1 Installation of VS Code:

Here are the steps to download and install Visual Studio Code on Windows 11 operating system ¹:  
Prerequisites:

* Ensure your Windows OS is updated, as Visual Studio Code requires a recent Windows build.  
  Steps:
* Download the Visual Studio Code installer for Windows.
* Run the installer (VSCodeUserSetup-{version}.exe).
* The installation process takes only a minute to complete.
* By default, VS Code will be installed under C:\Users{Username}\AppData\Local\Programs\Microsoft VS Code.  
  Alternative method:
* Download a Zip archive of Visual Studio Code.
* Extract the contents of the archive.
* Run Code from there.  
  Note:  
  Visual Studio Code provides both Windows user and system level setups. The user setup does not require Administrator privileges to run. However, when running VS Code as Administrator in a user setup installation, updates will be disabled. The system setup requires elevation to Administrator privileges to run.

2 First-time Setup:

After installing VS Code, here are some initial configurations and settings to adjust for an optimal coding environment:

1. **Theme and Color Scheme**: Personalize your editor's appearance by selecting a theme and color scheme that suits your preferences.
2. **Font and Size**: Adjust the font family and size to improve readability.
3. **Indentation and Formatting**: Set your preferred indentation size, format, and language-specific settings.
4. **Extensions**: Install essential extensions like:
   * **ESLint** for code linting and error detection
   * **Prettier** for code formatting
   * **Debugger for Chrome** or **Debugger for Firefox** for browser debugging
   * **Live Server** for a local development server
   * **Git Extension Pack** for version control
5. **Settings Sync**: Enable settings sync to synchronize your configurations across devices.
6. **Code Completion**: Enable IntelliSense for intelligent code completion.
7. **Terminal**: Configure the integrated terminal to your preferred shell and settings.
8. **Explorer**: Customize the file explorer to show or hide files and folders as needed.
9. **Status Bar**: Customize the status bar to display relevant information.
10. **Keyboard Shortcuts**: Familiarize yourself with VS Code's keyboard shortcuts and customize them to your liking.

Important settings to explore:

* **files.trimTrailingWhitespace**: Remove trailing whitespace from files
* **editor.formatOnSave**: Format code on save
* **editor.codeActionsOnSave**: Run code actions (like linting and formatting) on save

These configurations and extensions will enhance your coding experience in VS Code.

3 User Interface Overview:

The VS Code user interface consists of four main components:

**1. Activity Bar**:  
\* Located on the left side of the window  
\* Displays icons for various activities, such as:  
+ Explorer (file navigation)  
+ Search  
+ Debug  
+ Extensions  
\* Clicking an icon opens the corresponding panel or view

**2. Side Bar**:  
\* Also located on the left side, adjacent to the Activity Bar  
\* Displays the contents of the selected activity, such as:  
+ File Explorer (directory tree)  
+ Search results  
+ Debug console  
+ Extension marketplace

**3. Editor Group**:  
\* The main area where your code is displayed  
\* Can be split into multiple editor panels (horizontal or vertical)  
\* Each editor panel displays a file or document

**4. Status Bar**:  
\* Located at the bottom of the window  
\* Displays information about the current file or project, such as:  
+ File name and path  
+ Language and encoding  
+ Line and column numbers  
+ Errors and warnings  
\* Also provides shortcuts to common actions, like saving or debugging

These components work together to provide a flexible and customizable interface for coding, debugging, and managing projects in VS Code.

4 Command Palette:

The Command Palette, also known as the "Command Menu", is a central location in VS Code where you can access and execute various commands, tasks, and features. To access the Command Palette, you can use the following methods:

* Press Ctrl + Shift + P (Windows, Linux) or Cmd + Shift + P (macOS)
* Open the **View** menu and select **Command Palette**
* Use the keyboard shortcut F1 and then select **Command Palette**

Some common tasks that can be performed using the Command Palette include:

* **Switching editor layouts**: Use the Command Palette to switch between different editor layouts, such as splitting the editor into multiple panels or opening a new editor window.
* **Running tasks**: Access and run tasks defined in your tasks.json file, such as building or linting your code.
* **Debugging**: Start or stop debugging sessions, or access debugging-related commands like "Step Over" or "Step Into".
* **Git commands**: Perform Git-related tasks like committing changes, pulling or pushing code, or checking out branches.
* **Formatting code**: Format your code using the "Format Document" or "Format Selection" commands.
* **Changing settings**: Quickly change various VS Code settings, such as the theme, font size, or indentation settings.
* **Opening files**: Open files or folders using the "Open File" or "Open Folder" commands.
* **Extensions management**: Manage your extensions, including installing, updating, or uninstalling them.
* **Keybinding editor**: Open the Keybinding editor to customize your keyboard shortcuts.
* **Terminal commands**: Access terminal-related commands, such as creating a new terminal or closing the current terminal.

These are just a few examples of the many tasks you can perform using the Command Palette in VS Code.

5 Extensions in VS Code:

Extensions play a crucial role in VS Code, enhancing its functionality and allowing developers to customize their coding experience. Here's how users can find, install, and manage extensions:

**Finding Extensions:**

* Open the Extensions panel by clicking the Extensions icon in the left sidebar or pressing Ctrl + Shift + X (Windows, Linux) or Cmd + Shift + X (macOS)
* Search for extensions using the search bar
* Browse through the extensions marketplace, which features a wide range of extensions categorized by popularity, trending, and staff picks

**Installing Extensions:**

* Click the "Install" button next to the extension you want to install
* Wait for the extension to download and install
* Reload VS Code by clicking the "Reload" button or pressing F1 and selecting "Reload Window"

**Managing Extensions:**

* View installed extensions in the Extensions panel
* Enable or disable extensions as needed
* Update extensions by clicking the "Update" button next to the extension
* Uninstall extensions by clicking the "Uninstall" button next to the extension

Some essential extensions for web development include:

* **HTML Snippets**: Provides HTML code snippets for quick development
* **CSS Intellisense**: Offers intelligent code completion for CSS
* **JavaScript (ES6) code snippets**: Provides code snippets for JavaScript development
* **Debugger for Chrome**: Allows debugging of JavaScript code in the Chrome browser
* **ESLint**: Integrates ESLint into VS Code for code linting and formatting
* **Path Intellisense**: Offers intelligent code completion for file paths
* **Auto Close Tag**: Automatically closes HTML tags
* **Auto Rename Tag**: Automatically renames paired HTML tags
* **Live Server**: Launches a development server with live reload functionality

These extensions can enhance productivity, improve code quality, and streamline web development tasks in VS Code.

6 Integrated Terminal:

To open the integrated terminal in VS Code:

1. Press Ctrl + Shift + (Windows, Linux) or Cmd + Shift + (macOS)
2. Use the keyboard shortcut F1 and select "Terminal: Create New Terminal"
3. Click the "Terminal" icon in the left sidebar and select "New Terminal"

Once open, you can use the integrated terminal like any other terminal:

* Run commands and scripts
* Interact with your project directory
* Use command-line tools and utilities

Advantages of using the integrated terminal compared to an external terminal:

1. **Convenience**: No need to switch between VS Code and an external terminal window.
2. **Context**: The integrated terminal stays in the context of your project, making it easier to navigate and run commands.
3. **Seamless integration**: VS Code extensions can interact with the integrated terminal, providing features like automatic command execution and output rendering.
4. **Multi-terminal support**: You can open multiple terminals in VS Code, each with its own context and environment.
5. **Better debugging experience**: The integrated terminal allows for a more streamlined debugging experience, especially when used with extensions like Debugger for Chrome.
6. **Improved workflow**: The integrated terminal enables a more efficient workflow, as you can quickly switch between editing code and running commands.

Overall, the integrated terminal in VS Code provides a more streamlined and efficient development experience compared to using an external terminal.

7 File and Folder Management:

In VS Code, you can create, open, and manage files and folders using the following methods:

**Creating Files and Folders:**

1. Open the **Explorer** panel by clicking the Explorer icon in the left sidebar or pressing Ctrl + Shift + E (Windows, Linux) or Cmd + Shift + E (macOS).
2. Right-click in the Explorer panel and select **New File** or **New Folder**.
3. Alternatively, use the keyboard shortcuts Ctrl + N (Windows, Linux) or Cmd + N (macOS) to create a new file, or Ctrl + Shift + N (Windows, Linux) or Cmd + Shift + N (macOS) to create a new folder.

**Opening Files and Folders:**

1. Navigate to the desired file or folder in the Explorer panel.
2. Click on the file or folder to open it.
3. Alternatively, use the keyboard shortcuts Ctrl + O (Windows, Linux) or Cmd + O (macOS) to open a file, or Ctrl + K followed by Ctrl + O (Windows, Linux) or Cmd + K followed by Cmd + O (macOS) to open a folder.

**Managing Files and Folders:**

1. Rename files and folders by clicking on the name and typing a new name.
2. Delete files and folders by right-clicking and selecting **Delete**.
3. Move files and folders by dragging and dropping them into a new location.

**Navigating between Files and Directories:**

1. Use the **Explorer** panel to navigate through your project's directory structure.
2. Use the **Breadcrumbs** navigation bar (above the Editor) to quickly switch between files and directories.
3. Use the **Go to File** command (Ctrl + P (Windows, Linux) or Cmd + P (macOS)) to quickly search and open files.
4. Use the **Go to Folder** command (Ctrl + Shift + P (Windows, Linux) or Cmd + Shift + P (macOS)) to quickly search and open folders.
5. Use the **Recent Files** list (Ctrl + R (Windows, Linux) or Cmd + R (macOS)) to quickly access recently opened files.

By using these methods, users can efficiently create, open, and manage files and folders in VS Code, and navigate between different files and directories with ease.

8 Settings and Preferences:

Users can find and customize settings in VS Code by following these steps ¹:

* Navigate to **File** > **Preferences** > **Settings**.
* Open the Settings editor from the **Command Palette** with **Preferences: Open Settings**.
* Use the keyboard shortcuts **Ctrl + ,** (Windows, Linux) or **Cmd + ,** (macOS).  
  Once the Settings editor is opened, you can search and discover the settings you are looking for. When you search using the search bar, it not only shows and highlights the settings matching your criteria but also filters out those that are not matching.  
  Here are examples of how to change settings:
* **Theme:**  
  To change the theme in VS Code, search for "theme" in the settings search bar and select "Appearance: Theme" to choose from a list of available themes.
* **Font Size:**  
  To change the font size in VS Code, search for "font size" in the settings search bar and select "Editor: Font Size" to change the font size of the editor.
* **Keybindings:**  
  To change keybindings in VS Code, search for "keybindings" in the Command Palette and select "Preferences: Open Keyboard Shortcuts" to view and edit keybindings.

9 Debugging in VS Code:

Here are the steps to set up and start debugging a simple program in VS Code:

**Step 1: Create a launch configuration**

* Open the Run panel by clicking the Run icon in the left sidebar or pressing Ctrl + Shift + D (Windows, Linux) or Cmd + Shift + D (macOS)
* Click the "create a launch.json file" link
* Select the appropriate debugger and configuration

**Step 2: Set breakpoints**

* Open the file you want to debug
* Click in the margin next to the line of code where you want to set a breakpoint
* A red dot will appear, indicating the breakpoint

**Step 3: Start debugging**

* Click the "Run" button or press F5
* The debugger will start, and your program will run until it hits the breakpoint

**Key debugging features available in VS Code:**

* **Breakpoints**: Set breakpoints to pause the program at specific points
* **Step Over**: Step over a line of code to execute it and move to the next line
* **Step Into**: Step into a function or method to debug it
* **Step Out**: Step out of a function or method to return to the caller
* **Variables**: View and edit variable values in the Variables panel
* **Call Stack**: View the call stack to see the sequence of function calls
* **Debug Console**: Interact with the program using the debug console
* **Auto Attach**: Automatically attach the debugger to a running process
* **Conditional Breakpoints**: Set breakpoints that only trigger when a condition is met
* **Logpoints**: Set logpoints to log messages to the console without pausing the program

These features help you debug your program efficiently and effectively in VS Code.

10 Using Source Control:

Here is the process of integrating Git with VS Code ¹ ² ³:

* **Initialize a repository**: Open VS Code in a folder which you want to be the local repository. Then, select the **Initialize Repository** button in the **Source Control** view. You can also run the **Git: Initialize Repository** command from the **Command Palette**.
* **Make commits**: To make a commit, you first need to stage changes by right-clicking on the file in the **Source Control** view and clicking **Stage Changes**. Then, you can write a commit message and commit the changes.
* **Push changes to GitHub**: To push changes to GitHub, you need to link your GitHub account with VS Code. Then, you can push your changes to your GitHub repository. You can also publish your local repository to GitHub directly from VS Code.