Installing VS Code on Windows 11

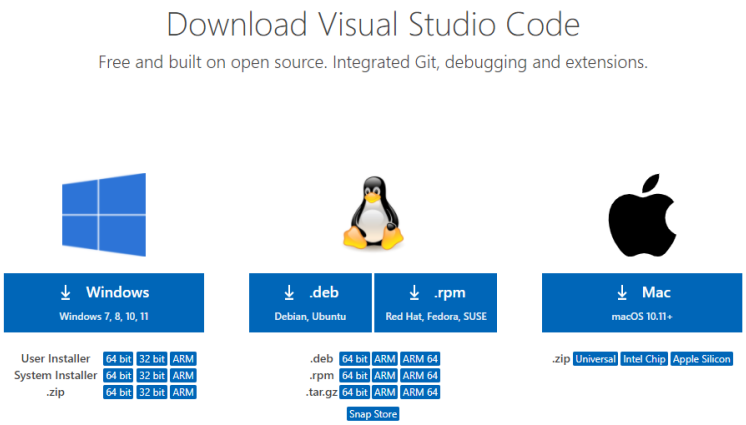
1.1 This guide will walk you through each step to ensure you get VS Code installed and ready to use.

Pre-requisites:

- Have a Windows 11 operating system installed on your computer.
- Ensure that your computer meets the minimum system requirements for Visual Studio Code. This typically includes a 1.6 GHz or faster processor, 1 GB of RAM, and 200 MB of available hard disk space.
- Make sure that your computer has the latest updates and patches installed.
- Have administrator privileges on your computer to install software.
- Check for any compatibility issues with other software or hardware on your computer before installing Visual Studio Code.
- Ensure that your internet connection is stable and reliable for downloading and installing Visual Studio Code.
- Consider backing up your important files and data before installing any new software.

Step 1.

Download the VS Code Installer (https://code.visualstudio.com/docs/?dv=win)

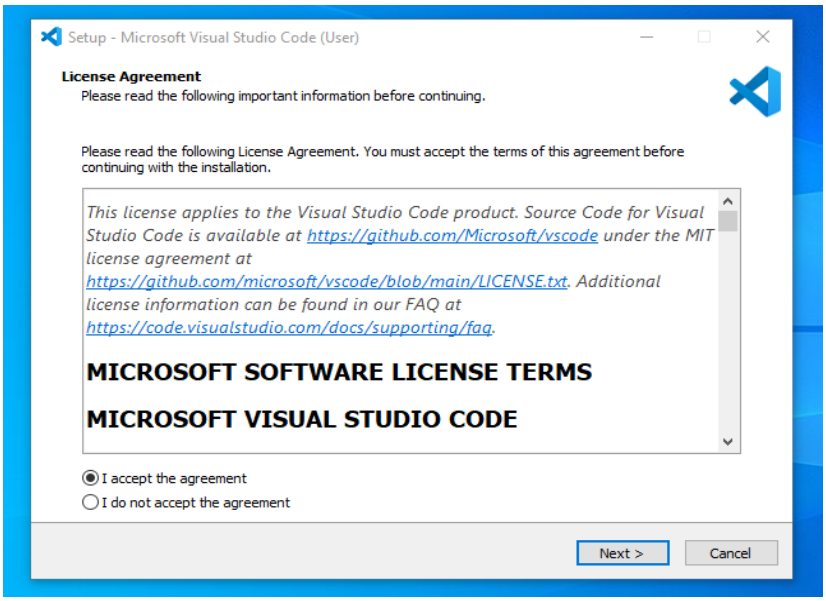


Step 2.

Run the Downloaded Installer

Step 3.

Accept the License Agreement

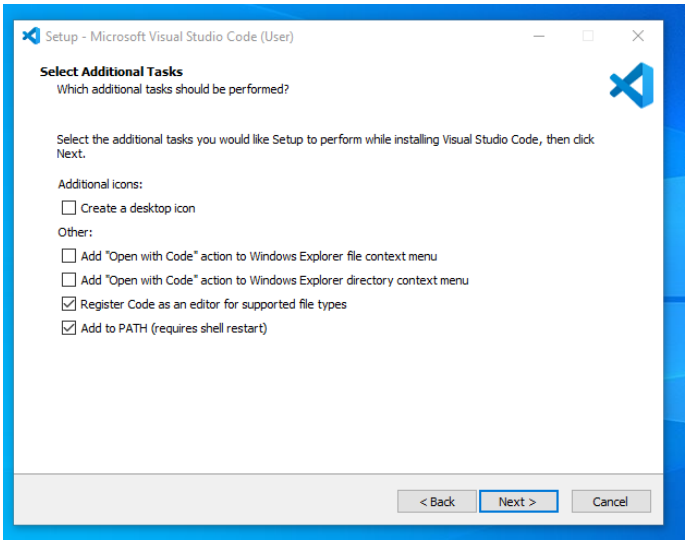


Step 4.

Choose Installation Location

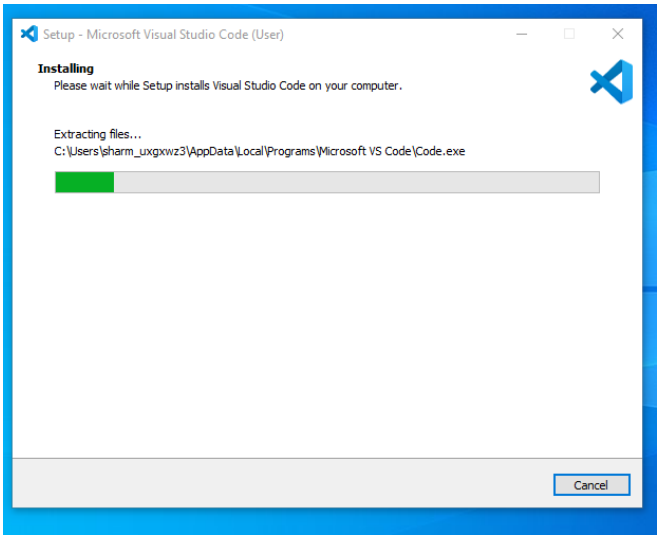
Step 5.

Select Additional Tasks



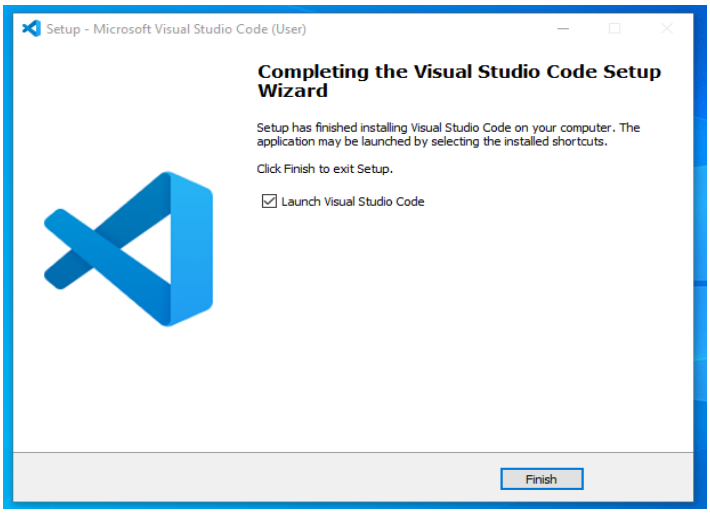
Step 6.

Waiting for the installation



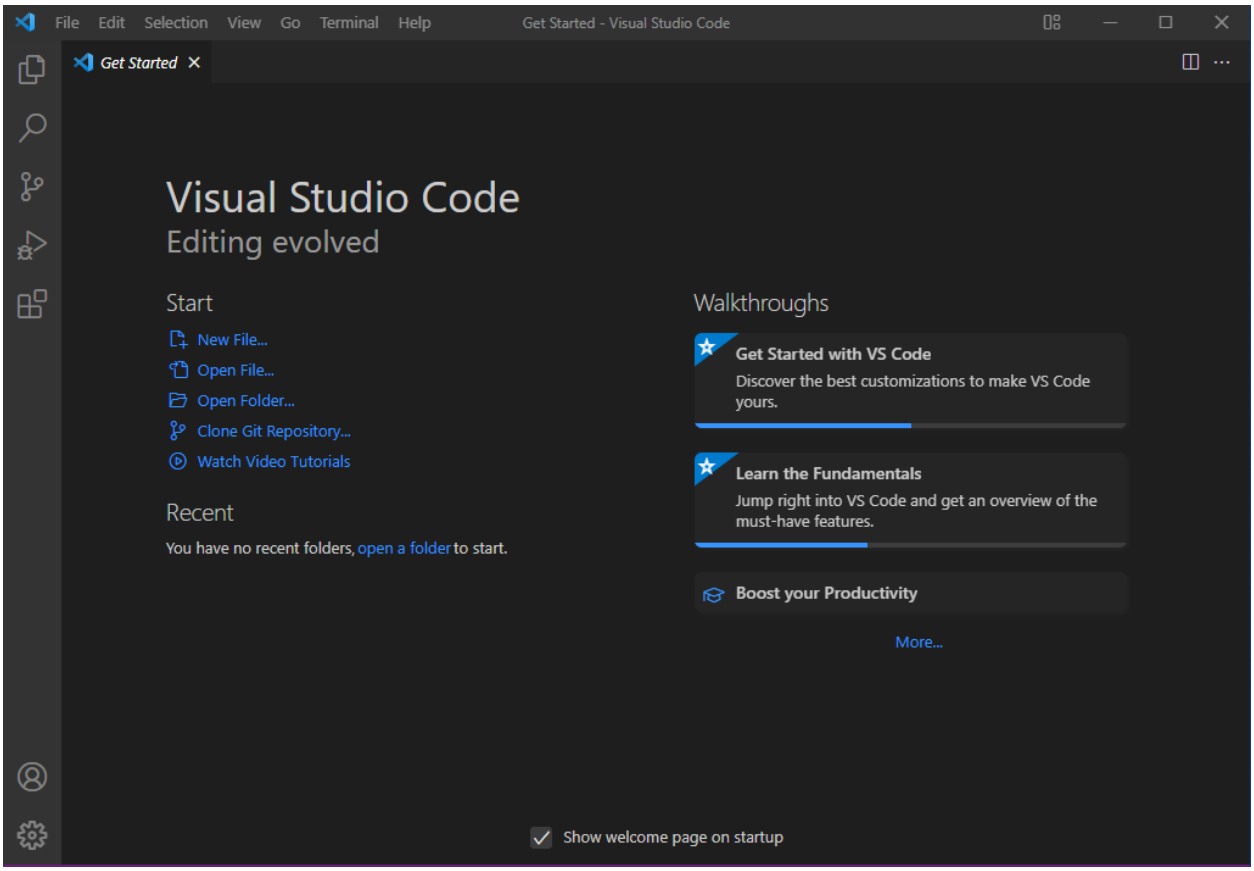
Step 7.

After the Installation setup for Visual Studio Code is finished, it will show a window like this below. Tick the “Launch Visual Studio Code” checkbox and then click Next.



Step 8.

After the previous step, the Visual Studio Code window opens successfully. Now you can create a new file in the Visual Studio Code window and choose a language of yours to begin your programming journey.



1.2 First Time Setup

- Theme: Choose a theme that is easy on the eyes and helps improve readability during coding sessions.

- Font Size and Family: Set the font size and family to your preference to ensure comfortable coding experience.

- Code Formatting: Install extensions like Prettier or ESLint for automatic code formatting to maintain consistency in your code.

- Git Integration: Enable version control by configuring Git settings in VS Code for seamless collaboration and code management.

- Extensions: Install extensions for tools and languages you use frequently to enhance productivity and streamline your workflow.

- Keybindings: Customize keybindings or shortcuts to suit your coding style and improve efficiency while coding.

- Workspace Settings: Configure workspace settings for specific projects to customize behavior and appearance as needed.

1.3. User Interface Overview

- ✓ Visual Studio Code (VS Code) user interface consists of several main components that help users navigate and interact with the editor effectively. These components include the Activity Bar, Side Bar, Editor Group, and Status Bar:
- ✓ Activity Bar: The Activity Bar is located on the far left side of the editor and contains icons for different views and functionalities, such as Explorer, Search, Source Control, and Extensions. The purpose of the Activity Bar is to provide quick access to these essential features and allow users to switch between them easily.
- ✓ Side Bar: The Side Bar is positioned next to the editor and typically displays additional information or navigation options related to the file or project being worked on. It includes views like File Explorer, Search results, and Git information. The Side Bar is useful for managing project files, navigating code, and accessing various functionalities within the editor.
- ✓ Editor Group: The Editor Group is the main area where users can view and edit their code. It consists of one or more editors that can be arranged horizontally or vertically within the editor window. Users can work on multiple files simultaneously by using tabs in the Editor Group, enabling better organization and multitasking while coding.
- ✓ Status Bar: The Status Bar is located at the bottom of the editor and provides information about the current file, project, and editor status. It displays details like line and column numbers, language mode, indentation settings, and Git branch information. The Status

Bar also includes quick actions and shortcuts for performing common tasks, such as changing the language mode or encoding of the file.

1.4. Command Palette:

Command Palette in Visual Studio Code (VS Code) is a powerful feature that allows users to access and execute various commands, settings, and actions within the editor. It serves as a centralized hub for performing tasks quickly and efficiently without needing to navigate through menus or memorize keyboard shortcuts.

Command Palette can be accessed by pressing `Ctrl+Shift+P` on Windows.
Some common tasks that can be performed using the Command Palette in VS Code include:

- ✓ Opening and switching between files: Users can quickly open files or switch between recently viewed files by typing the file name in the Command Palette.
- ✓ Running commands: Users can run various commands, such as formatting code, toggling word wrap, or commenting/uncommenting code blocks, by typing the command name in the Command Palette.
- ✓ Accessing settings: Users can access and modify editor settings by searching for specific settings in the Command Palette, such as changing tab size, enabling/disabling word wrap, or adjusting font settings.
- ✓ Installing extensions: Users can search for and install extensions directly from the Command Palette by typing `Extensions: Install Extensions`.
- ✓ Keyboard shortcuts: Users can view and customize keyboard shortcuts by searching for `Preferences: Open Keyboard Shortcuts` in the Command Palette.
- ✓ Git operations: Users can perform Git operations like staging changes, committing changes, and pushing/pulling from repositories by typing Git-related commands in the Command Palette.

1.5. Extensions in VS Code

Extensions play a crucial role in enhancing the functionality and customization options of VS Code. They allow users to extend the editor's capabilities by adding new features, language support, themes, and tools tailored to their specific needs and coding preferences.

To find, install, and manage extensions in VS Code by follow these steps:

Finding extensions:
- Click on the Extensions icon in the Activity Bar (or press `Ctrl+Shift+X`) to open the Extensions view.
- You search for extensions by name or category in the search bar.
- You can also browse the curated list of popular extensions, recommended extensions, and

new extensions.

Examples of extensions include;

- ✓ ESLint - a popular linter for JavaScript and TypeScript that helps identify and fix code errors and maintain coding standards.
- ✓ Prettier - a code formatter that automatically formats code to adhere to consistent styling rules, enhancing readability and maintainability.
- ✓ Live Server - a tool that launches a local development server with live reload capability, making it easier to preview and test web applications.
- ✓ Debugger for Chrome - an extension that allows users to debug JavaScript code running in the Google Chrome browser directly from the VS Code editor.
- ✓ HTML CSS Support - provides autocompletion and support for HTML and CSS code, improving productivity and reducing errors in web development.
- ✓ GitLens - enhances Git integration in VS Code by providing advanced features like blame information, history tracking, and repository exploration.

1.6 Integrated Terminal

To open and use the integrated terminal in VS Code, follow these steps:

Open the Integrated Terminal:
- Press `Ctrl+`` (backtick) on Windows/Linux or `CMD - Alternatively, click on the `View` menu in the top toolbar and select `Terminal`.

Use the Integrated Terminal:
- Once the terminal is open, users can run commands directly within the terminal window.
- Users can switch between multiple terminal instances by clicking on the dropdown next to the `+` icon in the terminal window.
- They can also customize the terminal settings, such as terminal shell, font size, color scheme, and more, by accessing the terminal settings from the dropdown menu.

Advantages of using the integrated terminal in VS Code compared to an external terminal include:

- ✓ Seamless Integration: The integrated terminal is a part of the VS Code editor, allowing users to switch between coding and terminal tasks seamlessly without switching between different applications.
- ✓ Productivity: By having the terminal integrated directly into the editor, users can run commands, build and test applications, and manage files without leaving the coding environment, leading to increased productivity and efficiency.

- ✓ Context Switching: Users can easily reference and interact with code snippets or output from commands in the terminal while coding in the editor, reducing the need for constant context switching between different windows.
- ✓ Customization: The integrated terminal in VS Code offers customization options such as terminal themes, shell preferences, and font settings, allowing users to personalize their terminal experience according to their preferences.
- ✓ Multi-Platform Support: The integrated terminal supports the same commands and features across different operating systems, providing a consistent experience for users working on various platforms.

1.7. File and Folder Management

Creating Files and Folders

- ✓ To create a new file, users can click on the `File` menu in the top toolbar, select `New File`, and then specify the file name.
- ✓ To create a new folder, users can click on the `File` menu, select `New Folder`, and provide a name for the folder.

Opening Files and Folders:

- ✓ Users can open an existing file by clicking on the `File` menu, selecting `Open...`, and choosing the file they want to open from their system.
- ✓ Users can open a folder (workspace) in VS Code by clicking on the `File` menu, selecting `Open Folder...`, and then selecting the folder location on their system.

Managing Files and Folders:

- ✓ Users can save their files by clicking on the `File` menu and selecting `Save` or `Save As. ` to save the file with a new name or location.
- ✓ To close a file, users can click on the `x` button on the tab corresponding to the file they want to close.
- ✓ Users can also move, copy, rename, or delete files and folders within the VS Code explorer by right-clicking on the file or folder and selecting the desired action from the context menu.

Navigating Between Files and Directories:

- ✓ To navigate between different files, users can click on the tabs at the top of the editor window to switch between open files.
- ✓ Users can also use keyboard shortcuts like `Ctrl+Tab` on Windows or `Cmd+Tab` on macOS to cycle through open files.
- ✓ To navigate between directories and files in the explorer, users can click on files and folders in the sidebar to open them or use the search feature to quickly find and open specific files.

1.8 Settings and Preference

Users can find and customize settings in Visual Studio Code (VS Code) by accessing the Settings menu. To do this, click on the gear icon in the lower-left corner of the VS Code window and select "Settings".

To change the theme in VS Code, you can search for "Color Theme" in the search bar of the Settings menu. From there, you can choose from a variety of pre-installed themes or install new ones by clicking on the "Install Additional Color Themes" button.

To change the font size in VS Code, search for "Font Size" in the search bar of the Settings menu. You can adjust the font size by using the slider or by entering a custom value.

To customize keybindings in VS Code, search for "Keybindings" in the search bar of the Settings menu. You can customize keybindings by clicking on the "Edit in settings.json" link or by clicking on the "keybindings.json" link in the right pane.

In "settings.json" or "keybindings.json", you can add or modify keybinding configurations. For example, to change the keybinding for a specific action, you can add a new keybinding entry with the desired key combination.

1.9 Debugging in VS Code

To set up and start debugging a simple program in VS Code, follow these steps:

- ✓ Install any necessary extensions: Depending on the programming language you are using, you may need to install a debugging extension. You can do this by opening the Extensions view (`Ctrl + Shift + X`) and searching for the appropriate extension.
- ✓ Open your project folder: Open your project folder in VS Code by clicking on "File" "Open Folder" and selecting the folder that contains your program files.
- ✓ Set breakpoints: Set breakpoints in your code by clicking on the line number where you want the program to stop executing. This will allow you to inspect variables and step through your code.
- ✓ Start debugging: Click on the "Run and Debug" icon in the Activity Bar on the side of the VS Code window. Then select the appropriate debug configuration for your program (e.g., Node.js, Python, etc.) and click the green play icon to start debugging.
- ✓ Use debugging features: Once debugging has started, you can use various debugging features in VS Code to analyze and troubleshoot your program. Some key debugging features available in VS Code include:
- ✓ Step Over: Execute the current line of code and move to the next line.
- ✓ Step Into: Execute the current line of code and move to the next line, but if the current line is a function call, move into the function.
- ✓ Step Out: Continue executing the program until the current function is completed.
- ✓ Watch and Local Variables: View and modify the values of variables in real-time while debugging.

- ✓ Call Stack: View the hierarchy of function calls leading up to the current point in the program.
- ✓ Debug Console: Interact with the program through a console interface while debugging.

10. Using Source Control

To integrate Git with Visual Studio Code for version control and collaborate with others via GitHub,

Do follow these steps:

Initialize a Git repository:

- ✓ Open your project folder in VS Code.
- ✓ Click on the 'Source Control' icon in the Activity Bar on the side of the VS Code window (or press `Ctrl + Shift + G`).
  - Click on the 'Initialize Repository' button to initialize a new Git repository for your project.

Stage and commit changes:

- ✓ Make changes to your project files.
- ✓ In the Source Control view, you'll see a list of changed files. Click the '+' icon next to the file(s) you want to include in the commit to stage them.
- ✓ Enter a commit message in the message box at the top of the Source Control view and press `Ctrl + Enter` to commit your changes.

Push changes to GitHub:

- ✓ Sign in to your GitHub account and create a new repository if you haven't already.
- ✓ In VS Code, open the Command Palette by pressing `Ctrl + Shift + P` and search for "Git: Push".
- ✓ Select "Git: Push" from the list and choose the remote repository (GitHub) you want to push your changes to.
- ✓ Enter your GitHub credentials if prompted.
- ✓ Your changes will be pushed to the remote repository on GitHub.

SOURCES

1. **https://www.tcpipacademy.com/install-vscode-windows10/**
2. How to Install VS Code in Windows 11: A Step-by-Step Guide - Support Your Tech
3. **https://code.visualstudio.com/docs/setup/windows**
4. **https://www.geeksforgeeks.org/how-to-install-visual-studio-code-on-windows/**