

# Beginners to Advanced

A stylized illustration of a person with dark hair, seen from the side, sitting at a desk and working on a laptop. The person is wearing a light blue long-sleeved shirt. The laptop screen displays a document with text and a small Python logo. The desk is dark blue and features a potted plant with green leaves and a white mug. The entire scene is set within a circular frame. Surrounding this central circle are numerous colorful icons representing various concepts: a lightbulb (idea), a cloud with a circular arrow (cloud services), a gear inside a head (thought), a magnifying glass (search), a circular arrow (refresh/cycle), a microchip (technology), a database cylinder (data), a key (security), a USB symbol, a folder, a document with a checkmark, a right-angle triangle, a square, a circle, a zigzag line, a square with a cross, a square with an arrow, a square with a plus, a square with a minus, a square with a multiply, a square with a divide, a square with a percent, a square with a hash, a square with a dollar sign, a square with a pound sign, a square with a yen sign, a square with a euro sign, a square with a ruble sign, a square with a won sign, a square with a new sheqel sign, a square with a new dollar sign, a square with a new peso sign, a square with a new real sign, a square with a new zloty sign, a square with a new forint sign, a square with a new koruna sign, a square with a new lek sign, a square with a new dracma sign, a square with a new tenge sign, a square with a new dirham sign, a square with a new sheqel sign, a square with a new dollar sign, a square with a new peso sign, a square with a new real sign, a square with a new zloty sign, a square with a new forint sign, a square with a new koruna sign, a square with a new lek sign, a square with a new dracma sign, a square with a new tenge sign, a square with a new dirham sign.

# 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



/kunchalavikram1427



# 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

The image shows two overlapping windows from a Python IDE. The background window is titled 'Python 3.5.1 Shell' and displays the Python interpreter's startup message: 'Python 3.5.1 (default, Jul 10 2016, 20:36:01) [GCC 6.1.1 20160621 (Red Hat 6.1.1-3)] on linux'. It prompts the user to type 'copyright', 'credits', or 'license()' for more information. The foreground window is titled 'hello.py - /home/user/hello.py (3.5.1)' and contains a simple Python script: `#!/usr/bin/python3` followed by `print("Hello Fedora 24")`. Both windows have standard menu bars (File, Edit, Shell, Debug, Options, Window, Help).



# Code Editors and IDEs

## Installing IDLE

### For Windows

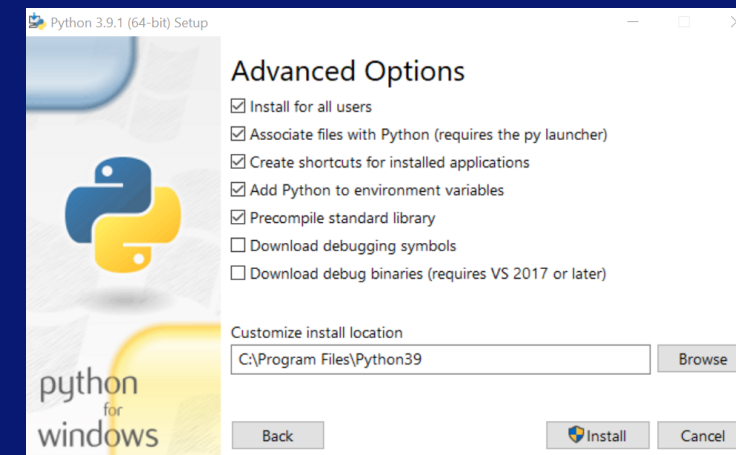
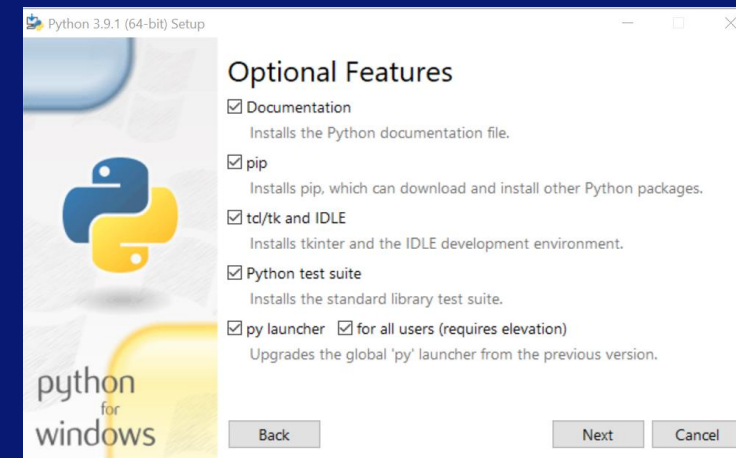
- Open a browser window and navigate to <https://www.python.org/downloads/> to download the latest python installer
- Double click on the downloaded file and click on Run/Install
- Select the checkboxes as shown in the images
- Click on 'Customize installation' button to customize the installation location 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

```
Command Prompt
Microsoft Windows [Version 10.0.18363.1316]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\428991>python --version
Python 3.9.1

C:\Users\428991>
```

 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](https://code.visualstudio.com/docs/python/python-tutorial#_prerequisites)



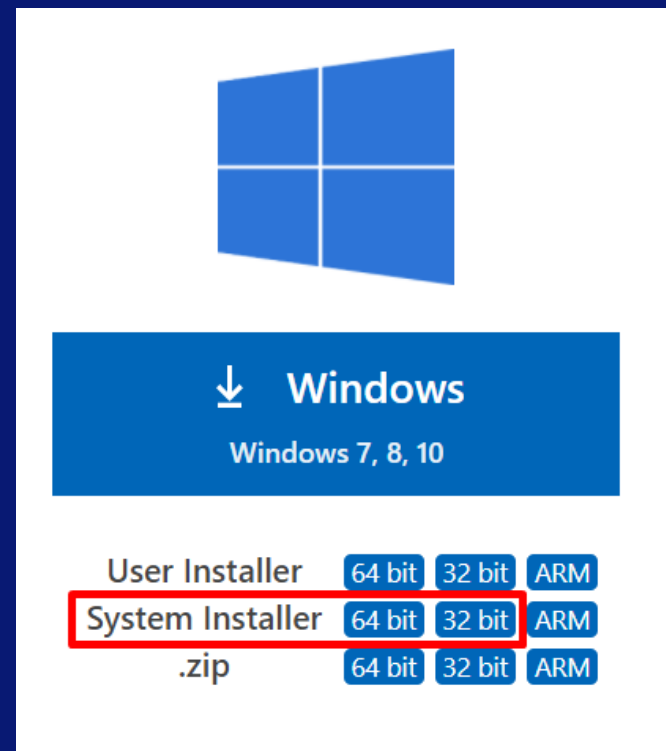
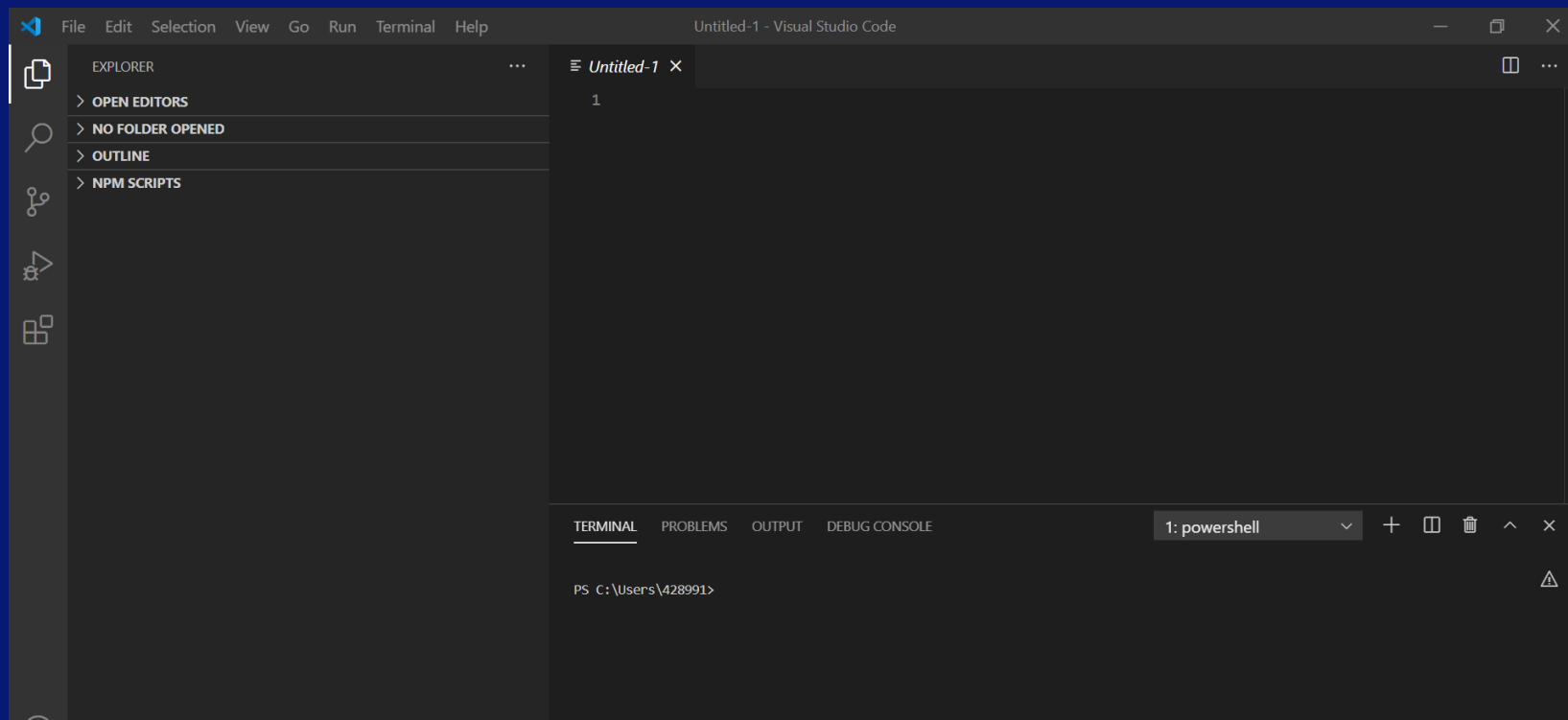




# Code Editors and IDEs

## 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>



/kunchalavikram1427

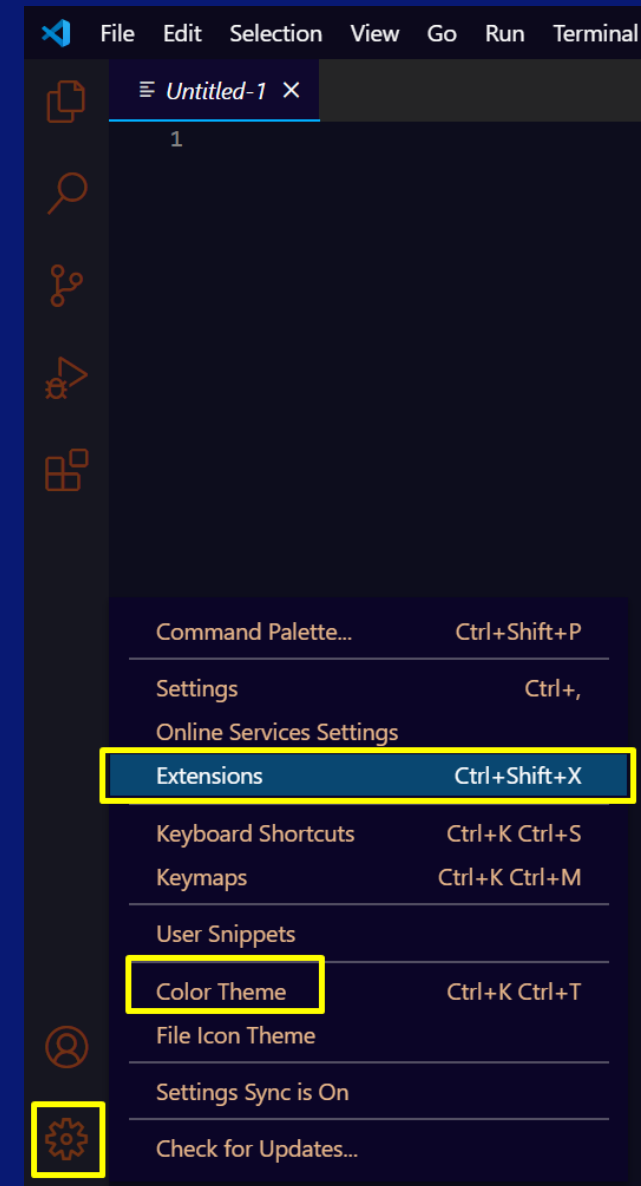




# Code Editors and IDEs

## 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>

```
File Edit Selection View Go Run Terminal Help
helloworld.py X
c: > Users > 428991 > Desktop > helloworld.py > ...
1 list = [1,2,3,4,5,6]
2 reverse_list = []
3 for element in list:
4     reverse_list.append(element)
5 print(f'Reverse of original list is {reverse_list}')
6
```

Python 3.9.1.final.0

```
1 list = [1,2,3,4,5,6]
2 reverse_list = []
3 for element in list:
4     reverse_list.append(element)
5 print(f'Reverse of original list is {reverse_list}')
```

line that has just executed  
next line to execute

Click a line of code to set a breakpoint; use the Back and Forward buttons to jump there.

Print output (drag lower right corner to resize)

Frames

Global frame

list

reverse\_list

element 1

Objects

list

0 1 2 3 4 5 6

1 2 3 4 5 6

list

0 1

1

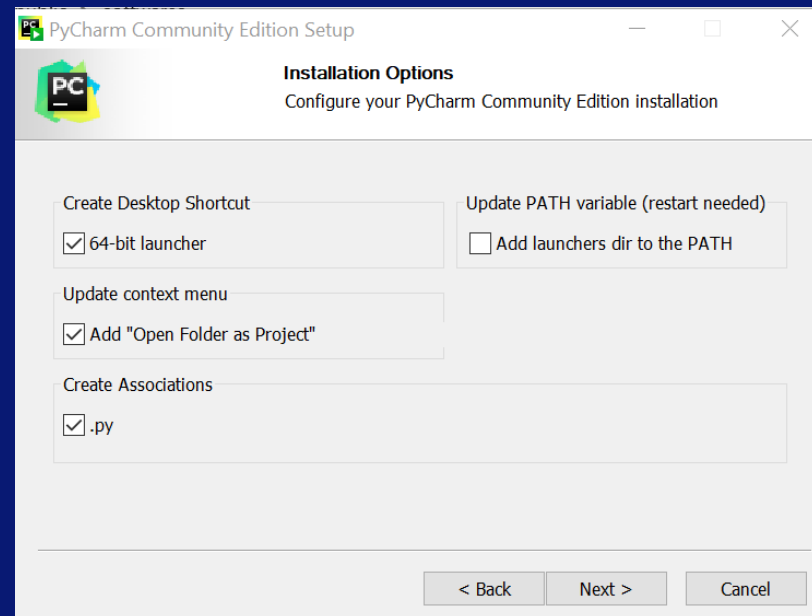
<< First < Back Step 5 of 16 Forward > Last >>



# 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

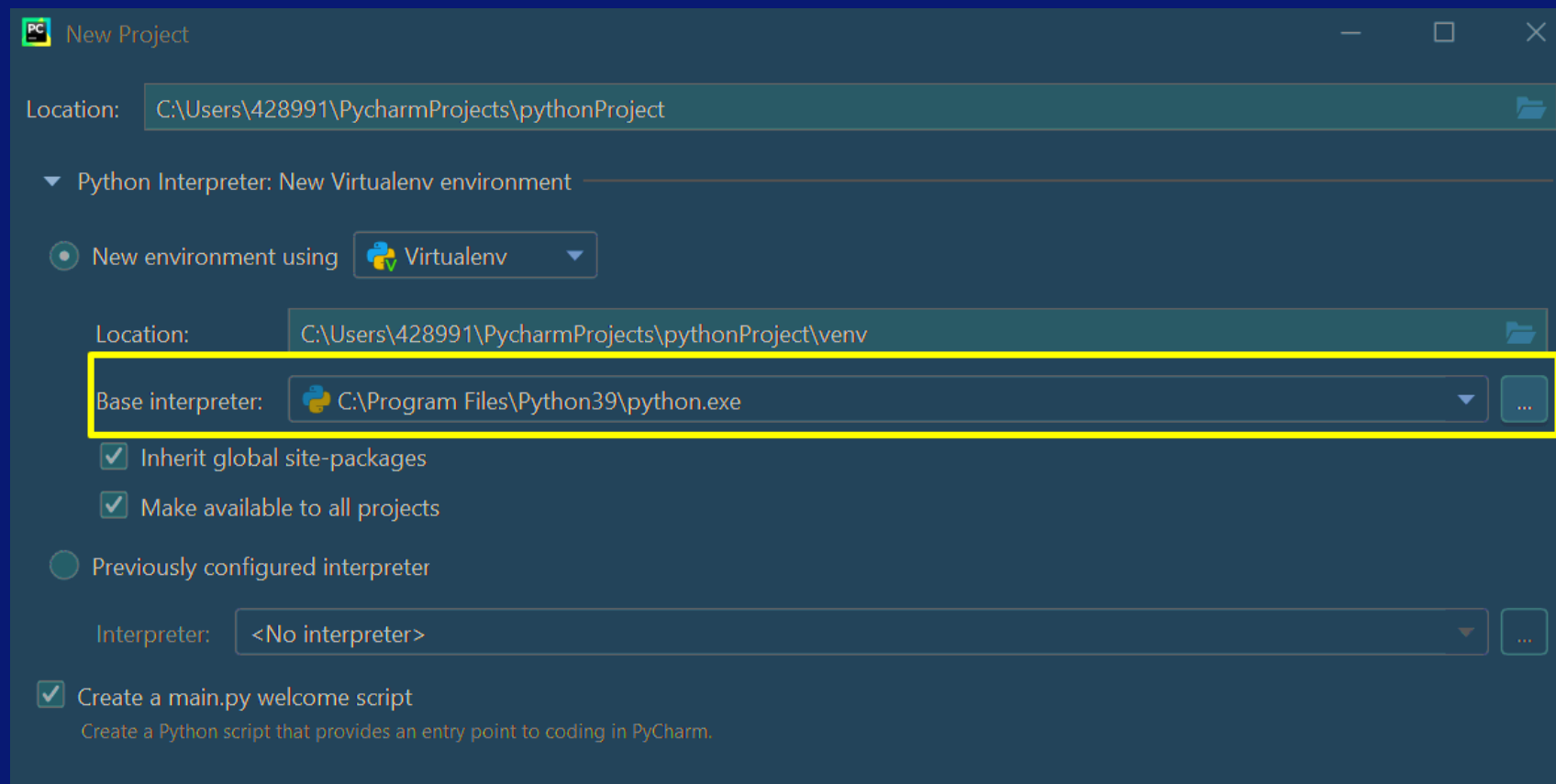
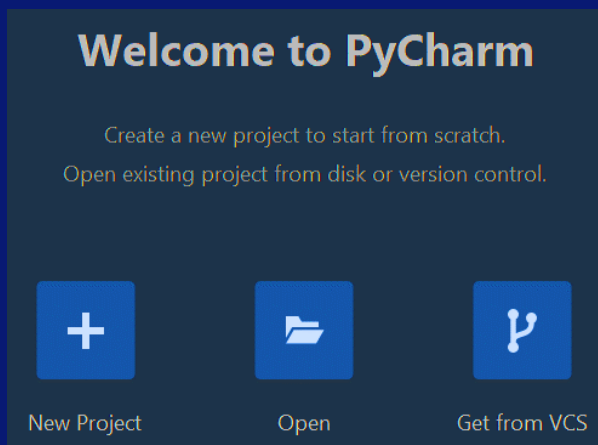




# Code Editors and IDEs

## PyCharm: Setting the Interpreter

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

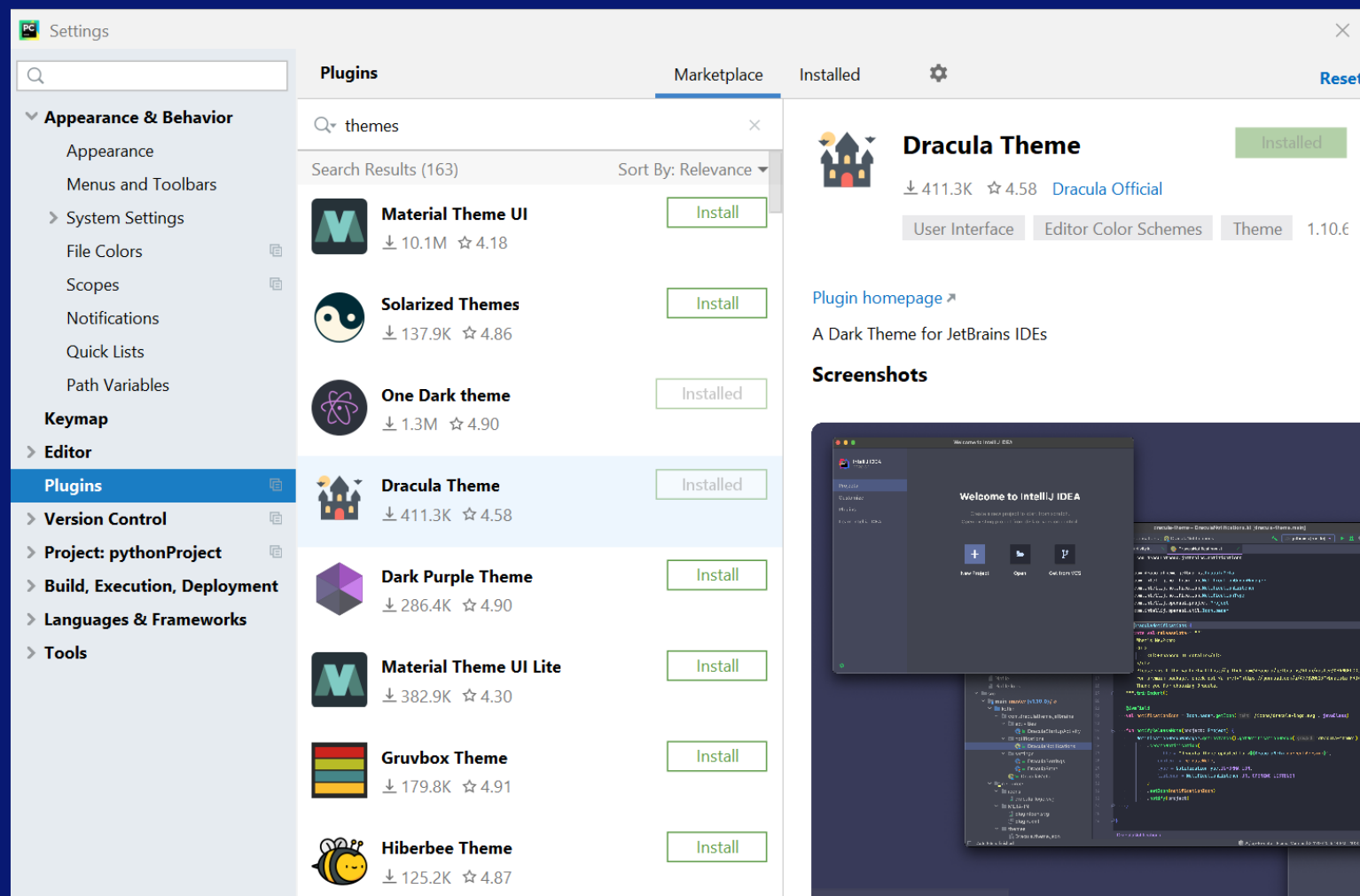




# Code Editors and IDEs

## 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

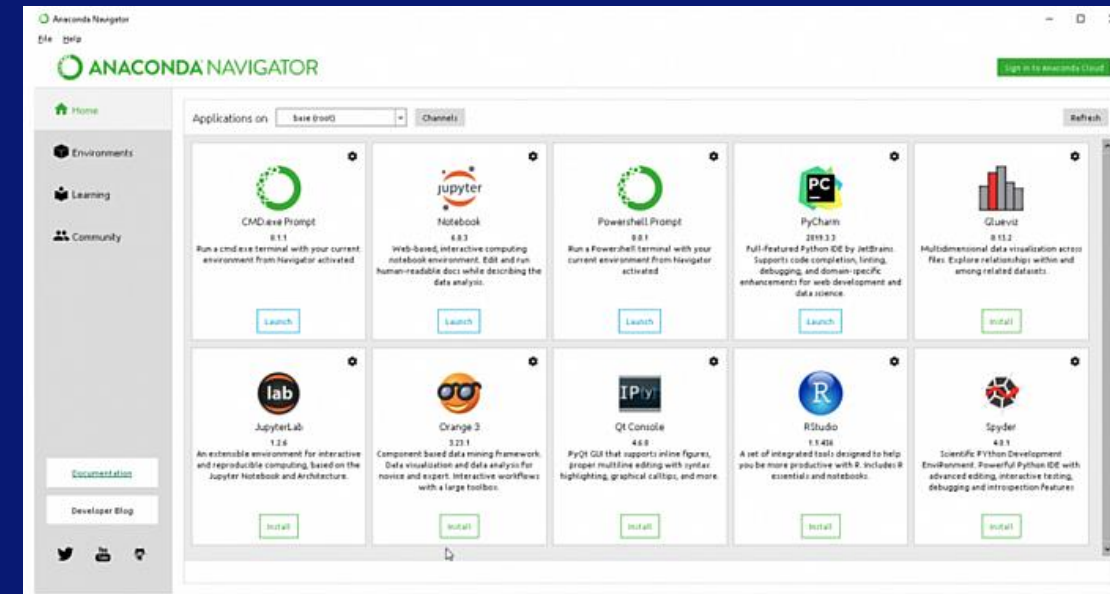




# Code Editors and IDEs

## 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

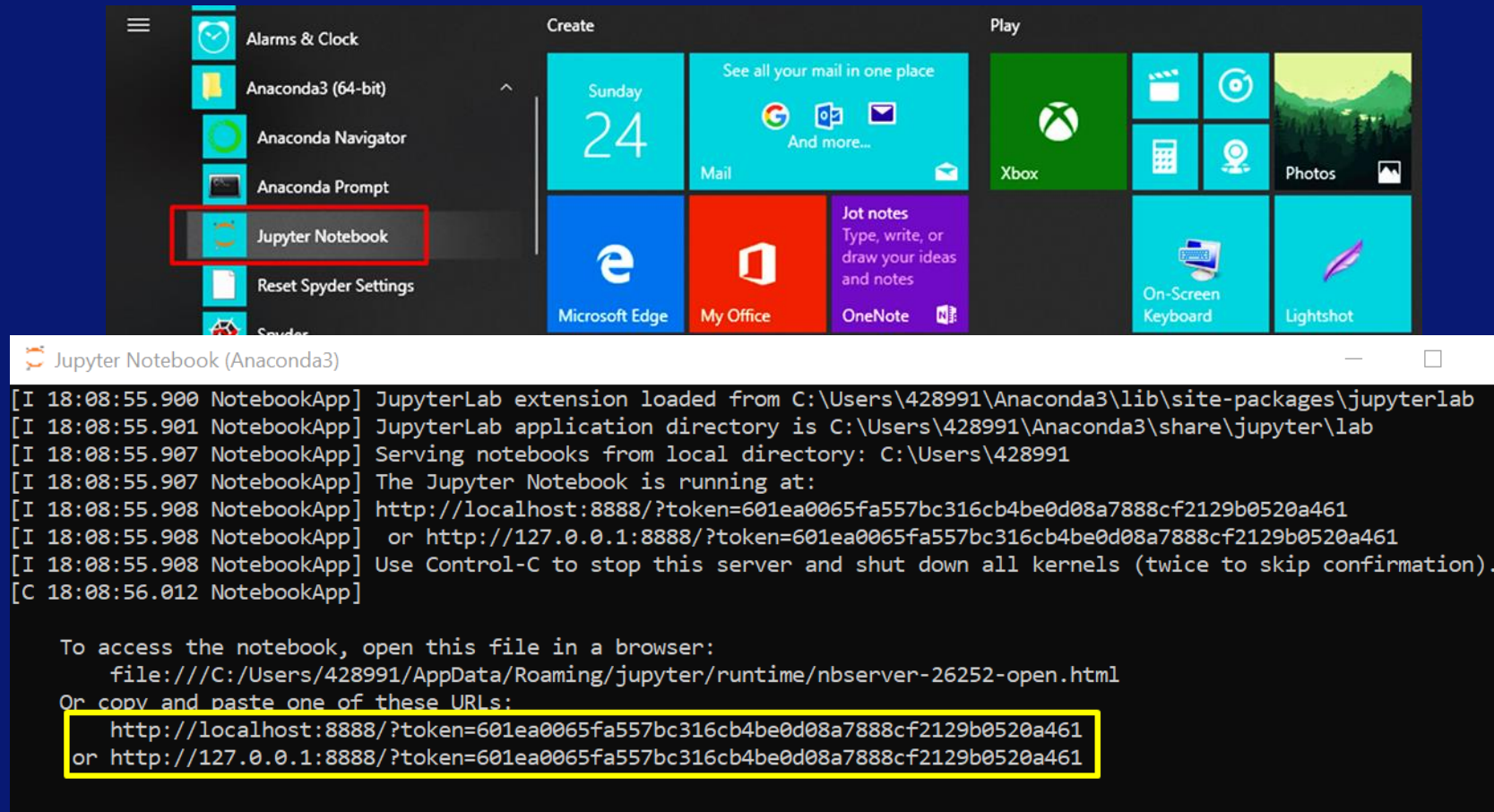




# Code Editors and IDEs

## 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!



The image shows a Windows Start menu with the 'Jupyter Notebook' application highlighted in a red box. Below it, a terminal window titled 'Jupyter Notebook (Anaconda3)' displays the following output:

```
[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]
```

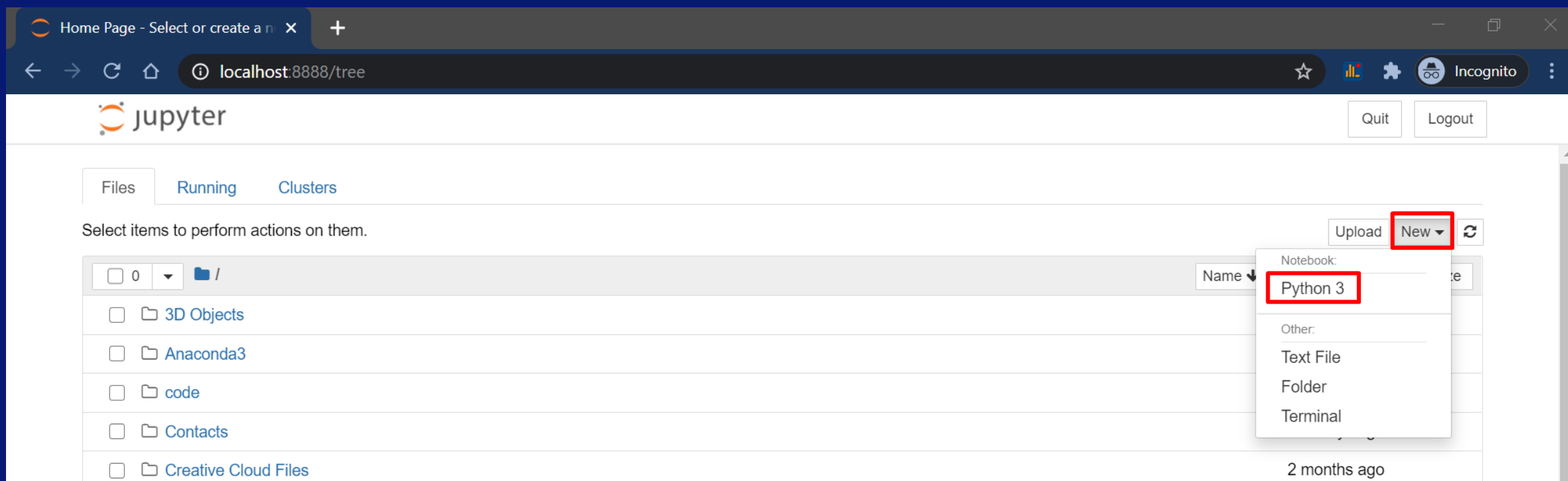
To access the notebook, open this file in a browser:  
file:///C:/Users/428991/AppData/Roaming/jupyter/runtime/nbserver-26252-open.html  
Or copy and paste one of these URLs:  
<http://localhost:8888/?token=601ea0065fa557bc316cb4be0d08a7888cf2129b0520a461>  
or <http://127.0.0.1:8888/?token=601ea0065fa557bc316cb4be0d08a7888cf2129b0520a461>



# 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

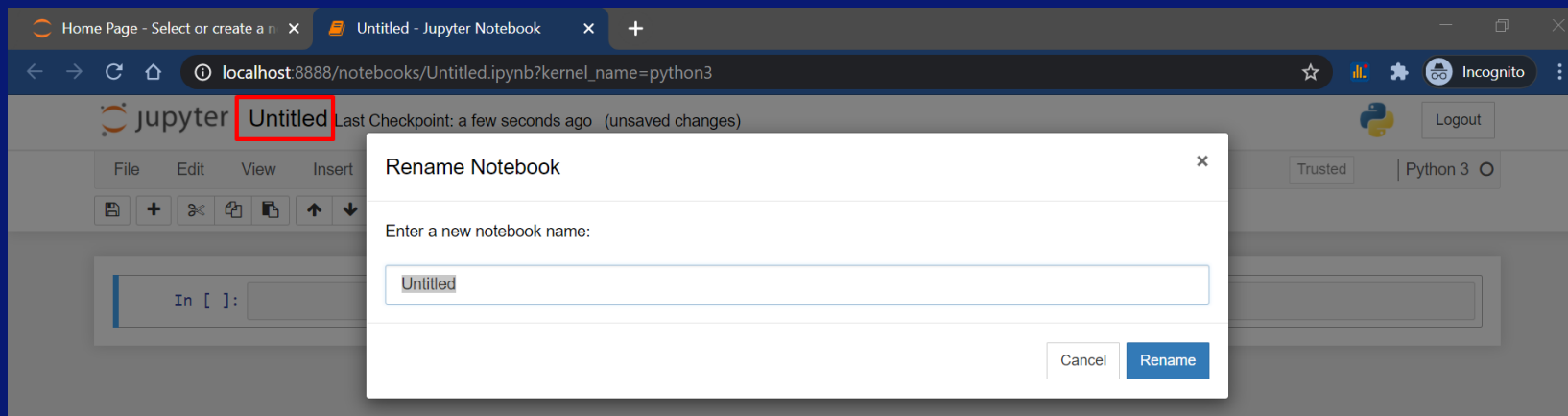




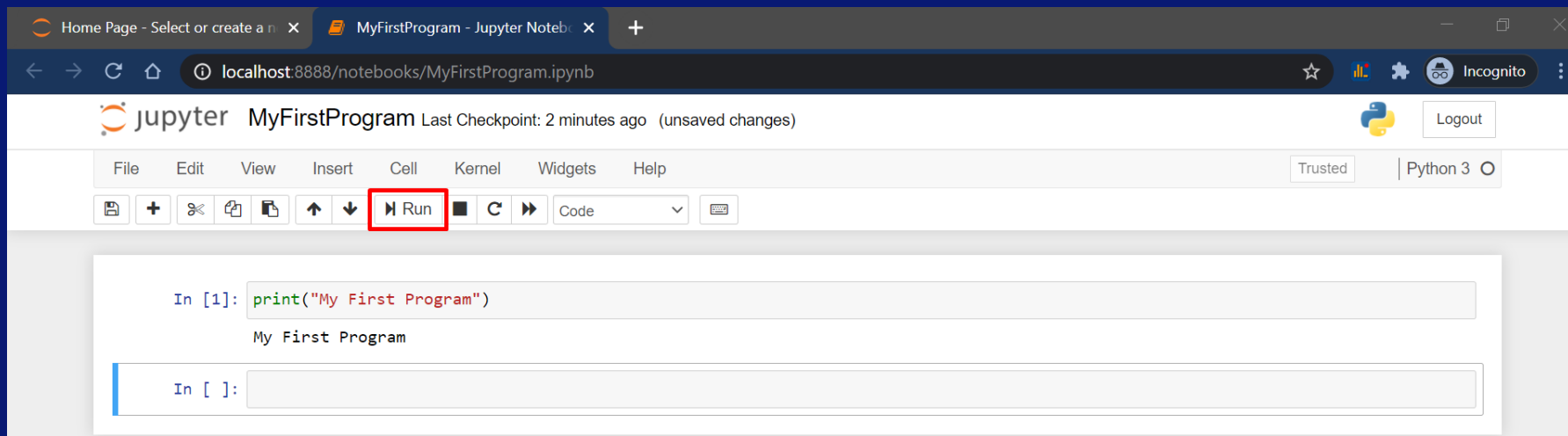
# Code Editors and IDEs

## 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

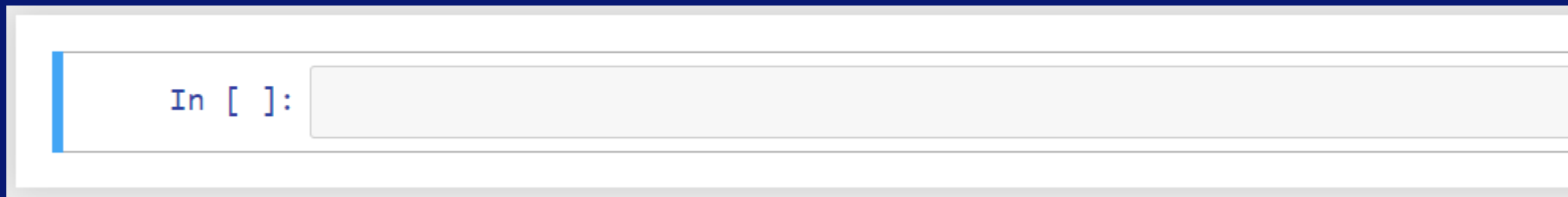


# Code Editors and IDEs

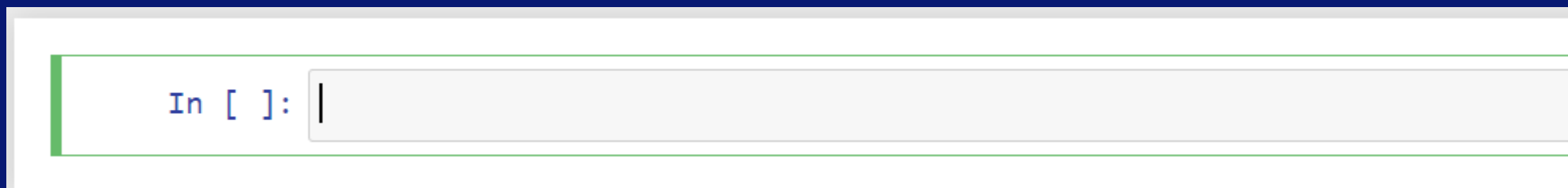
## Jupyter Notebooks tips

### 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

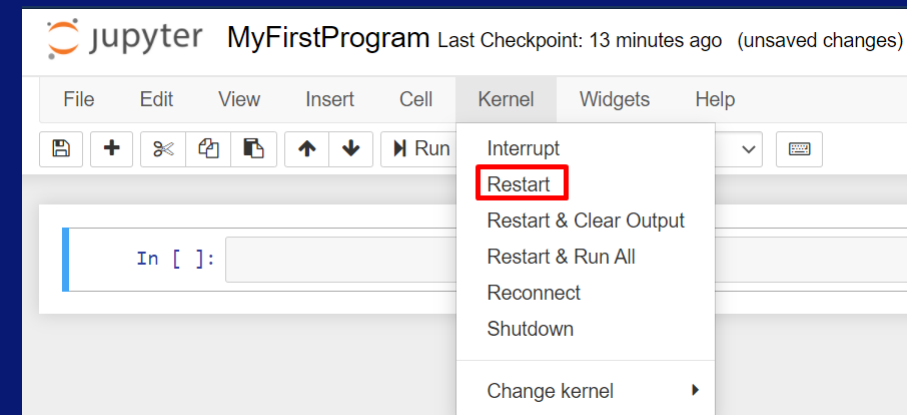


# Code Editors and IDEs

## 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
- OO - restart kernel
- M - change cell to markdown mode (good for documentation)

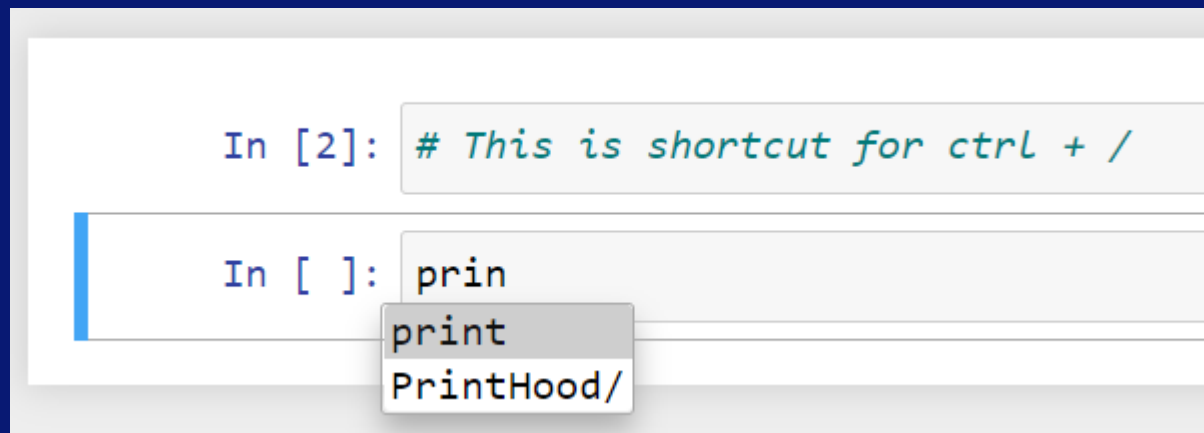


# Code Editors and IDEs

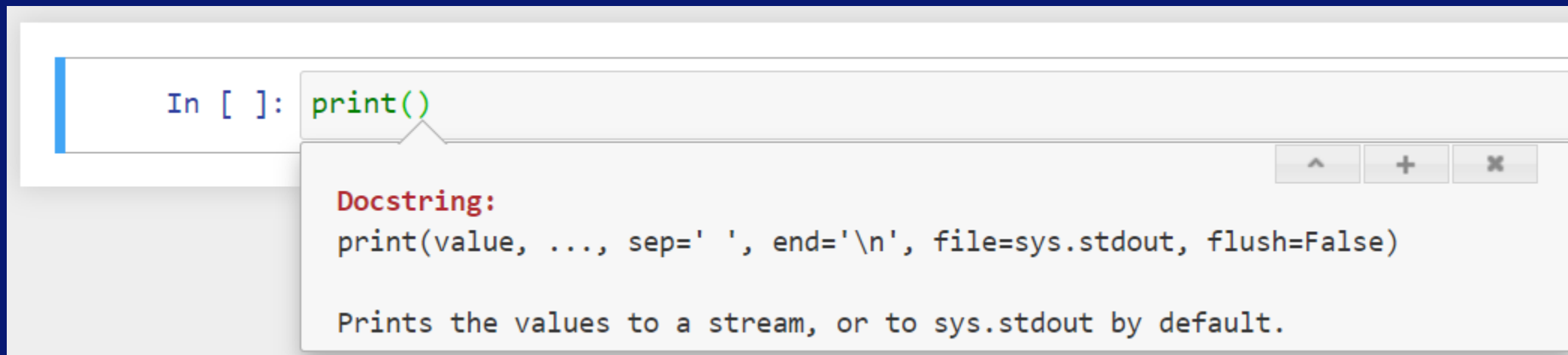
## Jupyter Notebooks tips

### Edit mode shortcuts

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



A screenshot of a Jupyter Notebook cell in edit mode. The first line of code is `In [2]: # This is shortcut for ctrl + /`. The second line is `In [ ]: prin`. A dropdown menu is visible below the second line, showing suggestions: `print` (highlighted), `print`, and `PrintHood/`.



A screenshot of a Jupyter Notebook cell in edit mode. The first line of code is `In [ ]: print()`. A tooltip is displayed below the code, showing the docstring for the `print()` function. The tooltip text is: **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

The image shows the Jupyter Notebook editor interface. The title bar reads "jupyter PythonProgramming (unsaved changes)". The menu bar includes File, Edit, View, Insert, Cell, Kernel, Widgets, and Help. The toolbar contains icons for saving, adding, deleting, copying, pasting, undo, redo, and a dropdown menu currently set to "Markdown". The main editing area contains the following markdown code:

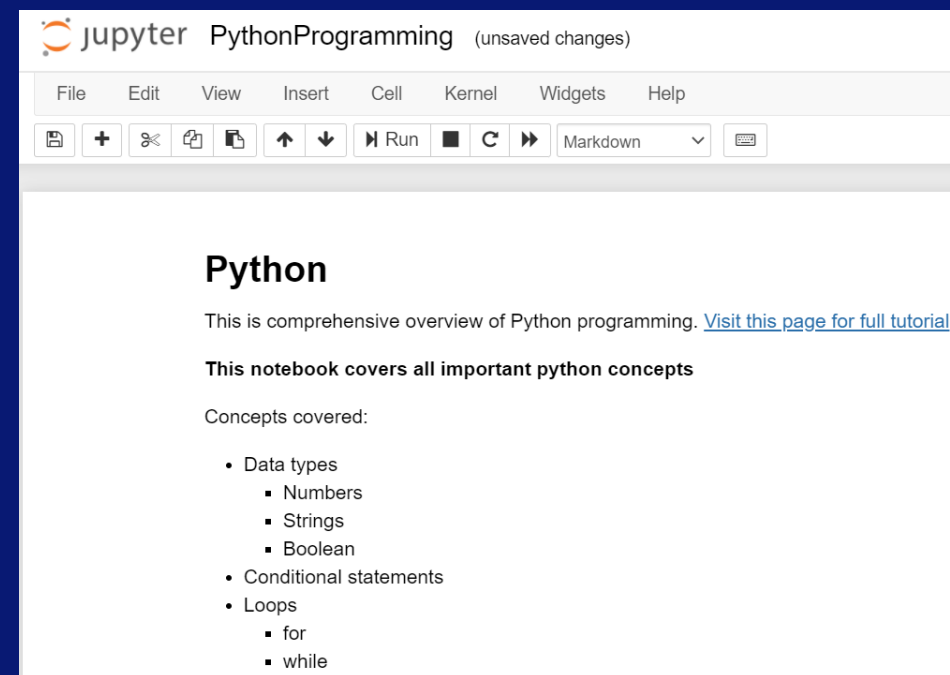
```
# Python

This is comprehensive overview of Python programming.
<a href="https://www.facebook.com/vikram.devops">Visit this page for full tutorial</a>

**This notebook covers all important python concepts**

Concepts covered:

* Data types
  * Numbers
  * Strings
  * Boolean
* Conditional statements
* Loops
  * for
  * while
```





# 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)
```

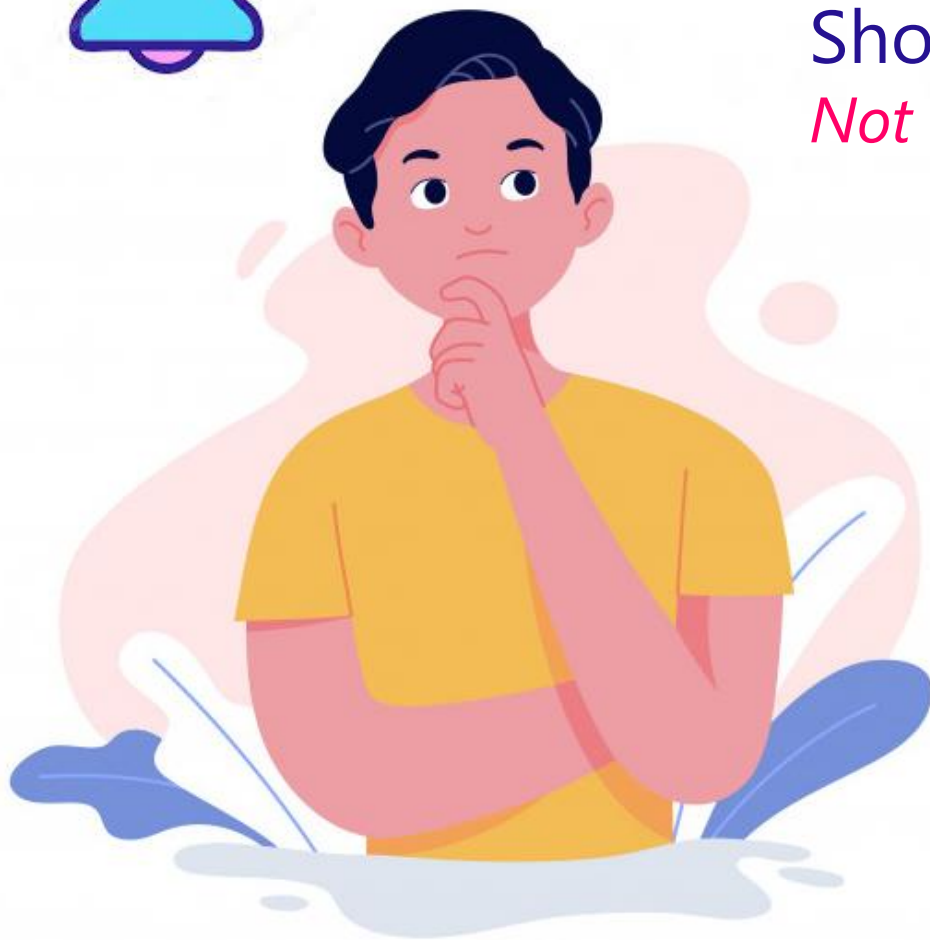
```
In [ ]:
```





## 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.

UP NEXT

# Introduction to Python

# | References

- <https://wiki.python.org/moin/BeginnersGuide/Programmers>
- [https://code.visualstudio.com/docs/python/python-tutorial# prerequisites](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

