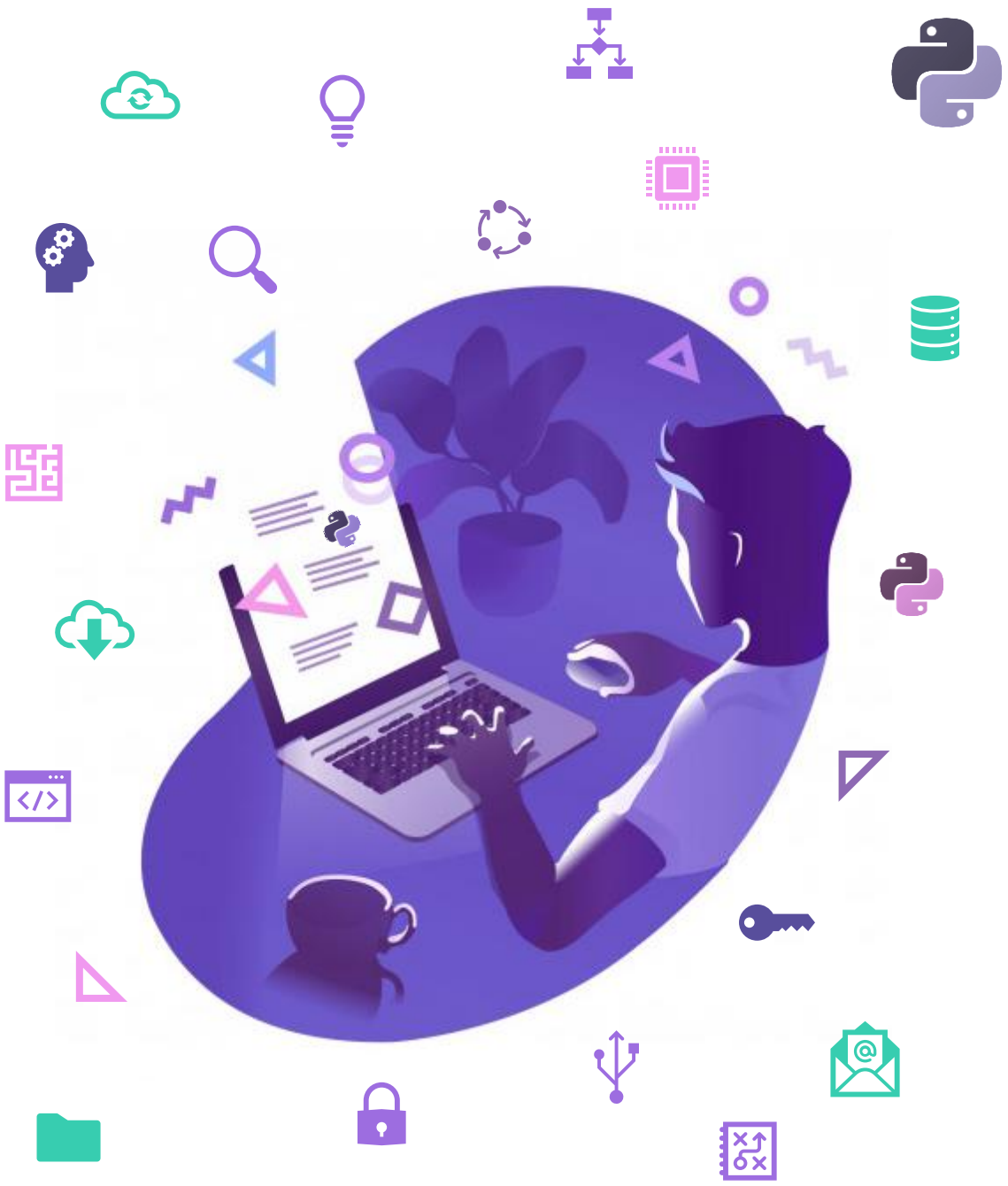
Introduction to **Python Programming**

Beginners to Advanced

Vikram
IoT Application Dev & DevOps



Course Contents

- Setup Development Tools
- Introduction to Python
- Syntax, Variables And Keywords
- Input And Output : Usage Of End Parameter & Formatted Printing
- Statements, Indentation And Comments In Python
- Operators
- Data Types And Type Casting
- Strings & its methods
- Decision Making
- Conditional Branching
- Loops, Range Function, Membership Operators
- Break, Continue and Return statements
- Switch Case
- Main Method
- Functions
- Global Vs Local Variables
- Lists, Tuples, Sets, Dictionary and Arrays
- Copy Methods
- Exception Handling
- OOPS : Class, Objects etc
- File Handling
- Modules: OS, JSON etc

Installing Tools

Code Editors and IDEs

Code Editors

- Code editors/text editors are great for writing and editing the code
- They are usually lightweight and suitable for development of small projects
- However, once your program gets larger, you need much more capabilities in the editors, like test and debug your code, other than editing. That's where IDEs come into the play
- Depending upon the language one codes on the editor, it highlights special keywords and gives some suggestions

  Ex: Visual Studio Code, Sublime Text, Atom, Vi, Vim etc

IDE (Integrated Development Environment)

- An IDE is a software that consists of common developer tools into a single user-friendly GUI (Graphical User interface)
- An IDE majorly consists of a source code editor for writing software code, local build automation for creating a local build of the software like compiling source code
- A good IDE must possess: Save and Reload Source Code , Execution from Within the Environment, Debugging Support, Syntax Highlighting, Automatic Code Formatting, Auto code Completion etc
- Some IDEs also include source control and support for package managers like PIP

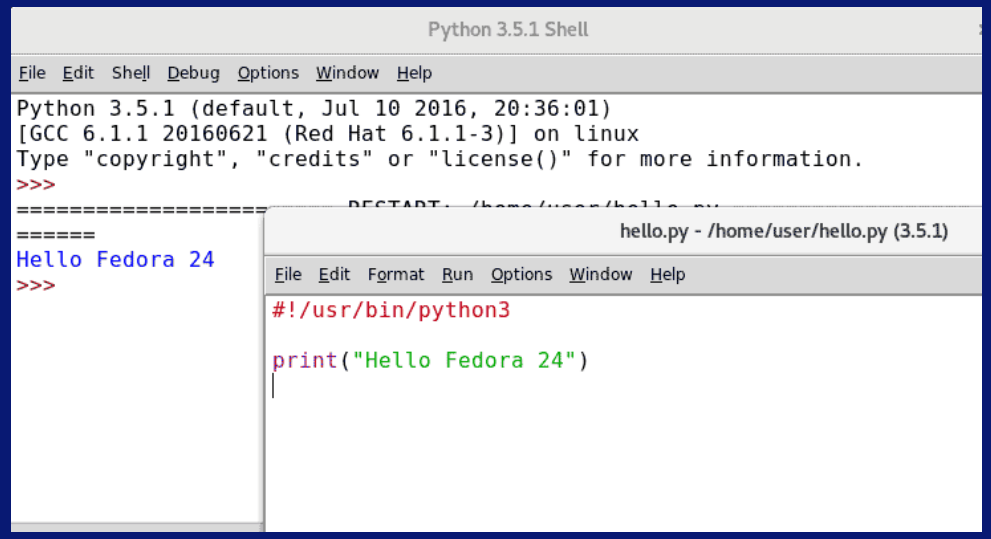
  Ex: IDLE, PyCharm, Jupyter, Spyder are some Python IDEs

Python interpreter should be installed before using any of the above editors and IDEs

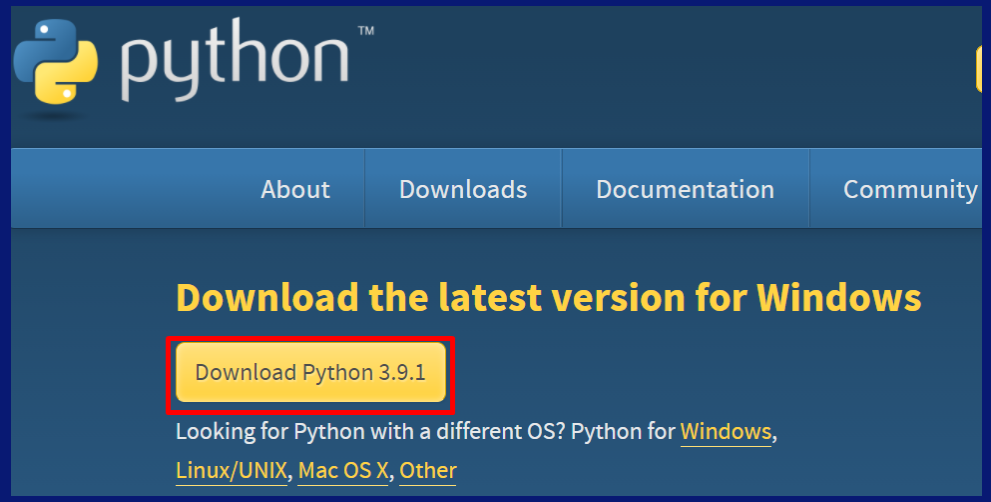
Code Editors and IDEs

Installing IDLE

- IDLE (Integrated Development and Learning Environment) is a default editor that comes with Python
- It is one of the best Python IDE software which helps a beginner to learn Python easily
- It is optional for many Linux distributions and can be installed by package managers like APT, YUM etc
- It consists of a multi-window text editor with syntax highlighting and integrated debugger with stepping, persistent breakpoints, and call stack visibility



- Open a browser window and navigate to https://www.python.org/downloads/ to download the latest python installer



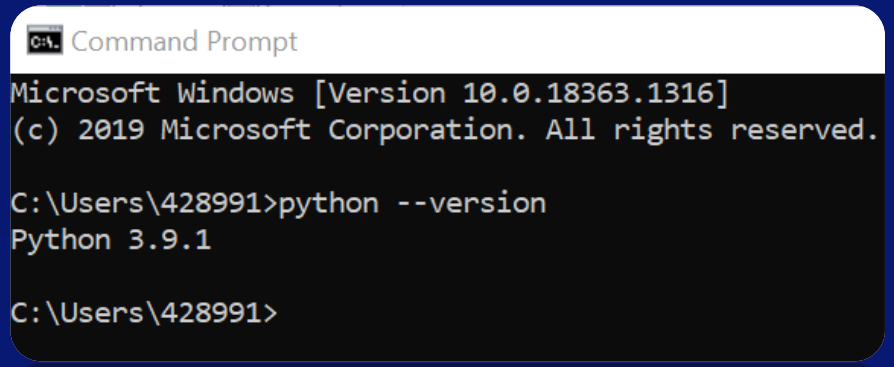
Code Editors and IDEs

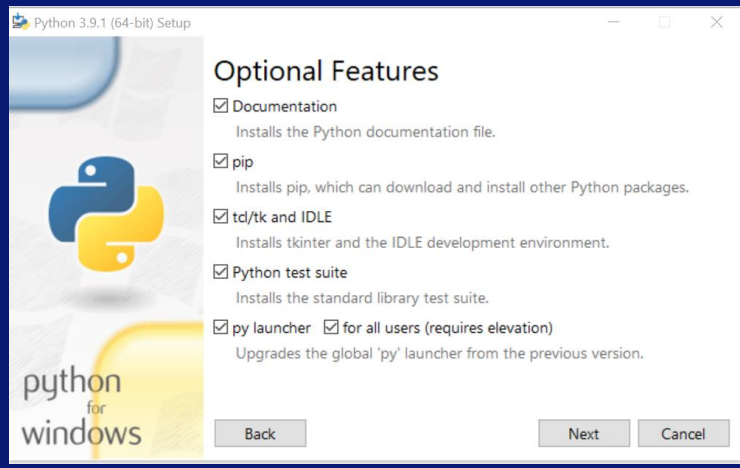
Installing IDLE

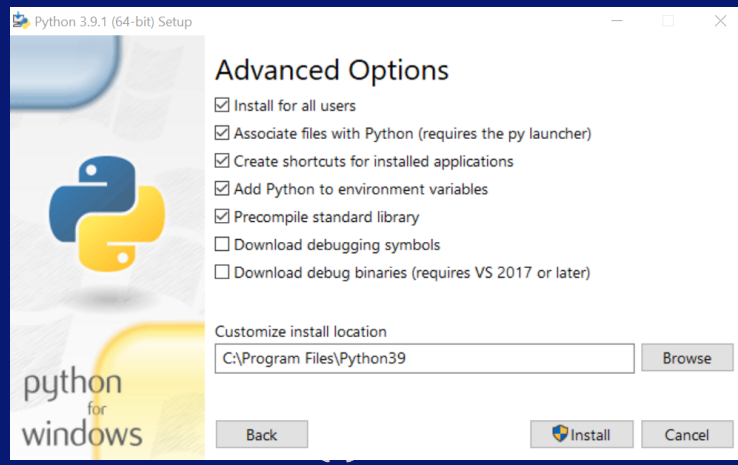
For Windows

- Open a browser window and navigate to https://www.python.org/downloads/ to download the latest python installer
- Double click on the downloaded file and click on Run/Install
- Select the checkboxes as shown in the images
- Click on 'Customize installation' button to customize the installation location and which additional features get installed, including pip and IDLE
- The Install launcher for all users (recommended) checkbox is checked default. This means every user on the machine will have access to the py.exe launcher.
- Click on 'Install' to install the software

- To check the installation, run `python --version` or `py -3 --version` in the windows command prompt, which should display the version of python installed







For Linux, follow the steps at https://realpython.com/installing-python/

Code Editors and IDEs

Visual Studio Code

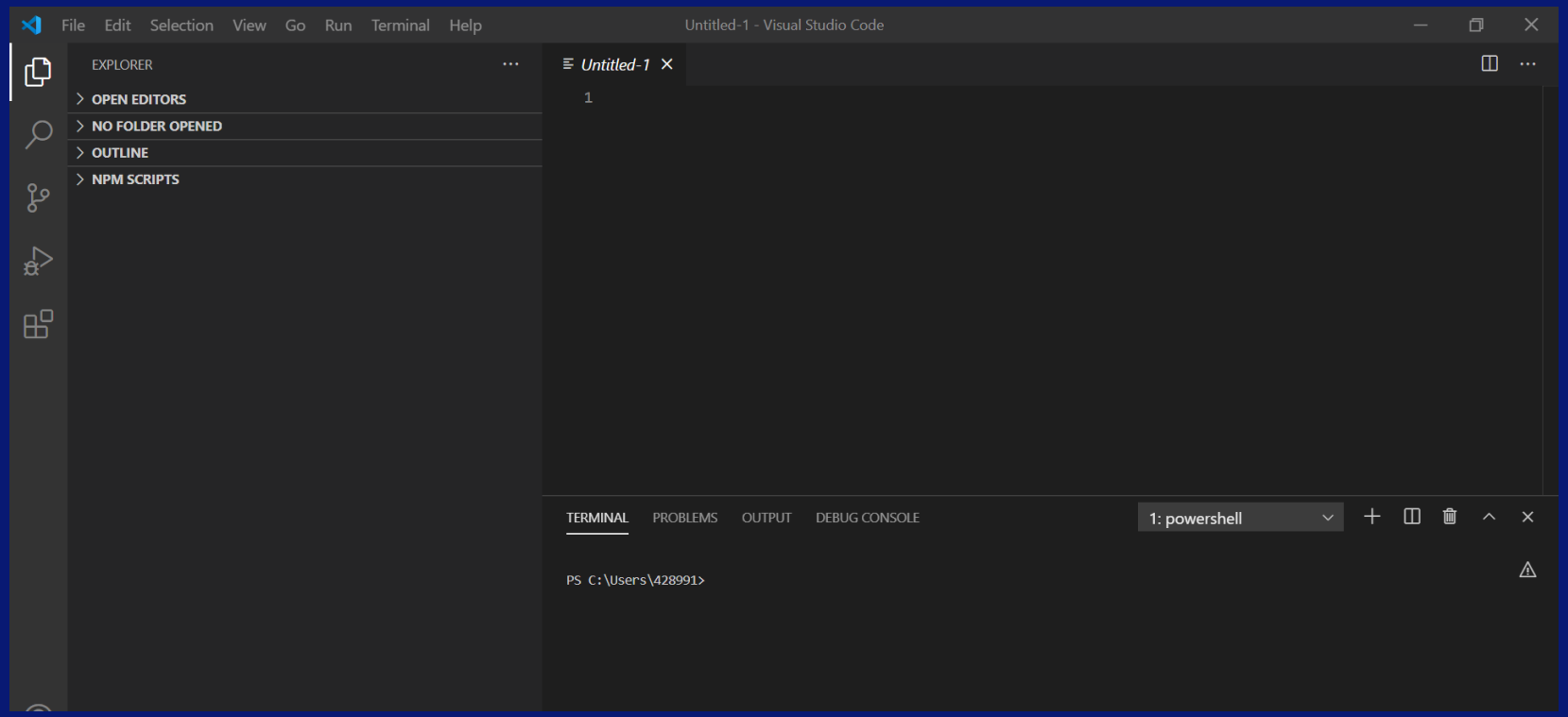
- Visual Studio Code (aka VS Code) is a full-featured code editor available for Linux, Mac OS X, and Windows platforms
- Small and light-weight, but full-featured, VS Code is open-source, extensible, and configurable for almost any task
- Has support for more than 4700 extensions/plugins
- You can add a new language to the environment, such as Python, via a plugin
- Simply download and install the corresponding plugin to adapt it to the environment
- It has integrated powerful code auto-completion engine (IntelliSense), a debugging console, and a terminal to launch server commands
- VS Code will recognize your Python installation and libraries automatically

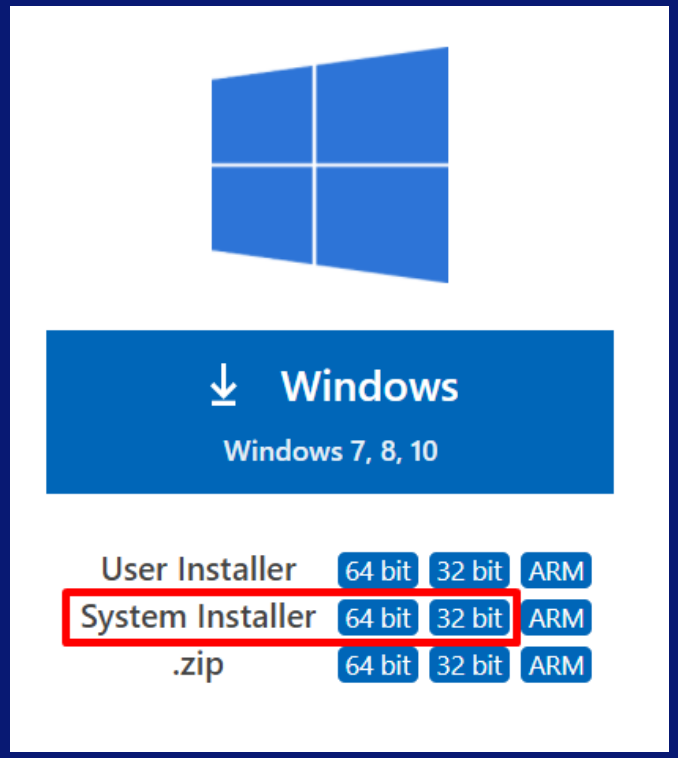
Using VS code for Python and setting up the editor, worth reading!
https://code.visualstudio.com/docs/python/python-tutorial#_prerequisites

Code Editors and IDEs

Installing VS Code

- Go to https://code.visualstudio.com/download to download the latest version for your OS
- Download System Installer 64/32 bit depending on your OS arch
- Install with default settings and open the VS Code



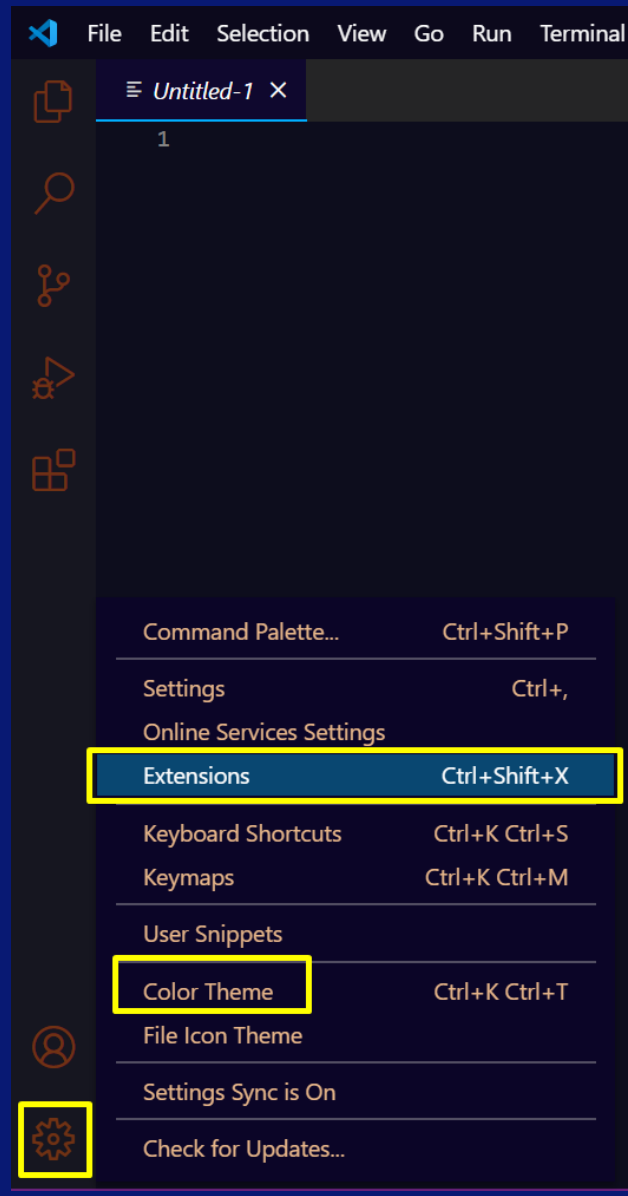


Checkout these awesome tools for VS Code: https://github.com/kunchalavikram1427/awesome-vscode
Themes: https://dev.to/thegeoffstevens/50-vs-code-themes-for-2020-45cc

Code Editors and IDEs

Installing VS Code Plugins

- VS Code comes with support for many plugins to extend its functionality for various languages
- For python, we need to install the following plugins from Extensions marketplace
  - Python
  - Pylance
  - Python Preview
  - Tabnine Autocomplete AI



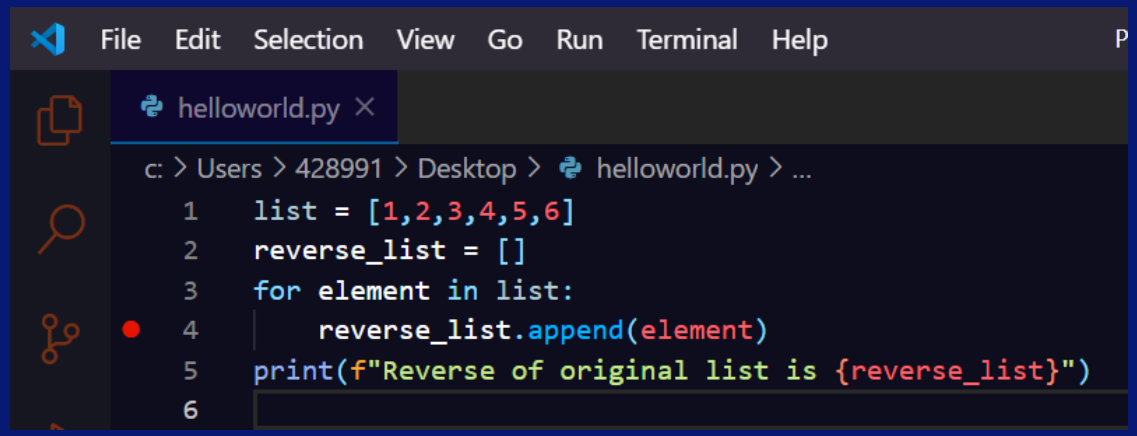
Details on how to use the plugin are in the installation page of the plugin.
Checkout how to use Python preview plugin to debug the code as shown in installation page

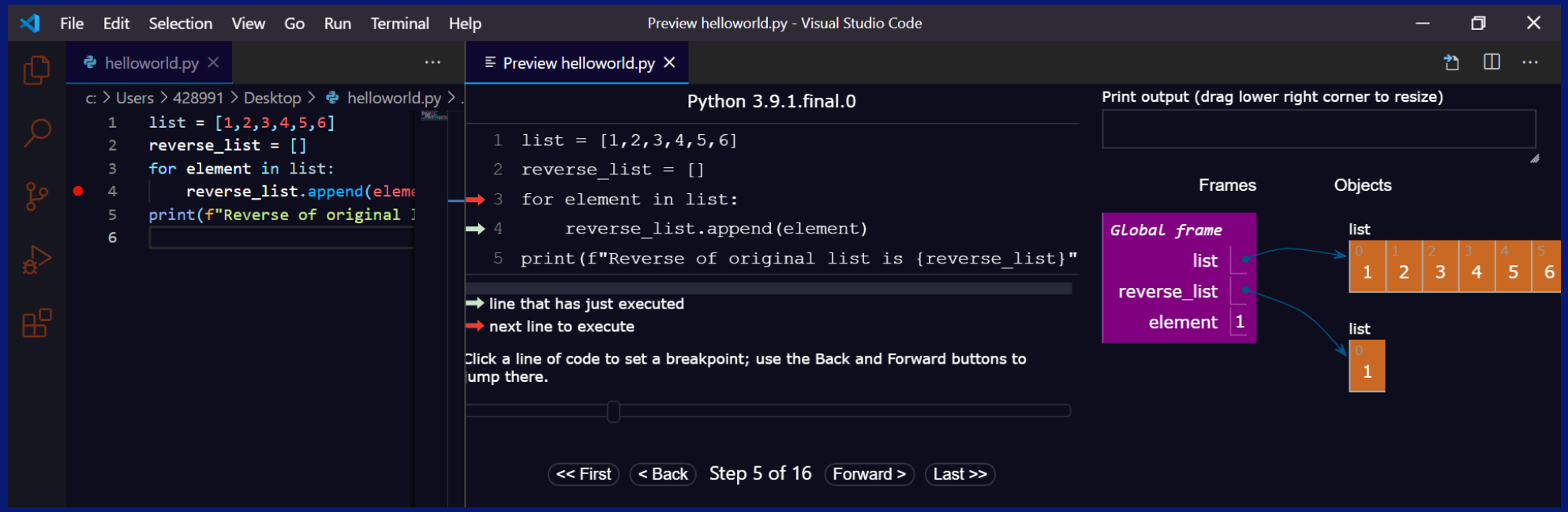
Code Editors and IDEs

Installing VS Code Plugins

- Python preview plugin lets you debug the code and see the variables and other data directly in the IDE

More info on usage here
https://marketplace.visualstudio.com/items?itemName=dongli.python-preview

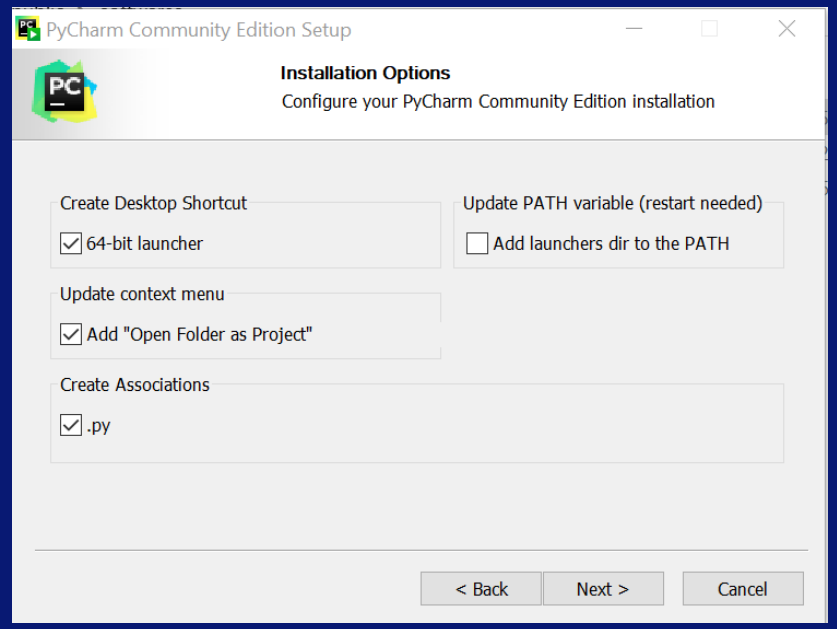




Code Editors and IDEs

PyCharm

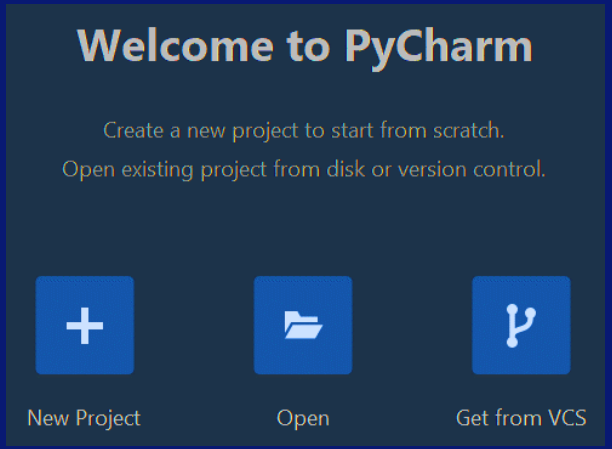
- PyCharm is an integrated development environment developed by JetBrains
- Offers Community version(free) and the Professional version(paid) which offers advanced features such as full database management and a multitude of important Frameworks such as Django, Flask, Google App, Engine, Pyramid, and web2py
- The Community version offers different features such as syntax highlighting, auto-completion, debugging and live code verification
- Download PyCharm from https://www.jetbrains.com/pycharm/download/ and install with below shown settings

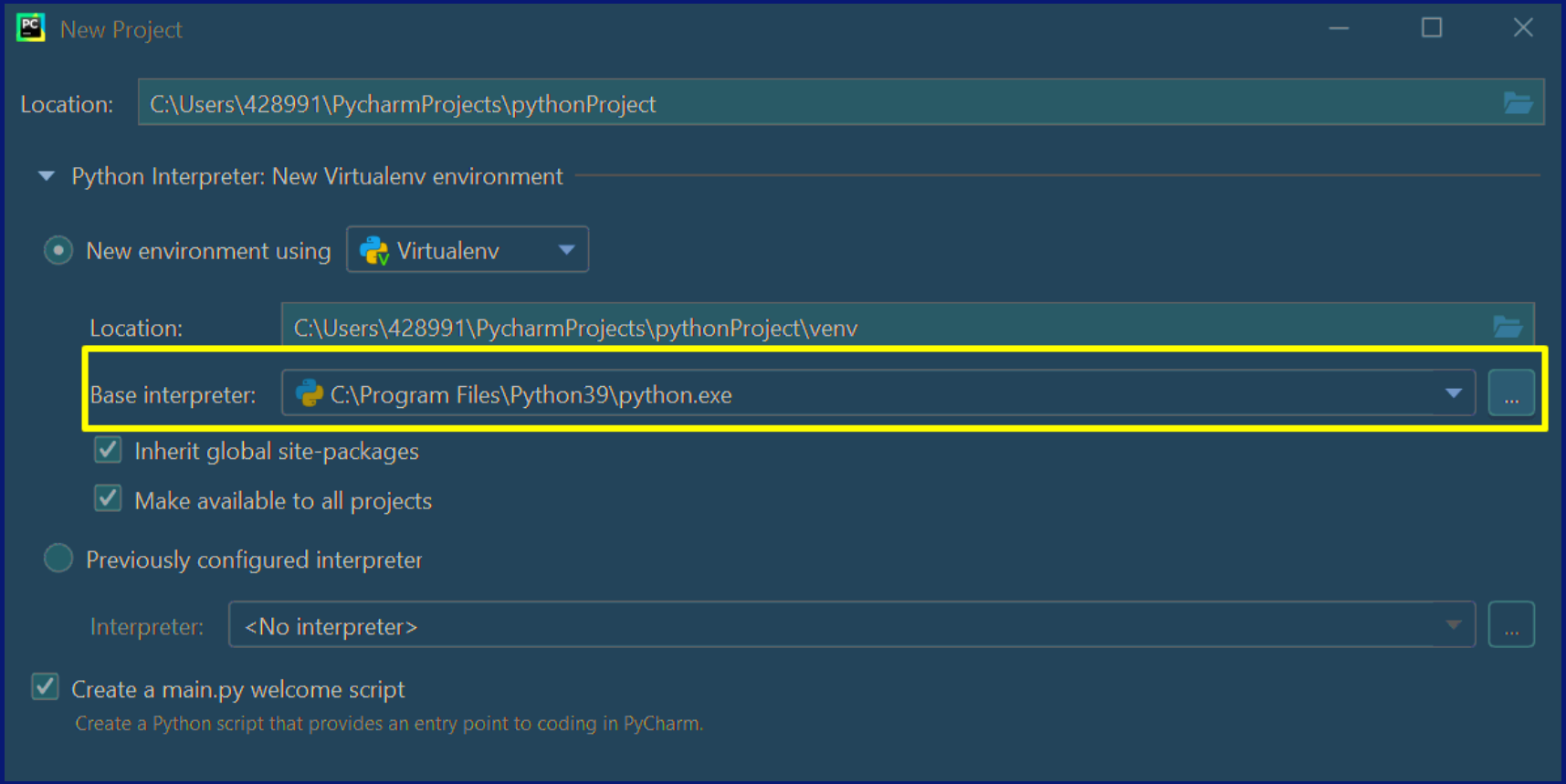


Code Editors and IDEs

PyCharm: Setting the Interpreter

- Open PyCharm and click on New Project
- Locate the base interpreter in the Python installation directory and click on create

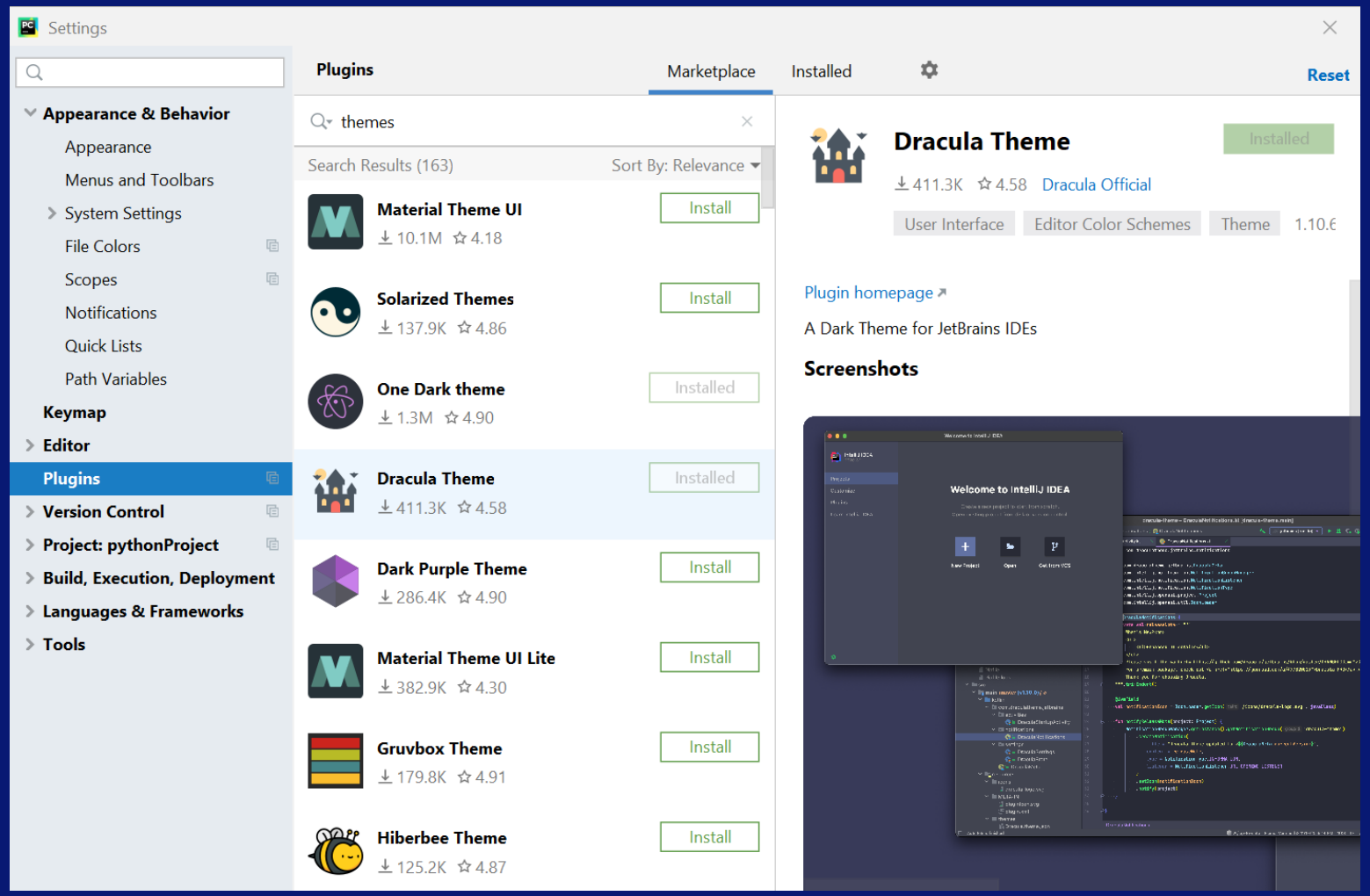




Code Editors and IDEs

PyCharm: Customize with plugins and themes

- Open PyCharm, click on File -> Settings



Code Editors and IDEs

Anaconda Distribution: Data science toolkit

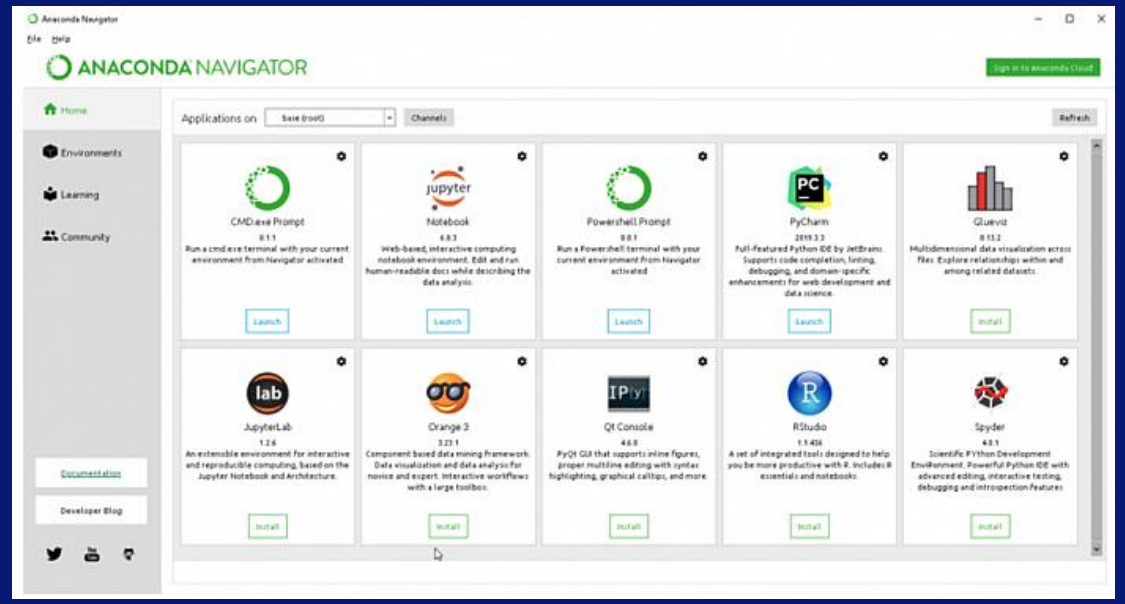
- Anaconda is a distribution of the Python and R programming languages for scientific computing i.e., data science, machine learning applications, large-scale data processing, predictive analytics, etc
- It is the toolkit with thousands of open-source packages & libraries and simplifies package management and deployment
- Package versions in Anaconda are managed by the package management system conda and can be installed by `conda install` command
- Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI using PIP as well as the conda package
- The difference between conda and the PIP(pip package manager) is in how package dependencies are managed
- When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages which results in erroneous results with existing code
- In contrast, conda analyses the current environment including everything currently installed, and shows a warning if this cannot be done

More info on PyPI and PIP in the later part of the course

Code Editors and IDEs

Anaconda Navigator

- Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands
- Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them
- The following applications are available by default in the Navigator
  - Jupyter Lab
  - Jupyter Notebook
  - Qt Console
  - Spyder
  - PyCharm
  - Orange
  - RStudio
  - Visual Studio Code



Installing Anaconda

- Download from https://www.anaconda.com/products/individual and install it

Code Editors and IDEs

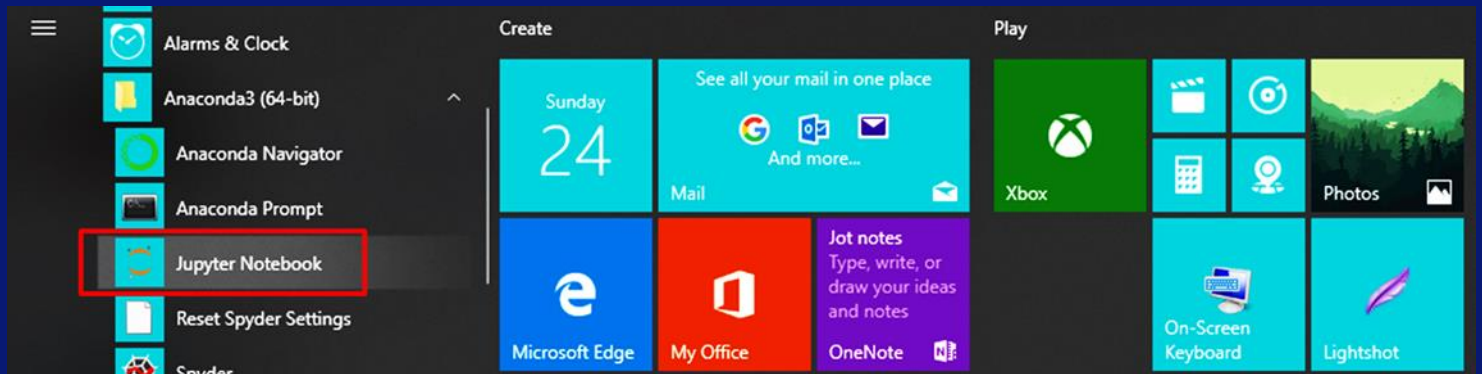
Jupyter Notebooks

- Jupyter notebook, formerly known as the IPython notebook, is a flexible tool that helps you create readable analyses, as you can keep code, images, comments, formulae and plots together for easy presentation
- It is one of the best Python IDE that supports for numerical simulation, data cleaning machine learning data visualization, statistical modelling and integrated data science libraries (matplotlib, NumPy, Pandas)
- Used for sharing documents that contain live code, equations, visualizations and narrative text
- Jupyter has support for over 40 different programming languages and Python is one of them

Code Editors and IDEs

Launch Jupyter Notebooks

- Launch Jupyter notebook from start menu in Windows or directly from Anaconda Navigator
- Once launched, paste the URL in a browser to access Jupyter Notebooks. In most cases, browser automatically opens!



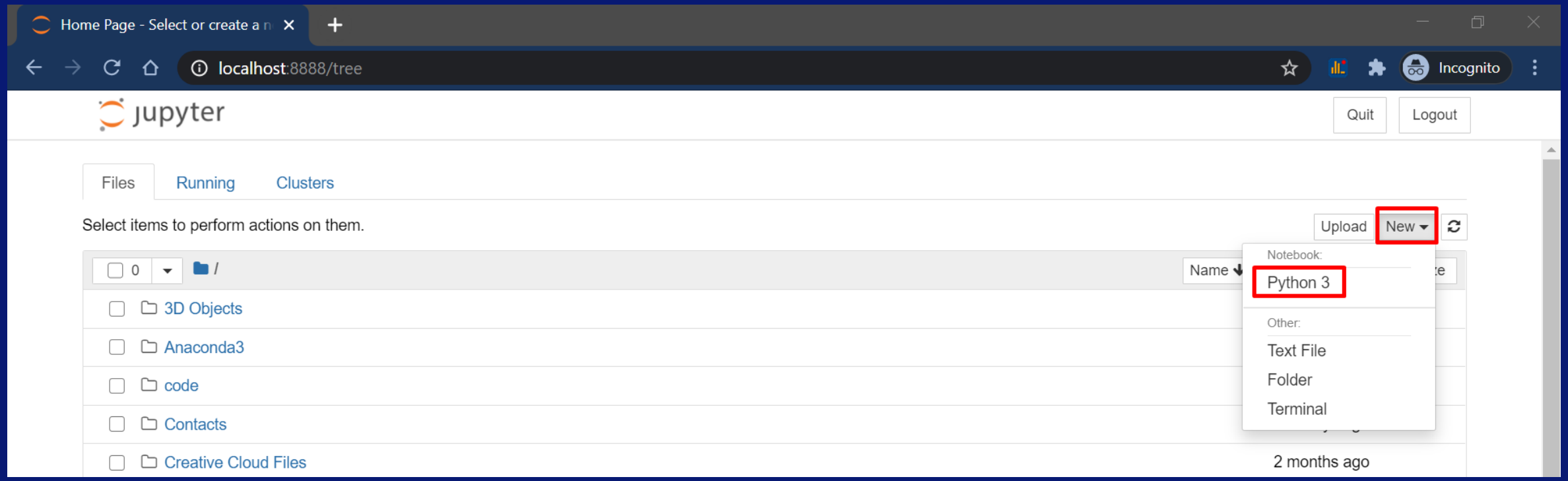
Jupyter Notebook (Anaconda3)

```
[I 18:08:55.900 NotebookApp] JupyterLab extension loaded from C:\Users\428991\Anaconda3\lib\site-packages\jupyterlab
[I 18:08:55.901 NotebookApp] JupyterLab application directory is C:\Users\428991\Anaconda3\share\jupyter\lab
[I 18:08:55.907 NotebookApp] Serving notebooks from local directory: C:\Users\428991
[I 18:08:55.907 NotebookApp] The Jupyter Notebook is running at:
[I 18:08:55.908 NotebookApp] http://localhost:8888/?token=601ea0065fa557bc316cb4be0d08a7888cf2129b0520a461
[I 18:08:55.908 NotebookApp]  or http://127.0.0.1:8888/?token=601ea0065fa557bc316cb4be0d08a7888cf2129b0520a461
[I 18:08:55.908 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 18:08:56.012 NotebookApp]

    To access the notebook, open this file in a browser:
        file:///C:/Users/428991/AppData/Roaming/jupyter/runtime/nbserver-26252-open.html
    Or copy and paste one of these URLs:
        http://localhost:8888/?token=601ea0065fa557bc316cb4be0d08a7888cf2129b0520a461
     or http://127.0.0.1:8888/?token=601ea0065fa557bc316cb4be0d08a7888cf2129b0520a461
```

Code Editors and IDEs

Launch Jupyter Notebooks

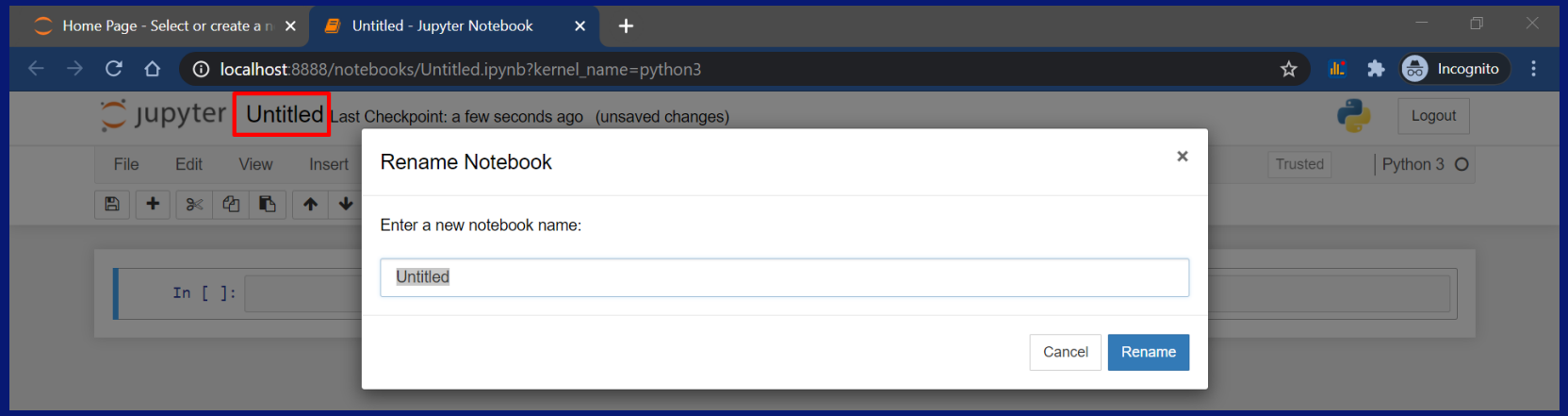
- When the notebook opens in your browser, you will see the Notebook Dashboard
- Dashboard will show a list of the notebooks, files, and subdirectories from the user's home directory
- Click on New and select Python3 to start a python notebook



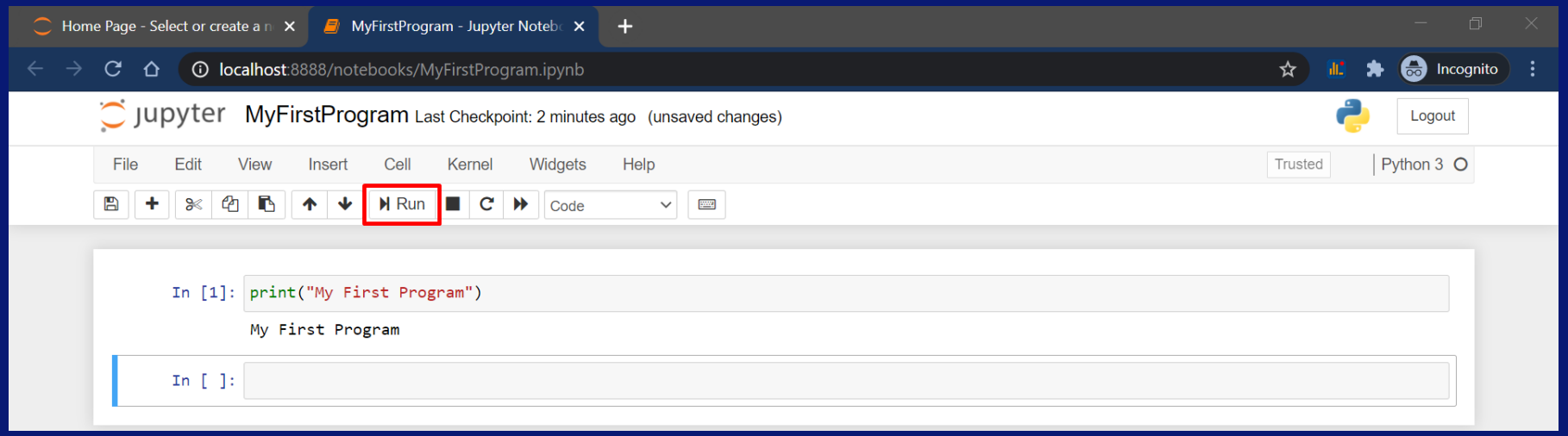
Code Editors and IDEs

Launch Jupyter Notebooks

- Double click on Untitled to rename the notebook



- Write a simple python statement and click on Run to execute the statement in the cell



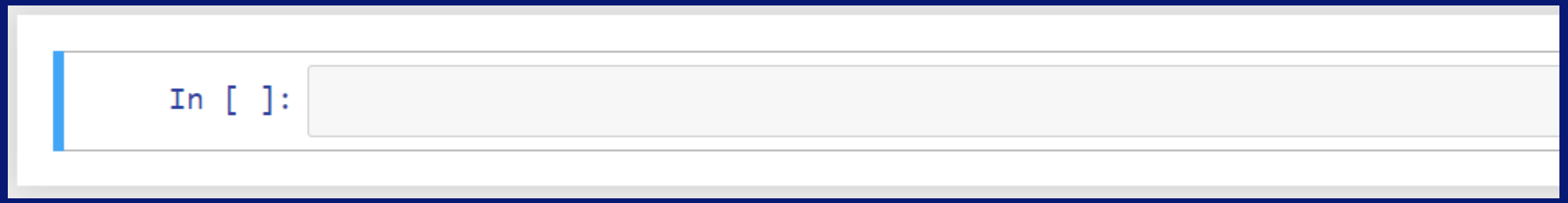
/kunchalavikram1427

Code Editors and IDEs

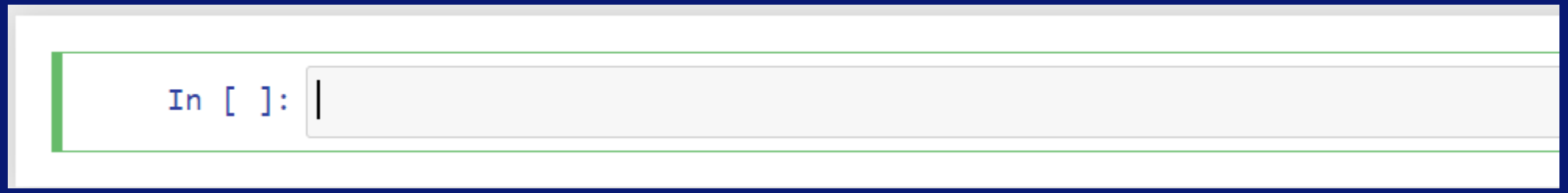
Jupyter Notebooks tips

Command mode vs. Edit mode

- **Command mode** binds the keyboard to notebook level actions. Actions like cell copy, delete, paste can be performed by keyboard keys. Indicated by a grey cell border with a blue left margin



- **Edit mode** when you're typing in a cell. Indicated by a green cell border. Keyboard keys are utilized for typing the instructions. To switch to command mode, click outside the cell active area

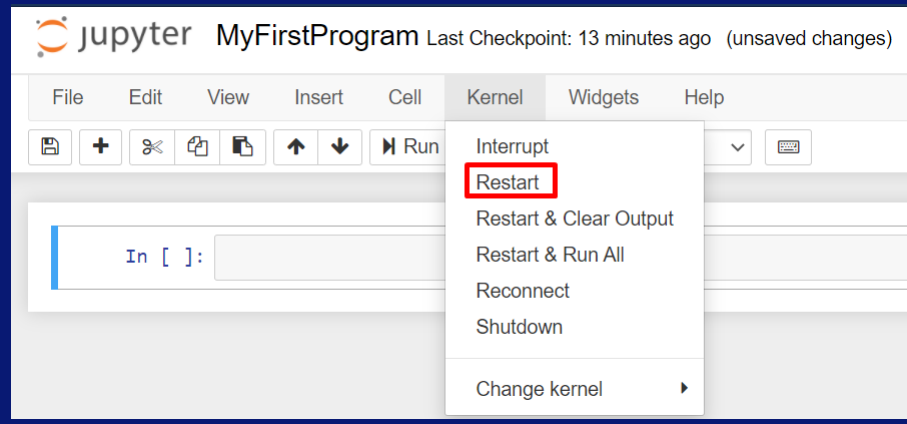


Code Editors and IDEs

Jupyter Notebooks tips

Command mode shortcuts

- Ctrl + enter - run cell and stay in same cell
- Alt + enter - run cell, select below cell in edit mode
- Shift + enter - run cell, select below cell in command mode
- A - insert cell above
- B - insert cell below
- C - copy cell
- V - paste cell
- DD or X - delete selected cell
- Shift + M - merge selected cells, or current cell with cell below if only one cell selected
- II - interrupt kernel, once interrupted restart the kernel as shown
- 00 - restart kernel
- M - change cell to markdown mode (good for documentation)



Code Editors and IDEs

Jupyter Notebooks tips

Edit mode shortcuts

- Ctrl+ / - toggle comment lines
- tab - code completion or indent
- shift + tab - tooltip

```
In [2]: # This is shortcut for ctrl + /

In [ ]: prin
        print
        PrintHood/
```

```
In [ ]: print()

Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
```

Code Editors and IDEs

Jupyter Notebooks tips

Pretty Display of Variables

- By default Jupyter notebooks displays output of only last variable
- Value of multiple statements can be displayed at once by using below command

```
from IPython.core.interactiveshell import InteractiveShell
InteractiveShell.ast_node_interactivity = "all"
```

```
In [1]: a = 1
        b = 2
        a
        b

Out[1]: 2

In [2]: from IPython.core.interactiveshell import InteractiveShell
        InteractiveShell.ast_node_interactivity = "all"

In [3]: a = 1
        b = 2
        a
        b

Out[3]: 1

Out[3]: 2

In [ ]:
```

Code Editors and IDEs

Jupyter Notebooks tips

Easy links to documentation

- Using ? before a command gives access to the Docstring for quick reference on syntax

Ex: `?print, ?input`

```
In [4]: ?print

In [ ]:
```

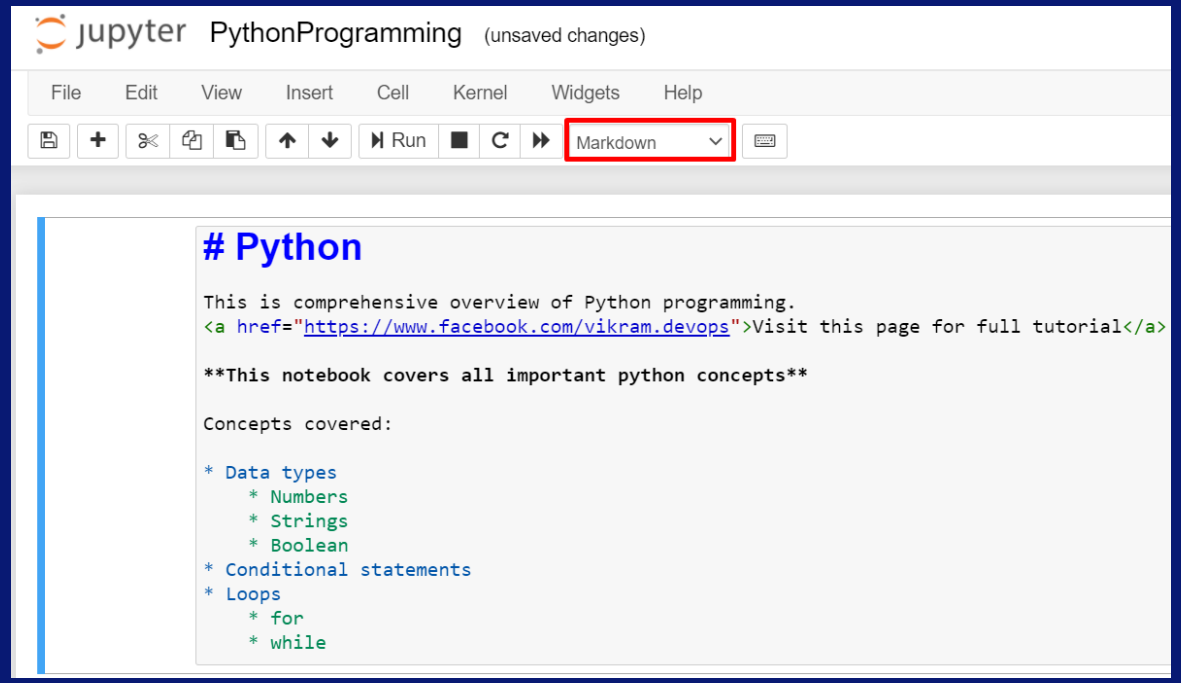
```
Docstring:
print(value, ..., sep=' ', end='\n', file=sys.stdout, flush=False)

Prints the values to a stream, or to sys.stdout by default.
Optional keyword arguments:
file:  a file-like object (stream); defaults to the current sys.stdout
sep:   string inserted between values, default a space.
end:   string appended after the last value, default a newline.
flush: whether to forcibly flush the stream.
Type:      builtin_function_or_method
```

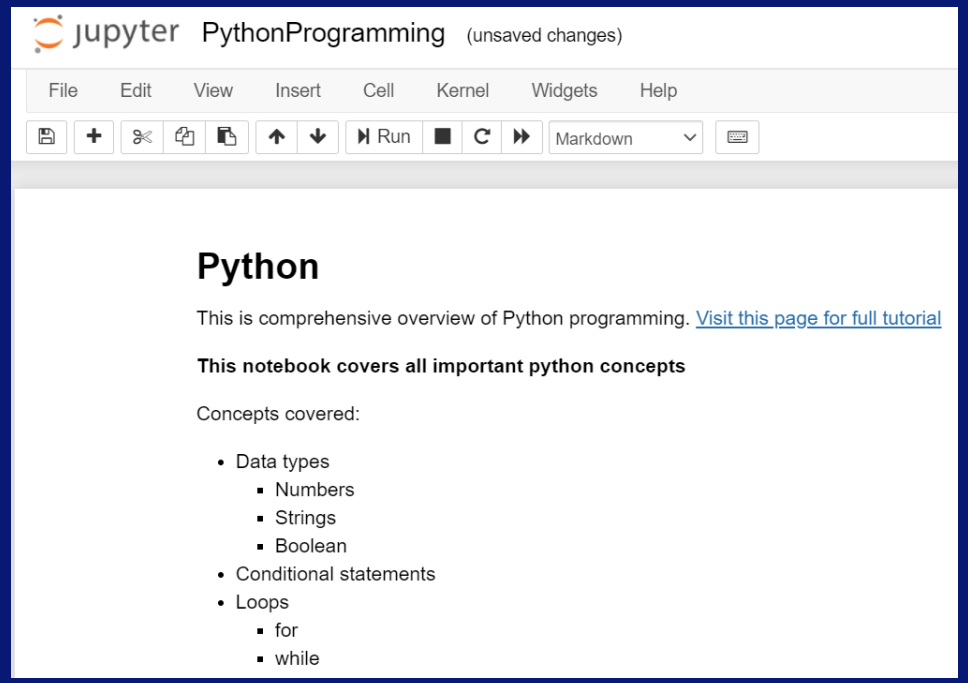
Code Editors and IDEs

Jupyter Notebooks tips

Mark down files to documentation

- Write markdown files for easy documentation and sharing
- Click on Run to render the file





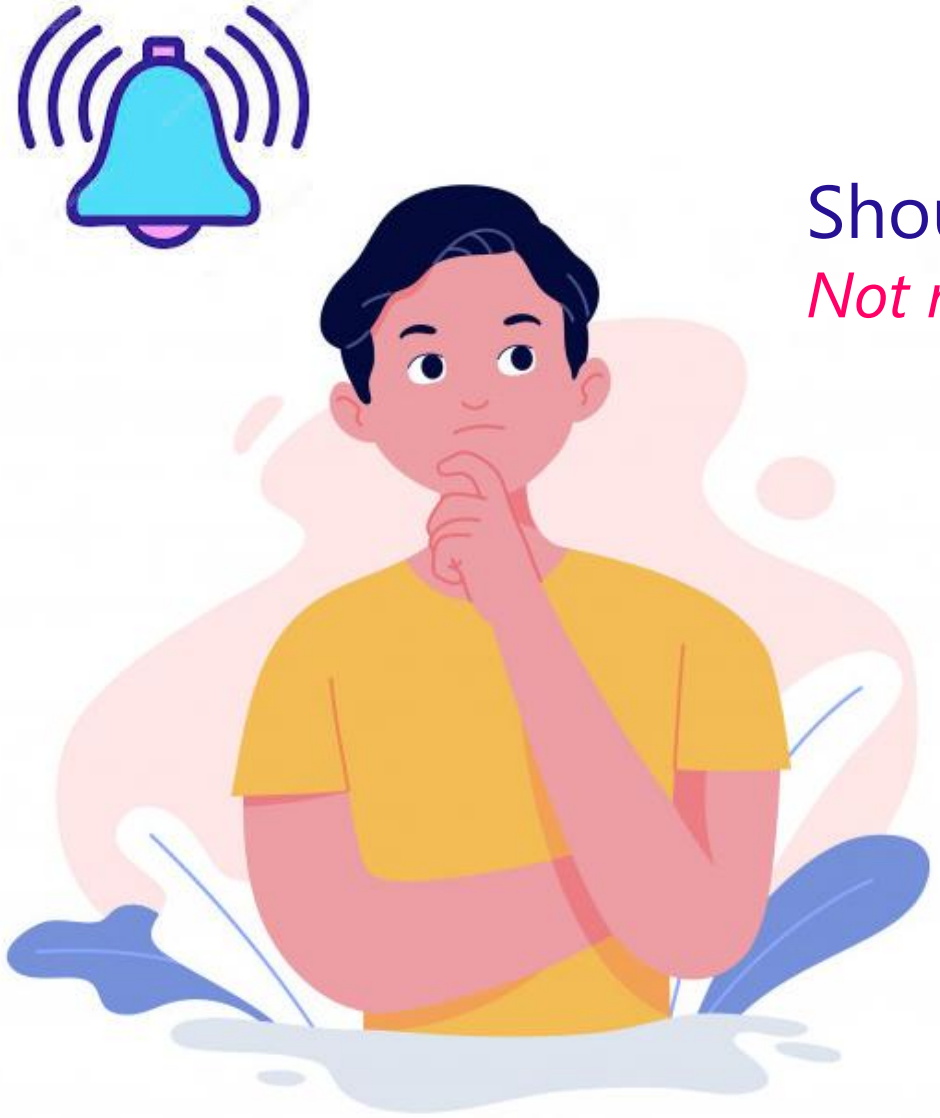
Code Editors and IDEs

Jupyter Notebooks tips

Execution time for code: %timeit

- %timeit automatically determine the execution time of the single-line Python statement that follows it
- For Multiline Python statements use %%timeit

```
In [1]: %timeit squares = [ num for num in range(1000)]

        42.5 µs ± 7.64 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)

In [2]: %%timeit
        squares = []
        for num in range(10000):
            squares.append(num ** 2)

        2.71 ms ± 130 µs per loop (mean ± std. dev. of 7 runs, 100 loops each)

In [ ]:
```



Should beginner programmers avoid using IDEs?

Not necessarily

Many text editors with language specific plugins or IDEs support syntax highlighting, auto-complete, or smart refactoring etc., which simplifies code building and speeds up the development.
IDEs are what probably most professional programmers use to write code, and they make programming easier to learn.
However, IDEs do hide things from you. They hide what software is really being used, for example, compiling and running the programs without using an IDE.

So, use an IDE or whatever makes you productive when you are in the business of software development. But try to stay away from them if you are a beginner and you really want to learn what you are doing.

http://blog.thefirehoseproject.com/posts/the-14-things-you-should-never-do-while-learning-to-code/

UP NEXT

Introduction to Python

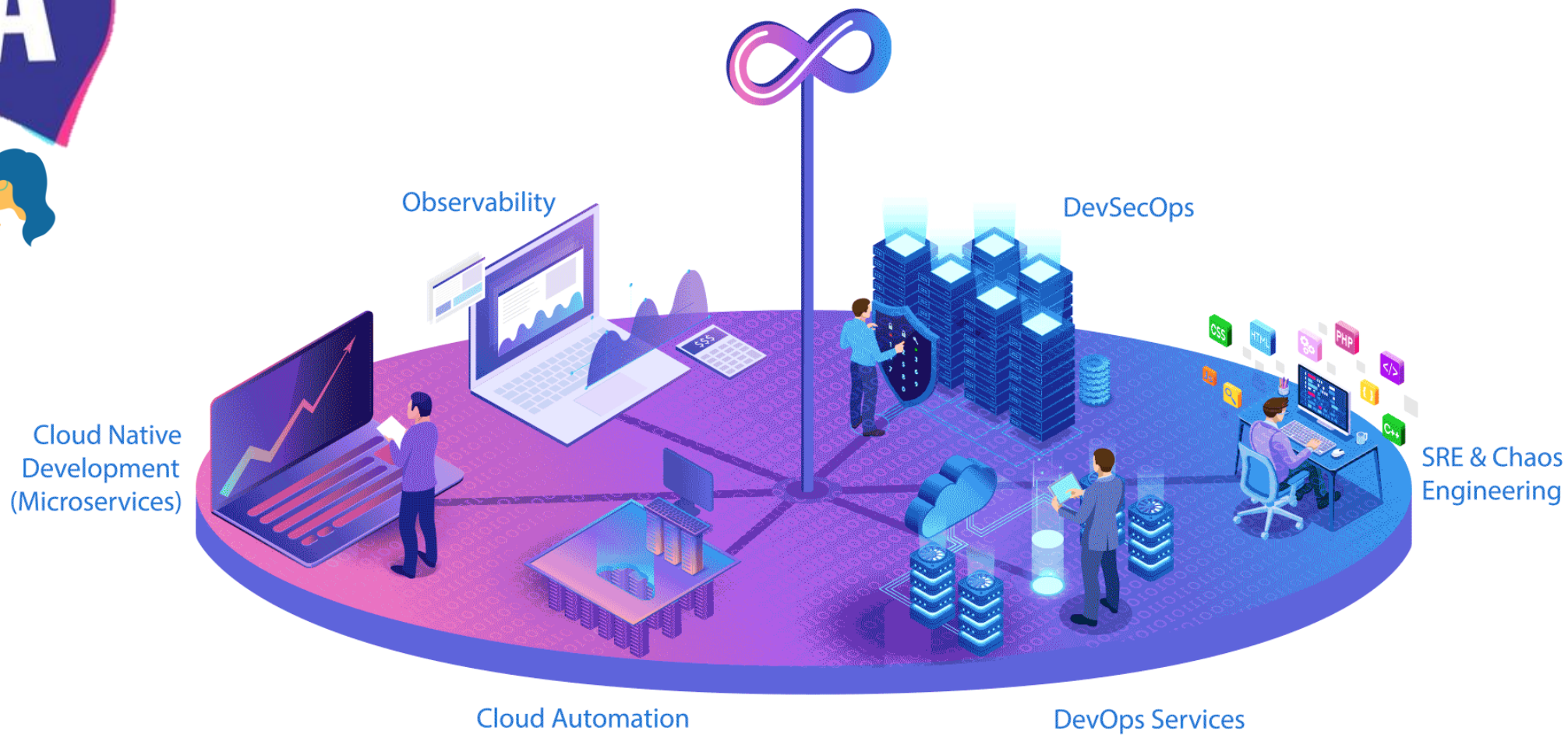
References

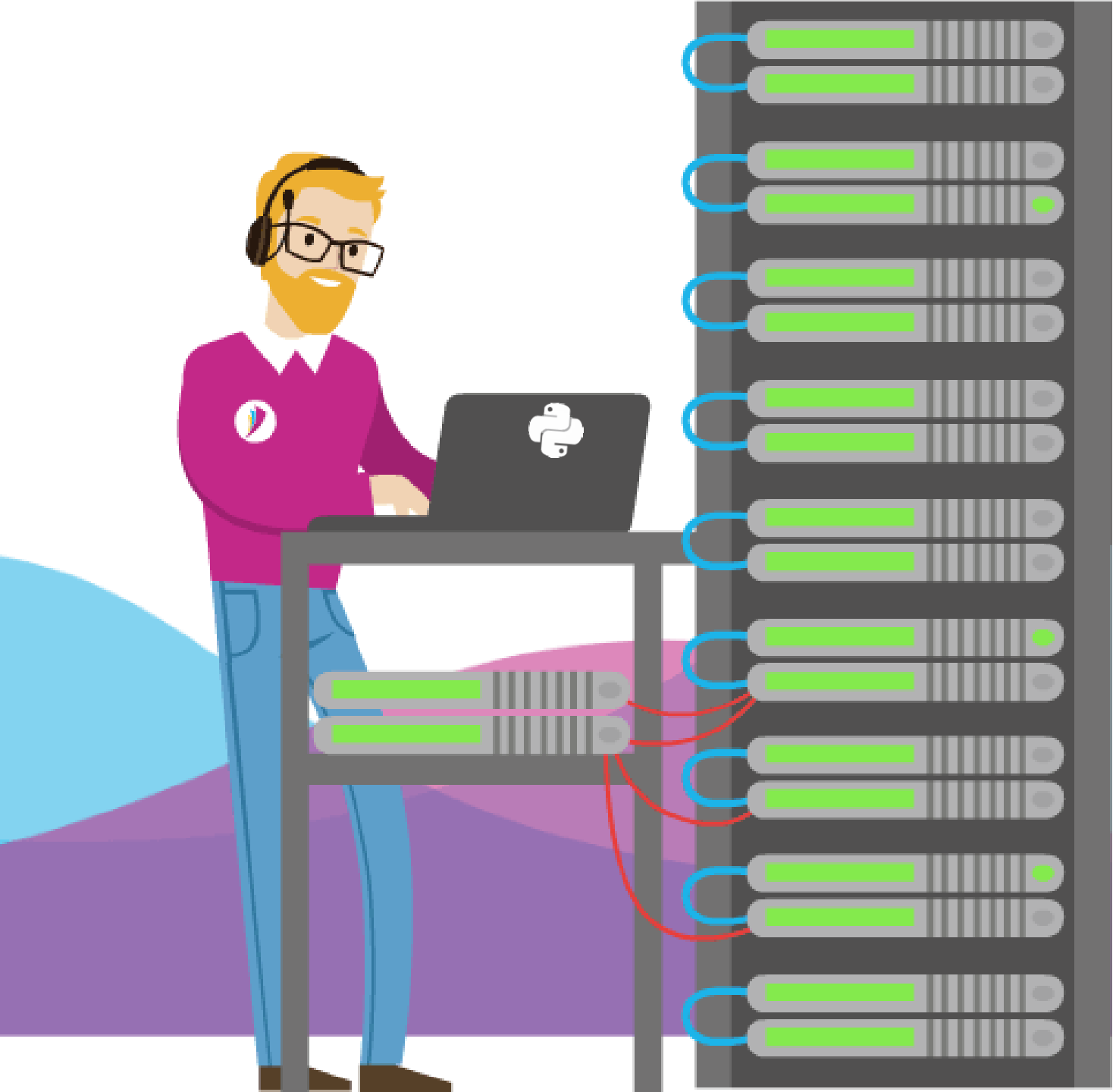
- https://wiki.python.org/moin/BeginnersGuide/Programmers
- https://code.visualstudio.com/docs/python/python-tutorial#_prerequisites
- https://cloudacademy.com/blog/python-what-is-it-and-why-is-it-so-popular/

Subscribe to my Facebook page:
https://www.facebook.com/vikram.devops
and join my group:
https://www.facebook.com/groups/171043094400359
for all latest updates on DevOps, Python and IoT

https://www.youtube.com/channel/UCE1cGZfooxT7-VbqVbuKjMg


Q&A


Observability
DevSecOps
Cloud Native Development (Microservices)
SRE & Chaos Engineering
Cloud Automation
DevOps Services




thank you !