Intro to coding environment

&

Intro to programming

A **GOVT. OF INDIA** SUPPORTED, **GOVT. OF KERALA** PARTNERED SOCIAL ENTERPRISE.

ICT ACADEMY OF KERALA

Introduction to Python Programming Environments

Visual Studio Code (VS Code)

Download: VS Code

- **Lightweight and Fast:** VS Code is known for its speed and responsiveness.
- **Extensible:** A rich marketplace of extensions for Python development.
- **Integrated Terminal:** Run Python scripts directly within the editor.
- **Git Integration:** Manage version control seamlessly.

A lightweight, open-source editor with extensive extension support.

Download Visual Studio Code - Mac, Linux, Windows

Visual Studio Code (VS Code) is a versatile code editor that supports a wide range of file types for various programming and development tasks.

Programming	Web Development	Data and Configuration	Scripting	Templates and Frameworks
JavaScript: `.js` Python: `.py` Java: `.java` C++: `.cpp` C#: `.cs` Ruby: `.rb` PHP: `.php` Go: `.go` Rust: `.rs`	HTML: `.html` CSS: `.css` JavaScript: `.js` TypeScript: `.ts` JSON: `.json` XML: `.xml` Markdown: `.md`	YAML: `.yaml`, `.yml` TOML: `.toml` INI: `.ini` CSV: `.csv` SQL: `.sql`	Bash: `.sh` PowerShell: `.ps1` Batch: `.bat` Git Config: `.gitignore`, `.gitattributes`	Django: `.html`, `.py`, `.json` React: `.jsx`, `.tsx`, `.js`, `.ts` Angular: `.html`, `.ts`

Jupyter Notebook and Google Colab

Pros:

- **Interactive:** Ideal for data exploration, visualization, and sharing.
- **Notebook Format:** Combine code, text, and visualizations.
- **Colab (Cloud-Based):** Free GPU/TPU support for machine learning.

Cons:

- **Not Full IDEs:** Lack features like debugging and project management.
- **Colab Requires Internet:** Works online, not offline.

An interactive web-based tool that allows you to run Python code in cells and view the output immediately.

Jupyter Notebook

Download

- **Best Suited For:**
  - Those who need full control over their environment.
  - Developers who prefer working locally.
- **Features**:
  - Free, open-source, and web-based.
  - Interactive computing platform.
  - Create and share computation documents.
  - Supports various programming languages.
  - Full control over environment.
  - Local execution.
  - Rich ecosystem of extensions.

Google Colab (Colaboratory)

Google Colab

- **Best Suited For:**
  - Collaboration in real-time.
  - Access to GPUs or TPUs.
- **Features:**
  - Web-based tool by Google Research.
  - Write and execute Python code in browsers.
  - Share computation files without downloads.
  - Collaborative features.
  - Free GPU/TPU support.
  - No installation required.
- **Cons:**
  - Limited control over environment.
  - Requires an internet connection.

Jupyter Notebook is an interactive computing environment that allows you to create and share documents containing live code, equations, visualizations, and narrative text. .

Main File Types - **Jupyter Notebooks**: `.ipynb` files, which can contain code (Python, R, Julia, etc.), text (Markdown), This file format combines live code, text, and visualizations. This format is highly interactive and widely used for data analysis, scientific research, and educational purposes.

Programming	Data	Exported Formats
Python: `.py` **R**: `.R` **Julia**: `.jl` **Scala**: `.scala` **JavaScript**: `.js` **Bash**: `.sh` (using bash kernel) **Ruby**: `.rb`	**CSV**: `.csv` **Excel**: `.xls`, `.xlsx` **JSON**: `.json` **Parquet**: `.parquet` **HDF5**: `.h5`, `.hdf5` **SQL**: `.sql`	**HTML**: `.html` **PDF**: `.pdf` **Markdown**: `.md` **ReStructuredText**: `.rst` **LaTeX**: `.tex` **Executable Scripts**: `.py` (Python script), `.R` (R script), etc.

Visualization Files
Images: `.png`, `.jpg`, `.jpeg`, `.svg`
Plots: Generated within notebooks using libraries like Matplotlib, Seaborn, Plotly, etc.

Environment Files: `environment.yml`, `requirements.txt`

Other

Text Files: `.txt` **Markdown**: `.md`

LaTeX: `.tex` **Log Files**: `.log`

ICT ACADEMY OF KERALA

Introduction to ANACONDA

- ➔ Anaconda is a distribution of Python and R for scientific computing and data science.
- ➔ It includes over 1,500 packages and the Conda package and environment manager.
- ➔ Popular for its simplicity in package management and deployment.

Why Use Anaconda?

- Simplifies package management.
- Easily create and manage environments.
- Includes popular data science libraries (NumPy, Pandas, Matplotlib, etc.).
- Cross-platform: Works on Windows, macOS, and Linux.
- Install and update packages easily with Conda.

ICT ACADEMY OF KERALA

Introduction to Python Virtual Environments

What is a Virtual Environment?

- A **virtual environment** is an isolated workspace for Python projects.
- It allows you to manage dependencies separately for each project.
- Think of it as a sandbox where you can install packages without affecting the system-wide Python installation.

Why Use Virtual Environments?

1. **Preventing Version Conflicts:**
   - Installing packages system-wide can lead to conflicts.
   - Different projects may require different package versions.
   - Virtual environments isolate dependencies for each project.
2. **Easy Reproducibility:**
   - Define exact package versions using a requirements.txt file.
   - Ensure your project works consistently across environments.
3. **Works Everywhere:**
   - Even on shared hosts where you lack administrator rights.
   - Ideal for university servers or web hosting providers.

Virtual environments vs. other options

There are other options to isolate your project:

1. In the most extreme case, you could buy a second PC and run your code there. Problem fixed! It was a bit expensive, though!
2. A virtual machine is a much cheaper option but still requires installing a complete operating system—a bit of a waste as well for most use cases.
3. Next in line is containerization, with the likes of Docker and Kubernetes. These can be very very powerful and are a good alternative.

virtual environments are a great way to isolate your project's dependencies.

Creating a Virtual Environment

1. Open your terminal or command prompt.
2. Navigate to your project directory.
3. Run the following command:

   python -m venv myenv

   If you are running Python 3.4+, you can use the venv module)

   (Replace myenv with your preferred environment name.)

All other Python versions

The alternative that works for any Python version is using the **virtualenv** package. You may need to install it first with pip install:

pip install virtualenv

Once installed, you can create a virtual environment with:

virtualenv [directory]

virtualenv myenv

Activating the Virtual Environment

- On Windows:

  myenv\Scripts\activate

- On macOS/Linux:

  source myenv/bin/activate

Installing Packages

- While the virtual environment is active:

  pip install package_name

Deactivating the Virtual Environment

Simply type:

```
deactivate
```

```
# If your virtual environment is
in a directory called 'venv':
remove the directory with all its
content.
rm -r venv
```

Understanding the PATH Variable and Virtual Environments

What is the PATH Variable?

- is a list of directories that your operating system (OS) searches through when you type a command in the terminal or command prompt.
- If you try to run a command that's not in your current working directory, the OS will look in the directories listed in the PATH variable to find it.

How Does the PATH Variable Work for Python?

- When you import a library in Python, it also looks through the directories in the PATH variable to find where the library is located.
- If your virtual environment (venv) is listed first in the PATH variable, Python will check there first before looking in system-wide directories like `/usr/bin`.

Why is This Important for Virtual Environments?

- **Isolation of Dependencies**: Python uses the packages installed in the venv first. This way, you avoid conflicts with system-wide packages.
- **Override System Packages**: This ensures that the specific versions of libraries your project needs are used, without interfering with other projects or system tools.