

**INSTITUTO FEDERAL  
DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA  
DE SÃO PAULO**



ELECTRON



ELECTRON

**CURSO: ANÁLISE E DESENVOLVIMENTO DE SISTEMAS**

**PERIODO: 4º SEMESTRE**

**DISCIPLINA: DESENVOLVIMENTO WEB II**

**PROFESSOR: JOHNATAN SANTICIOLI**

**ALUNO: WILSON BRANDÃO**



O que é Electron



Onde é usado



Arquitetura: Como funciona



Estrutura: Projeto Electron



Hello World!



Conclusão



## O que é Electron?

🚀 Electron é um framework, open Source e multiplataforma, desenvolvido pelo **Github**.

🚀 Foi criado em 2013 pelo engenheiro do Github **Cheng Zhao**.

🚀 O projeto era chamado no inicio de Atom Shell.

🚀 Em 2014 o projeto se tornou open Source, com licença MIT.

🚀 Atualmente é mantido pela OpenJS Foundation, organização com objetivo de apoiar projetos OS baseados em Javascript.







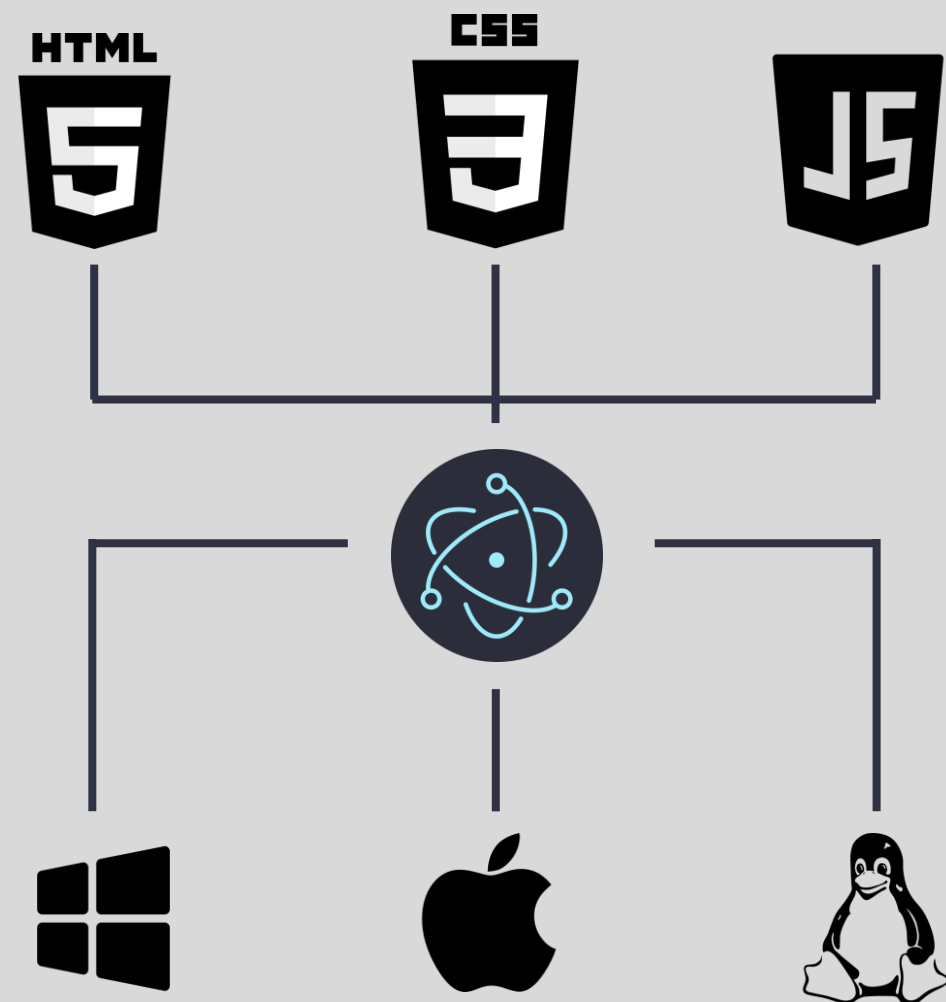
## O que é Electron?

Mas afinal, o que é esse framework?

Electron é um shell multiplataforma, ou seja uma interface do usuário para acessar serviços do sistema operacional tanto via linha de comando (CLI) e interface gráfica (GUI)

 Utilizado para criar aplicações desktop usando tecnologias web (HTML, CSS e Javascript).

 Possui integração com Node.js, possibilitando a construção de lógica do BackEnd, acessando recursos do SO, diretórios, banco de dados, etc.





Onde é usado?

E que tipo de aplicações pode ser feita com Electron?

## Algumas aplicações feitas com Electron



Visual Studio Code



Discord



Figma  
Desktop



Whatsapp  
Desktop



Skype



Onde é usado?



Visual Studio Code

The screenshot displays the Visual Studio Code interface with a workspace named 'main.js - default (Workspace)'. The Explorer sidebar on the left shows a project structure for 'theCatSaidNo' with folders like '.github', '.vscode', 'assets', 'static', 'images', 'javascript', 'stylesheets', and 'templates'. The main editor area is split into three panes: 'home.html' (HTML), '# style.css' (CSS), and 'JS main.js' (JavaScript). The HTML file contains a basic page structure with a title 'The Cat said No!', links to Google Fonts, and a button that triggers a JavaScript function. The CSS file defines styles for the button and a container. The JavaScript file contains logic for toggling the button state and updating the page content. The status bar at the bottom indicates the current file is 'main.js' at line 11, column 1, with 4 spaces, UTF-8 encoding, and LF line endings.

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <title>The Cat said No!</title>
6   <link href="https://fonts.googleapis.com/css?family=Montserrat:400,700" rel="stylesheet" type="text/css">
7   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css">
8   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
9   <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js">
10  </head>
11
12 <body class="preload">
13   <div class="centered">
14     <button type="button" class="btn btn-primary" id="togglePaw">
15       <div class="handle"></div>
16     </button>
17     <div class="catpaw-container">
18       
19     </div>
20     <div>
21       <h1 style="text-align: center;">The Cat said No!</h1>
22     </div>
23     <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
24     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
25     <script>
26       $(window).load(function () {
27         $(".preload").removeClass("preload");
28       });
29     </script>
30     <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js" integrity="sha384-Tc5IQib026qysyN4RybnjKihBNtdfKQ7nnI642Oi8mkYzo0f1b616KC7fytIHXD5" crossorigin="anonymous"></script>
31   </div>
32 </body>
```

```
1 body {
2   font-family: 'Montserrat', 'Lato', 'Helvetica Neue', sans-serif;
3   color: #6b7381;
4   background: #fff;
5   -webkit-touch-callout: none;
6   -webkit-user-select: none;
7   -khtml-user-select: none;
8   -moz-user-select: none;
9   -ms-user-select: none;
10  user-select: none;
11  transition: background-color 0.25s;
12 }
13
14 .jumbotron {
15   background: #6b7381;
16   color: #bdc1c8;
17 }
18 .jumbotron h1 {
19   color: #fff;
20 }
21 .example {
22   margin: 4rem auto;
23 }
24 .example > .row {
25   margin-top: 2rem;
26   height: 5rem;
27   vertical-align: middle;
28   text-align: center;
29   border: 1px solid #181818;
30 }
31 .example > .row:first-of-type {
32   border: none;
33   height: auto;
34   text-align: left;
35 }
36 .example h3 {
37   font-weight: 400;
```

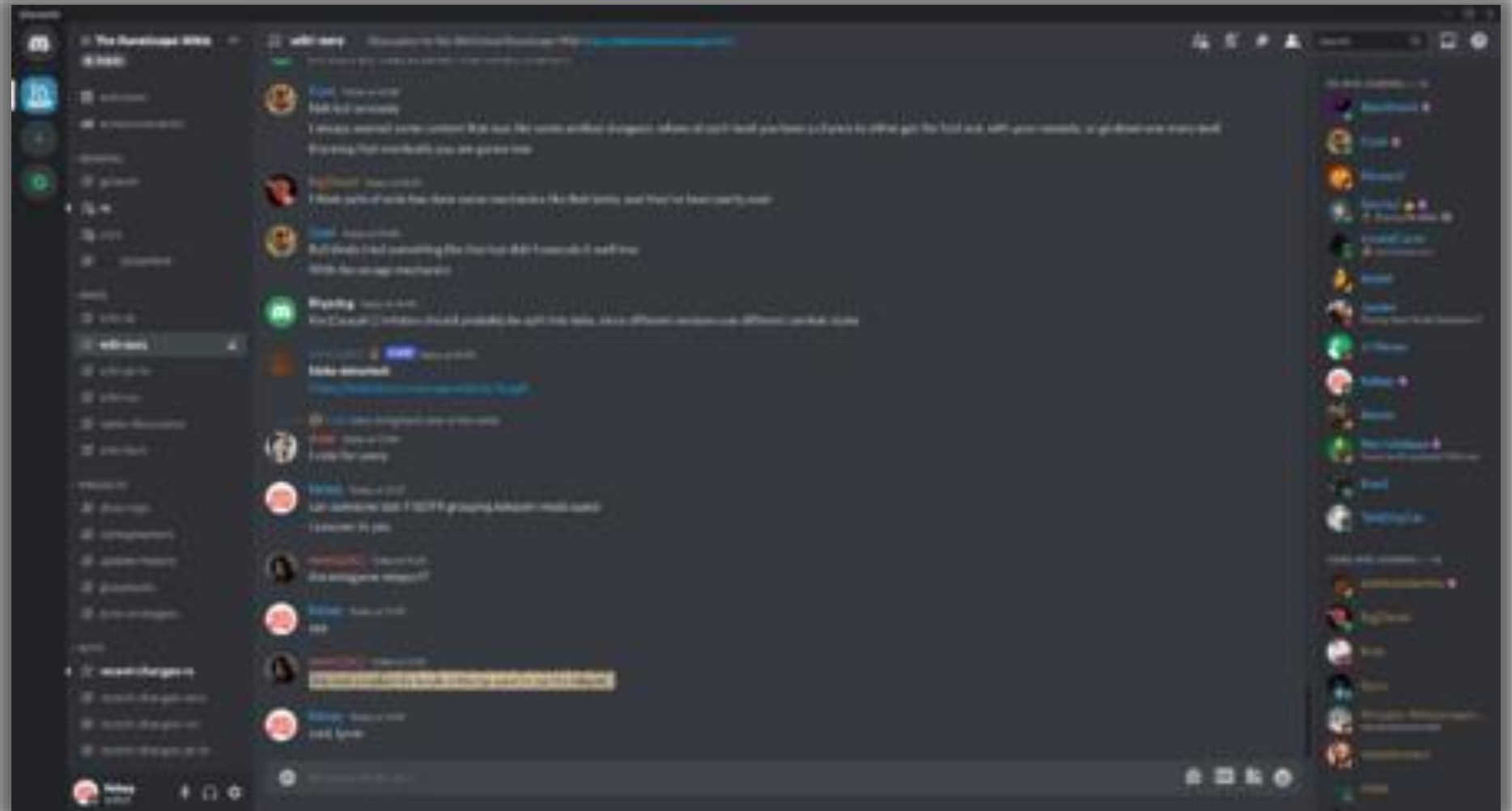
```
1 pawToggled = false;
2 var myTimeout;
3
4 function callbackToggle() {
5   return function () {
6     if (pawToggled) {
7       document.getElementById("togglePaw").innerHTML = "Paw";
8     }
9   }
10 }
11
12 function togglePaw() {
13   if (!pawToggled) {
14     // Runs when we toggle the button
15     document.getElementById("togglePaw").innerHTML = "Paw";
16     myTimeout = setTimeout(callbackToggle, 1000);
17   } else {
18     document.getElementById("togglePaw").innerHTML = "Paw";
19     clearTimeout(myTimeout);
20   }
21
22   pawToggled = !pawToggled;
23 }
24
```



Onde é usado?



Discord







## Arquitetura: Como funciona?

O Electron é construído baseado nas libs do chromium, um projeto open Source de onde surgiu o Google Chrome, e com runtime do Node.js.



**Chromium**

Camada de FrontEnd, Renderização da parte visual, janelas, etc.



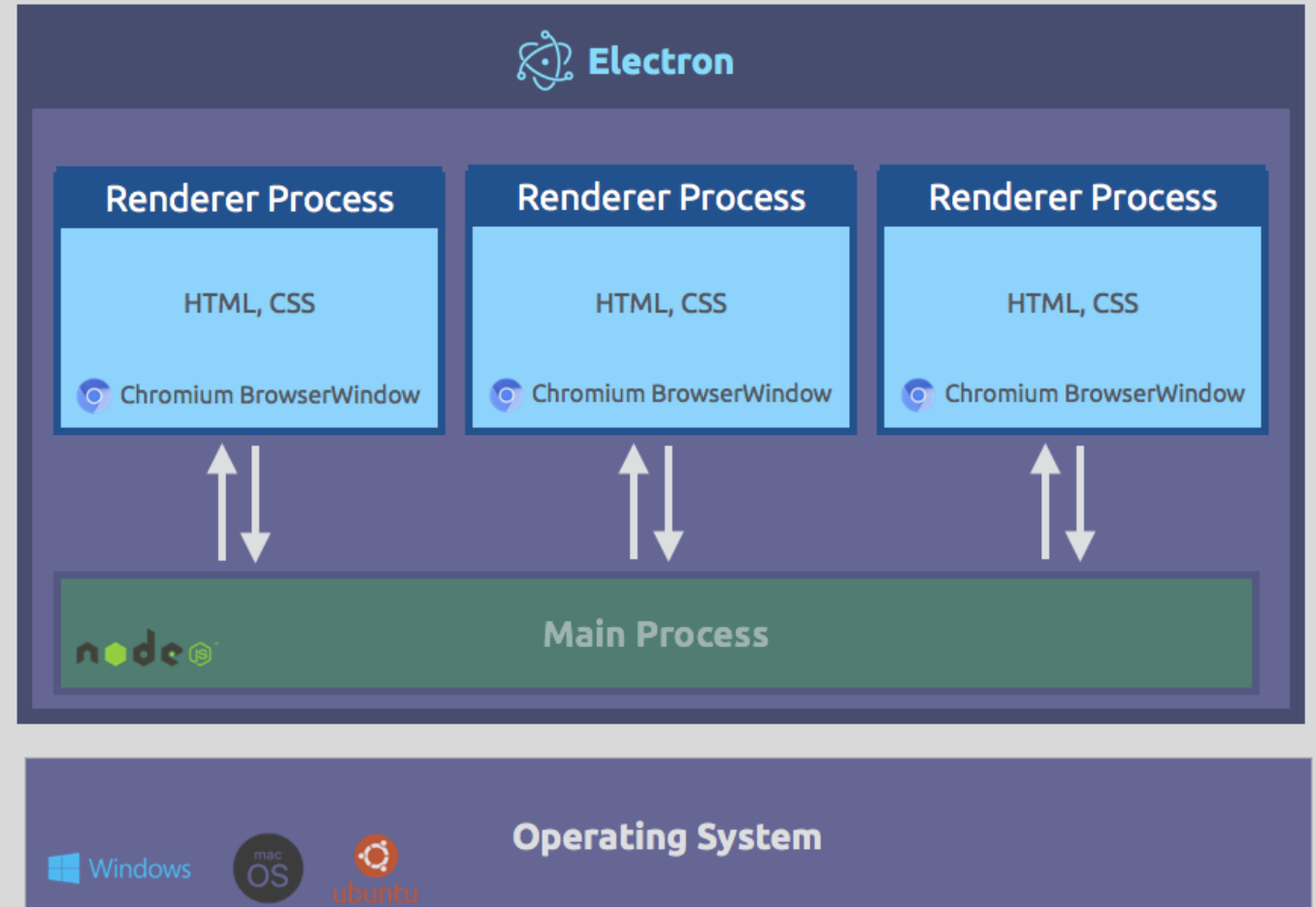
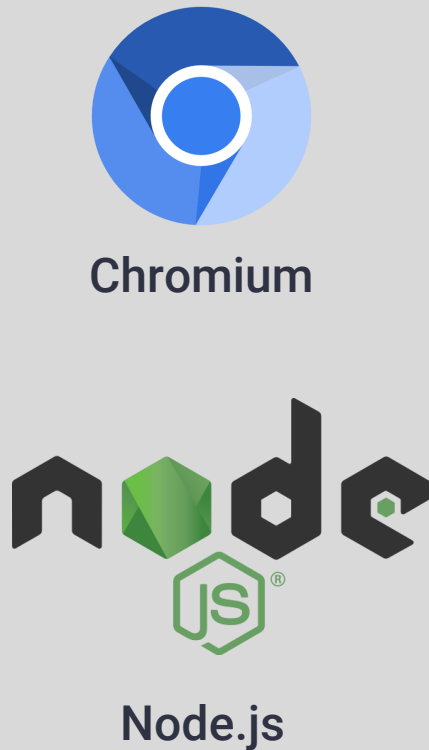
**Node.js**

Camada de BackEnd, acesso ao sistema operacional e banco de dados.



## Arquitetura: Como funciona?

O Electron é construído baseado na macro arquitetura do chromium, um projeto open Source de onde surgiu o Google Chrome, e com runtime do Node.js.

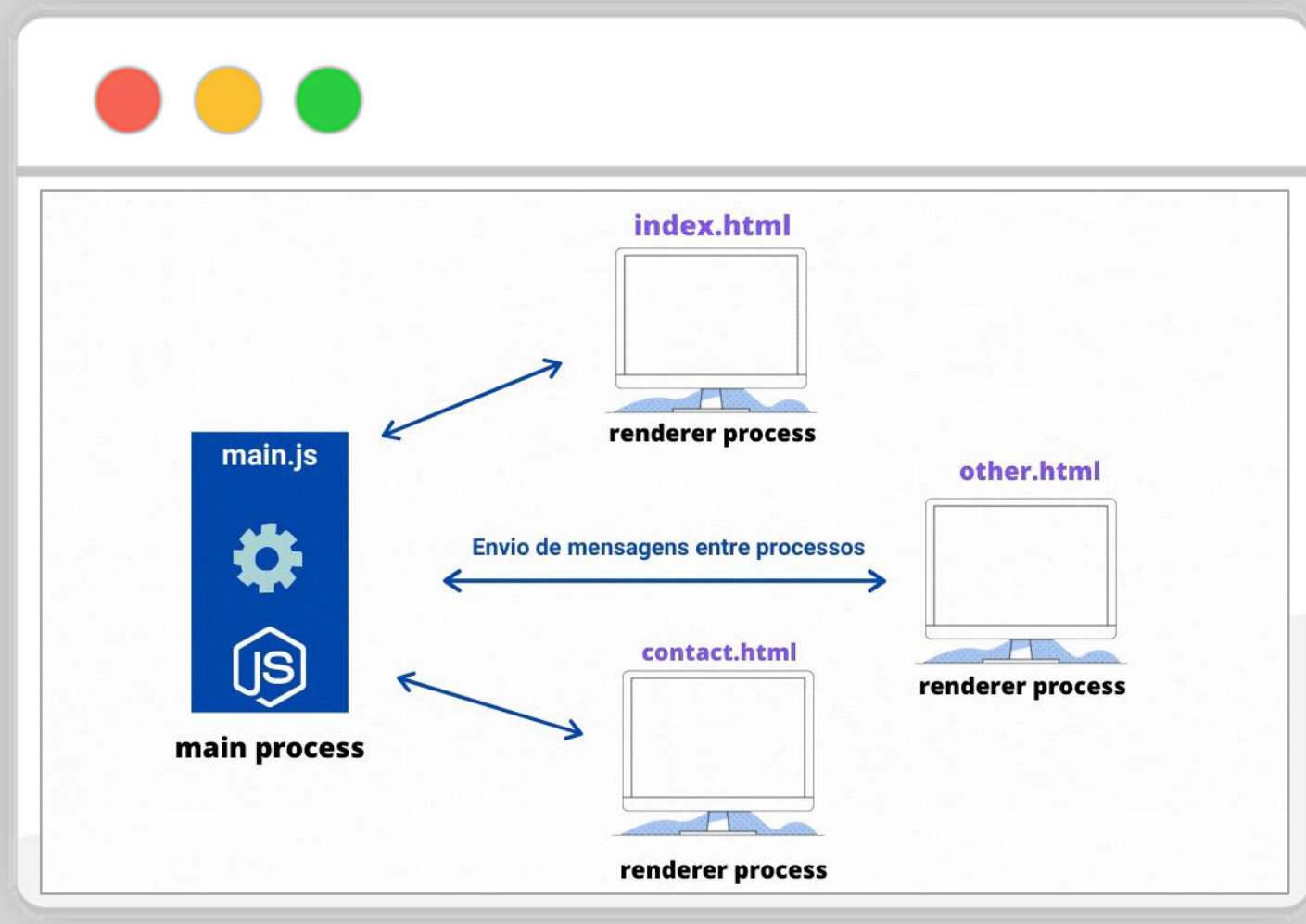




## Arquitetura: Como funciona?

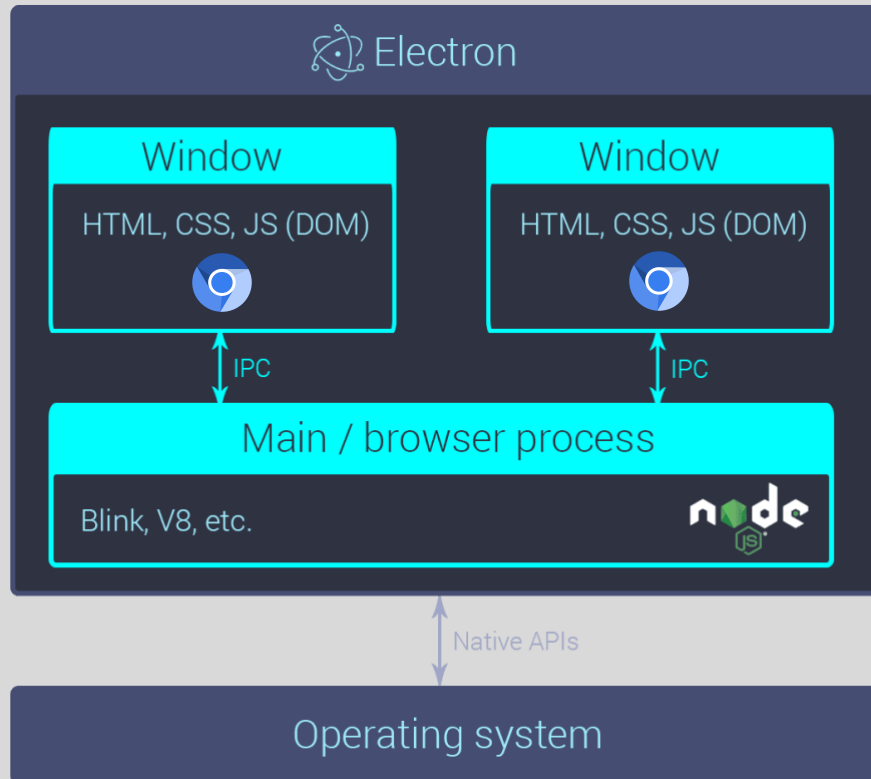
A arquitetura de um projeto Electron é composta por três estruturas principais:

- main process
- renderer process
- comunicação entre esses processos





## Arquitetura: Como funciona?



### Main process:

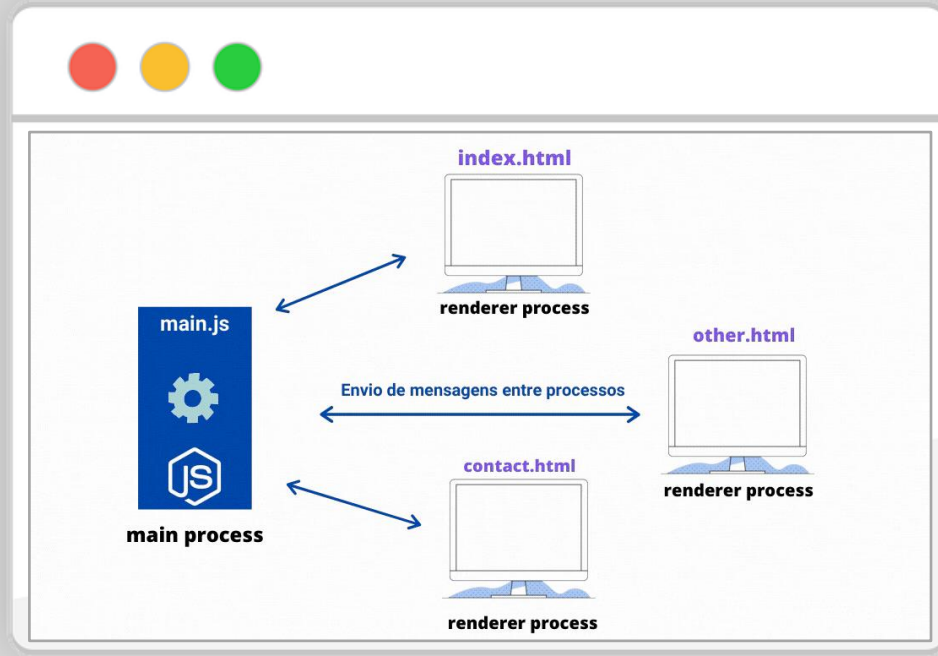
O main.js é o arquivo principal do electron, e é único em toda aplicação

Ele é responsável por criar as janelas da aplicação(render process) através de instancias `BrowserWindow`.

O main.js deve ser definido dentro do arquivo `package.json` na propriedade "main".



## Arquitetura: Como funciona?



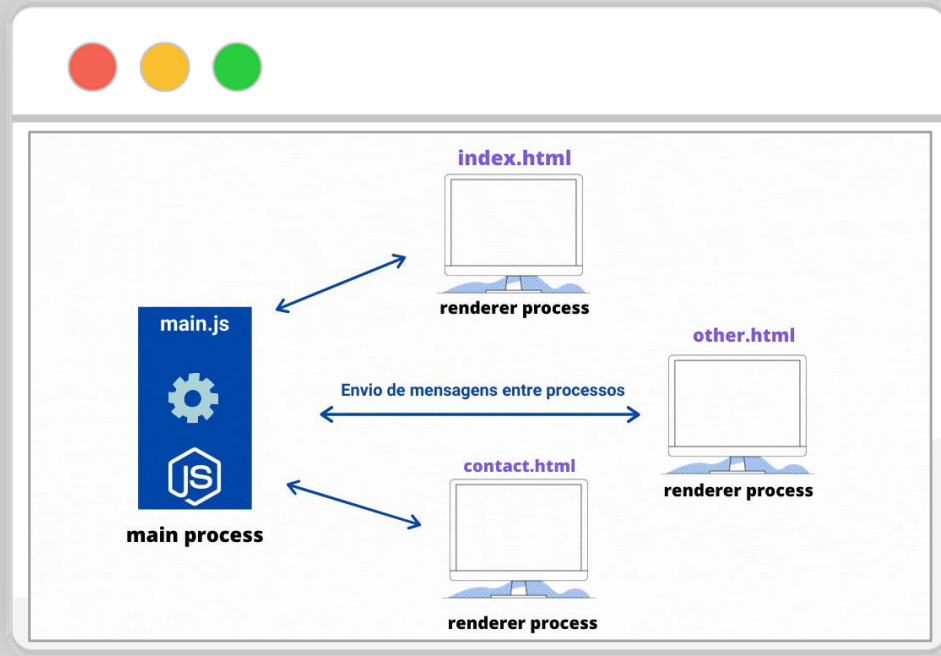
### Renderer process:

São as páginas da web (janela desktop), que utilizam a arquitetura de multiprocessos do chromium.

Cada janela desktop do Electron (pagina web) é renderizada em seu próprio processo chamado de **renderer process**



## Arquitetura: Como funciona?



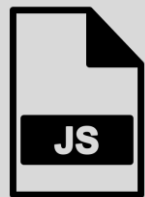
### IPC e RPC: Comunicação entre processos

A comunicação entre **main process** os **renderer**, é feita através de um conceito de comunicação entre processos (IPC) usando o método RPC - Remote Procedure Call (Chamada de Procedimentos Remotos).

Para isso temos as APIs `ipcRender` e `ipcMain`, que faz a comunicação entre arquivos `.js` e `.html`



## Estrutura do projeto Electron



main.js



Inicia o app e cria uma janela do navegador para renderizar HTML



index.html



Página da web que será renderizada.



package.json



Aponta para o arquivo principal do app e lista as suas dependências



gerenciador de pacotes npm (ou yarn)



## Estrutura do projeto Electron



/meDaNota10



/meDaNota10/node\_modules



window title

```
usuario@pc ~ $ mkdir meDaNota10 && npm init -y  
usuario@pc ~ $ cd meDaNota10
```





## Estrutura do projeto Electron



/meDaNota10



/meDaNota10/node\_modules



/meDaNota10/package.json



/meDaNota10/main.js



/meDaNota10/index.html



window title

```
usuario@pc ~ $ mkdir meDaNota10 && npm init -y
usuario@pc ~ $ cd meDaNota10
usuario@pc ~/meDaNota10 $ npm install --save-dev electron
usuario@pc ~/meDaNota10 $ touch main.js index.html
```



## Estrutura do projeto Electron



/meDaNota10



/meDaNota10/build



icon.png



/meDaNota10/node\_modules



/meDaNota10/package.json



/meDaNota10/main.js



/meDaNota10/index.html



window title

```
usuario@pc ~ $ mkdir meDaNota10 && npm init -y
usuario@pc ~ $ cd meDaNota10
usuario@pc ~/meDaNota10 $ npm install --save-dev electron
usuario@pc ~/meDaNota10 $ touch main.js index.html
usuario@pc ~/meDaNota10 $ mkdir build
usuario@pc ~/meDaNota10 $ mv ~/icon.png /build
usuario@pc ~/meDaNota10 $ code .
```



## Estrutura do projeto Electron



/meDaNota10/package.json

```
package.json > ...
1  {
2    "name": "medanota10",
3    "version": "1.0.0",
4    "description": "",
5    "main": "main.js",
6    "scripts": {
7      "start": "electron main.js"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC"
12 }
```



## Estrutura do projeto Electron



/meDaNota10/index.html

window title

```
File Edit Selection View Go Run Terminal Help
index.html - meDaNota10 - Visual Studio Code
index.html U x
index.html > html
1  <!DOCTYPE html>
2  <html Lang="en">
3  <head>
4    <meta charset="UTF-8">
5    <meta http-equiv="X-UA-Compatible" content="IE=edge">
6    <meta name="viewport" content="width=device-width, initial-scale=1.0">
7    <title>Hello World</title>
8  </head>
9  <body>
10   <h1>Hello World - Web 2 </h1>
11 </body>
12 </html>
```



## Estrutura do projeto Electron



/meDaNota10/main.js

```
main.js > ...
1  const { app, BrowserWindow, nativeImage } = require("electron");
2
3  // Função que cria uma janela desktop
4  function createWindow() {
5      // Adicionando um ícone na barra de tarefas/dock
6      const icon = nativeImage.createFromPath(`${app.getAppPath()}/build/icon.png`);
7
8      if (app.dock) {
9          app.dock.setIcon(icon);
10     }
11
12     // Cria uma janela de desktop
13     const win = new BrowserWindow({
14         icon,
15         width: 800,
16         height: 600,
17         webPreferences: {
18             // habilita a integração do Node.js no FrontEnd
19             nodeIntegration: true,
20         },
21     });
22
23     // carrega a janela com o conteúdo dentro de index.html
24     win.loadFile("index.html");
25 }
```



## Estrutura do projeto Electron



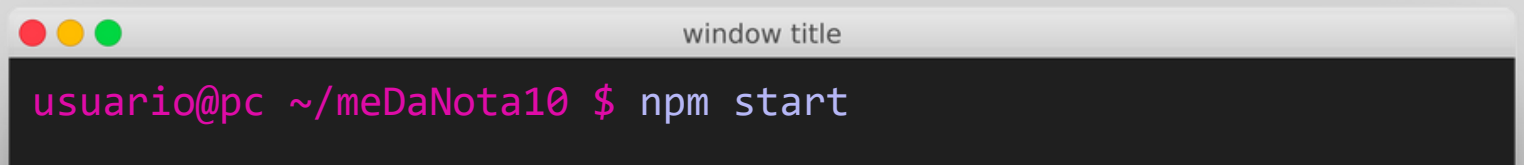
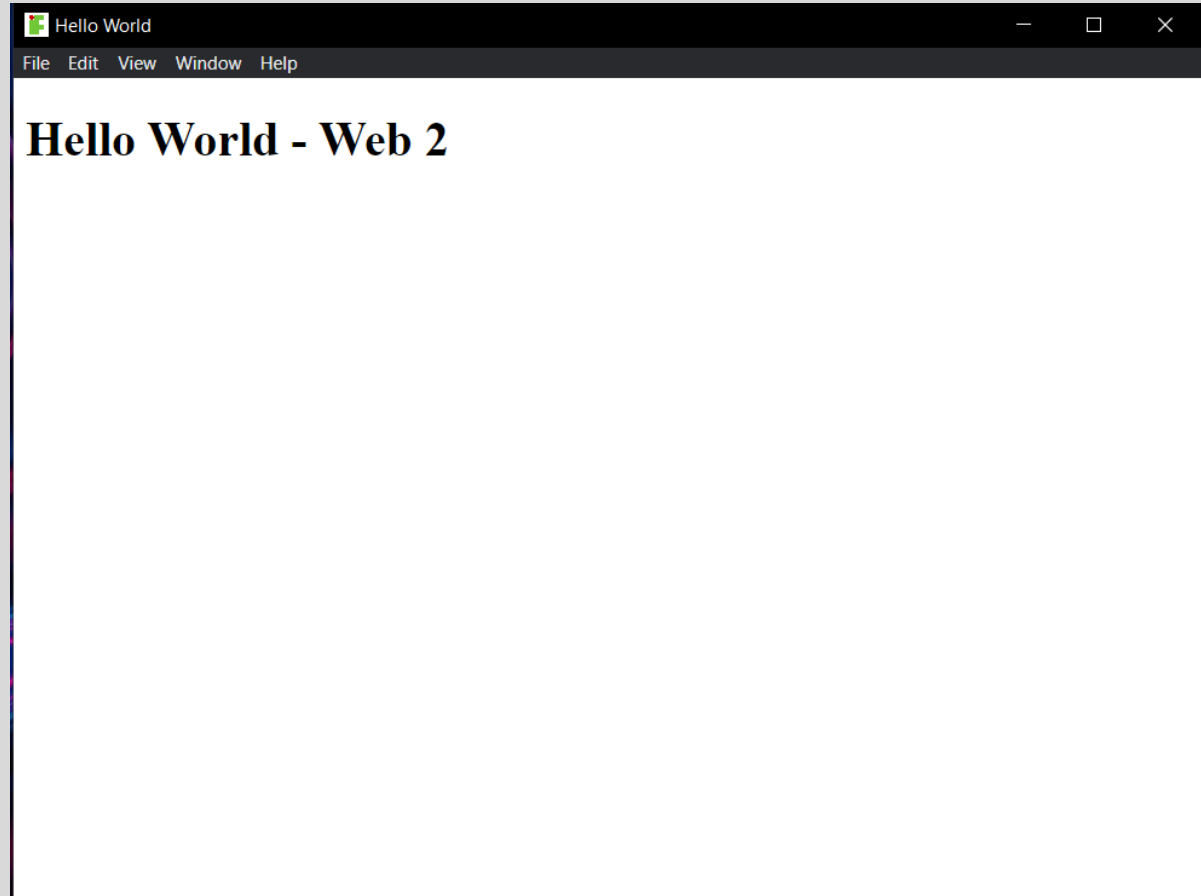
/meDaNota10/main.js

```
window title
File Edit Selection View Go Run Terminal Help
main.js - meDaNota10 - Visual Studio Code
main.js
26
27 // Método vai ser chamado assim que o Electron finalizar sua inicialização
28 // e estiver pronto para abrir e manipular o nosso código.
29 // Algumas APIs podem ser usadas somente depois que este evento ocorre.
30 app.whenReady().then(createWindow);
31
32 // Quando clicarmos no botão de fechar a janela no app desktop
33 // O evento vai ser ouvido aqui no arquivo main.js e algum procedimento pode ser realizado
34 // tipo fechar alguma conexão de banco de dados por exemplo.
35 app.on("window-all-closed", () => {
36   // No MacOS quando fecha uma janela, na verdade ela é "minimizada"
37   if (process.platform !== "darwin") {
38     app.quit();
39   }
40 });
41
42 app.on("activate", () => {
43   // Esse evento é disparado pelo MacOS quando clica no ícone do aplicativo no Dock.
44   // Basicamente cria a janela se não foi criada.
45   if (BrowserWindow.getAllWindows().length === 0) {
46     createWindow();
47   }
48 });
49
```



Hello World

Aplicativo Electron iniciado em  
janela do sistema operacional  
Windows 10





Hello World Compilado



/meDaNota10/package.json

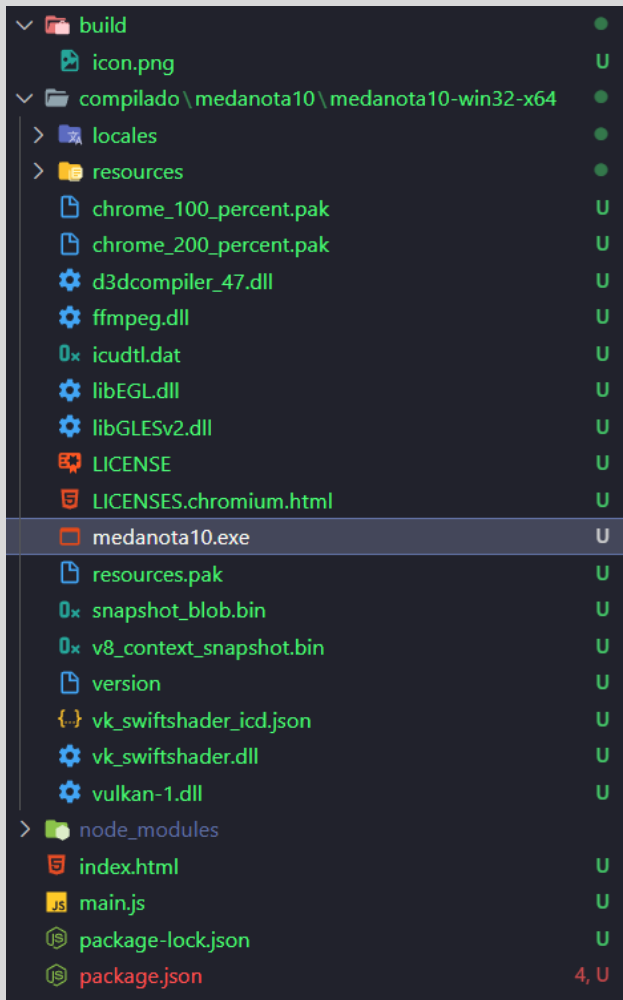
```
window title
usuario@pc ~/meDaNota10 $ npm install electron-packager -g
```

```
window title
package.json > ...
1  {
2    "name": "medanota10",
3    "version": "1.0.0",
4    "description": "",
5    "main": "main.js",
6    "scripts": {
7      "start": "electron main.js",
8      "compilar": "electron-packager ./ medanota10 --plataform=windows
          --arch=x64 --out ./compilado/medanota10 --app-version 1.0.0
          --electron-version 21.1.1 --overwrite"
9    },
10   "keywords": [],
11   "author": "",
12   "license": "ISC"
13 }
14
15
```





# Hello World Compilado



Arquivo  
.exe





### Vantagens

- Multiplataforma
- Open Source
- Tecnologias Web como HTML, CSS e JS
- Interfaces nativas que possibilitam se adaptar ao SO em execução.

### Desvantagens

- Tamanho do apps
- Chromium tem mais de 20 milhões de linhas
- Um hello world pode chegar a 100MB
- Proteção: O código não é criptografado

**OBRIGADO!**



ELECTRON

**WILSON BRANDÃO!**