Prerequisites

FOLLOW ALONG THE TUTORIAL

This is **part 1** of the Electron tutorial.

Electron is a framework for building desktop applications using JavaScript, HTML, and CSS. By embedding **Chromium** and **Node.js** into a single binary file, Electron allows you to create cross-platform apps that work on Windows, macOS, and Linux with a single JavaScript codebase.

This tutorial will guide you through the process of developing a desktop application with Electron and distributing it to end users.

Goals

This tutorial starts by guiding you through the process of piecing together a minimal Electron application from scratch, then teaches you how to package and distribute it to users using Electron Forge.

If you prefer to get a project started with a single-command boilerplate, we recommend you start with Electron Forge's `create-electron-app` command.

Assumptions

Electron is a native wrapper layer for web apps and is run in a Node.js environment. Therefore, this tutorial assumes you are generally familiar with Node and front-end web development basics. If you need to do some background reading before continuing, we recommend the following resources:

- **Getting started with the Web (MDN Web Docs)**
- **Introduction to Node.js**

Required tools

Code editor

You will need a text editor to write your code. We recommend using **Visual Studio Code**, although you can choose whichever one you prefer.

Command line

Throughout the tutorial, we will ask you to use various command-line interfaces (CLIs). You can type these commands into your system's default terminal:

- Windows: Command Prompt or PowerShell
- macOS: Terminal
- Linux: varies depending on distribution (e.g. GNOME Terminal, Konsole)

Most code editors also come with an integrated terminal, which you can also use.

Git and GitHub

Git is a commonly-used version control system for source code, and GitHub is a collaborative development platform built on top of it. Although neither is strictly necessary to building an Electron application, we will use GitHub releases to set up automatic updates later on in the tutorial. Therefore, we'll require you to:

- **Create a GitHub account**
- **Install Git**

If you're unfamiliar with how Git works, we recommend reading GitHub's **Git guides**. You can also use the **GitHub Desktop** app if you prefer using a visual interface over the command line.

We recommend that you create a local Git repository and publish it to GitHub before starting the tutorial, and commit your code after every step.

> **INSTALLING GIT VIA GITHUB DESKTOP**
>
> GitHub Desktop will install the latest version of Git on your system if you don't already have it installed.

Node.js and npm

To begin developing an Electron app, you need to install the **Node.js** runtime and its bundled npm package manager onto your system. We recommend that you use the latest long-term support (LTS) version.

> **TIP**
>
> Please install Node.js using pre-built installers for your platform. You may encounter incompatibility issues with different development tools otherwise. If you are using macOS, we recommend using a package manager like **Homebrew** or **nvm** to avoid any directory permission issues.

To check that Node.js was installed correctly, you can use the `-v` flag when running the `node` and `npm` commands. These should print out the installed versions.

```
$ node -v
v16.14.2
$ npm -v
8.7.0
```

> **CAUTION**

Although you need Node.js installed locally to scaffold an Electron project, Electron **does not use your system's Node.js installation to run its code**. Instead, it comes bundled with its own Node.js runtime. This means that your end users do not need to install Node.js themselves as a prerequisite to running your app.

To check which version of Node.js is running in your app, you can access the global `process.versions` variable in the main process or preload script. You can also reference **https://releases.electronjs.org/releases.json**.

Edit this page