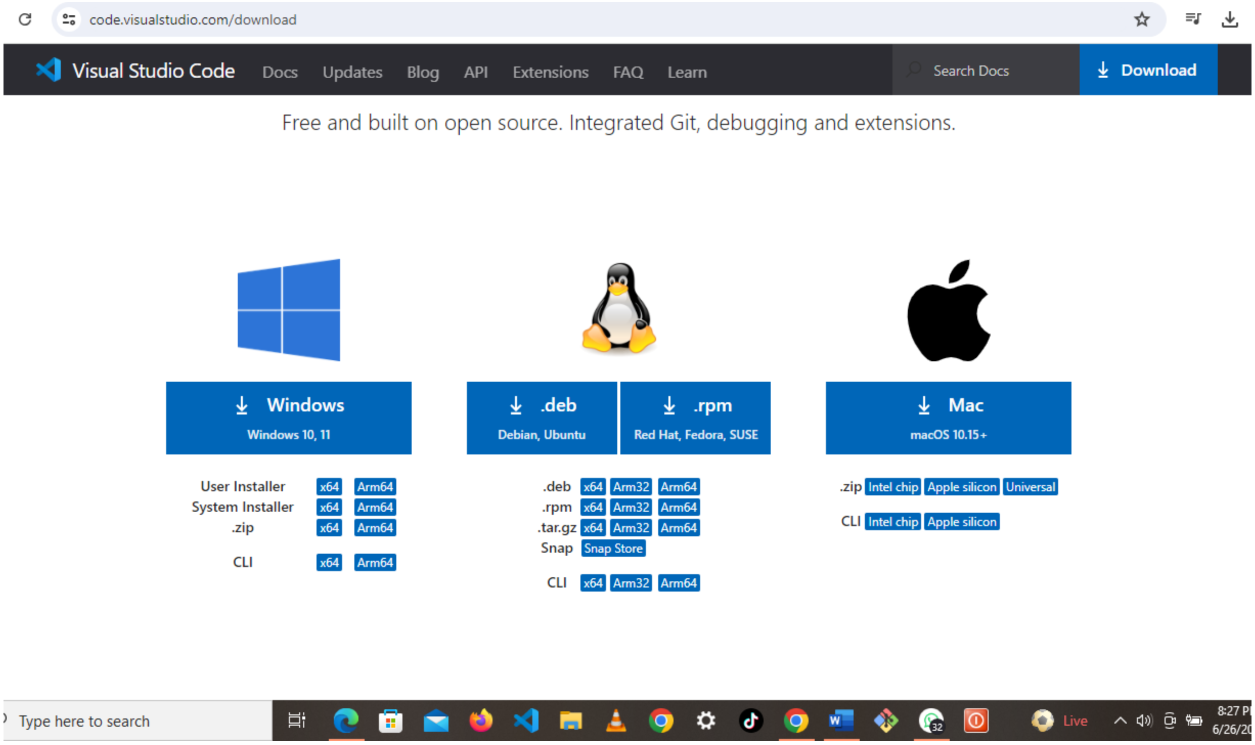
installation of VS Code:

1. **Download VS Code**:
   - Visit the official [VS Code download page](https://code.visualstudio.com/download).
   - Choose the version specific to your OS (Windows).
   - The download process will start automatically.
2. **Installation**:
   - Once the download is complete, open the downloaded file.
   - Accept the license agreement.
   - Click on “Next” and then “Install.”
   - VS Code will be installed under the default directory: `C:\Users\`



Free and built on open source. Integrated Git, debugging and extensions.

User Installer	x64	Arm64
System Installer	x64	Arm64
.zip	x64	Arm64
CLI	x64	Arm64

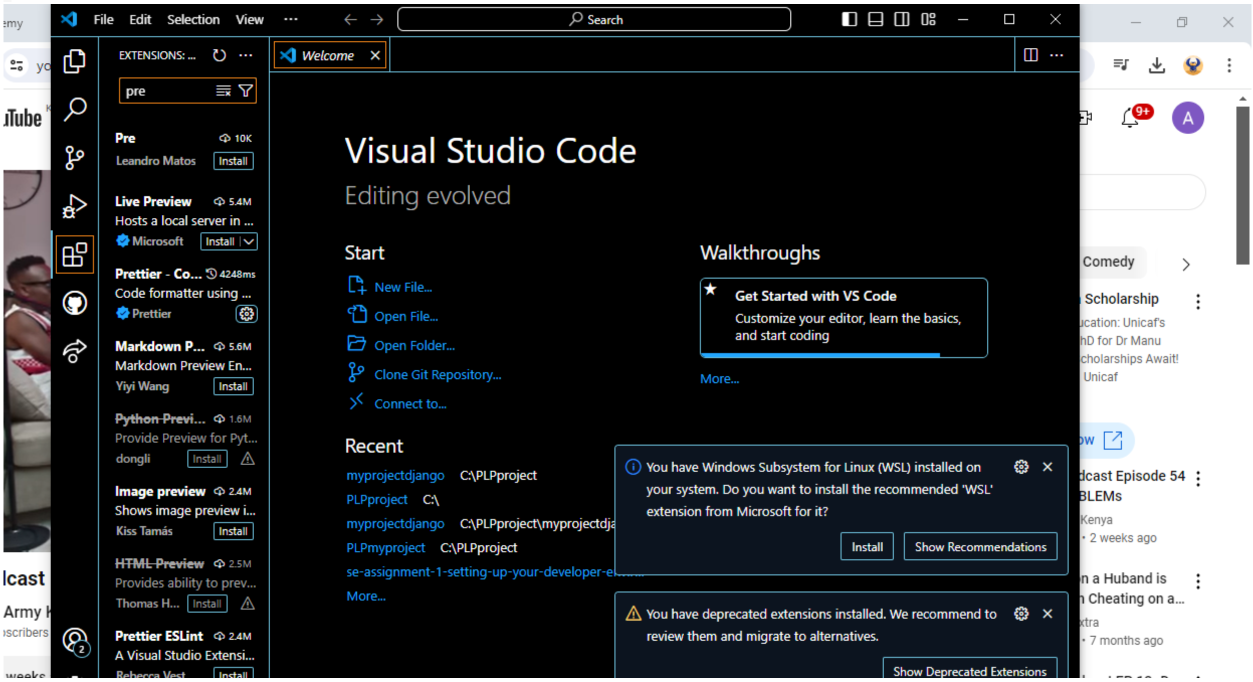
.deb	x64	Arm32	Arm64
.rpm	x64	Arm32	Arm64
.tar.gz	x64	Arm32	Arm64
Snap	Snap Store		
CLI	x64	Arm32	Arm64

.zip	Intel chip	Apple silicon	Universal
CLI	Intel chip	Apple silicon	

After installing VS Code, what initial configurations and settings should be adjusted for an optimal coding environment? Mention any important settings or extensions

1. **Extensions**:
    - Install relevant extensions based on your programming language and workflow. Some popular ones include:
        - **Python**: Install the "Python" extension for Python development.
        - **Git**: Install the "GitLens" extension for better Git integration.
        - **Live Share**: Install "Live Share" for collaborative coding sessions.
        - **Bracket Pair Colorizer**: Helps visualize matching brackets.
        - **Prettier**: For code formatting.
        - **Material Theme**: Choose a theme you like.
    - You can explore and install extensions from the Extensions sidebar in VS Code.
2. **Settings**:
    - Access settings by clicking the gear icon in the lower-left corner and selecting "Settings."
    - Customize settings like font size, tab size, and line wrapping.
    - Set your preferred theme (light or dark).
    - Configure auto-save behavior and line numbers.
3. **Keybindings**:
    - Customize keybindings to match your preferences. Go to "File" > "Preferences" > "Keyboard Shortcuts."
4. **Workspace Settings**:
    - If you're working on a specific project, create a workspace and set project-specific settings.
5. **Integrated Terminal**:
    - VS Code has an integrated terminal. Customize it by going to "View" > "Terminal."
6. **Version Control**:
    - Connect VS Code to your Git repository. Use the Source Control sidebar.

.



You have Windows Subsystem for Linux (WSL) installed on your system. Do you want to install the recommended 'WSL' extension from Microsoft for it?

You have deprecated extensions installed. We recommend to review them and migrate to alternatives.

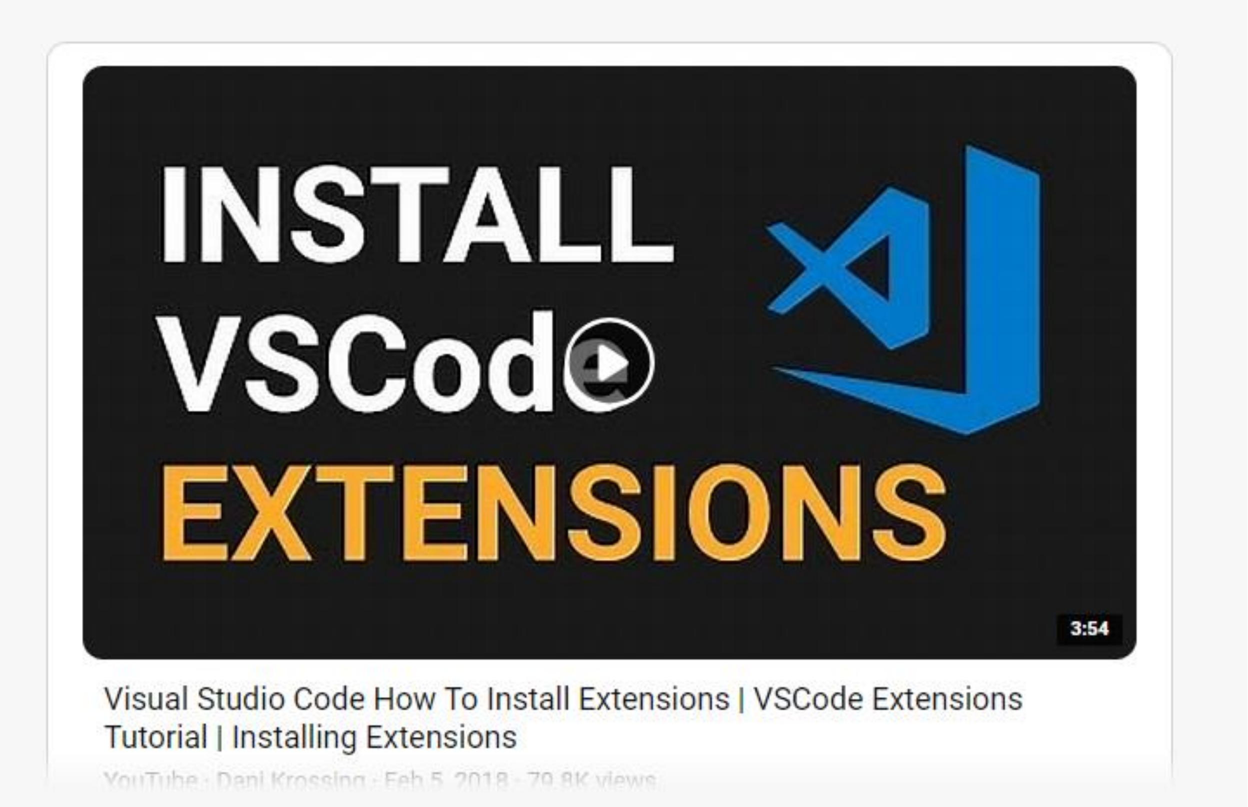
1. **Activity Bar**:
   - Located on the side (usually on the left), the Activity Bar provides quick access to different functionalities:
     - **Explorer**: Navigates files and folders in your workspace.
     - **Search**: Allows you to search across files.
     - **Source Control**: Integrates with Git for version control.
     - **Run and Debug**: Manages debugging sessions and tasks.
     - **Extensions**: Access installed extensions.
2. **Side Bar**:
   - Adjacent to the Activity Bar, the Side Bar contains panels:
     - **Explorer**: Displays your project's file structure.
     - **Source Control**: Shows Git changes and branches.
     - **Extensions**: Lists installed extensions.
     - **Search**: Provides search results.
     - **Outline**: Shows symbols and functions in the current file.
     - **Debug**: Helps with debugging tasks.
3. **Editor Group**:
   - The central area where you write code.
   - You can split it into multiple editors (tabs) for different files.
   - Each editor has its own set of tabs, allowing you to work on multiple files simultaneously.
4. **Status Bar**:
   - Located at the bottom, the Status Bar provides essential information:
     - **Line and Column Numbers**: Shows your cursor position.
     - **Language Mode**: Indicates the programming language.

        - **Git Branch**: Displays the active Git branch.
        - **Notifications**: Alerts about tasks or extensions.
        - **Extensions**: Quick access to installed extension

5. Command Palette:
    - What is the Command Palette in VS Code, and how can it be accessed? Provide examples of common tasks that can be performed using the Command Palette.

Extensions in VS Code: Discuss the role of extensions in VS Code. How can users find, install, and manage extensions? Provide examples of essential extensions for web development.

1. **Role of Extensions**:
    - **Enhancement**: Extensions add languages, debuggers, and tools to your VS Code installation, tailoring it to your needs.
    - **Extensibility**: VS Code's rich model allows extension authors to plug directly into the UI and contribute functionality.
2. **Finding and Installing Extensions**:
    - Open the **Extensions view** by clicking the icon in the Activity Bar or using the shortcut (⇧⌘X or Windows/Linux Ctrl+Shift+X).
    - Browse popular extensions on the **VS Code Marketplace**.
    - Search for an extension, click **Install**, and wait for it to download.
    - For example, install the **TODO Highlight** extension to find and highlight "TODO:" and "FIXME:" comments in your code[1].
3. **Essential Extensions for Web Development**:
    - Here are some must-have extensions:
        - **JavaScript (ES6) Code Snippets**: Provides code snippets for JavaScript, TypeScript, Vue, React, and HTML[2].
        - **CSS Peek**: Allows you to jump to CSS code using classes and IDs[2].
        - **Auto Close Tag**: Automatically adds closing tags for HTML and XML[2].
        - **REST Client**: Test APIs directly within VS Code[2].
        - **ESLint**: Linting utility for JavaScript to catch common errors[2].
        - **Prettier**: Auto-formats code consistently[2].
        - **Live Server**: Runs and reloads HTML files in the browser[3].
        - **Path Intellisense**: Autocompletes file paths[2].
        - **GitLens**: Enhances Git integration[2].



Visual Studio Code How To Install Extensions | VSCode Extensions Tutorial | Installing Extensions

YouTube · Dani Krossing · Feb 5, 2018 · 79.8K views

Describe how to open and use the integrated terminal in VS Code. What are the advantages of using the integrated terminal compared to an external terminal?

1. **Opening the Integrated Terminal**:
   - To open the terminal, press `Ctrl + Backtick` (or `Cmd + Backtick` on macOS).
   - Alternatively, go to the menu: `View` > `Terminal`.
2. **Using the Integrated Terminal**:
   - Once open, you'll see a terminal at the bottom of the VS Code window.
   - You can type commands directly into the terminal and press `Enter` to execute them.
   - Use standard terminal commands like `cd`, `ls`, `mkdir`, etc.
   - You can also run build scripts, start servers, or run any other command-line tasks.
3. **Advantages of the Integrated Terminal**:
   - **Seamless Integration**: The terminal is part of VS Code, so you don't need to switch between different applications.
   - **Context Awareness**: The terminal opens in the current workspace directory, making it contextually relevant.
   - **Split Views**: You can split the terminal into multiple panes, allowing simultaneous execution of different commands.
   - **Customization**: Configure the terminal's appearance, font, and shell (e.g., PowerShell, Bash, or Command Prompt).

- **Debugging**: Debug directly from the terminal using breakpoints and other debugging features.

File and Folder Management: Explain how to create, open, and manage files and folders in VS Code. How can users navigate between different files and directories efficiently?

1. **Opening a Folder (Workspace)**:
   - To open a folder (workspace), use **File > Open Folder…** from the menu.
   - Alternatively, if you launch VS Code from a terminal, pass the folder path as an argument (e.g., `code .` opens the current folder).
2. **Creating Files and Folders**:
   - Use the **Explorer view** (usually on the left):
     - Click the **New File** button (paper with a plus sign) to create a file.
     - Click the **New Folder** button (folder with a plus sign) to create a folder[1].
   - You can also drag and drop files and folders.
3. **Managing Files and Folders**:
   - Right-click to:
     - **Rename**, **delete**, or **move** files/folders.
     - **Open with VS Code** (or double-click) to edit files.
     - Use keyboard shortcuts (e.g., `Ctrl+O` to open a file).
4. **Multi-root Workspaces** (Advanced):
   - Combine multiple folders into one workspace using a `.code-workspace` JSON file.
   - Useful for complex projects with distinct folders[2].
   -

Settings and Preferences: Where can users find and customize settings in VSCode? Provide examples of how to change the theme, font size, and keybindings.

1.
- **Open the Settings Editor**:
  - Navigate to **File > Preferences > Settings**.
  - Alternatively, use the Command Palette (⇧⌘P or Windows/Linux Ctrl+Shift+P) and search for **Preferences: Open Settings (JSON)** or **Preferences: Open Settings (UI)**[1].
- **Examples of Customization**:
  - **Change Theme**:
    - Go to the **Settings editor**.
    - Search for “Color Theme” and choose your preferred theme (e.g., Dark+ or Light+).
  - **Adjust Font Size**:
    - Search for “Font Size” and modify the value (e.g., 14px).

Debugging in VS Code: Outline the steps to set up and start debugging a simple program in VS Code. What are some key debugging features available in VS Code?

1. **Create a Simple Application**:
    - First, create a sample Node.js application (e.g., a “Hello World” JavaScript program).
    - Ensure you have Node.js installed.
2. **Open the Run and Debug View**:
    - Click the **Run and Debug icon** in the Activity Bar (or use the shortcut ⇧⌘D on macOS, Ctrl+Shift+D on Windows/Linux).
    - This view displays all relevant information for running and debugging.
3. **Launch Configurations**:
    - VS Code uses a `launch.json` file to configure debugging settings.
    - If no `launch.json` exists, VS Code shows the **Run start view**.
    - Create a `launch.json` file (manually or automatically) to specify your debug environment.
4. **Set Breakpoints**:
    - Place breakpoints in your code where you want to pause execution.
    - Use breakpoints to inspect variables, view call stacks, and understand program flow.
5. **Start Debugging**:
    - Press **F5** or select **Run and Debug**.
    - VS Code will try to run your currently active file based on the launch configuration.
6. **Debugging Features**:
    - **Step Over**: Execute code line by line.
    - **Inspect Variables**: View variable values during execution.
    - **Call Stack**: Understand method invocation order.
    - **Breakpoints**: Pause execution at specific points.
    - **Watch Expressions**: Monitor specific variables.
    - **Restart Quickly**: Resume debugging without re-launching.
    - **Interactive Playground**: Experiment with code during debugging

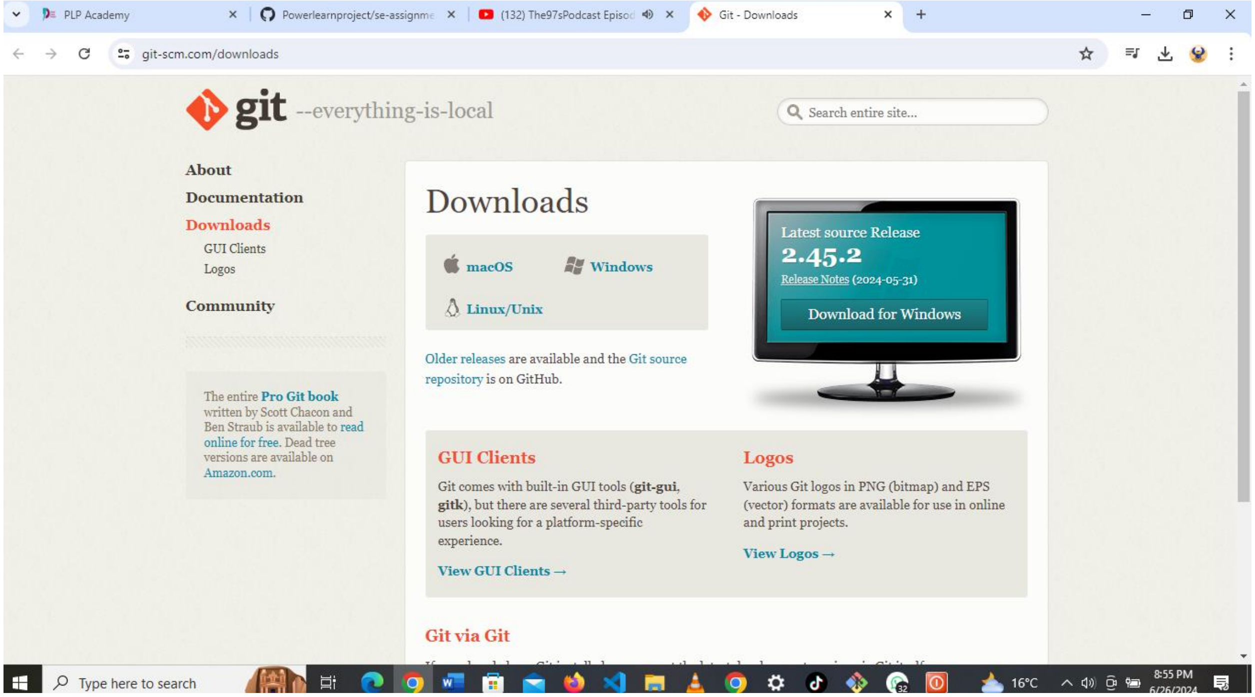
Using Source Control:

- How can users integrate Git with VS Code for version control? Describe the process of initializing a repository, making commits, and pushing changes to GitHub.

7. **Install Git**:
    - Ensure Git is installed on your computer. If not, download and install it from the official Git website.
8. **Open a Project Folder**:
    - Open your project folder in VS Code (File > Open Folder…).
    - VS Code will automatically detect Git if it’s initialized within the folder.
9. **Initialize a Repository**:

   - If your project doesn’t have a Git repository yet, create one:
      1. Click the Source Control icon in the Activity Bar (on the left).
      2. Select “Initialize Repository.”
      3. Choose the project folder.
   - VS Code will create a `.git` folder to track changes.

10. **Stage and Commit Changes**:
    - Make changes to your files (e.g., edit code).
    - In the Source Control view:
       1. Click the “+” icon next to changed files to stage them.
       2. Enter a commit message.
       3. Click the checkmark icon to commit.
    - Your changes are now saved locally.
11. **Push to GitHub**:
    - If you want to sync your local changes with a remote repository (e.g., GitHub):
       1. Create a repository on GitHub.
       2. Copy the repository URL.
       3. In VS Code:
          1. Click the three dots (…) in the Source Control view.
          2. Choose “Push” and paste the repository URL.
          3. Authenticate with your GitHub credentials.
          4. Confirm the push.
12. **Cloning a Repository**:
    - To work on an existing GitHub repository:
       1. Click the Source Control icon.
       2. Choose “Clone Repository.”
       3. Paste the repository URL.
       4. Select a local folder to clone into.



Downloads

- macOS
- Windows
- Linux/Unix

Older releases are available and the Git source repository is on GitHub.

Latest source Release
2.45.2
Release Notes (2024-05-31)

Download for Windows

The entire **Pro Git book** written by Scott Chacon and Ben Straub is available to read online for free. Dead tree versions are available on Amazon.com.

GUI Clients

Git comes with built-in GUI tools (**git-gui**, **gitk**), but there are several third-party tools for users looking for a platform-specific experience.

View GUI Clients →

Logos

Various Git logos in PNG (bitmap) and EPS (vector) formats are available for use in online and print projects.

View Logos →

Git via Git