

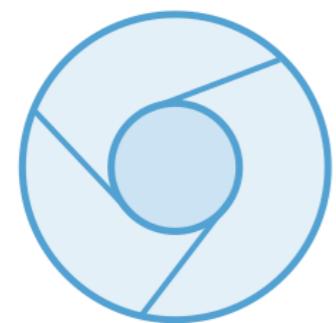
ELECTRON

@codebytere

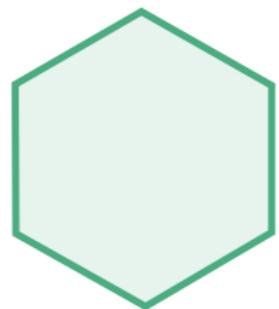


What is Electron?

Electron is a [library](#) you can use to build desktop applications with [JavaScript](#), [HTML](#) and [CSS](#). These applications can be packaged to run on Mac, Windows and Linux computers as well as be placed in the Mac and Windows app stores.



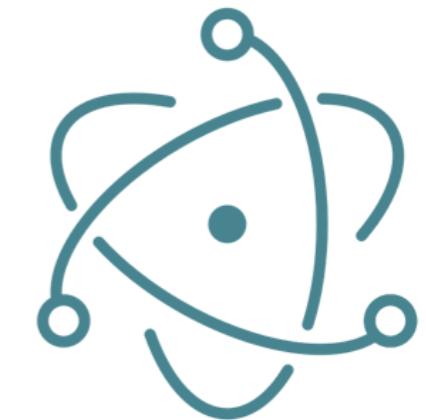
+



+



=



Chromium
for making
web pages

Node.js
for filesystems
and networks

Native APIs
for three
systems

ELECTRON



Main Process



You get:

- Node.js APIs
- Electron **main process modules**

Common tasks:

- Create Renderer Processes
- Call native elements
- Start and quit app

Renderer Process



index.html

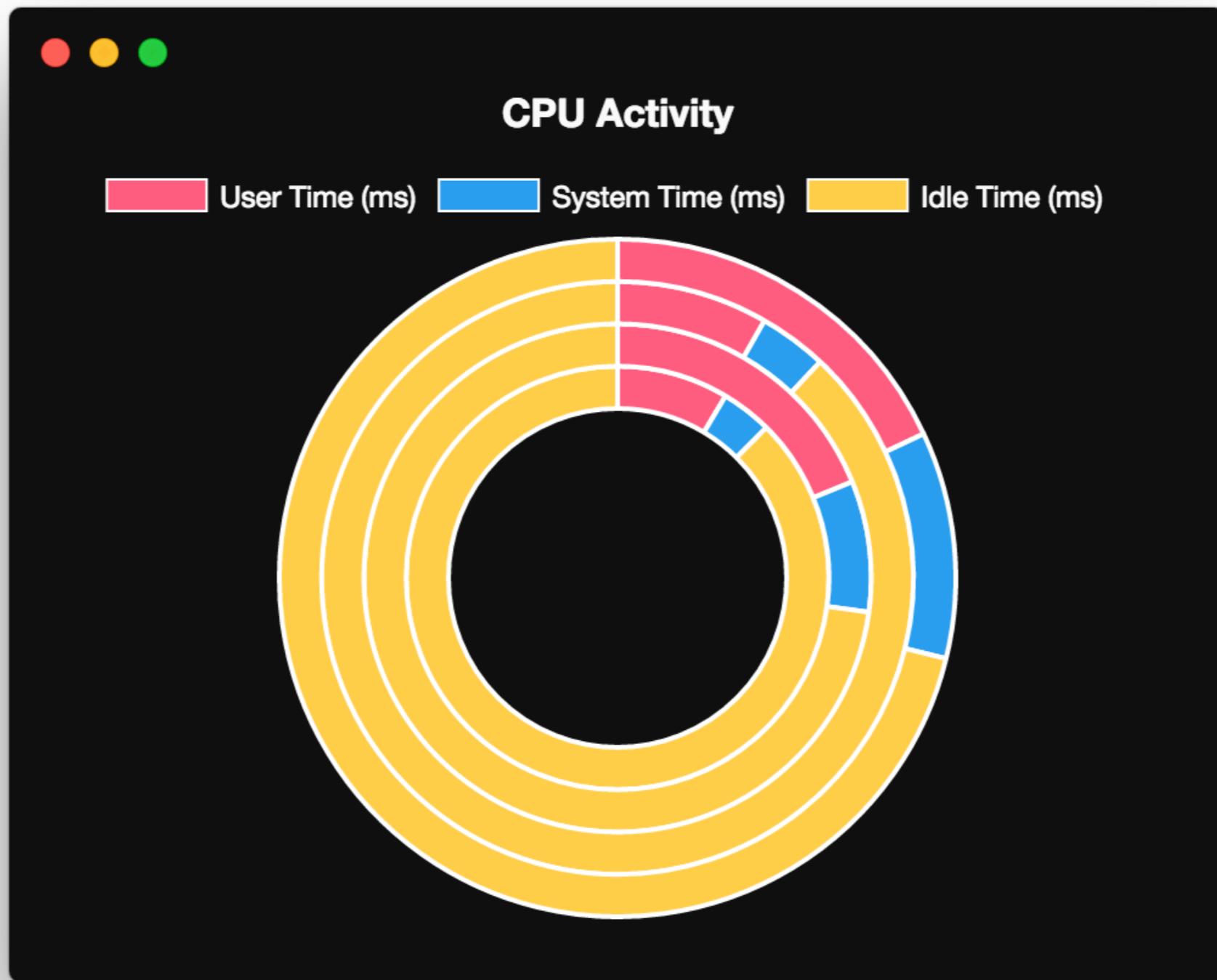
You get:

- Node.js APIs
- DOM APIs
- Electron **renderer process modules**

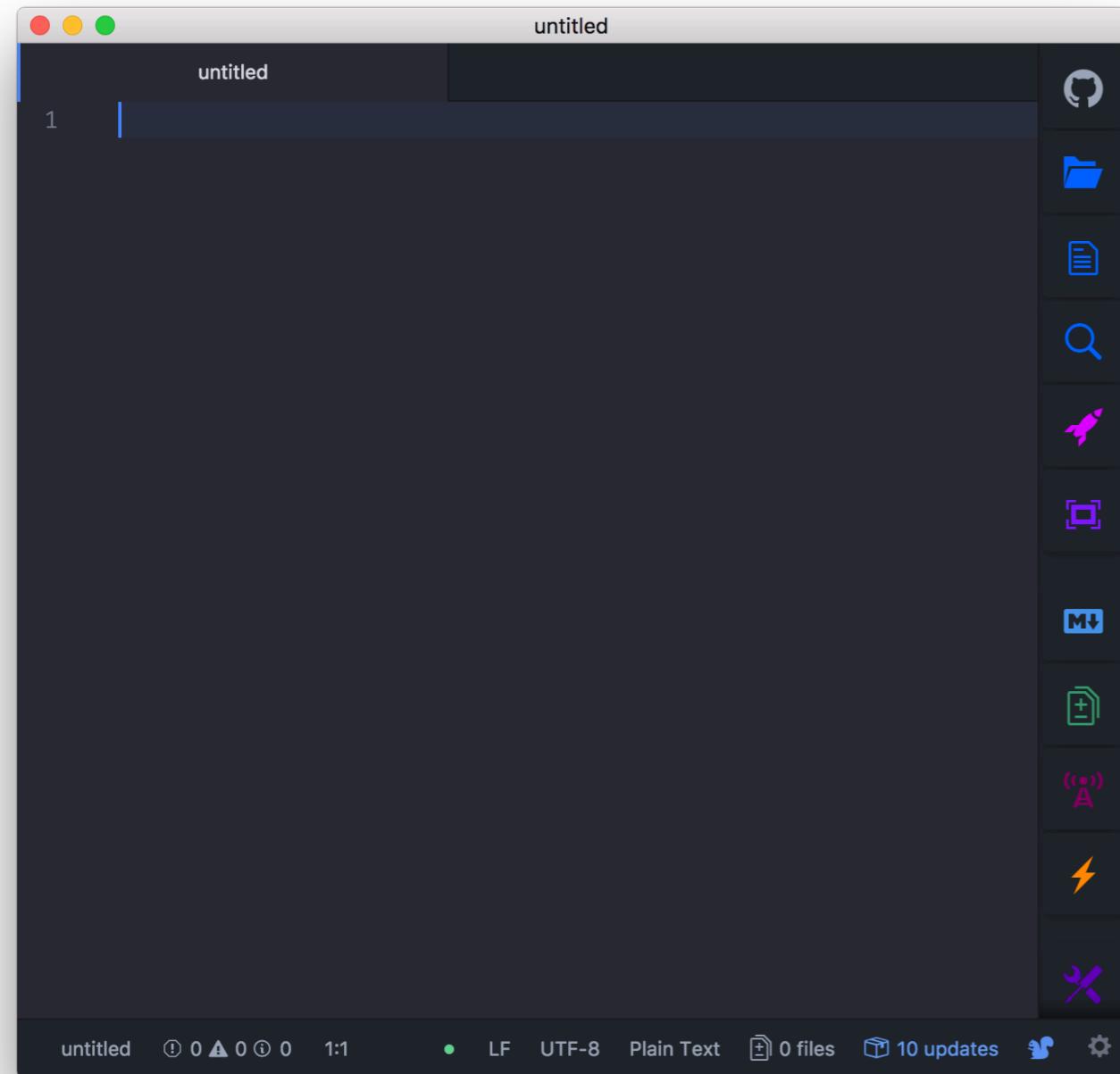
Common tasks:

- Design your page with HTML & CSS
- JavaScript page interactions

Activity Monitor



Prerequisites



```
codebytere@codebytere: ~ 1:07PM
```

Creating the Directory



codebytere@codebytere: ~/activity-monitor

```
codebytere ~ > mkdir activity-monitor && cd activity-monitor
codebytere activity-monitor > ls
codebytere activity-monitor > /* nothing here yet */
```

1:10PM

1:10PM

1:10PM

Creating package.json

```
● ● ●          npm init  
  
codebytere activity-monitor > npm init      1:12PM  
This utility will walk you through creating a package.json file.  
It only covers the most common items, and tries to guess sensible defaults.  
  
See `npm help json` for definitive documentation on these fields  
and exactly what they do.  
  
Use `npm install <pkg>` afterwards to install a package and  
save it as a dependency in the package.json file.  
  
Press ^C at any time to quit.  
package name: (activity-monitor)  
version: (1.0.0) 0.1.0  
description: app that shows a doughnut chart of the CPU system, user, and idle activity time.  
entry point: (index.js)  
test command:  
git repository:  
keywords:  
author: Shelley Vohr  
license: (ISC) MIT
```

Installing Electron

```
● ● ● codebytere@codebytere: ~/activity-monitor  
codebytere activity-monitor > npm i electron --save 1:16PM  
  
> electron@1.7.9 postinstall /Users/codebytere/activity-monitor/node_modules/electron  
> node install.js  
  
npm notice created a lockfile as package-lock.json. You should commit this file.  
npm WARN activity-monitor@0.1.0 No repository field.  
  
+ electron@1.7.9  
added 155 packages in 8.96s  
codebytere activity-monitor > | 1:17PM
```

Adding Files

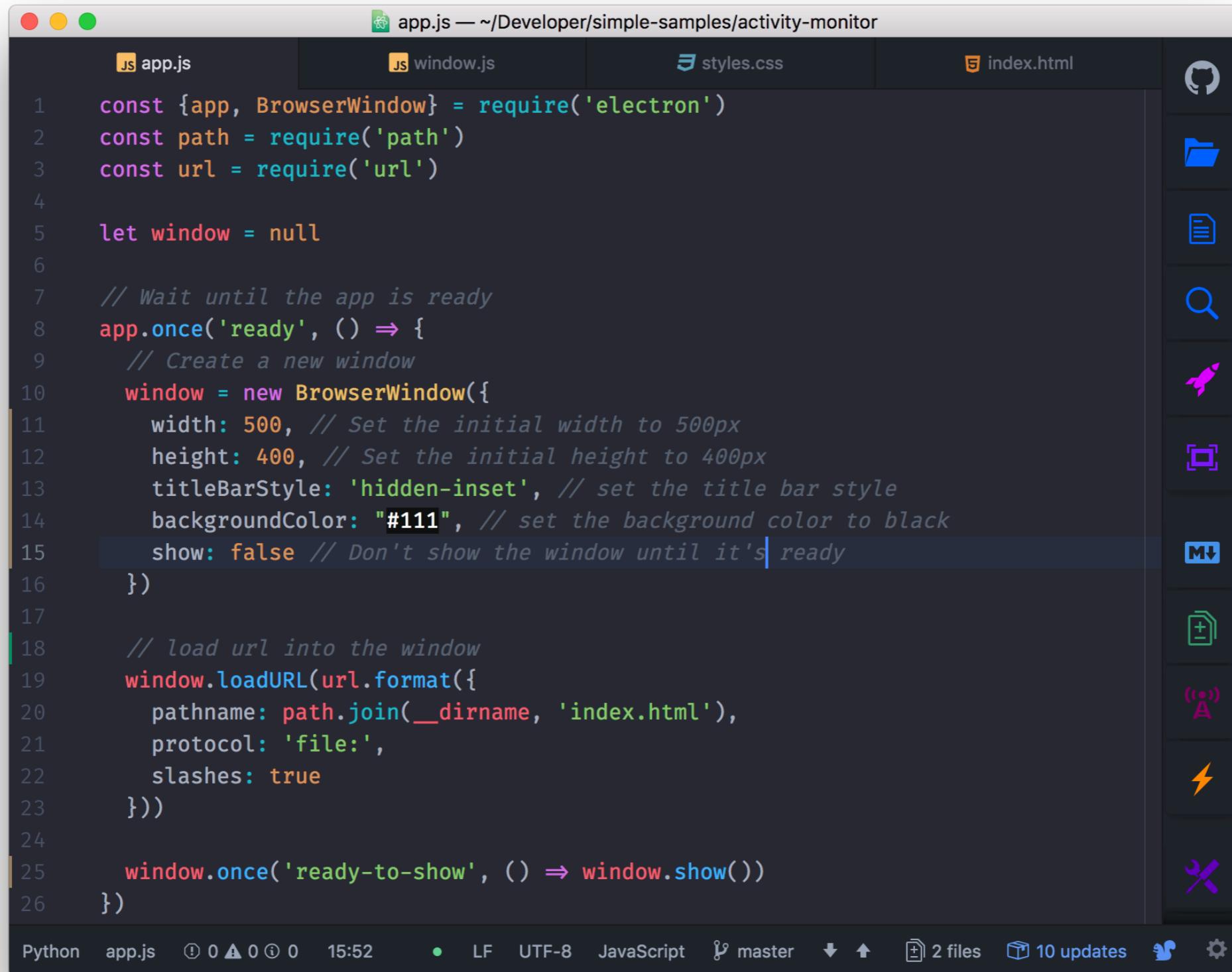


codebytere@codebytere: ~/activity-monitor

```
codebytere activity-monitor > touch index.js window.js style.css index.html
codebytere activity-monitor > ls                                         1:19PM
index.html          node_modules      package.json      window.js
index.js            package-lock.json  style.css
codebytere activity-monitor >                                              1:19PM
```

Adding The Code

Adding app.js

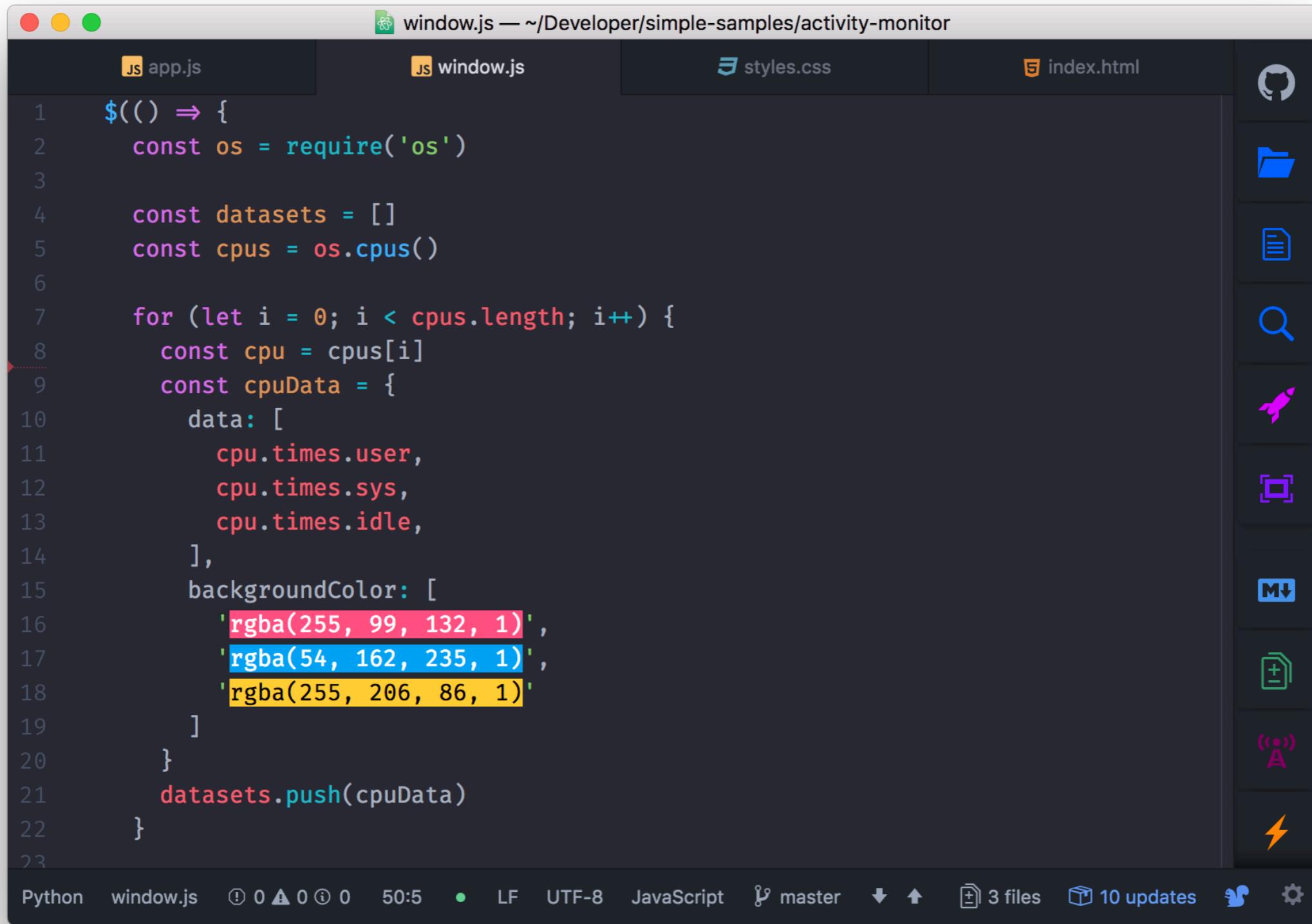


The screenshot shows a code editor window titled "app.js — ~/Developer/simple-samples/activity-monitor". The editor displays the following JavaScript code:

```
1 const {app, BrowserWindow} = require('electron')
2 const path = require('path')
3 const url = require('url')
4
5 let window = null
6
7 // Wait until the app is ready
8 app.once('ready', () => {
9     // Create a new window
10    window = new BrowserWindow({
11        width: 500, // Set the initial width to 500px
12        height: 400, // Set the initial height to 400px
13        titleBarStyle: 'hidden-inset', // set the title bar style
14        backgroundColor: "#111", // set the background color to black
15        show: false // Don't show the window until it's ready
16    })
17
18    // load url into the window
19    window.loadURL(url.format({
20        pathname: path.join(__dirname, 'index.html'),
21        protocol: 'file:',
22        slashes: true
23    }))
24
25    window.once('ready-to-show', () => window.show())
26})
```

The code uses Electron's API to create a new window and load an HTML file. The window is initially hidden (show: false) and will be shown once it's ready.

Adding window.js



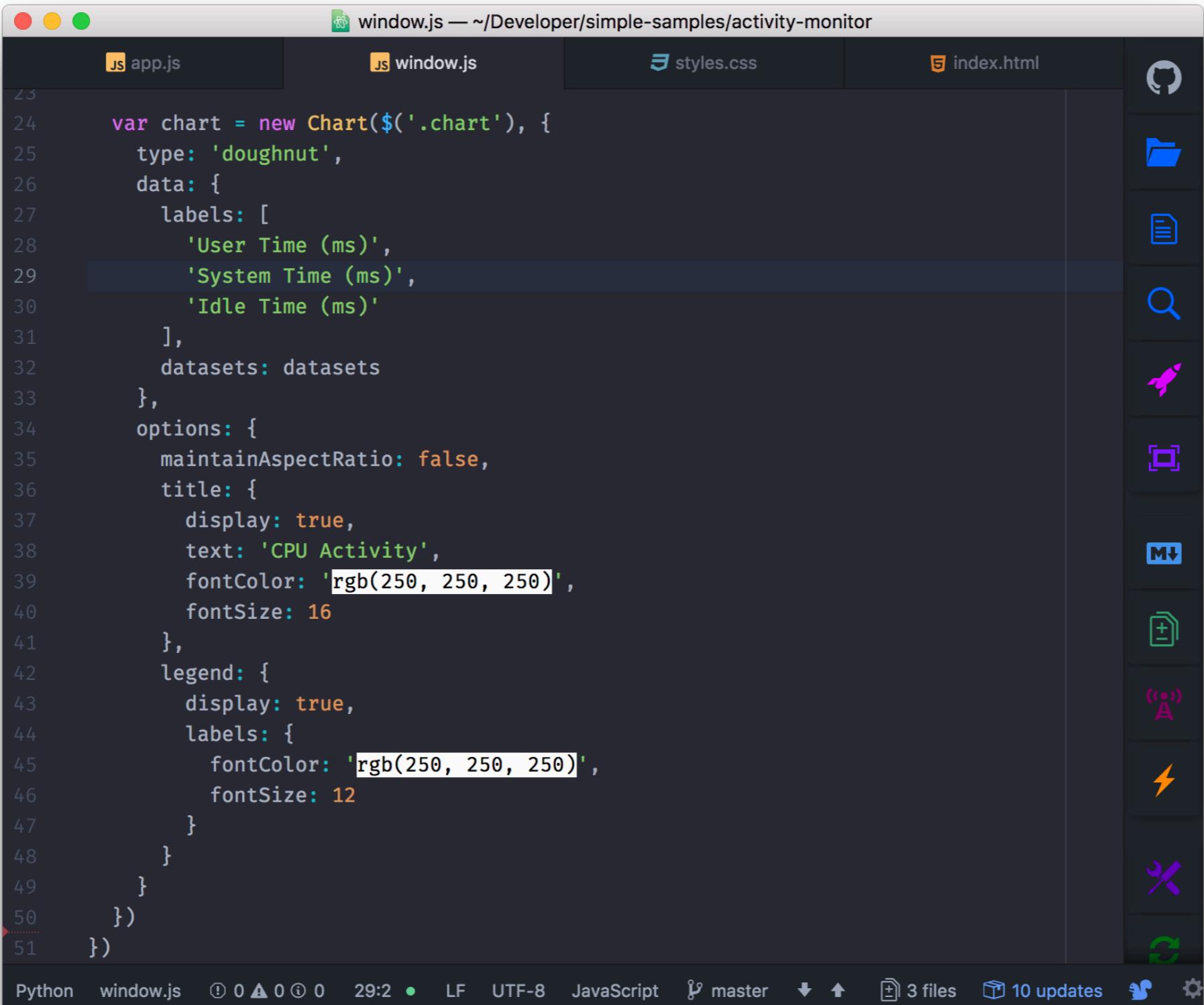
```
1 $(() => {
2     const os = require('os')
3
4     const datasets = []
5     const cpus = os.cpus()
6
7     for (let i = 0; i < cpus.length; i++) {
8         const cpu = cpus[i]
9         const cpuData = {
10             data: [
11                 cpu.times.user,
12                 cpu.times.sys,
13                 cpu.times.idle,
14             ],
15             backgroundColor: [
16                 'rgba(255, 99, 132, 1)',
17                 'rgba(54, 162, 235, 1)',
18                 'rgba(255, 206, 86, 1)'
19             ]
20         }
21         datasets.push(cpuData)
22     }
23 }
```

The screenshot shows a code editor window titled "window.js — ~/Developer/simple-samples/activity-monitor". The "window.js" tab is active. The code in the editor is as follows:

```
1 $(() => {
2     const os = require('os')
3
4     const datasets = []
5     const cpus = os.cpus()
6
7     for (let i = 0; i < cpus.length; i++) {
8         const cpu = cpus[i]
9         const cpuData = {
10             data: [
11                 cpu.times.user,
12                 cpu.times.sys,
13                 cpu.times.idle,
14             ],
15             backgroundColor: [
16                 'rgba(255, 99, 132, 1)',
17                 'rgba(54, 162, 235, 1)',
18                 'rgba(255, 206, 86, 1)'
19             ]
20         }
21         datasets.push(cpuData)
22     }
23 }
```

The code uses the 'os' module to get CPU data and 'd3' to handle the data. It creates three datasets for each CPU, each with a different background color.

Adding The Chart



```
23
24     var chart = new Chart($('.chart'), {
25         type: 'doughnut',
26         data: {
27             labels: [
28                 'User Time (ms)',
29                 'System Time (ms)',
30                 'Idle Time (ms)'
31             ],
32             datasets: datasets
33         },
34         options: {
35             maintainAspectRatio: false,
36             title: {
37                 display: true,
38                 text: 'CPU Activity',
39                 fontColor: 'rgb(250, 250, 250)',
40                 fontSize: 16
41             },
42             legend: {
43                 display: true,
44                 labels: {
45                     fontColor: 'rgb(250, 250, 250)',
46                     fontSize: 12
47                 }
48             }
49         }
50     })
51 }
```

Adding Styles

```
1 html, body, .container-fluid {  
2     height: 100%;  
3     background-color: #111;  
4 }  
5  
6 html {  
7     -webkit-app-region: drag;  
8 }  
9  
10 .container-fluid {  
11     padding: 25px;  
12 }  
13
```

Adding index.html

The screenshot shows a code editor window titled "index.html — ~/Developer/simple-samples/activity-monitor". The editor displays the contents of the "index.html" file, which is a basic HTML document structure. The code includes meta tags for charset and title, links to external stylesheets and scripts, and a canvas element for a chart. The editor interface features tabs for other files like "app.js", "window.js", and "styles.css", and a sidebar with various icons for file operations. The bottom status bar shows file statistics and a terminal-like interface.

```
<!DOCTYPE html>
<html>

<head>
  <meta charset="utf-8">
  <title>Activity Monitor</title>
  <!-- Stylesheets -->
  <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
  <link rel="stylesheet" href=".//styles.css">
  <!-- Scripts -->
  <script> delete module.exports </script>
  <script src="https://code.jquery.com/jquery-3.2.1.js"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.1.4/Chart.bundle.min.js"></script>
  <script src=".//window.js"></script>
</head>

<body>
  <div class="container-fluid">
    <canvas class="chart"></canvas>
  </div>
</body>

</html>
```

Python index.html ⚡ 0 ▲ 0 ⚡ 0 18:7 • LF UTF-8 HTML ⚡ master ⚡ 4 files 10 updates ⚡ ⚡

Thank You!

