Python Tools Tutorials

Read/Watch the required knowledge, and complete each section’s activity

Contents

[1: Python Linting 101 with Black 1](#_Toc112137126)

[2: Debugging with vscode 2](#_Toc112137127)

[3: SAST semgrep and trufflehog 3](#_Toc112137128)

[4: SCA pip-audit 4](#_Toc112137129)

[5: Back to Git 5](#_Toc112137130)

# 1: Python Linting 101 with Black

**Required Knowledge:**  
Watch the black tutorial  
<https://www.youtube.com/watch?v=j1MbEYhYj_Y>

Read through PEP8 standard

<https://peps.python.org/pep-0008/>

**Activities:**

Install black locally on your machine

<https://black.readthedocs.io/en/stable/>

Run it against your python code from previous practical’s

**Bonus Activity:**

Look through and use the vscode linting for python

<https://code.visualstudio.com/docs/python/linting>

# 2: Debugging with vscode

**Required Knowledge:**

**Watch vscode debugger tutorial**

<https://www.youtube.com/watch?v=cqtIKryI_Cc> **Read through vscode debugging help pages**

<https://code.visualstudio.com/docs/python/debugging>

**Activities:**

Go back to your python fib code from several examples ago

Set a breakpoint in your fib for loop

Run python in debug mode, and set it stop at the breakpoint

# 3: SAST semgrep and trufflehog

**Required Knowledge:**  
Watch the semgrep 101 video (about using with jenkins but it’s basically the same thing)  
<https://www.youtube.com/watch?v=X7_jiKsLkHs>

Watch the truffle hog 101 video

<https://www.youtube.com/watch?v=C__PTsoITqY>

**Activity:**  
Load up your git project for using databases and docker  
  
Run trufflehog against your code

<https://github.com/trufflesecurity/truffleHog>

Run semgrep against your code

<https://semgrep.dev/docs/cli-reference/>

Complete/fix found secrets or anti patterns discovered

# 4: SCA pip-audit

**Required Knowledge:**  
Watch the pip audit 101 video  
<https://www.youtube.com/watch?v=OL5eo4cJFh8>

**Activity:**  
Load up requirements.txt created during course and run pip audit on it

<https://pypi.org/project/pip-audit/>

Update any vulnerable packages found

***Hint***: if you haven’t created any requirement files

You can download the vulnerable requirements.txt from eze goat

<https://github.com/RiverSafeUK/eze-goats/tree/main/goat-python/insecure>

# 5: Back to Git Hooks with Black

**Required Knowledge:**  
Re-Read sections on git hooks  
(covered in GIT basics sections)

<https://www.atlassian.com/git/tutorials/git-hooks>   
<https://git-scm.com/book/en/v2/Customizing-Git-Git-Hooks>

**Read up on precommit ( python git hook manager )**

<https://pre-commit.com/>

**Activity:**  
Load up project used to test black on, in tutorial 1 of this document

Add precommit to the project’s requirements.txt

Add black to automatically run, when doing a git commit, using precommit

Check your .git/hooks folder, and test the hooks are working

Well done, you’ve done typical production git hook, one many projects use to enforce code standards and keep code clean