Streamlit Tutorial

This tutorial will guide you through the process of installing Streamlit on Mac and Windows, setting up your project files, creating a small example application, and learning some basic commands.

1. Installing Streamlit on MacOS

1.1. Ensure Python is Installed

First, you need to ensure that Python is installed on your computer. To do this, open your terminal and navigate to the folder where you plan to create your project:

```
cd /path/to/your/project
```

Then enter the following command to check the version of Python installed:

```
python3 --version
```

If Python is not installed, you can download and install it from python.org. Having Python installed is essential because Streamlit is a Python library.

1.2. Create a Virtual Environment

A **virtual environment** is an isolated environment for Python projects. It ensures that the libraries you install for one project don't interfere with those of another project. This is particularly useful when working with multiple Python projects that may require different versions of the same library. You can read more about virtual environments in the Python documentation.

To create a virtual environment, enter this command in your terminal:

```
python3 -m venv streamlit_env
```

This will create a folder named `streamlit_env` in your project directory, which contains the isolated environment.

Next, activate your virtual environment using the following command:

```
source streamlit_env/bin/activate
```

When the virtual environment is activated, you'll notice your terminal prompt changes to indicate you are now working within the `streamlit_env`.

1.3. Install Streamlit

Once your virtual environment is activated, you can install Streamlit using `pip`, Python's package installer:

```
pip install streamlit
```

This installs the Streamlit library and its dependencies within your virtual environment. After installation, you're ready to start building your first Streamlit app!

---

2. Installing Streamlit on Windows

2.1. Ensure Python is Installed

First, ensure that Python is installed on your computer. To verify this, open the Command Prompt and navigate to the folder where you want to create your project:

```
cd \path\to\your\project
```

Then, enter the following command to check the version of Python installed:

```
python --version
```

If Python is not installed, download and install it from python.org. During the installation process, ensure the "Add Python to PATH" option is selected to simplify command-line usage.

2.2. Create a Virtual Environment

A **virtual environment** is an isolated environment for Python projects, ensuring libraries installed for one project don't interfere with others. It's especially

useful when working on multiple Python projects requiring different library versions. For more details, refer to the Python virtual environment documentation.

To create a virtual environment, run the following command in your project folder:

```
python -m venv streamlit_env
```

This will create a folder named `streamlit_env` in your project directory, containing the isolated environment.

Next, activate your virtual environment using the following command:

```
streamlit_env\Scripts\activate
```

When the virtual environment is activated, you'll see `(streamlit_env)` at the beginning of your terminal prompt, indicating you are working within the virtual environment.

2.3. Install Streamlit

With your virtual environment activated, install Streamlit using `pip`:

```
pip install streamlit
```

This installs the Streamlit library and its dependencies within your virtual environment. Once installed, you are ready to create your first Streamlit app!

---

3. Setting Up Your Project

Once Streamlit is installed, you need to organize your project files to begin building your app. A well-structured project ensures that your code is easy to navigate and maintain.

3.1. Create a Project Folder

Start by creating a folder for your Streamlit app. You can name this folder something descriptive, like `streamlit_app`. This folder will contain all the files for your project.

Example structure:

```
streamlit_app/
├── app.py
```

3.2. Create a Main Python File

Inside your project folder, create a Python file named `app.py`. This file will serve as the main entry point for your Streamlit application. All the code for your app will go into this file.

---

4. Writing Your First Streamlit App

Now that your project is set up, let's create a simple "Hello, Streamlit!" app to get started with Streamlit.

4.1. Writing the Code

Open the `app.py` file and write the following code:

```
import streamlit as st

# Title
st.title("Hello, Streamlit!")

# Text Input
name = st.text_input("Enter your name:")

# Button
if st.button("Say Hello"):
    st.write(f"Hello, {name}! Welcome to Streamlit!")
```

4.2. Running the App

To run your app:

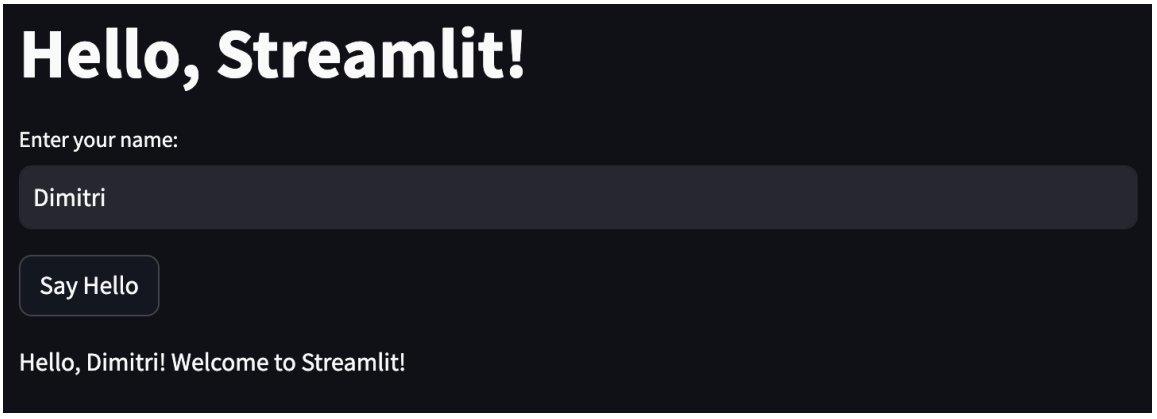
1. Navigate to your project folder in the terminal or command prompt:

```
cd /path/to/streamlit_app
```

2. Use the `streamlit` command to start the app:

```
streamlit run app.py
```

3. A new tab will open in your default web browser, displaying your Streamlit app at `http://localhost:8501`. You can now interact with the app.



---

5. Exploring Basic Streamlit Commands

Now that you've created and run your first Streamlit app, let's explore some of the basic Streamlit commands. These commands are the building blocks for creating interactive and dynamic applications. You can find a comprehensive list of Streamlit components and their usage in the Streamlit API reference.

5.1. Widgets

Widgets allow users to interact with the app. Here are some commonly used widgets:

Text Input: Creates a text box for user input.

```
st.text_input("Enter your name:")
```

Button: Adds a clickable button.

```
st.button("Click me!")
```

Slider: Allows users to select a value from a range.

```
st.slider("Select a number:", 0, 100)
```

Checkbox: Adds a checkbox for yes/no options.

```
st.checkbox("I agree")
```

5.2. Display Elements

Streamlit provides simple ways to display text, data, and other elements:

Write Text or Data: Dynamically displays any type of content (text, dictionaries, dataframes, etc.).

```
st.write("This is a simple text.")
st.write({"key": "value"})
```

Title and Headers: Adds formatted titles and headers for sections.

```
st.title("Streamlit App")
st.header("This is a header")
```

5.3. Layout Management

Streamlit supports layout elements to structure your app:

Sidebar: Moves widgets to a collapsible sidebar for better organization.

```
st.sidebar.title("Sidebar")
```

```
st.sidebar.slider("Choose a value:", 0, 50)
```

Columns: Splits the app into multiple columns for better visual arrangement.

```
col1, col2 = st.columns(2)
col1.write("Content in column 1")
col2.write("Content in column 2")
```

---

6. Building an Interactive Application

Let’s combine the commands we’ve learned to build a more interactive app with multiple pages.

6.1. Writing the Code

Update your `app.py` file with the following code:

```
import streamlit as st

# Sidebar Navigation
page = st.sidebar.radio("Navigate to", ["Home", "About"])

if page == "Home":
    st.title("Home Page")
    st.write("Welcome to the Home Page!")
    age = st.slider("Select your age:", 0, 100)
    st.write(f"Your age is: {age}")

elif page == "About":
    st.title("About Page")
    st.write("This app demonstrates Streamlit’s basic featu
res.")
```

6.2. Running the App

1. Navigate to your project folder in the terminal or command prompt:

```
cd /path/to/streamlit_app
```

2. Run the app:

```
streamlit run app.py
```

3. Navigate between the "Home" and "About" pages using the sidebar. Interact with widgets like the slider to see how the app responds dynamically.