1

Introducing Visual Studio Code

WHAT'S IN THIS CHAPTER?

- Installing and getting started with Visual Studio Code
- Understanding the cross-platform components that make up Visual Studio Code

GETTING STARTED

The choice of the editor used by any developer is an incredibly personal one. The reason to pick one over the rest depends on a collection of attributes typically related to the tasks they perform on a daily basis. Developers look for functionality, keystroke shortcuts, code snippets, colorations, and more that allow them to stay productive.

Dislodging developers from their choice is not easy. Any change in editors is going to result in an immediate loss of productivity. After all, it takes time to become familiar with the features offered and have them become a natural part of the coding "flow." As a result, it takes a special level of "better" for a developer to switch editors.

For this reason, the success of Visual Studio Code speaks volumes for its features and functionality. Although it has been officially released for just three years (it left public preview in April 2016), it has quickly become one of the top editors in terms of popularity, competing with Sublime Text, Atom, and UltraEdit for the top spot.

But that doesn't matter to you, the reader. What you care about more is what Visual Studio Code can do to help you be productive. As a developer, it is frequently the small things that make the biggest difference—knowing how to add code with a single keyboard chord, being able to do client and server debugging on your `Node.js` project, or language-sensitive code completion. Any, all, or none of those might matter, but the goal of this book is to help you find the five or ten features that matter to you and that will make you excited to use Visual Studio Code.

Installing Visual Studio Code

Visual Studio Code is a cross-platform editor. In this instance, cross-platform means that a version is available to run on Windows (7, 8, and 10), macOS, and Linux. The installation process is similar for each, and the starting point is the same in all cases: `https://code.visualstudio.com/Download`. Figure 1-1 shows what the download page looks like presently, but it's naturally subject to change.

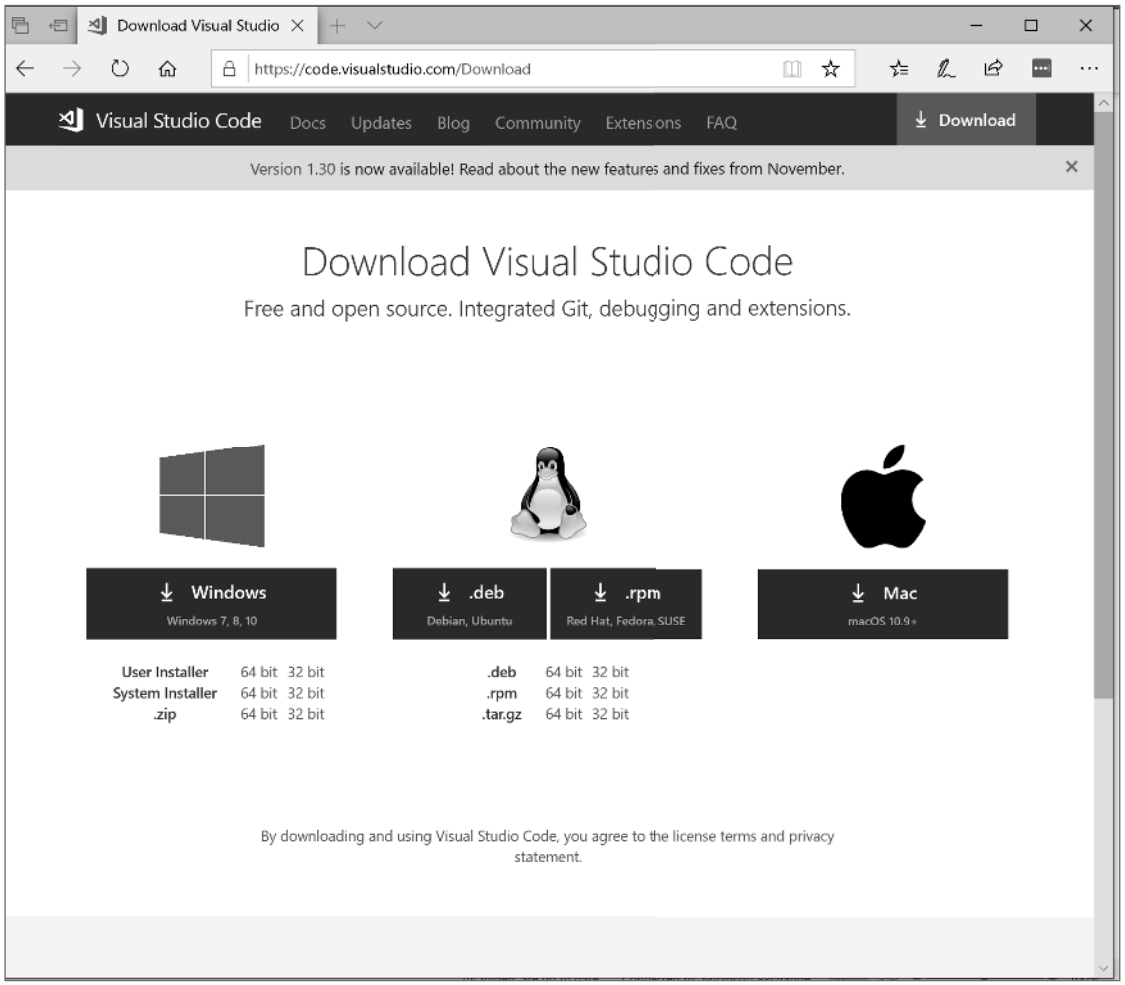

FIGURE 1-1

Windows

The most efficient starting point for a Windows installation is to download the desired installer. You have six possible options—three different installers each of which is available in 32-bit and 64-bit

formats, divided along two separate attributes. First, you can choose from three installer types—System, User, and Zip:

- *System Installer*—This was the original installer for Visual Studio Code. It requires local Administrator privileges and places the executable and supporting files in the Program Files directory structure.
- *User Installer*—A more recent addition to Visual Studio Code, this installer does not require Administrator permission to be successful. Instead of placing the files into the Program Files structure, you can find the files in `AppData\Local\Programs` in your user directory. Because this is actually the preferred installer, you will be asked if you want to uninstall any versions that had been installed using the System Installer.

  When Visual Studio Code is installed using one of these two installers, you will automatically be notified when an update is available. The pace of change for Visual Studio Code means that updates are delivered approximately once a month.
- *Zip*—The Zip option is just a ZIP compressed file containing the contents that are placed into `AppData\Local\Programs` by the User Installer. Once you have opened the file, you can copy the contents to whatever location you prefer. However, you're responsible for creating any links to the executable (called `code.exe`) that you want to place on your desktop or task bar. Also, you won't automatically receive updates. If you wish to use a more recent version, you'll need to download a new ZIP file and copy the files over again.

The second attribute available for the installer has a value of either 32-bit or 64-bit. This refers to the width of the data units supported by the CPU on your device. If you are running 32-bit Windows, choose the 32-bit version of the desired installer. If you are running 64-bit Windows, you can choose either the 32-bit or 64-bit version.

> **NOTE** *It will come as a disappointment to some, but this option does* not *mean that Visual Studio Code is a 64-bit application. It's not. Visual Studio Code is a 32-bit application regardless of whether or not it's running on a 64-bit version of Windows.*

For both the System and User Installers, running the installation program provides a similar experience. The following screens make up the installation process:

- *Welcome Screen*—Describes what you're about to do (that being, installing Visual Studio Code). The description indicates whether you are using the User or System Installer.
- *License Agreement*—The license agreement for Visual Studio Code is presented. You must accept the agreement before you can install the software.
- *Select Destination Location*—In this screen, shown in Figure 1-2, you choose the directory into which Visual Studio Code will be installed. The default is different for the User and System versions. For the User installation, it is placed in `C:\Users\<your username>\AppData\Local\Programs\Microsoft VS Code`. For the System installation, it is placed in `C:\Program Files (386)\Microsoft VS Code`. The (386) part of the directory is left off if you're running on a 32-bit operating system.

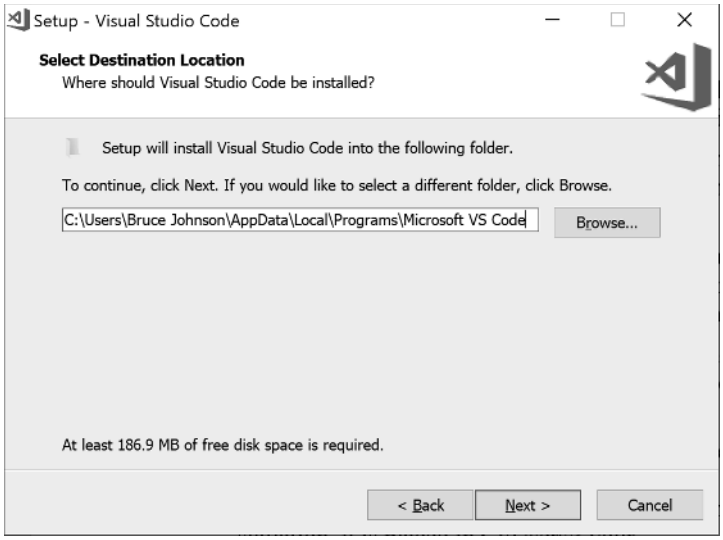

FIGURE 1-2

➤ *Select Start Menu Folder*—Once you have specified where Visual Studio Code will be installed, you get to identify where the links to the application are placed within your Start menu. The dialog to do this is shown in Figure 1-3. The default folder is Visual Studio Code, but you can provide another name, browse within your existing Start menu folders, or create a new one if you prefer. If you don't want to have any Start menu items added, check the Don't Create A Start Menu Folder checkbox at the bottom left of the dialog.

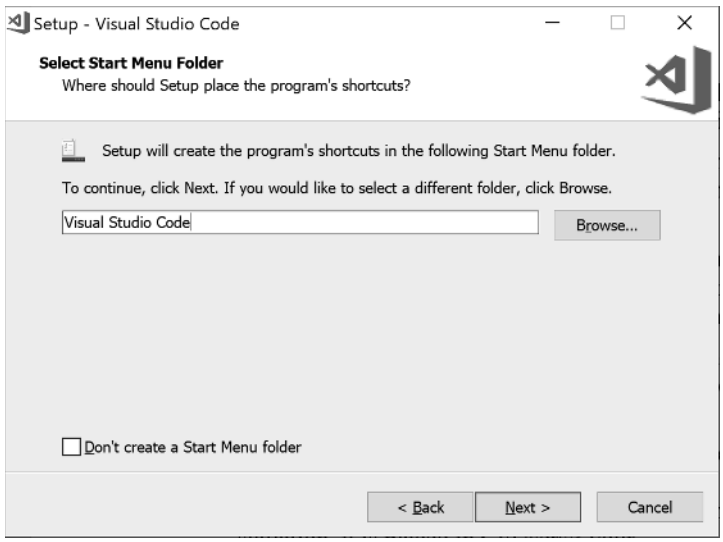

FIGURE 1-3

- *Select Additional Tasks*—A number of additional tasks can be performed as part of the installation process. The choices, available on the dialog shown in Figure 1-4, include:
  - Creating a desktop icon that launches Visual Studio Code.
  - Adding an Open With Code option to the context menu for Windows Explorer files and directories.
  - Registering Visual Studio Code as an editor for any supported file types. This causes Visual Studio Code to appear in the Open With list of options in the context menu in Windows Explorer.
  - Add the installation directory for Visual Studio Code to the `PATH` environment variable. This allows Visual Studio Code commands to be invoked from within a command-line tool. Keep in mind that it takes a restart of your computer for any changes to `PATH` to take effect.

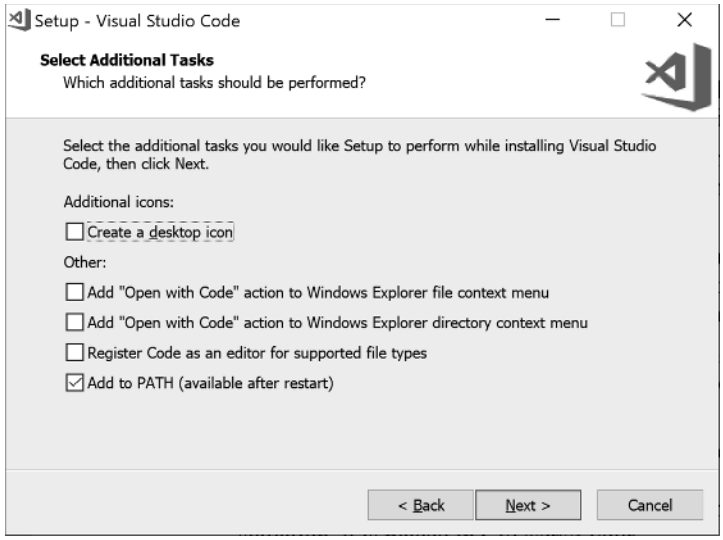

FIGURE 1-4

- *Ready To Install*—The final step in the installation process, as shown in Figure 1-5, is a summary of the options that you selected on the other screens. If you click Install, the installation commences.

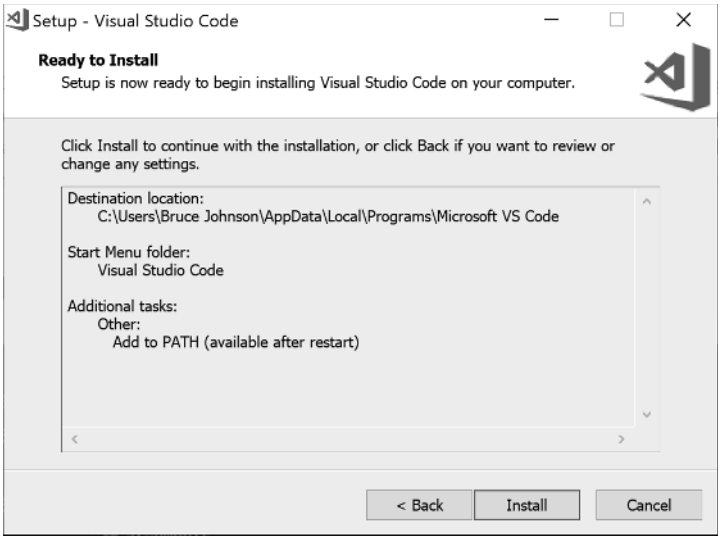

FIGURE 1-5

Linux

The basic steps involved with installing Visual Studio Code on Linux are the same regardless of the distribution you are using:

1. Download the appropriate software package.
2. Install it using the tools appropriate for your distribution.

However, the specifics within each of these steps do depend on your distribution. The precise instructions for a number of different distributions are described in the following sections.

All of the figures in this section were captured from within Ubuntu, so you might see something slightly different in your own environment. Also, be aware that you need to have a desktop installed in your Linux environment in order to run Visual Studio Code.

Ubuntu and Debian Distributions

The installation flow for Ubuntu or Debian is quite similar to Windows or macOS:

1. Open your favorite browser and navigate to `https://code.visualstudio.com/download`. Figure 1-6 shows what this web page looks like.
2. Identify the desired installation package and download it. 64-bit and 32-bit versions of Visual Studio Code are available, as well as a gzipped TAR file. Use the version suitable for your platform (that is, not the TAR file).

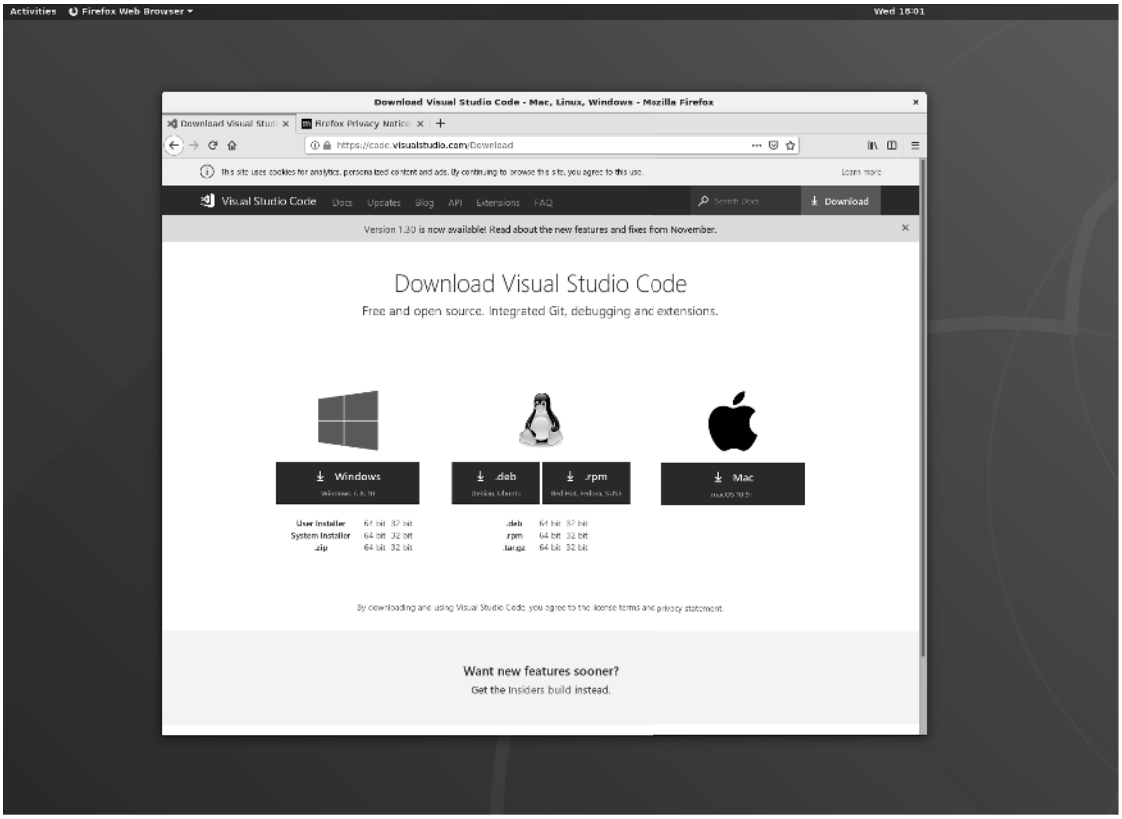

FIGURE 1-6

3. You will be prompted for what to do with the file. The dialog is shown in Figure 1-7. Though it might seem like opening with Software Install is the obvious choice, instead select the Save File radio button and click OK.

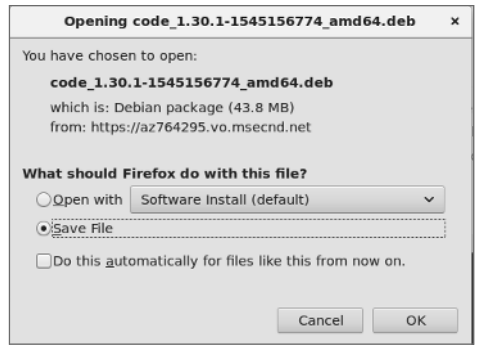

FIGURE 1-7

4. Once the file has been downloaded, open a Terminal window. Change the directory to the location where the file was downloaded. By default, the command to do this is:

```
cd ~/Downloads
```

5. Install the package using the `apt` utility:

```
sudo apt install ./<file>.deb
```

> **NOTE** *In some cases, it might be necessary to install some additional dependencies. If that happens, you will see the missing dependencies in the output from that last command. You can install the dependencies manually (using the same* `apt install` *command), or you can execute the following command to install all of the missing dependencies:*
>
> ```
> sudo apt-get install -f
> ```

At this point, Visual Studio Code should be ready to go. Execute **`code`** from within Terminal to launch the application (see Figure 1-8).

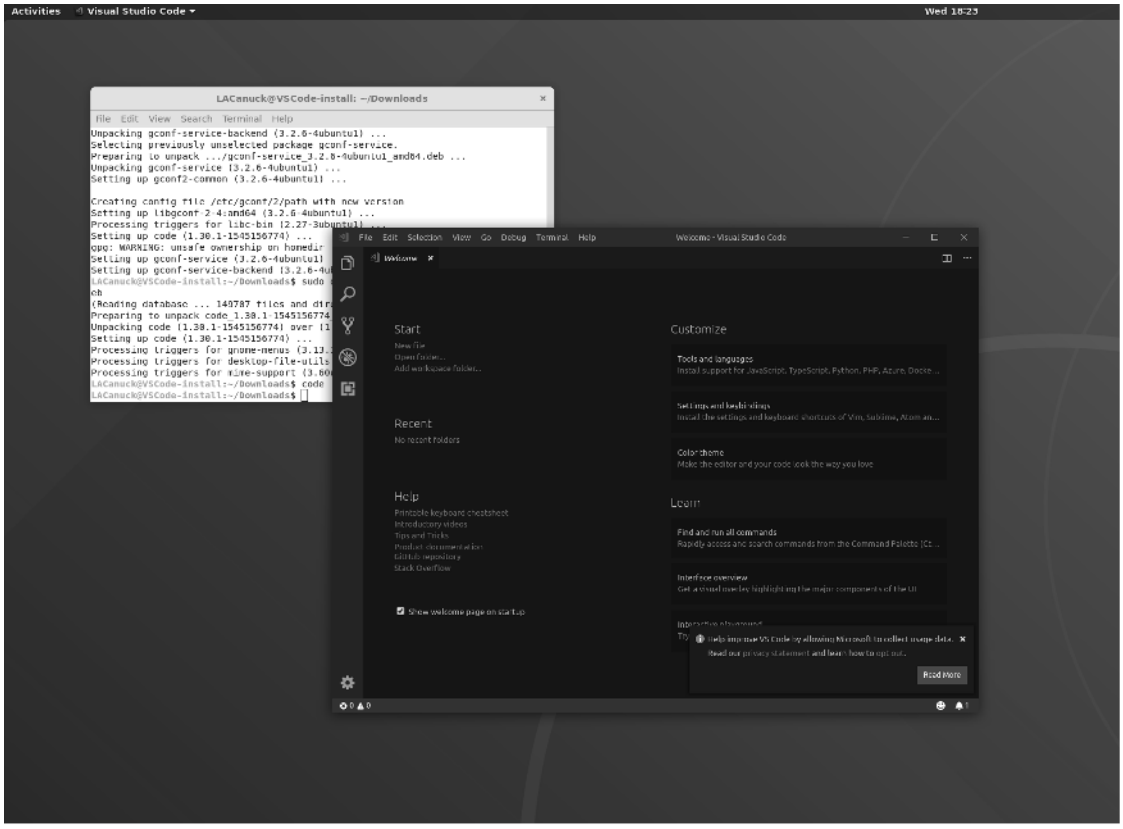

FIGURE 1-8

CentOS, Fedora, and RHEL Distributions

Another way to get the installation files for Visual Studio Code is through a YUM repository. Microsoft maintains a YUM repository that contains the current stable 64-bit version of Visual Studio Code.

> **NOTE** *A YUM (Yellowdog Updater, Modified) repository is a storehouse of RPM (Red Hat Package Manager) files. The purpose of the repository is to provide a simple mechanism through which software can be installed, dependencies resolved, and updates delivered. It's conceptually the same as NuGet or NPM (Node Package Manager).*

To install the key and the YUM repository, execute the following commands on your Linux machine:

```
sudo rpm --import https://packages.microsoft.com/keys/microsoft.asc
sudo sh -c 'echo -e "[code]\nname=Visual Studio
Code\nbaseurl=https://packages.microsoft.com/yumrepos/vscode\nenabled=1\
ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" >
/etc/yum.repos.d/vscode.repo'
```

The first command imports the key to the repository into your environment. The second adds the Microsoft repository as a recognized YUM repository. This is accomplished by creating a file (called `vscode.repo`) and putting it in the `/etc/yum.repos.d` directory.

Once this is done, you can update the YUM package cache by executing the following command:

```
yum check-update
```

Now you're ready to actually install Visual Studio Code. The `yum install` command initiates the installation:

```
sudo yum install code
```

When the command is finished, Visual Studio Code is ready to run.

> **NOTE** *Microsoft uses a manual signing process for the package placed in the YUM repository. As a result, the system that is used to publish Visual Studio Code is not able to deliver an updated package at the same time as it is delivered to other distribution points (like the Visual Studio Code website); there might be a delay between updates being available on other platforms and in the YUM repository.*

OpenSUSE and SLE Distributions

For these distributions, the YUM repository is the best choice for getting Visual Studio Code. What differs are the commands that need to be executed. The following commands install the key and repository:

```
sudo rpm --import https://packages.microsoft.com/keys/microsoft.asc
sudo sh -c 'echo -e "[code]\nname=Visual Studio
Code\nbaseurl=https://packages.microsoft.com/yumrepos/vscode\nenabled=1\ntype=r-
pm-md\ngpgcheck=1\ngpgkey=https://packages.microsoft.com/keys/microsoft.asc" >
/etc/zypp/repos.d/vscode.repo'
```

These next two commands will update the package cache and install Visual Studio Code:

```
sudo zypper refresh
sudo zypper install code
```

Nix Package Manager

Another package manager that is occasionally found in the Linux world is Nix. Although Microsoft doesn't maintain a Nix package, a community-managed version is available at `https://github.com/NixOS/nixpkgs/blob/master/pkgs/applications/editors/vscode/vscode.nix`.

To install Visual Studio Code using Nix, set the `allowUnfree` option in your `config.nix` file to true. Then execute the following command:

```
nix-env -i vscode
```

macOS

The starting point for installing Visual Studio Code on macOS is, like Windows, the Visual Studio Code download page (found at `https://code.visualstudio.com/Download` and shown on Safari in Figure 1-9).

FIGURE 1-9

Click the Mac option on the right-hand side to start the download. When it has finished, open the download section on your browser and open the downloaded file. You will be greeted with a warning message indicating that the file you're trying to open came from the Internet (Figure 1-10).

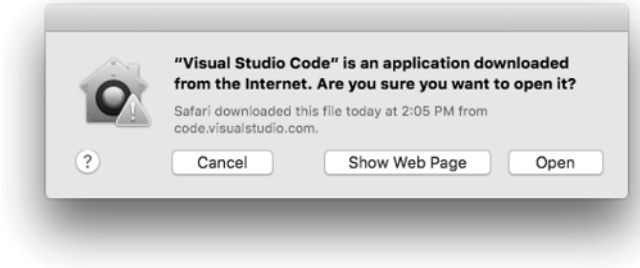

FIGURE 1-10

Click the Open button to expand the downloaded file. Because the download is a ZIP file, this means that the contents become visible. Drag the Visual Studio `Code.app` file to the `Applications` folder. This makes Visual Studio Code available in your Launchpad.

If you want to add Visual Studio Code to your dock, right-click the icon and choose Options ➢ Keep In Dock (Figure 1-11).

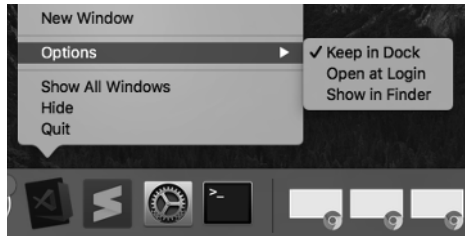

FIGURE 1-11

> **NOTE** *If you upgrade to macOS Mojave after installing Visual Studio Code, it's possible that you might start seeing dialogs suggesting that Visual Studio Code needs to access some common file locations. Specifically, you might see any or all of calendar, photos, and contacts in the dialog. These dialogs were a security addition to Mojave and, as such, are not specific to Visual Studio Code. Running other applications might generate the same dialogs. You are safe to answer Don't Allow to these requests. There is no need for Visual Studio Code to access these folders and there will be no impact on any functionality if you deny the request.*

How Cross-Platform Works

Although it is not essential to being able to use Visual Studio Code, it is instructive to understand the underpinnings that allow it to work across the disparate platforms that it supports. That is especially true if you have any thoughts about trying to create extensions for Visual Studio Code, a topic that is covered in Chapter 10, "Creating Your Own Extensions."

First, Visual Studio Code is an open-source project. This means that the source code is readily available for anyone to look at. Or even contribute to, if you have the desire. You can find the source at `https://github.com/Microsoft/vscode`.

Visual Studio Code is based on a framework called Electron. Electron was designed to allow for the creation of desktop applications using the front- and backend components of a web application. On the frontend side, JavaScript, HTML, and CSS are used to create the user interface functionality. The interface is rendered using Chromium.

> **NOTE** *Chromium is an open-source web browser project that is headed by Google. It is the foundation on which Chrome is built and will also be the basis for future versions of Microsoft's Edge.*

On the backend, Node.js is used to serve the user interface component and provide any additional functionality. This combination of technologies allows it to run on the desktop of three major platforms: Windows, macOS, and Linux.

Electron was originally created by GitHub and, ostensibly, its primary purpose was to support the Atom text editor. But its usage didn't stop there. It is currently used in a wide variety of applications, ranging from integrated development environments (IDEs) to music players to games.

Though Electron is the foundation on which Atom was built, Visual Studio Code does not use Atom as its editor. Instead, it uses the same text editor that is found in Azure DevOps (formerly named Visual Studio Team Services, which was formerly Visual Studio Online). The text editor is called Monaco. It is also open-source and available separately from Visual Studio Code (you can find the repository at `https://microsoft.github.io/monaco-editor`).

Additional common functionality is implemented through a number of other channels. For example, colorization is provided through Monarch, a library that allows you to specify a syntax highlighter using a declarative syntax. What's even better is that the syntax is expressed using JSON, something that is likely to be familiar to most web developers.

As another example, language-specific functionality, like auto-completion, formatting, and diagnostics, are implemented in a separate language server. This server runs in a process separate from Visual Studio Code, the goal being to minimize any impact on the IDE that performing continual lexographical analysis could have. Visual Studio Code communicates with these servers (there is typically one for each language you are editing) using the Language Server Protocol (LSP).

SUMMARY

You've seen how to install Visual Studio Code on the different operating systems that it supports. Once the application is ready, the user experience is very consistent across the platforms. That has to do with how Visual Studio Code was built (that is, on the Electron platform).

In subsequent chapters, you'll learn how to customize the IDE to more closely fit your own working style. You'll also see how Visual Studio Code works hard to make your development process easier and faster.