

ELECTRON WORKSHOP

DAVID NEAL



@REVERENTGEEK

Up Ahead

- Intro to Electron
- Create an App
- Live Reload
- Debug
- IPC
- Package
- Tests



Set up and initialize your project

<https://bit.ly/electron-workshop>

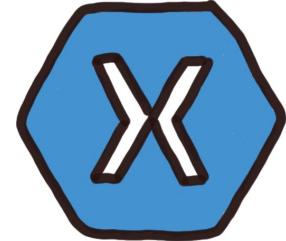
- or -

<https://github.com/reverentgeek/electron-workshop/wiki>



After completing Exercise 1, Part 1

.NET + Mono + Xamarin

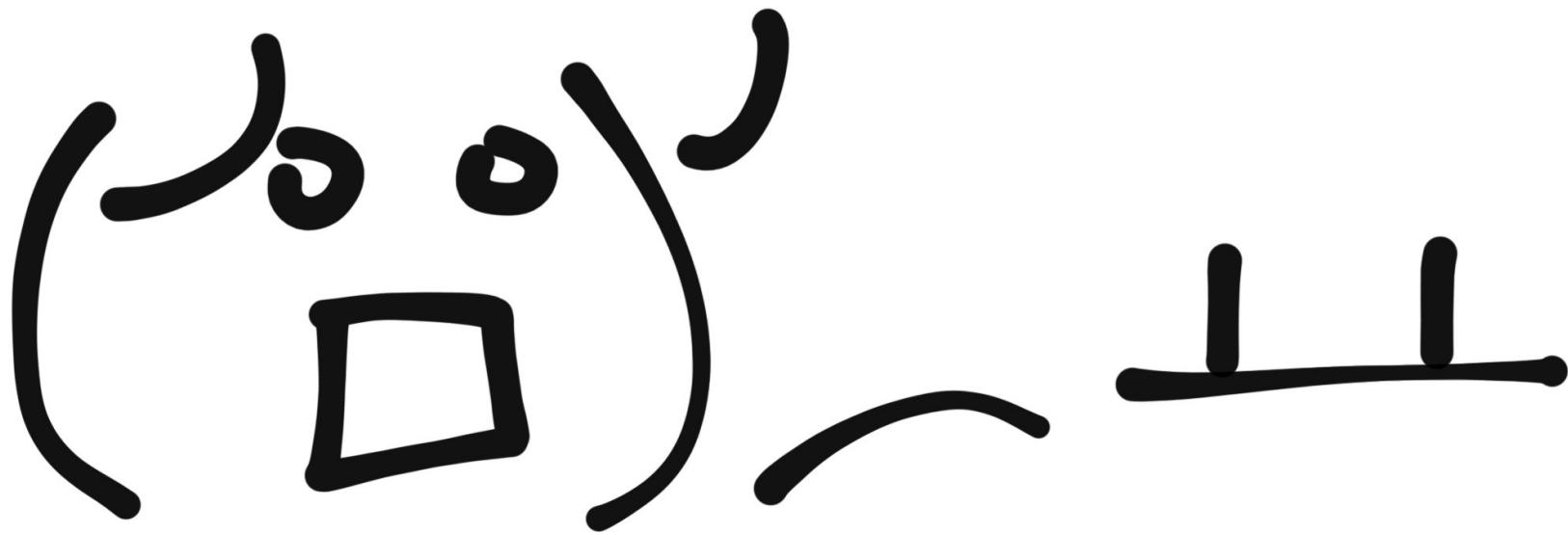


Pros:

- Shared .NET code base

Cons:

- Xamarin Mac != Xamarin iOS/Android
- Native UI is hard
- Deployment
- ~~Licensing~~

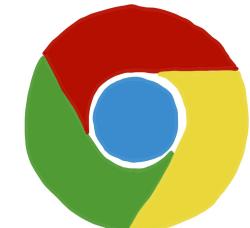
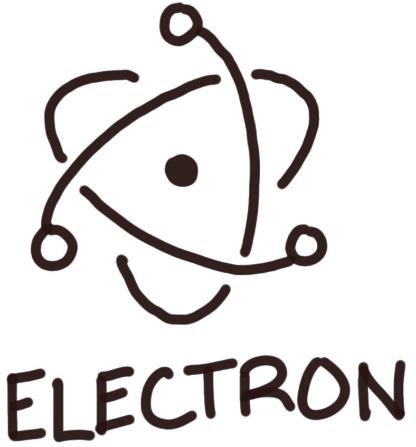
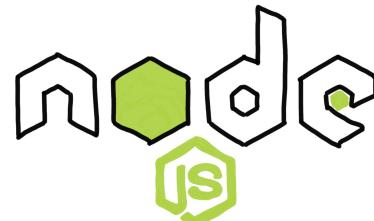
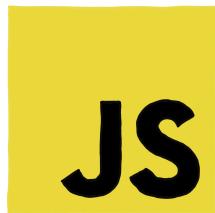


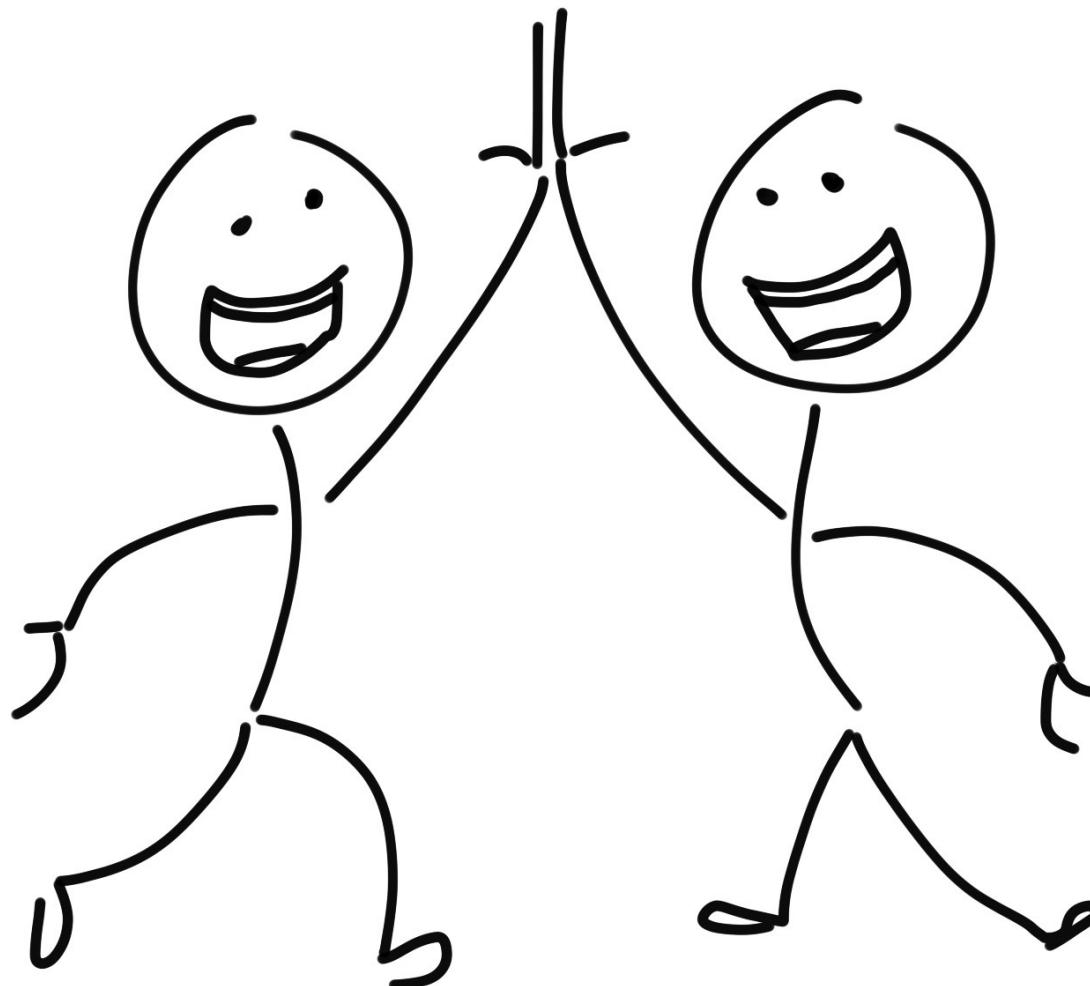
DAVID NEAL @REVERENTGEEK

Electron

Pros:

- HTML + CSS + JavaScript
- Node.js + Chromium
- No deployment dependencies





DAVID NEAL @REVERENTGEEK

Not Your Dad's



JAVASCRIPT

DAVID NEAL @REVERENTGEEK

Electron

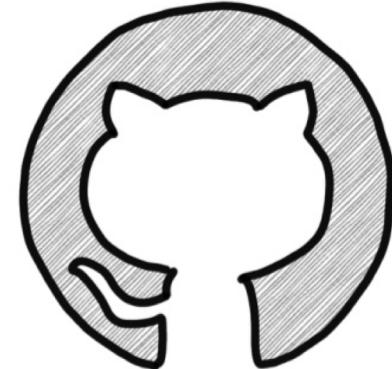
Cons:

- HTML + CSS + JavaScript



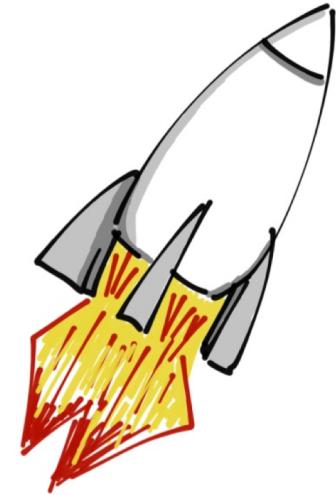
History

- Created by GitHub for Atom
- Formerly Atom Shell
- Active since January 2013
- 1.0 release May 2016
- 2.0 release May 2018



Electron Features

- Rapid development
- Themes
- Shared code/UI
- Deployment + “silent” updates
- Native UX



Why Desktop Apps?

- Offline
- Printers, devices, other local hardware
- On-premises
- Internal, LOB
- Edit local files
- App Store
- Kiosk
- Desktop > Intranet
- Sometimes it “just feels right”



Desktop App Ideas

- Disconnected data entry
 - Editor
 - Time management
 - Media player
 - Email client
 - Messaging, collaboration
 - Kiosk
 - Mapping, route planner
 - Social media client
 - Calendar
- Bulk media editor
 - File management, backup
 - Document generation, reading
 - Audio/video conferencing
 - Games



Atom

The screenshot shows the Atom code editor interface. The left sidebar displays a project structure for a folder named 'upsync' containing files like .git, specs, .eslintrc.js, .gitignore, index.js, LICENSE, package.json, README.md, and upsync.png. The right pane shows the content of 'index.js'. The code is written in JavaScript and defines a function to wrap callback-style functions into promises.

```
1 const AsyncFunction = Object.getPrototypeOf( async function() {} ).construct
2
3 const nameFunction = ( originalName, func ) => {
4     const name = originalName.length === 0 ? "Anonymous" : originalName.charAt(0).toUpperCase() + originalName.slice(1);
5     const fname = `async${ name }`;
6     return { async [ fname ]( ...args ) {
7         return func( ...args );
8     } }[ fname ];
9 }
10
11 module.exports = callBackStyleFunction => {
12     if ( !callBackStyleFunction ) {
13         throw new Error( "Must be a callback-style function" );
14     }
15
16     if ( AsyncFunction.prototype.isPrototypeOf( callBackStyleFunction ) ) {
17         return callBackStyleFunction;
18     }
19
20     if ( typeof callBackStyleFunction !== "function" ) {
```

index.js 1:1

Project — ~/Projects/upsync

LF UTF-8 JavaScript master 0 files

Slack

Slack - LeanKit

LeanKit ▾ David Neal

All Unreads All Threads Starred

all-hands-play
all-hands-work
analytics-squad
company-announceme...
franklin-visits

gifs

pd
pd-analytics
pd-deadpool
random
speakerships

Channels Direct Messages Apps

#gifs

★ | 8 20 | 0 | http://media.giphy.com/media/1

Tuesday, September 12th



John Campbell 4:16 PM
[http://i.imgur.com/b8EXeAE.gif \(2MB\)](http://i.imgur.com/b8EXeAE.gif) ▾



Visual Studio Code

The screenshot shows the Visual Studio Code interface with the following details:

- File Explorer (Left):** Shows the project structure. Opened files include `reportHandler.spec.js` and `consoleLogger.spec.js`. Other files visible in the `spec/plugins` folder are `apiVersion.spec.js`, `azureStorage.spec.js`, `cache.spec.js`, `consoleLogger.spec.js`, `csvExport.spec.js`, and `database.spec.js`.
- Editor (Right):** The current file is `consoleLogger.spec.js`. The code is as follows:

```
1  const code = require( "code" );
2  const Lab = require( "lab" );
3  const lab = exports.lab = Lab.script();
4  const describe = lab.describe;
5  const expect = code.expect;
6  const it = lab.it;
7
8  const ConsoleLogger = require( "../../app/plugins/consoleLogger" );
9
10 const Stream = require( "stream" );
11
12 class Writer extends Stream.Writable {
13   constructor() {
14     super( { objectMode: true } );
15     this.data = [];
16   }
17   _write( chunk, enc, callback ) {
18     this.data.push( chunk );
19     callback( null );
20   }
21 }
22
23 class Reader extends Stream.Readable {
24   constructor() {
```

The status bar at the bottom indicates the file is `feature-exceptions-export*`, has 7 lines, 11 characters, 0 errors, and 0 warnings. The tab size is 4, encoding is UTF-8, and the language is JavaScript. ESLint is enabled.

Postman

The screenshot shows the Postman application interface. At the top, there's a dark header bar with the word "Postman" in white. Below it, a navigation bar has "Builder" highlighted in orange, and "Team Library" is also visible. On the far right of the header, there are icons for a user profile ("David"), notifications, and other settings.

The main workspace is titled "Get Board". On the left, a sidebar titled "Collections" lists several collections: "LeanKit API" (54 requests), "Bandit Software", "Board API", and "Get Board". The "Get Board" collection is currently selected and highlighted in orange. It contains five API requests: "Get Boards", "Get Board", "Get Board Identifiers", "Get Board Archive", "Get Newer if Exists", "Check for Updates", "Get Board History Since", and "Get Board Backlog". The "Get Board" request is specifically highlighted with an orange background.

The central area displays the "Get Board" request details. The method is set to "GET" and the URL is "https://reverentgeek.leankit.com/kanban/api/boards/256064019". The "Headers" tab is active, showing three headers: "Authorization", "Accept", and "Content-Type", each with their respective values. The "Body" tab is also present below the headers.

At the bottom of the interface, the response status is shown as "Status: 200 OK Time: 327 ms". The response body is displayed in a JSON editor, showing the following structure:

```
1 {  
2   "ReplyData": [  
3     {  
4       "id": 1,  
5       "name": "Project A",  
6       "status": "In Progress",  
7       "last_update": "2023-10-01T12:00:00Z"  
8     }  
9   ]  
10 }  
11 }
```

On the far right of the bottom bar, there are buttons for "Save Response" and a "Save" button with a dropdown arrow.

DAVID NEAL @REVERENTGEEK

GitHub Desktop

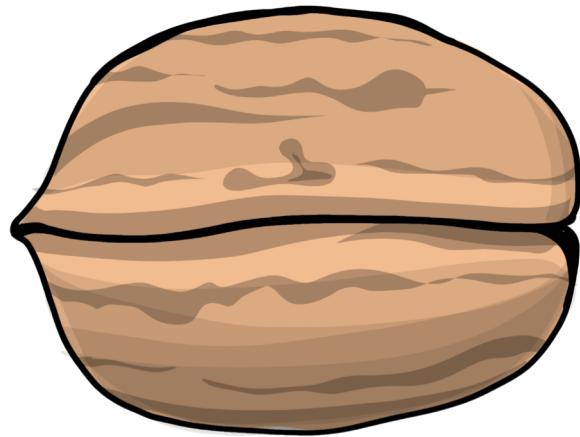
The screenshot shows the GitHub Desktop application interface. At the top, there are three tabs: "Changes" (selected), "History", and a third tab that is mostly obscured. Below these tabs, the repository name "desktop" and the current branch "progress-reporting" are displayed. To the right of the branch, there is a "Publish branch" button with the sub-instruction "Publish this branch to GitHub".

The main area displays a code diff for the file "app/src/ui/app.tsx". The left column shows line numbers and the right column shows the corresponding code. The code diff highlights changes made to the file:

```
@@ -956,6 +956,8 @@ export class App extends React.Component<IAppProps, IAppState> {
 956     956
 957     957     const state = selection.state
 958     958     const remoteName = state.remote ? state.remote.name : null
 959 +    959     +     const progress = state.pushProgress || state.pullProgress
 960 +    960
 959     961         return <PushPullButton
 960     962             dispatcher={this.props.dispatcher}
 961     963                 repository={selection.repository}
@@ -963,7 +965,7 @@ export class App extends React.Component<IAppProps, IAppState> {
 963     965                     remoteName={remoteName}
 964     966                     lastFetched={state.lastFetched}
 965     967                     networkActionInProgress={state.pushPullInProgress}
 966 -    966             -     progress={state.pushProgress}
 968 +    968             +     progress={progress}
 967     969         />
 968     970     }
 969     971 }
```

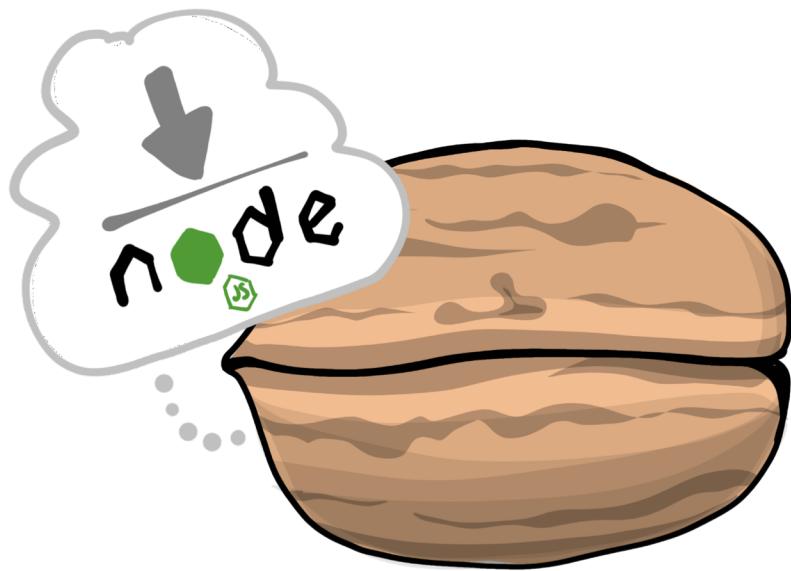
DAVID NEAL @REVERENTGEEK

How To ELECTRoN



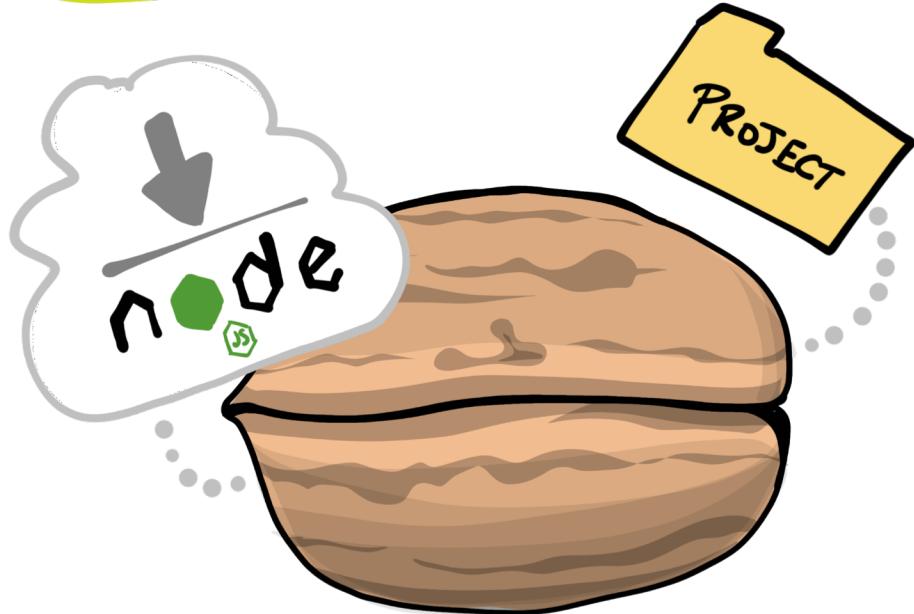
DAVID NEAL @REVERENTGEEK

How To ELECTRoN



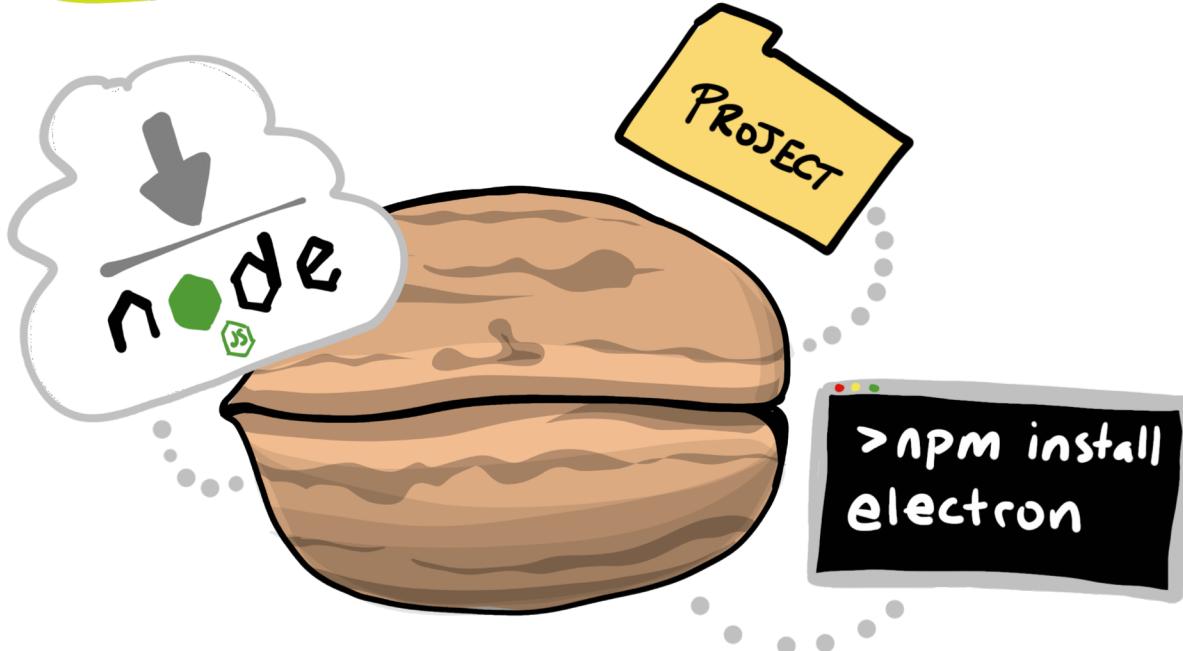
DAVID NEAL @REVERENTGEEK

How To ELECTRoN



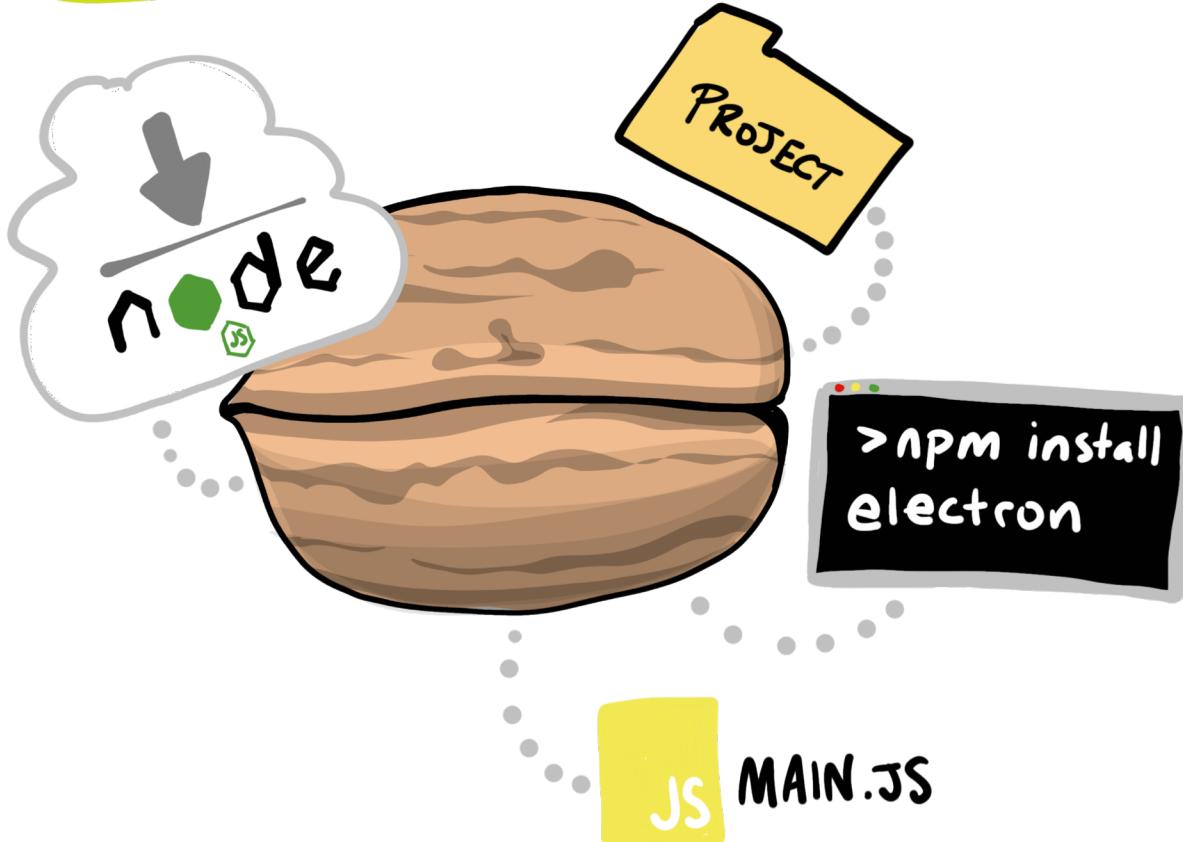
DAVID NEAL @REVERENTGEEK

How To ELECTRoN



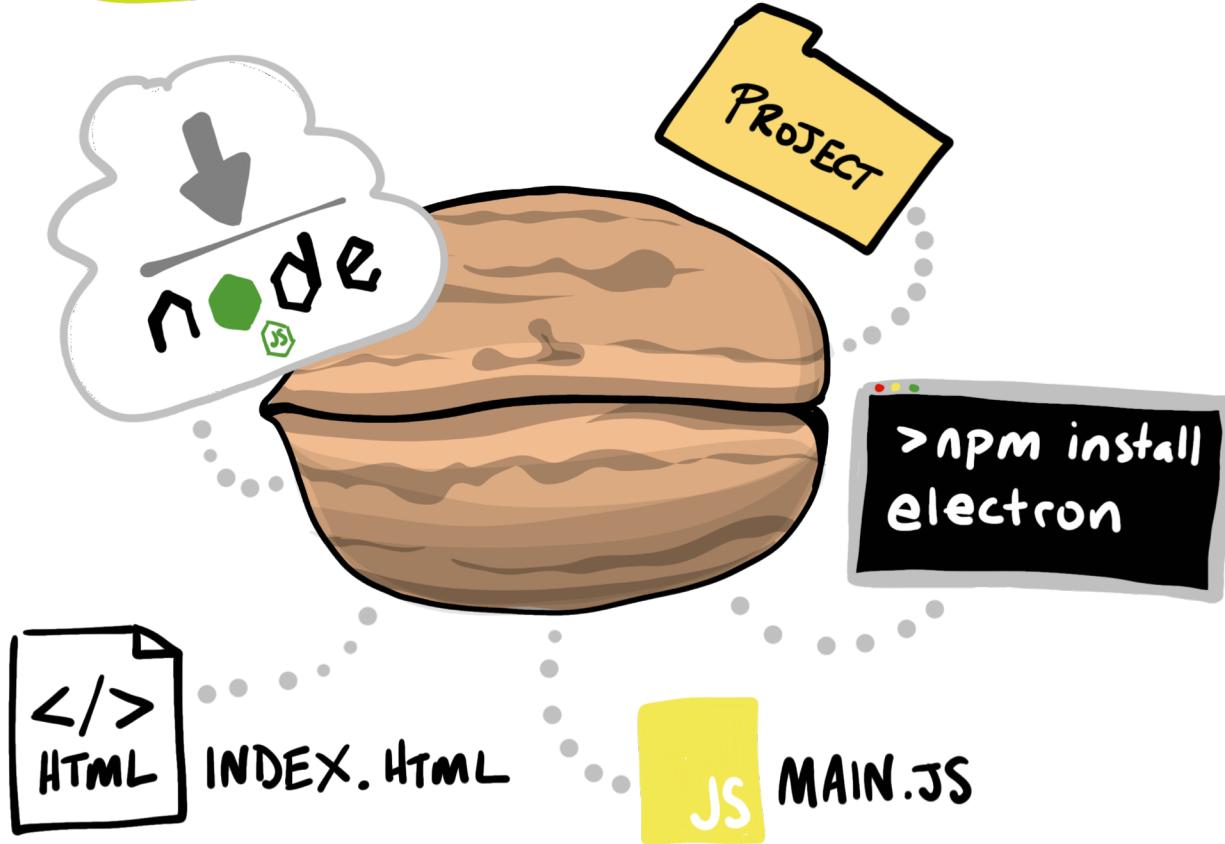
DAVID NEAL @REVERENTGEEK

How To ELECTRoN



DAVID NEAL @REVERENTGEEK

How To ELECTRoN



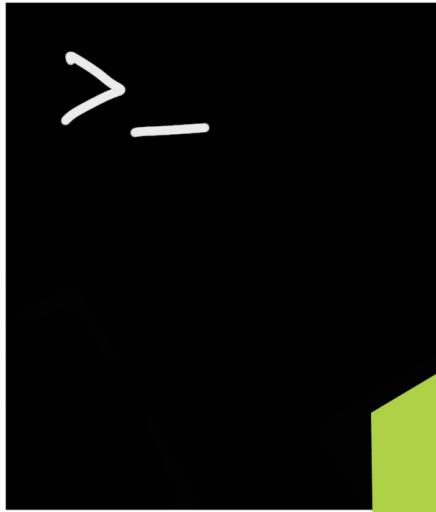
DAVID NEAL @REVERENTGEEK



PROCESS

(main.js)

DAVID NEAL @REVERENTGEEK



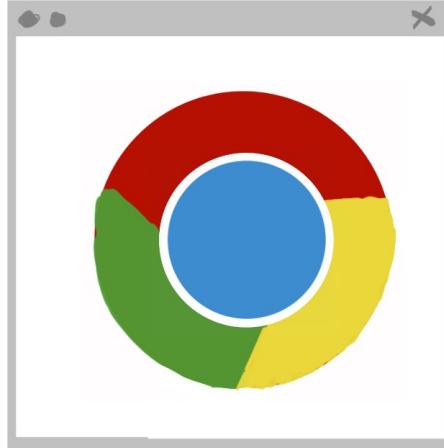
PROCESS

(main.js)

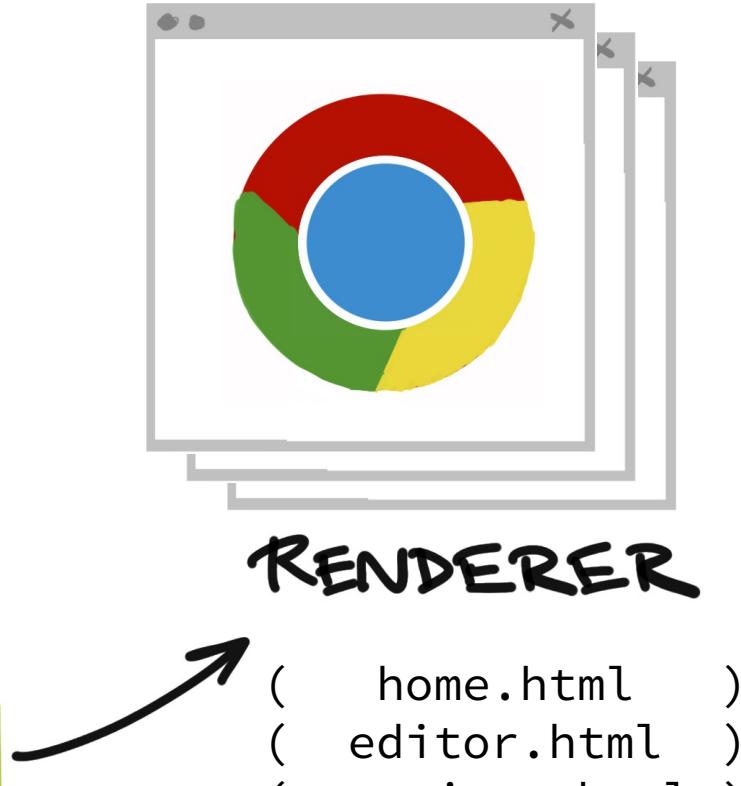
DAVID NEAL @REVERENTGEEK



(main.js)



RENDERER
(index.html)





DAVID NEAL @REVERENTGEEK

main.js

```
1 const electron = require( "electron" );
2 const app = electron.app;
3 const BrowserWindow = electron.BrowserWindow;
4 let mainWindow;
5
6 app.on( "ready", () => {
7     mainWindow = new BrowserWindow( {
8         width: 400,
9         height: 300
10    } );
11    mainWindow.loadURL( `file://${ __dirname }/index.html` );
12} );
```

main.js

```
1 const electron = require( "electron" );
2 const app = electron.app;
3 const BrowserWindow = electron.BrowserWindow;
4 let mainWindow;
5
6 app.on( "ready", () => {
7     mainWindow = new BrowserWindow( {
8         width: 400,
9         height: 300
10    } );
11    mainWindow.loadURL( `file://${ __dirname }/index.html` );
12} );
```

main.js

```
1  const electron = require( "electron" );
2  const app = electron.app;
3  const BrowserWindow = electron.BrowserWindow;
4  let mainWindow;
5
6  app.on( "ready", () => {
7      mainWindow = new BrowserWindow( {
8          width: 400,
9          height: 300
10     } );
11     mainWindow.loadURL( `file://${ __dirname }/index.html` );
12 } );
```

main.js

```
1 const electron = require( "electron" );
2 const app = electron.app;
3 const BrowserWindow = electron.BrowserWindow;
4 let mainWindow;
5
6 app.on( "ready", () => {
7     mainWindow = new BrowserWindow( {
8         width: 400,
9         height: 300
10    } );
11    mainWindow.loadURL( `file://${ __dirname }/index.html` );
12 } );
```

main.js

```
1 const electron = require( "electron" );
2 const app = electron.app;
3 const BrowserWindow = electron.BrowserWindow;
4 let mainWindow;
5
6 app.on( "ready", () => {
7     mainWindow = new BrowserWindow( {
8         width: 400,
9         height: 300
10    } );
11    mainWindow.loadURL( `file://${ __dirname }/index.html` );
12 } );
```

main.js

```
1  const electron = require( "electron" );
2  const app = electron.app;
3  const BrowserWindow = electron.BrowserWindow;
4  let mainWindow;
5
6  app.on( "ready", () => {
7      mainWindow = new BrowserWindow( {
8          width: 400,
9          height: 300
10     } );
11     mainWindow.loadURL( `file://${ __dirname }/index.html` );
12 } );
```

main.js

```
1 const electron = require( "electron" );
2 const app = electron.app;
3 const BrowserWindow = electron.BrowserWindow;
4 let mainWindow;
5
6 app.on( "ready", () => {
7     mainWindow = new BrowserWindow( {
8         width: 400, // x, y, minWidth, minHeight, movable,
9         height: 300 // resizable, alwaysOnTop, kiosk
10    } );
11    mainWindow.loadURL( `file://${ __dirname }/index.html` );
12 } );
```

main.js

```
1  const electron = require( "electron" );
2  const app = electron.app;
3  const BrowserWindow = electron.BrowserWindow;
4  let mainWindow;
5
6  app.on( "ready", () => {
7      mainWindow = new BrowserWindow( {
8          width: 400,
9          height: 300
10     } );
11     mainWindow.loadURL( `file://${ __dirname }/index.html` );
12 } );
```

index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Electron Demo</title>
6  </head>
7  <body>
8      <h1>Hello World!</h1>
9  </body>
10 </html>
```

index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Electron Demo</title>
6  </head>
7  <body>
8      <h1>Hello World!</h1>
9  </body>
10 </html>
```

```
> npm start
```

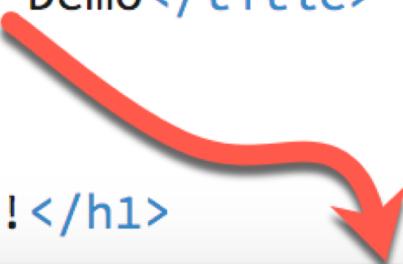
index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Electron Demo</title>
6  </head>
7  <body>
8      <h1>Hello World!</h1>
9  </body>
10 </html>
11
```



index.html

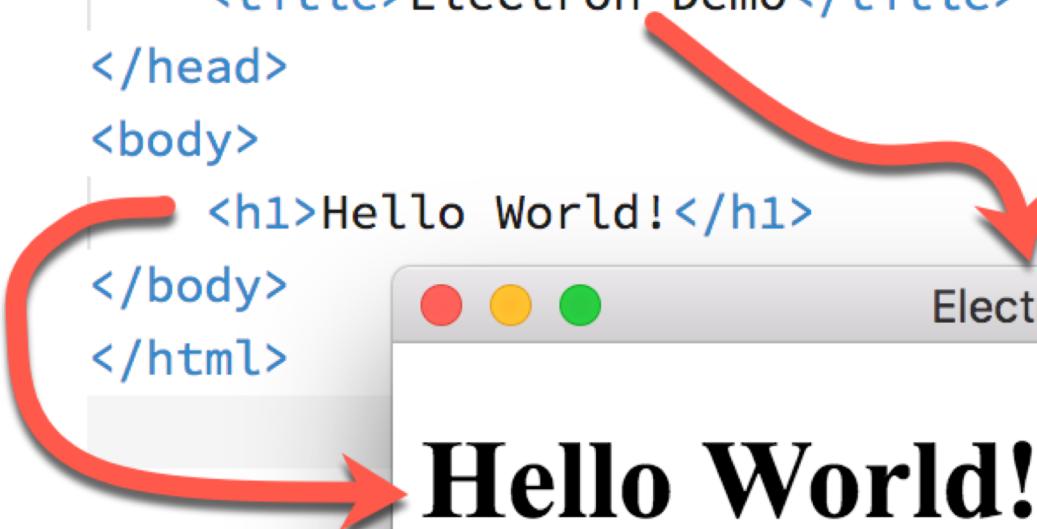
```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Electron Demo</title>
6  </head>
7  <body>
8      <h1>Hello World!</h1>
9  </body>
10 </html>
11
```



Hello World!

index.html

```
1  <!DOCTYPE html>
2  <html>
3  <head>
4      <meta charset="UTF-8">
5      <title>Electron Demo</title>
6  </head>
7  <body>
8      <h1>Hello World!</h1>
9  </body>
10 </html>
```



index.html

```
7 <body>
8     <h1>Hello World!</h1>
9     <div id="info"></div>
10    <script>
11        const msg = `Node.js: ${process.versions.node},
12                      Electron: ${ process.versions.electron},
13                      Chrome: ${ process.versions.chrome}`;
14        document.getElementById("info").textContent = msg;
15    </script>
16 </body>
```

index.html

```
7 <body>
8     <h1>Hello World!</h1>
9     <div id="info"></div>
10    <script>
11        const msg = `Node.js: ${process.versions.node},
12                      Electron: ${ process.versions.electron},
13                      Chrome: ${ process.versions.chrome}`;
14        document.getElementById("info").textContent = msg;
15    </script>
16 </body>
```

index.html

```
18      const fs = require( "fs" );
19      fs.readdir( process.env.HOME, ( err, entries ) => {
20          entries.forEach( entry => {
21              if ( !entry.startsWith( "." ) ) {
22                  const li = document.createElement( "li" );
23                  li.textContent = entry;
24                  homeList.appendChild( li );
25              }
26          } );
27      } );
28  </script>
29 </body>
```

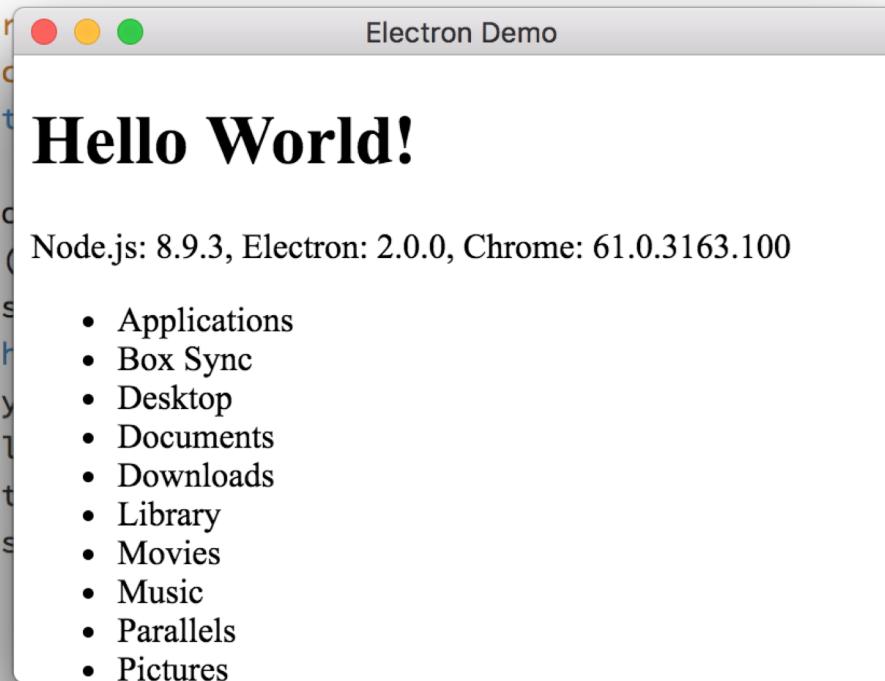
index.html

```
18  const fs = require( "fs" );
19  fs.readdir( process.env.HOME, ( err, entries ) => {
20      entries.forEach( entry => {
21          if ( !entry.startsWith( "." ) ) {
22              const li = document.createElement( "li" );
23              li.textContent = entry;
24              homeList.appendChild( li );
25          }
26      } );
27  } );
28  </script>
29  </body>
```

index.html

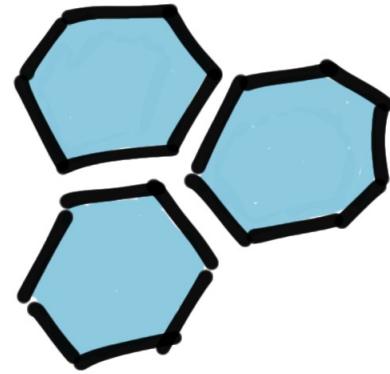
```
18  const fs = require( "fs" );
19  fs.readdir( process.env.HOME, ( err, entries ) => {
20      entries.forEach( entry => {
21          if ( !entry.startsWith( "." ) ) {
22              const li = document.createElement( "li" );
23              li.textContent = entry;
24              homeList.appendChild( li );
25          }
26      } );
27  } );
28  </script>
29  </body>
```

```
7 <body>
8   <h1>Hello World!</h1>
9   <div id="info"></div>
10  <ul id="homeList"></ul>
11  <script>
12    const msg = `Node.js: ${process.versions.node},
13      Electron: ${ process.versions.electron },
14      Chrome: ${ process.versions.chrome }`;
15  document.getElementById('info').innerHTML = msg;
16
17  const homeList = document.querySelector('#homeList');
18  const fs = require('fs');
19  fs.readdir( process.env.HOME, { withFileTypes: true } )
20    .then( entries => entries.forEach( entry =>
21      if ( !entry.isDirectory() ) return;
22      const li = document.createElement('li');
23      li.textContent = entry.name;
24      homeList.appendChild(li);
25    ) );
26  } );
27
28 </script>
29 </body>
```



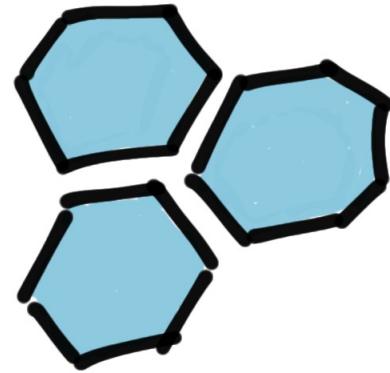
Process modules

- app
- ipc
- dialog
- menu, menu-item
- power-monitor
- tray



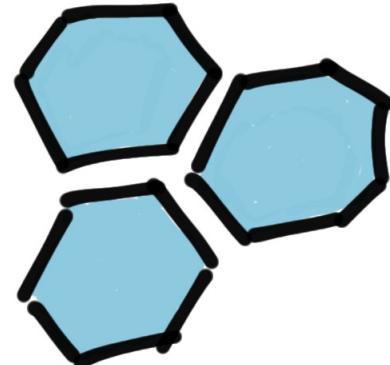
Render modules

- ipc
- remote
- web-frame



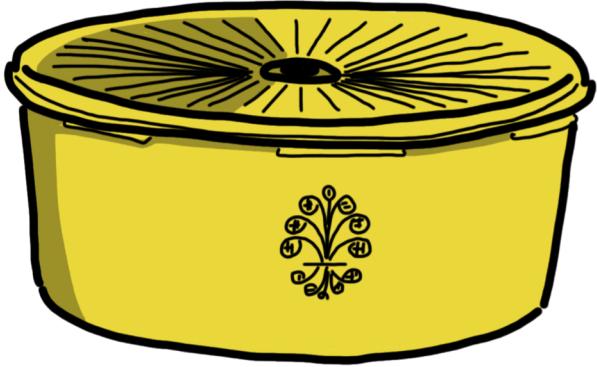
Modules available to both

- clipboard
- crash-reporter
- native-image
- screen
- shell



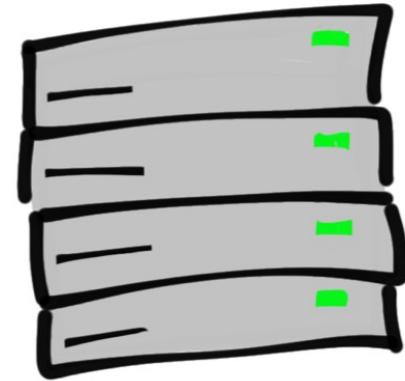
Local Storage

- Read/write .json files
 - Pro tip: use fs-jetpack
- nedb
- pouchdb



Relational DB Storage

- seriate (SQL Server)
- pg
- mysql
- oracledb



Let's Get Started!

<https://bit.ly/electron-workshop>

- or -

<https://github.com/reverentgeek/electron-workshop/wiki>

You Don't Need
PERMISSION
TO BE
AWESOME



THANK
YOU!

DAVID NEAL
@REVERENTGEEK