Installation & Environment Setup

MUKESH KUMAR

AGENDA

- Installing
  - ✓Python
  - ✓Jupyter Notebook & Lab
  - ✓Anaconda
- Setting up Google Colab
- Running Python scripts:
  - ✓In Jupyter Notebook.
  - ✓.py files from the terminal.
  - ✓Google Colab basics.

SETTING UP PYTHON

Python Installation

- Installing Python:
  - Available for Windows, macOS, and Linux.
- Download from the official Python website (python.org).

- https://www.python.org/downloads/

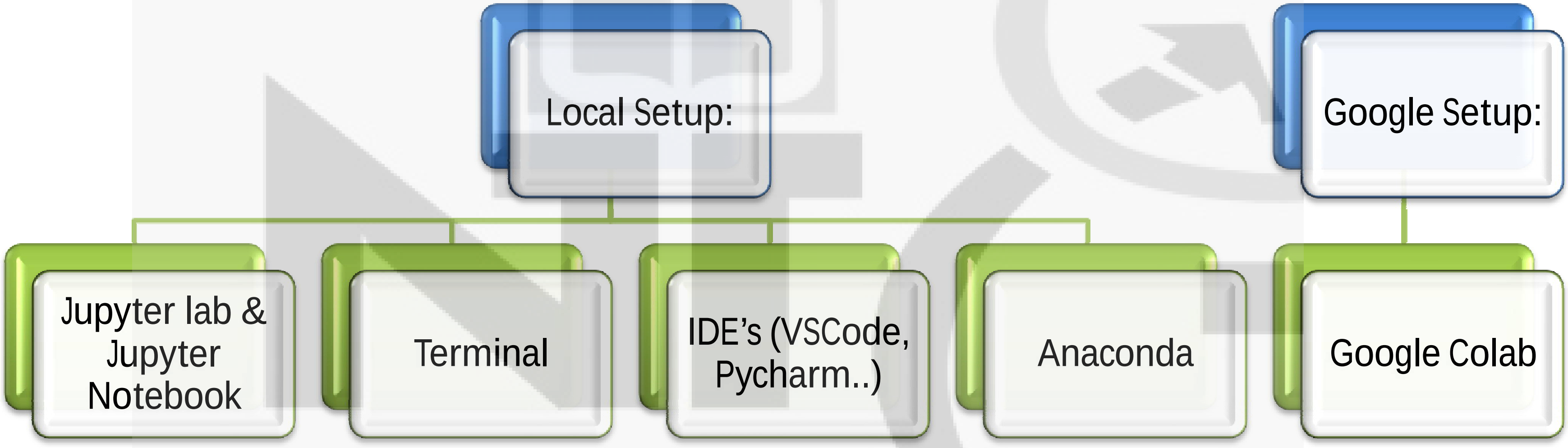
Verify Installation

- Open the terminal (Command Prompt/PowerShell on Windows, Terminal on macOS/Linux).

- Run ***python --version*** or ***python3 --version*** to verify.

SETTING UP DEVELOPMENT ENVIRONMENTS

Development Environments



Local Setup:

- Jupyter lab & Jupyter Notebook
- Terminal
- IDE’s (VSCode, Pycharm..)
- Anaconda

Google Setup:

- Google Colab

SETUP IN LOCAL ENV

TERMINAL/COMMAND PROMPT

YOUR FIRST PYTHON CODE

Hello, World!

```
print("Hello, World!")
```

- **Explanation:**
  - – print() is a function that displays output.
  - – Text within quotes is a string.

Jupyter Installation

- Jupyter Notebook
- Jupyter Lab

Setting Up Jupyter Notebook

- **Install pip (if not already installed):**
  - Comes pre-installed with Python. Check with **pip --version**.
- **Install Jupyter Notebook:**
  - Run **pip install notebook**.
- **Launch Jupyter Notebook:**
  - In the terminal, run jupyter notebook.
  - It will open in your default web browser.

Setting Up Jupyter Lab

- **Install JupyterLab:**
  - Run **pip install jupyterlab**.
- **Launch JupyterLab:**
  - In the terminal, type ***jupyter lab*** and press Enter.
  - JupyterLab will open in your default web browser with an enhanced interface.
- **Features of JupyterLab:**
  - Tabbed interface for working on multiple notebooks, text files, and terminals.
  - Integrated file browser.
  - Support for extensions to enhance functionality.

Running Python Scripts from the Terminal

- Open any text editor (VS Code, Notepad++, etc.).
- Save the file with the .py extension, e.g., hello.py.
- Run the script in the terminal:
  - On Windows: python hello.py.
  - On macOS/Linux: python3 hello.py.

Setting Up IDE’s

- VS Code
- PyCharm

What is an IDE

- **Definition**:
  - An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to programmers for software development.
- **Features**:
  - Code editor with syntax highlighting and autocomplete.
  - Debugging tools.
  - Integrated terminal or command prompt.
  - Build and testing tools.

IDE Features

- **Code Editor:**
  - Syntax highlighting, autocompletion, code formatting, and refactoring tools.
- **Compiler/Interpreter:**
  - Translates your code into machine-readable instructions.
- **Debugger:**
  - Helps you find and fix errors in your code by stepping through execution, inspecting variables, and setting breakpoints.
- **Build Automation:**
  - Automates tasks like compiling, testing, and packaging your code.
- **Project Management:**
  - Organizes your project files, manages dependencies, and provides a central workspace.

Benefits of Using an IDE:

- **Increased Productivity:** Streamlines development workflows and saves time.
- **Improved Code Quality:** Helps you write cleaner, more efficient, and bug-free code.
- **Better Collaboration:** Facilitates teamwork and code sharing.
- **Enhanced Learning:** Provides a structured environment for learning and experimenting.

Examples of Popular IDEs:

- **Python:** PyCharm, VS Code, Jupyter Notebook
- **Java:** Eclipse, IntelliJ IDEA
- **C/C++:** Visual Studio, Code::Blocks
- **Web Development:** Visual Studio Code, WebStorm

Visual Studio Code

A powerful Code Editor

VSCode

What is VS Code?

- A lightweight yet powerful code editor.
- Supports a wide range of programming languages, including Python.
- Highly customizable with extensions.

Installation:

- Download and install from the official VS Code website (link provided).

VSCode

Key Features:

- **Intelligent Code Completion:**
  - Autocompletes code, saving time and reducing errors.
- **Debugging:**
  - Built-in debugger for stepping through code, setting breakpoints.
- **Git Integration:**
  - Easily manage Git repositories within VS Code.
- **Extensions:**
  - Vast marketplace of extensions for adding new languages, themes, and functionalities.

Setting Up VSCode

- **Download VS Code:**
  - Visit Visual Studio Code and download the installer for your operating system.

- **Install VS Code:**
  - Run the installer and follow the instructions.

- **Install Python Extension:**
  - Open VS Code and navigate to the Extensions view (Ctrl + Shift + X).
  - Search for "Python" and install the official extension by Microsoft.

Setting Up VSCode

- **Configure Python Interpreter:**
  - Open any Python file or create a new one.
  - Select the interpreter by clicking on the interpreter name in the bottom-left corner or using Ctrl + Shift + P and typing "Python: Select Interpreter".

- **Run Python Code in VS Code:**
  - Open a Python file and press F5 to run the script.
  - Alternatively, use the integrated terminal to run python script_name.py.

VSCode Additional Features

- Use the debugger to set breakpoints and step through code.

- Leverage IntelliSense for code suggestions and error highlighting.

PyCharm

What is PyCharm?

- A powerful Integrated Development Environment (IDE) specifically designed for Python.
- Offers intelligent code completion, debugging, code inspection, and more.

PyCharm Key Features

- **Smart Code Editor:**
  - Intelligent code completion, error highlighting, and quick fixes.
  - Refactoring tools for improving code quality.
- **Debugger:**
  - Step through code, set breakpoints, and inspect variables.
- **Version Control:**
  - Seamless integration with Git and other version control systems.
- **Project Management:**
  - Create and manage Python projects easily.
- **Scientific Tools:**
  - Support for NumPy, Pandas, Matplotlib, and other scientific libraries.

Setting Up PyCharm

- **Download PyCharm:**
  - Visit JetBrains PyCharm and download the Community Edition (free) or Professional Edition.
- **Install PyCharm:**
  - Run the installer and follow the instructions.
- **Create a New Project:**
  - Open PyCharm and click "New Project."
  - Select the location and configure a virtual environment for your project.

Setting Up PyCharm

- **Configure Python Interpreter:**
  - Go to File -> Settings -> Project -> Python Interpreter.
  - Add or select the desired Python interpreter.
- **Run Python Code in PyCharm:**
  - Write Python code in the editor.
  - Right-click and select "Run" or use the green play button.

PyCharm Additional Features

- Integrated debugger for step-by-step execution.
- Version control tools (Git) built-in.
- Support for plugins to extend functionality.

Anaconda

What is Anaconda?

- A free and open-source distribution of Python and R.
- Includes over 1,500 popular data science and machine learning packages.
- Simplifies package and environment management.

Why Use Anaconda?

- **Easy Installation:** Includes Python and essential libraries in a single package.
- **Package Management:** conda command for installing, updating, and removing packages easily.
- **Environment Management:** Create isolated environments for different projects to avoid conflicts.
- **Jupyter Notebook:** Included for interactive coding and data exploration.

Setting Up Anaconda

- **Download Anaconda:**
  - Visit Anaconda and download the installer for your operating system.

- **Install Anaconda:**
  - Run the installer and follow the instructions.
  - Ensure you check the option to add Anaconda to your PATH during installation.

Setting Up Anaconda

- **Launch Anaconda Navigator:**
  - Open the Anaconda Navigator from your start menu or applications folder.
- **Create a New Environment:**
  - In Navigator, go to the "Environments" tab.
  - Click "Create" to set up a new environment with specific Python versions and libraries.
- **Launch Jupyter Notebook or Spyder:**
  - From Navigator, launch tools like Jupyter Notebook or Spyder IDE for Python coding.

Running Python in Anaconda

- **Using the Terminal (Anaconda Prompt):**
  - Open the Anaconda Prompt.
  - Activate your environment: conda activate <environment_name>.
  - Run Python scripts: python script_name.py.
- **Using Jupyter Notebook:**
  - Launch Jupyter Notebook from Navigator or Anaconda Prompt.
  - Create a new notebook and run Python code interactively.

Google Colab

Why Use Google Colab?

- Free to use, no installation required.
- Runs on Google’s cloud servers.
- Includes a ready-to-use Python environment.

Steps to Access Google Colab:

- Go to Google Colab.
- Log in with your Google account.
- Create a new notebook (File -> New Notebook).

Running Python Code in Jupyter Notebook

- *Running Python Code in Jupyter Notebook*
- Open Jupyter Notebook in your browser.
- Create a new notebook (New -> Python 3).
- Type Python code in a cell and **press Shift + Enter** to execute.

Google Colab Basics

Using Google Colab Effectively

- **Code Cells and Execution:**
  - Write Python code in cells and run by pressing **Ctrl + Enter.**
- **Importing Libraries:**
  - Libraries like NumPy and Pandas are pre-installed.
  - Use **!pip install <package_name>** for additional libraries.
- **Uploading Files:**
  - Use the upload button or from google.colab import files to upload local files.
- **Saving Your Work:**
  - Save the notebook to your Google Drive (File -> Save a copy in Drive).

Summary

- Python is a versatile programming language used in multiple domains.
- Set up Python locally or use cloud-based tools like Google Colab.
- Run Python code in Jupyter Notebook, the terminal, or Google Colab.

Choosing a development Environment

- **Jupyter Notebook:**
  - Interactive, great for learning.

- **Other Options:**
  - VS Code, PyCharm (briefly mention)