Setup Development Environment

Setting Up Your Developer Environment

Objective: This aims to familiarization with the tools and configurations necessary to set up an efficient developer environment for software engineering projects. This will give you the skills required to set up a robust and productive workspace conducive to coding, debugging, version control, and collaboration.

1. Installing Operating System (Windows 11)

- Installing Windows 11 involves a few steps, whether you are upgrading from a previous version of Windows or performing a clean installation. Below is a step-by-step guide on how to install Windows 11.

Step 1: Check System Requirements

Before starting, ensure your PC meets the minimum system requirements for Windows 11:

- Processor: 1 GHz or faster with at least 2 cores on a compatible 64-bit processor or System on a Chip (SoC)
- RAM: 4 GB or more
- Storage: 64 GB or larger
- TPM: Trusted Platform Module (TPM) version 2.0
- System firmware: UEFI, Secure Boot capable
- Graphics Card: DirectX 12 compatible graphics / WDDM 2.x
- Display: >9” with HD Resolution (720p)
- Internet connection: Internet connectivity is necessary to perform updates and to download and take advantage of some features.

Troubleshooting

If you encounter any issues during the installation process, refer to the following:

- **Check compatibility**: Ensure your PC meets all the Windows 11 system requirements.

  Use the PC Health Check tool to verify if your computer meets the requirements for Windows 11:

  - Download the PC Health Check tool from the Microsoft website.
  - Run the tool and follow the instructions to check compatibility.

- **Update drivers and firmware**: Make sure all your drivers and system firmware (BIOS/UEFI) are up to date.
- **Use the Windows 11 Installation Assistant or Media Creation Tool**: If Windows Update does not work, use these tools to perform the installation.

For detailed steps and more troubleshooting, refer to the official Microsoft Windows 11 installation guide.

Step2: Backup Your Data

Ensure you have a backup of all important data. Use an external drive or a cloud storage service to back up your files.

Step 3: Download Windows 11 Installation Media

- Go to the official Microsoft Windows 11 download page.
- Choose the appropriate installation media creation option:
  - **Windows 11 Installation Assistant**: This is the simplest option for upgrading your current PC.
  - **Create Windows 11 Installation Media**: This option allows you to create a bootable USB drive or DVD.
  - **Download Windows 11 Disk Image (ISO)**: For advanced users who want to create installation media with specific tools.

Step 4: Create Installation Media

If you chose to create installation media:

- Download the Media Creation Tool.
- Run the tool and accept the license terms.
- Select "Create installation media (USB flash drive, DVD, or ISO file) for another PC" and click Next.
- Choose your preferred language, edition, and architecture (64-bit).
- Select the media you want to use (USB flash drive or ISO file). If you choose a USB drive, it must be at least 8 GB and will be formatted.
- Follow the prompts to create the media.

Step 5: Install Windows 11

Option 1: Using Windows 11 Installation Assistant

- Run the Windows 11 Installation Assistant.
- Follow the on-screen instructions to upgrade your current Windows version to Windows 11.

Option 2: Using Installation Media

- Insert the USB flash drive or DVD with the Windows 11 installation files into your PC.
- Restart your PC.
- Press the key that opens the boot-device selection menu for the computer, such as the Esc/F10/F12 keys. Select the option that boots the PC from the USB flash drive or DVD.
- Windows Setup will start. Follow the prompts to start the installation:
    - Select your language, time, and keyboard preferences, and click Next.
    - Click "Install now."
    - Enter your product key if prompted, or select "I don't have a product key" to enter it later.
    - Accept the license terms.
    - Choose the type of installation: "Upgrade" to keep your files and settings or "Custom" for a fresh installation.
    - If you chose Custom, select the partition where you want to install Windows 11. You may need to delete existing partitions to create space for the new installation.

Step 6: Complete the Setup

- The installation process will copy files and restart your PC multiple times. Follow the on-screen instructions.
- Once the installation is complete, you'll be prompted to customize your settings:
    - Choose your region and keyboard layout.
    - Connect to a network.
    - Set up a Microsoft account or a local account.
    - Adjust privacy settings.

Step 7: Install Updates and Drivers

- Once Windows 11 is installed, check for updates: Go to Settings > Windows Update > Check for updates.
- Install any available updates and restart if necessary.
- Visit your PC manufacturer's website to download and install the latest drivers for your hardware.

Step 8: Restore Data and Reinstall Applications

- Restore your backed-up data.
- Reinstall your applications.

Following these steps should help you successfully install Windows 11 on your PC.

2. Install a Text Editor or Integrated Development Environment (IDE)

Step 1: Download Visual Studio Code

- Open your web browser and go to the Visual Studio Code download page.
- Click the **"Windows"** button to download the installer.

Step 2: Install Visual Studio Code

- Once the download is complete, locate the installer file (usually in your Downloads folder) and double-click on it to run the installer.
- You might see a User Account Control (UAC) prompt asking for permission to allow the installer to make changes to your device. Click **"Yes"**.
- The Visual Studio Code Setup Wizard will open. Click **"Next"** to start the installation process.
- Review and accept the license agreement by checking the box and clicking **"Next"**.
- Choose the destination folder where you want Visual Studio Code to be installed, or leave the default location, and click **"Next"**.
- Select additional tasks you want to perform. It's recommended to check the following options for convenience:
    - **Create a desktop icon**
    - **Add to PATH** (allows you to open VS Code from the command line)
    - **Register Code as an editor for supported file types**

  Click **"Next"**.
- Click **"Install"** to begin the installation.
- Once the installation is complete, check the box to **"Launch Visual Studio Code"**, and click **"Finish"**.

Step 3: Launch Visual Studio Code

- If you didn't check the option to launch Visual Studio Code in the previous step, you can open it manually:
    - Use the desktop shortcut created during installation.
    - Or, search for "Visual Studio Code" in the Start menu and click on it.

Step 4: Customize Visual Studio Code

- **Install Extensions:** Click on the Extensions view icon on the Sidebar or press **`Ctrl+Shift+X`** to open the Extensions view. Search for and install extensions that fit your development needs. For example, if you are working with Python, search for the Python extension and install it.
- **Configure Settings:** Access settings via `File > Preferences > Settings` to customize Visual Studio Code to your preferences. You can also press **`Ctrl+,`** to open the settings directly.

Step 5: Verify Installation

- To verify the installation, open Visual Studio Code and create a new file. You can do this by clicking on `File > New File` or pressing **`Ctrl+N`**.
- Try writing a simple piece of code in the new file and save it with an appropriate extension for your programming language (e.g., `.py` for Python, `.js` for JavaScript).
- Use the integrated terminal in Visual Studio Code to run your code. Open the terminal by clicking `Terminal > New Terminal` or pressing `Ctrl+` (backtick).

Now you have Visual Studio Code installed and ready to use for your development projects on Windows 11!

3. Set Up Version Control System:

```
Install Git and configure it on your local machine. Create a
GitHub account  for hosting your repositories. Initialize a
Git repository for your project and make your first commit.
https://github.com
```

To set up a version control system using Git and GitHub, follow these steps:

Step 1: Install Git

1. **Download Git:**
    - Go to the Git download page.
    - Click the **"Windows"** button to download the installer.
2. **Install Git:**
    - Once the download is complete, locate the installer file (usually in your Downloads folder) and double-click on it to run the installer.
    - Follow the installation wizard, accepting the default options unless you have specific preferences. The defaults are generally suitable for most users.

Step 2: Configure Git

1. **Open Git Bash:** After installing Git, open Git Bash. You can find it by searching "Git Bash" in the Start menu.
2. **Set your username and email:**
    - In Git Bash, set your username by entering:

```sh
git config --global user.name "Your Name"
```

    - Set your email address by entering:

```sh
git config --global user.email "your.email@example.com"
```

    - These details will be used to identify you in your commits.

Step 3: Create a GitHub Account

1. **Sign up for GitHub:**
    - Go to the GitHub sign-up page.
    - Follow the instructions to create a new GitHub account.
2. **Verify your email:** After signing up, GitHub will send you a verification email. Click the link in the email to verify your account.

Step 4: Initialize a Git Repository

1. **Create a new project folder:**
    - Navigate to the location where you want to create your project folder. For example, in Git Bash, you can use:

```sh
cd path/to/your/directory

mkdir my-project

cd my-project
```

2. **Initialize the Git repository:**
    - Inside your project folder, initialize a Git repository by running:

```sh
git init
```

3. **Create a new file and make your first commit:**
    - Create a new file in your project folder. For example, create a `README.md` file:

```sh
echo "# My Project" > README.md
```

    - Add the file to the staging area:

```sh
git add README.md
```

    - Make your first commit:

```sh
git commit -m "Initial commit"
```

Step 5: Create a New Repository on GitHub

1. **Create a new repository:**
    - Go to your GitHub account and click the **"+"** icon in the top right corner, then select "**New repository**".
    - Fill in the repository name (e.g., `my-project`) and description.
    - Choose to make the repository public or private.
    - Do not initialize with a README, .gitignore, or license since you already have a local repository.
    - Click "**Create repository**".

Step 6: Push Your Local Repository to GitHub

1. **Copy the repository URL:** After creating the repository, GitHub will provide you with the repository URL. It will look something like this: `https://github.com/your-username/my-project.git`.

2. **Add the remote repository and push your local commits:**

    - In Git Bash, add the remote repository:

```sh
git remote add origin https://github.com/your-username/my-project.git
```

    - Push your commits to GitHub:

```sh
git push -u origin master
```

Step 7: Verify Your Repository on GitHub

- **Go to your repository on GitHub:** Navigate to `https://github.com/your-username/my-project`.

- **Check your files:** You should see your `README.md` file and your initial commit message.

Now you have set up Git, created a GitHub account, initialized a Git repository, made your first commit, and pushed it to GitHub!

4. Install Necessary Programming Languages and Runtimes:

```
Install Python from http://wwww.python.org programming language required for your
project and install their respective compilers, interpreters, or runtimes.
Ensure you have the necessary tools to build and execute your code.
```

To install Python and set up the necessary environment for your projects, follow these steps:

Step 1: Download Python

1. **Visit the Python website:**
    - Go to the Python download page.
2. **Select the version to download:**
    - Click the **"Download Python [version number]"** button. This will download the latest stable release of Python.

Step 2: Install Python

1. **Run the installer:**
    - Once the download is complete, locate the installer file (usually in your Downloads folder) and double-click on it to run the installer.
2. **Select installation options:**
    - Check the box that says **"Add Python [version number] to PATH"**. This is important as it allows you to run Python from the command line.
    - Click on **"Customize installation"** if you want to specify the installation directory or add/remove optional features. Otherwise, click **"Install Now"** to proceed with the default installation.
3. **Complete the installation:**
    - Follow the prompts to complete the installation process. The installer will also install `pip`, which is Python's package installer.

Step 3: Verify the Installation

1. **Open a Command Prompt:**
    - Search for "Command Prompt" in the Start menu and open it.
2. **Check the Python installation:**
    - Type **`python --version`** and press Enter.
        - This should display the installed Python version.
    - Type **`pip --version`** and press Enter.
        - This should display the installed `pip` version.

Step 4: Install Necessary Packages and Tools

1. **Install packages using pip:**

    - You can install necessary Python packages using `pip`. For example, to install popular packages like `numpy`, `pandas`, and `requests`, run:

```sh
pip install numpy pandas requests
```

2. **Set up a virtual environment (optional but recommended):**

    - It's a good practice to use virtual environments to manage dependencies for different projects. To create a virtual environment, navigate to your project directory and run:

```sh
python -m venv venv
```

    - Activate the virtual environment:

        - On Windows:

```sh
venv\Scripts\activate
```

Step 5: Verify Package Installation

1. **Check installed packages:**

    - With the virtual environment activated (if you're using one), run:

```sh
pip list
```

    - This will display a list of installed packages and their versions.

Step 6: Install Additional Tools (if necessary)

Depending on your specific needs, you might need additional tools or libraries. Here are some common ones:

1. **IDEs and Text Editors:**
    - **Visual Studio Code:** Already covered in a previous step. You can install Python extensions for better support.
    - **PyCharm:** A dedicated Python IDE. Download and install from JetBrains PyCharm.
2. **Version Control:**
    - **Git:** Already covered in a previous step.
3. **Build Tools:**
    - Install `build-essential` on Linux (if you're using Linux):

```sh
sudo apt-get update
sudo apt-get install build-essential
```

4. **Common Python Libraries:**
    - **Django/Flask:** For web development.

```sh
pip install django
pip install flask
```

    - **Jupyter Notebook:** For interactive data analysis.

```sh
pip install notebook
```

By following these steps, you will have Python installed and set up on your system, along with necessary packages and tools to build and execute your code effectively.

5. Install Package Managers:

If applicable, install package managers like pip (Python).

For Python, the primary package manager is `pip`. If you followed the previous steps to install Python, `pip` should already be installed. However, we'll verify its installation and cover the basics of using `pip`.

Step 1: Verify pip Installation

- **Open a Command Prompt:**
    - Search for "Command Prompt" in the Start menu and open it.
- **Check the `pip` installation:**
    - Type `pip --version` and press Enter. This should display the installed `pip` version, confirming that `pip` is installed correctly.

Step 2: Upgrade pip (Optional but recommended)

Upgrade `pip` to the latest version:

- In the Command Prompt, run:

```
sh
python -m pip install --upgrade pip
```

- This will ensure that you have the latest version of `pip`, which can provide improved features and security.

Step 3: Basic pip Usage

- **Install a package:**
    - To install a package, use the `install` command. For example, to install the `requests` package:

```sh
pip install requests
```

- **Uninstall a package:**
    - To uninstall a package, use the `uninstall` command. For example, to uninstall the `requests` package:

```sh
pip uninstall requests
```

- **List installed packages:**
    - To see a list of all installed packages, use the `list` command:

```sh
pip list
```

- **Show package information:**
    - To get more information about an installed package, use the `show` command. For example, to see details about the `requests` package:

```sh
pip show requests
```

- **Search for packages:**
    - To search for packages in the Python Package Index (PyPI), use the `search` command. For example, to search for packages related to "requests":

```sh
pip search requests
```

Step 4: Using ***virtualenv*** (Optional but recommended)

virtualenv is a tool to create isolated Python environments, which helps manage dependencies for different projects.

1. **Install `virtualenv`:**
    - Use `pip` to install `virtualenv`:

```sh
pip install virtualenv
```

2. **Create a virtual environment:**
    - Navigate to your project directory and create a virtual environment:

```sh
virtualenv venv
```

3. **Activate the virtual environment:**
    - On Windows:

```sh
venv\Scripts\activate
```

4. **Deactivate the virtual environment:**
    - To deactivate the virtual environment, simply run:

```sh
deactivate
```

By following these steps, you will have ***pip*** installed and configured on your system, enabling you to manage Python packages effectively. Additionally, using tools like ***virtualenv*** can help you manage project-specific dependencies in isolated environments.

6. Configure a Database (MySQL):

```
Download and install MySQL database.
https://dev.mysql.com/downloads/windows/installer/5.7.html
```

To configure a MySQL database on Windows, follow these steps to download, install, and set up MySQL:

Step 1: Download MySQL

1. **Visit the MySQL download page:**
    - Go to the MySQL Community Downloads page.
2. **Download the MySQL Installer:**
    - Click on **"MySQL Installer 5.7 for Windows"**.
    - Choose the appropriate installer based on your system architecture (x86 for 32-bit, x64 for 64-bit).
    - Click the **"Download"** button next to the installer file. You may be prompted to log in or sign up for a free Oracle web account, but you can also click **"No thanks, just start my download"**.

Step 2: Install MySQL

1. **Run the installer:**
    - Once the download is complete, locate the installer file (usually in your Downloads folder) and double-click on it to run the installer.
2. **Choose Setup Type:**
    - You will be prompted to choose a setup type. Select **"Custom"** if you want to choose specific components to install, or **"Developer Default"** for a typical setup that includes MySQL Server, MySQL Workbench, and other tools.
3. **Select Products and Features:**
    - If you chose "Custom," select the products you want to install. Make sure that **"MySQL Server"** and **"MySQL Workbench"** are selected.
4. **Download and Install MySQL:**
    - Click **"Next"** to proceed with the installation.
    - The installer will check for any required dependencies and will download and install them as needed.
    - Click **"Execute"** to start the installation process. This may take a few minutes.

Step 3: Configure MySQL Server

1. **Configuration:**
    - After the installation is complete, the MySQL Installer will launch the configuration wizard.
    - Click **"Next"** to begin the configuration.
2. **Server Configuration:**
    - Choose the **"Standalone MySQL Server"** option.
    - Select the appropriate configuration type based on your use case (Development Machine, Server Machine, Dedicated Machine). The "Development Machine" option is typically suitable for most users.
3. **Connectivity:**
    - Ensure that the **"TCP/IP"** option is checked and leave the default port (3306) unless you have a specific reason to change it.
    - Click **"Next"**.
4. **Authentication Method:**
    - Choose the authentication method you prefer. The recommended option is "Use Strong Password Encryption (RECOMMENDED)".
    - Click **"Next"**.
5. **Account and Roles:**
    - Set a strong password for the root account.
    - You can also add additional MySQL user accounts if needed.
    - Click **"Next"**.
6. **Windows Service:**
    - Choose to run MySQL as a Windows Service (recommended) and optionally, select "Start the MySQL Server at System Startup".
    - Click **"Next"**.
7. **Apply Configuration:**
    - Review the configuration settings and click **"Execute"** to apply them.
    - Once the configuration is applied, click **"Finish"**.

Step 4: Verify the Installation

1. **Open MySQL Workbench:**
    - Launch MySQL Workbench from the Start menu.
2. **Connect to MySQL Server:**
    - Click on the **"Local instance MySQL56"** (or similar) connection.
    - Enter the root password you set during configuration.
    - Click **"OK"** to connect.
3. **Verify the connection:**
    - If the connection is successful, you will see the MySQL Workbench interface with access to your databases.

Step 5: Basic MySQL Commands

1. **Open a new SQL tab in MySQL Workbench:**
    - Click on the **"SQL"** tab in MySQL Workbench to open a new query tab.
2. **Create a new database:**
    - Run the following SQL command to create a new database:

```sql
CREATE DATABASE my_database;
```

    - You should see a confirmation message that the database was created successfully.
3. **Create a new table:**
    - Run the following SQL command to create a new table in your database:

```sql
USE my_database;
CREATE TABLE users (
        id INT AUTO_INCREMENT PRIMARY KEY,
        username VARCHAR(50) NOT NULL,
        email VARCHAR(100) NOT NULL );
```

    - You should see a confirmation message that the table was created successfully.
4. **Insert data into the table:**
    - Run the following SQL command to insert data into the table:

```sql
USE my_database;
INSERT INTO users (username, email)
VALUES ('john_doe', 'john@example.com');
```

5. **Query the data:**
    - Run the following SQL command to query the data:

```sql
SELECT * FROM users;
```

By following these steps, you will have MySQL installed and configured on your Windows 11 machine, ready for use with your projects.

7. Set Up Development Environments and Virtualization (Optional):

```
Consider using virtualization tools like Docker or virtual machines to isolate project dependencies and ensure consistent environments across different machines.
```

Setting up development environments and using virtualization tools can significantly enhance your workflow by isolating project dependencies and ensuring consistency. Below are the steps to set up virtual machines using VirtualBox and Vagrant.

Step 1: Install VirtualBox

VirtualBox is a free and open-source virtualization tool that allows you to create and manage virtual machines.

1. **Download VirtualBox:**
    - Go to the VirtualBox download page.
    - Click on the **"Windows hosts"** link to download the installer.
2. **Install VirtualBox:**
    - Run the downloaded installer and follow the installation wizard.
    - Accept the default settings unless you have specific requirements.
    - Complete the installation process.

Step 2: Install Vagrant

Vagrant is a tool for building and managing virtual machine environments. It simplifies the process of setting up development environments.

1. **Download Vagrant:**
    - Go to the Vagrant download page.
    - Click on the **"Windows"** link to download the installer.
2. **Install Vagrant:**
    - Run the downloaded installer and follow the installation wizard.
    - Accept the default settings unless you have specific requirements.
    - Complete the installation process.

Step 3: Set Up a Virtual Machine with Vagrant and VirtualBox

1. **Create a Project Directory:**
    - Open a command prompt and navigate to the directory where you want to create your project.

```sh
mkdir my_vagrant_project
cd my vagrant project
```

2. **Initialize Vagrant:**
    - Run the following command to initialize Vagrant in your project directory:

```sh
vagrant init
```

    - This command creates a `Vagrantfile` in your project directory. The `Vagrantfile` is a configuration file used to define the virtual machine.

3. **Configure the Vagrantfile:**
    - Open the `Vagrantfile` in a text editor (such as Visual Studio Code).
    - Modify the `Vagrantfile` to specify the base box and other configurations. For example:

```ruby
Vagrant.configure("2") do |config|
 config.vm.box = "ubuntu/bionic64"
 config.vm.network "private_network", type: "dhcp"
 config.vm.provider "virtualbox" do |vb|
    vb.memory = "1024"
    vb.cpus = 2
 end
end
```

    - This configuration sets up an Ubuntu 18.04 (Bionic) virtual machine with 1GB of RAM and 2 CPUs.

4. **Start the Virtual Machine:**
    - In the command prompt, run:

```sh
vagrant up
```

    - Vagrant will download the specified base box (if not already downloaded) and start the virtual machine.

Step 4: Access the Virtual Machine

1. **SSH into the Virtual Machine:**
    - Run the following command to access the virtual machine via SSH:

```sh
vagrant ssh
```

2. **Configure the Environment:**
    - Once logged into the virtual machine, you can install the necessary dependencies and configure the environment for your project. For example, to install Python:

```sh
sudo apt update
sudo apt install python3 python3-pip
```

Step 5: Manage the Virtual Machine

1. **Suspend the Virtual Machine:**
    - To suspend the virtual machine, run:

```sh
vagrant suspend
```

2. **Resume the Virtual Machine:**
    - To resume a suspended virtual machine, run:

```sh
vagrant resume
```

3. **Stop the Virtual Machine:**
    - To stop the virtual machine, run:

```sh
vagrant halt
```

4. **Destroy the Virtual Machine:**
    - To destroy the virtual machine and remove all associated resources, run:

```sh
vagrant destroy
```

Optional: Use Docker for Containerization

Instead of virtual machines, you can use Docker to create lightweight, portable containers for your development environments. Containers share the same OS kernel, making them more efficient than full virtual machines.

1. **Install Docker Desktop:**
    - Go to the Docker Desktop download page.
    - Download and install Docker Desktop for Windows.
    - Follow the installation instructions and start Docker Desktop.
2. **Create a Dockerfile:**
    - Create a `Dockerfile` in your project directory to define the container environment. For example:

```
Dockerfile

FROM python:3.8-slim-buster
WORKDIR /app
COPY . /app
RUN pip install --no-cache-dir -r requirements.txt
CMD ["python", "app.py"]
```

3. **Build and Run the Container:**
    - Build the Docker image:

```
sh
docker build -t my-python-app
```

    - Run the Docker container:

```
sh
docker run -d -p 5000:5000 my-python-app
```

Using VirtualBox with Vagrant or Docker for virtualization can help you maintain consistent and isolated development environments, reducing the "works on my machine" problem and making it easier to manage dependencies.

8. Explore Extensions and Plugins:

```
Explore available extensions, plugins, and add-ons for your chosen text editor or
IDE to enhance functionality, such as syntax highlighting, linting, code
formatting, and version control integration.
```

Exploring and installing extensions and plugins can greatly enhance the functionality of your text editor or IDE. For Visual Studio Code (VS Code), a popular and highly customizable text editor, here are some steps to explore and install useful extensions:

Step 1: Open Visual Studio Code

1. **Launch VS Code:**
    - Open Visual Studio Code from the Start menu or by searching for "Visual Studio Code".

Step 2: Access the Extensions View

1. **Open Extensions View:**
    - Click on the Extensions icon in the Activity Bar on the side of the window, or press **`Ctrl+Shift+X`**.

Step 3: Explore Popular Extensions

1. **Search for Extensions:**
    - Use the search bar in the Extensions view to find extensions. For example, you can search for "Python" to find Python-related extensions.
2. **Browse Recommended Extensions:**
    - VS Code provides a list of recommended extensions based on your current setup and the languages you are using. You can find these recommendations in the Extensions view.

Step 4: Install Essential Extensions

Here are some essential extensions you might consider installing to enhance your development workflow:

1. **Python:**
    - Provides support for Python development, including linting, debugging, IntelliSense, and code navigation.

```sh
ms-python.python
```

2. **Pylance:**
    - An extension for Python that provides fast, feature-rich language support including type checking, IntelliSense, and more.

```sh
ms-python.vscode-pylance
```

3. **ESLint:**
    - Integrates ESLint into VS Code, providing real-time linting for JavaScript and TypeScript.

```sh
dbaeumer.vscode-eslint
```

4. **Prettier - Code formatter:**
    - An opinionated code formatter that supports many languages and integrates with VS Code.

```sh
esbenp.prettier-vscode
```

5. **GitLens:**
    - Enhances the built-in Git capabilities in VS Code, providing powerful features like Git blame annotations, code lens, and more.

```sh
eamodio.gitlens
```

6. **Docker:**
    - Provides tools to work with Docker containers and images directly from VS Code.

```sh
ms-azuretools.vscode-docker
```

7. **Live Server:**
    - Launches a local development server with live reload feature for static and dynamic pages.

```sh
ritwickdey.LiveServer
```

Step 5: Install Extensions

1. **Install an Extension:**
    - Click the **"Install"** button next to the extension you want to install. For example, to install the Python extension, click "Install" next to "Python" by Microsoft.
2. **Manage Installed Extensions:**
    - You can manage your installed extensions by clicking the gear icon next to an installed extension and selecting options like "Disable" or "Uninstall".

Step 6: Configure Extensions

1. **Access Extension Settings:**
    - Many extensions have customizable settings. To access these, go to the Extensions view, click the gear icon next to the installed extension, and select "Extension Settings".
2. **Configure Settings:**
    - Adjust the settings to match your workflow. For example, for the Python extension, you might configure the interpreter path or adjust linting settings.

Step 7: Explore Additional Plugins

1. **Explore Themes:**
    - VS Code also supports a variety of themes to customize the look and feel of the editor. Search for "themes" in the Extensions view to find and install new themes.
2. **Install Language Support:**
    - For other programming languages, search for and install language-specific extensions. For example, for JavaScript/TypeScript, you might install `` `TypeScript Hero` `` or `` `JavaScript (ES6) code snippets` ``.
3. **Enhance Productivity:**
    - Explore productivity extensions like `` `Todo Tree` `` for managing TODO comments in your code, or `` `Code Spell Checker` `` to help with spelling errors.

By exploring and installing these extensions and plugins, you can significantly enhance the functionality of VS Code, tailor it to your specific development needs, and streamline your workflow.