a, b)`

`data.demo.speed(distance: int, time_seconds: int) → int`

This function calculates speed given distance and time.

Parameters: • `distance (int)` – Distance variable
 • `time_seconds (int)` – Time (in seconds) variable

Raises: `ValueError` – Specifies if input distance is less than zero.

Returns: Speed variable, which is calculated using distance and time.
Return type: int

[Lucy Dickinson: Code Autodocumentation](#), expected input and output in source file codes, similar to creating a javadoc but markdowns

OTHER EXAMPLES (MORE OBJECT)

Sample Project 0.0.1 documentation » trees » trees package » trees.binary_tree...

[previous](#) | [modules](#) | [index](#)

Table of Contents

trees.binary_trees package

- Submodules
- [trees.binary_trees.avl_tree module](#)
- [trees.binary_trees.binary_search_tree module](#)
- [trees.binary_trees.binary_tree module](#)
- [trees.binary_trees.red_black_tree module](#)
- [trees.binary_trees.threaded_binary_tree module](#)
- [trees.binary_trees.traversal module](#)
 - Routines
- Module contents

Previous topic

[trees.bin package](#)

This Page

[Show Source](#)

Quick search

 Go

trees.binary_trees package

Submodules

trees.binary_trees.avl_tree module

AVL Tree.

```
class trees.binary_trees.avl_tree.AVLNode(key: Any, data: Any, left: Optional[trees.binary_trees.avl_tree.AVLNode] = None, right: Optional[trees.binary_trees.avl_tree.AVLNode] = None, parent: Optional[trees.binary_trees.avl_tree.AVLNode] = None, height: int = 0)
```

Bases: [trees.binary_trees.binary_tree.Node](#)

AVL Tree node definition.

height: *int* – 0

left: *Optional[AVLNode]* = None

parent: *Optional[AVLNode]* = None

right: *Optional[AVLNode]* = None

```
class trees.binary_trees.avl_tree.AVLTree
```

Bases: abc.ABC, Generic[[binary_tree.NodeType](#)]

AVL Tree.

root

The root node of the binary search tree.

Type: *Optional[AVLNode]*

OTHER EXAMPLES (MORE OBJECT)

Creating recipes

To retrieve a list of random ingredients, you can use the `lumache.get_random_ingredients()` function:

`lumache.get_random_ingredients(kind=None)`

Return a list of random ingredients as strings.

PARAMETERS

kind (*list[str]* or *None*) – Optional “kind” of ingredients.

RETURNS

The ingredients list.

RETURN TYPE

list[str]

Read the Docs

- Sphinx documentation can be hosted on Read the Docs
- Theme was chosen from this platform
- Another resourceful documentation tool
- Sign-up using email or GitHub account
- Should be able to clone and/or import repository

The screenshot shows the Read the Docs homepage. At the top, there's a navigation bar with links for Product, Pricing, Resources, Log in, and Sign up. Below the navigation, a main heading says "Documentation simplified" with the subtext "Build, host, and share documentation, all with a single platform." A "Sign up now" button is visible. To the right, there's a graphic illustrating a workflow: a central black box labeled "\$ git commit \$ git push" is connected by dashed lines to several colored boxes representing different versions and pull requests: "Version 3.1" (yellow), "PR #379" (green), "PR #378" (red), "Version 3.0" (green), and "Version 2.2" (purple). Below this graphic, three cards describe features: "Automatic deploy previews" (grey card with a robot icon), "Ideal developer experience" (dark grey card with a greater-than symbol icon), and "Work privately or publicly" (purple card with a key icon). At the bottom, a footer states "Trusted by businesses worldwide since 2010".



In-Class Activity

Looking at a sample README...

https://github.com/bunnhimaybe/DocumentationWorkshop/blob/master/samples/BAD_README.md

What makes it good or not good?

What changes could be made?

Something's kinda weird about it....



Deliverables

Each pod chooses at least 1 documentation tool to create and deliver:

- GitHub README
- GitHub User or Project Page
- GitHub Wiki