Electron

Construa aplicativos desktop multiplataforma com JavaScript, HTML e CSS

Fernando M. Tenório

@dotenorio

JAVASCRIPT, HTML e CSS!!!

É fácil!
Se você já sabe fazer um site..

Mas qual é mágica?

Não é mágica, é tecnologia..

Tecnologias Web

- Chromium
- NodeJS
- HTML, CSS, Javascript

Código Aberto

- Atom Shell (2013)
- Electron, MIT e Mantido pelo GitHub (2014)

Multiplataforma

- Mac
- Windows
- Linux

Até as partes difíceis são fáceis

É sério!

- Atualizações automáticas
- Menus nativos & notificações
- Relatório de falhas
- Depuração & criação de perfil
- Instaladores
- Windows Store e Mac App Store

E muitos outros recursos


ELECTRON Doadores Aplicativos Documentação Blog Comunidade Governance Versões Buscar Português
Documentação do Electron 5.0.0
Veja toda a documentação em uma só página ou verifique as perguntas frequente.
Guias
Sobre o Electron
Acessibilidade
Electron App Feedback Program
Arquitetura de Aplicativos Electron
Depuração de Aplicativos
Distribuição de Aplicativos
Empacotamento de Aplicativos
Automatizando Teste com Driver Personalizado
Boilerplates e CLIs
Assinando Código
O Processo Principal de Depuração
Depuração do Processo Principal em VSCode
Integração com Ambiente de Trabalho
Ambiente de Desenvolvimento
Extensão de DevTools
5.0.0 Release Schedule
Versionamento do Electron
Escrevendo Seu Primeiro Aplicativo com Electron
Compra no app (macOS)
Instalação
Atalhos do Teclado
Custom Linux Desktop Launcher Actions
Guia para Mac App Store
Dock do MacOS
Referência da API
Aceleradores
app
autoUpdater
Contrato da API
BrowserView
BrowserWindow
BrowserWindowProxy
Suporte ao Terminal de Comando do Chrome
ClientRequest
clipboard
contentTracing
Cookies
crashReporter
Debugger
desktopCapturer
dialog
DownloadItem
Variáveis de Ambiente
Objeto File
Frameless Window
globalShortcut
inAppPurchase
IncomingMessage
ipcMain
ipcRenderer
Localização
Estruturas da API
BluetoothDevice Object
Objeto Certificado
CertificatePrincipal Object
Cookie Object
CPUUsage Object
Objeto de Relatório de Erro
DesktopCapturerSource Object
Objeto Display
FileFilter Object
GPUFeatureStatus Object
Objeto IOCounters
JumpListCategory Object
JumpListItem Object
Objeto MemoryInfo
MemoryUsageDetails Object
Objeto MimeTypedBuffer
Objeto NotificationAction
Point Object
Objeto PrinterInfo
Processamento de Objeto
Objeto do produto
Rectangle Object
Referrer Object
Objeto RemoveClientCertificate
Objeto RemovePassword
Objeto ScrubberItem
Avançado
Diferentes Técnicas entre o Electron e NW.js (formalmente node-webkit)
Instruções de Compilação
Instruções para Configurar (Linux)
Instruções para Compilação (macOS)
Instruções para Configuração (Windows)
Construir Resumo do Sistema
Desenvolvimento do Chromium
Usando clang-format em Código C++
Estilo de Codificação
Depuração no Windows
Depuração no macOS
Debugging with XCode
Questões com o Electron
Pull Requests
Desenvolvendo com Electron
Lançamento
Setting Up Symbol Server in Debugger
Estrutura de Diretório do Código Fonte
Testando
Atualizando o Chromium
Upgrading Crashpad
Atualizando NodeJS
V8 Desenvolvimento

Muitos mesmo..

- Mojave Dark Mode
- Multitarefa
- Nativo Arquivo Drag & Drop (Arrastar e Soltar)
- Notificações (Windows, Linux, macOS)
- Renderização fora da tela
- Detecção de Evento Online/Offline
- Barra de Progresso na Barra de Tarefas (Windows, macOS, Unity)
- Início Rápido
- Recent Documents (Windows & macOS)
- REPL
- Representação de Arquivo para o macOS BrowserWindows
- Segurança, Capacidades Nativas e Suas Responsabilidades
- Snapcraft Guide (Ubuntu Software Center & More)
- Electron Support
- Testing on Headless CI Systems (Travis CI, Jenkins)
- Testando Widevine CCDM
- Atualizando Aplicativos
- Usando Módulos Nativos do Node
- Usando o Plugin Pepper Flash
- Usando Selenium e WebDriver
- Guia para Windows Store
- Windows Taskbar

- Menu
- MenuItem
- nativeImage
- net
- netLog
- Notificações
- powerMonitor
- powerSaveBlocker
- process
- protocol
- remote
- sandbox Option
- screen
- session
- shell
- Sinopse
- systemPreferences
- TouchBar
- TouchBarButton
- TouchBarColorPicker
- TouchBarGroup
- TouchBarLabel
- TouchBarPopover
- TouchBarScrubber
- TouchBarSegmentedControl
- TouchBarSlider
- TouchBarSpacer
- Tray
- webContents
- webFrame
- WebRequest
- <webview> Tag
- Função window.open

- Objeto ShortcutDetails
- Objeto Size
- Objeto StreamProtocolResponse
- Task Object
- Objeto ThumbarButton
- TraceCategoriesAndOptions Object
- Objeto TraceConfig
- Objeto de Transação
- Objeto UploadBlob
- Objeto UploadData
- Objeto UploadFile
- Objeto UploadRawData
- Objeto WebSource

Todos Documentos em Uma Página ou Veja as perguntas frequentes.

dotenorio.com http://bit.ly/electron-tdc

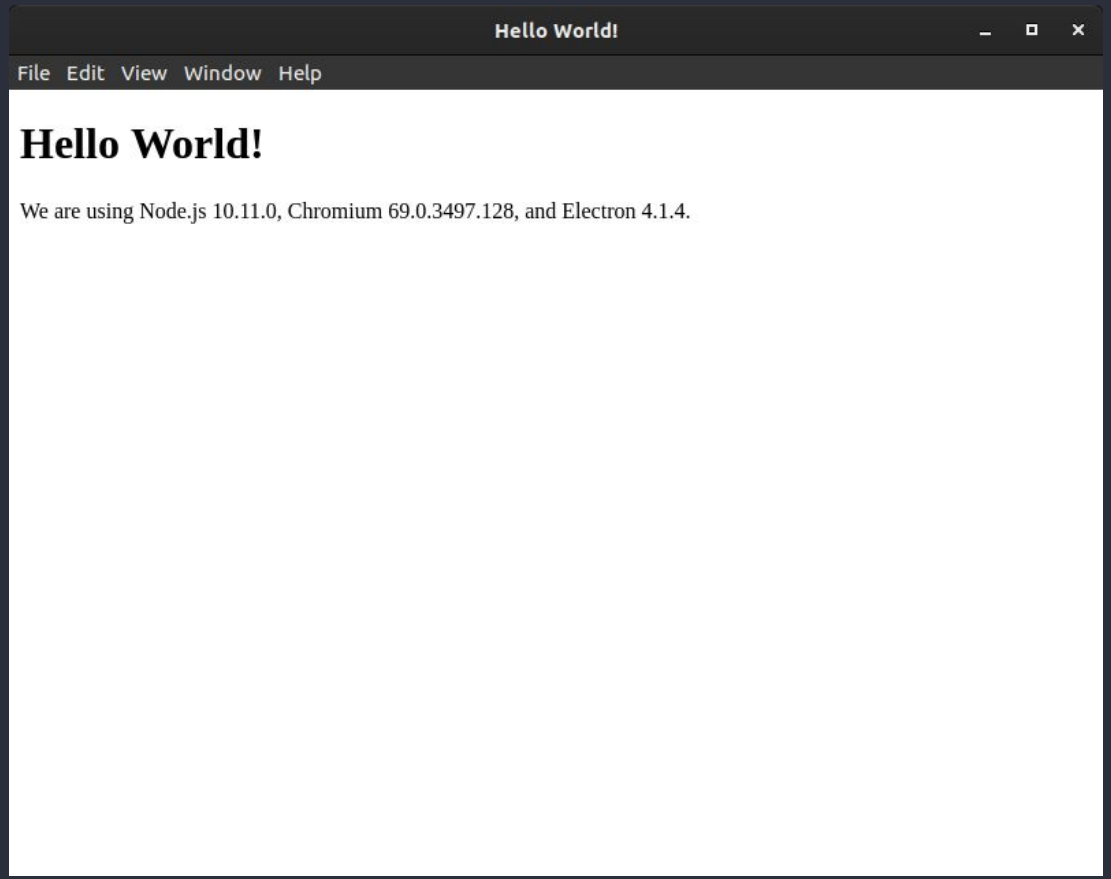
Quick Start

```
# Clone o repositório Quick Start
$ git clone https://github.com/electron/electron-quick-start

# Ir para o repositório
$ cd electron-quick-start

# Instale as dependências e execute
$ npm install && npm start
```

Só 3 linhas e já temos uma aplicação..


Hello World!
File Edit View Window Help
Hello World!
We are using Node.js 10.11.0, Chromium 69.0.3497.128, and Electron 4.1.4.

Olá Mundo!

```
{
  "name": "electron-quick-start",
  "version": "1.0.0",
  "description": "A minimal Electron application",
  "main": "main.js",
  "scripts": {
    "start": "electron ."
  },
  "repository": "https://github.com/electron/electron-quick-start",
  "keywords": [
    "Electron",
    "quick",
    "start",
    "tutorial",
    "demo"
  ],
  "author": "GitHub",
  "license": "CC0-1.0",
  "devDependencies": {
    "electron": "^4.1.4"
  }
}
```

package.json

main.js (1/3)

```
function createWindow () {
  mainWindow = new BrowserWindow({
    width: 800,
    height: 600,
    webPreferences: {
      nodeIntegration: true
    }
  })

  mainWindow.loadFile('index.html')

  // Open the DevTools.
  // mainWindow.webContents.openDevTools()

  mainWindow.on('closed', function () {
    mainWindow = null
  })
}

app.on('ready', createWindow)
```

main.js (2/3)

```
app.on('window-all-closed', function () {
  if (process.platform !== 'darwin') app.quit()
})

app.on('activate', function () {
  if (mainWindow === null) createWindow()
})
```

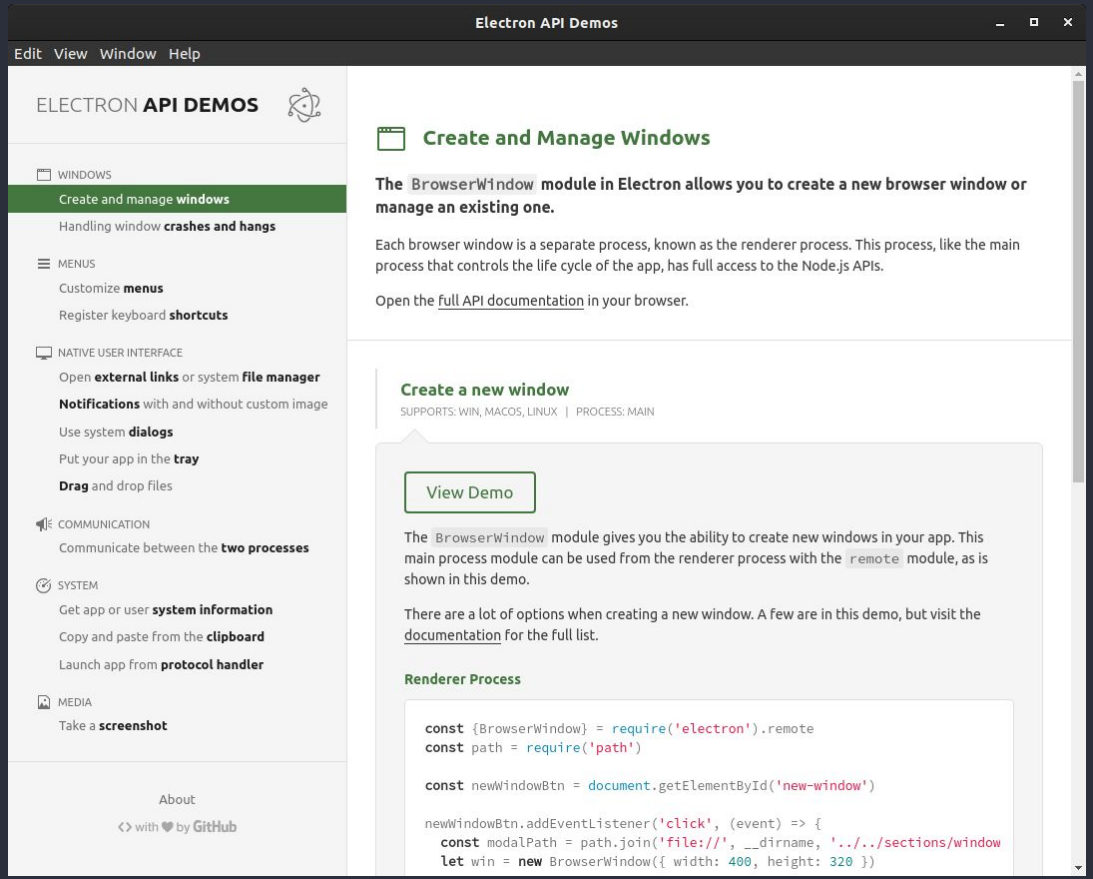
main.js (3/3)

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8">
    <title>Hello World!</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <!-- All of the Node.js APIs are available in this renderer process. -->
    We are using Node.js <script>document.write(process.versions.node)</script>,
    Chromium <script>document.write(process.versions.chrome)</script>,
    and Electron <script>document.write(process.versions.electron)</script>.

    <script>
      // You can also require other files to run in this process
      require('./renderer.js')
    </script>
  </body>
</html>
```

index.html

O app de demos da API Electron



Electron pra falar de Electron

E tem como ficar mais fácil ainda..

ELECTRON

FIDDLE

Maneira mais fácil de começar com Electron

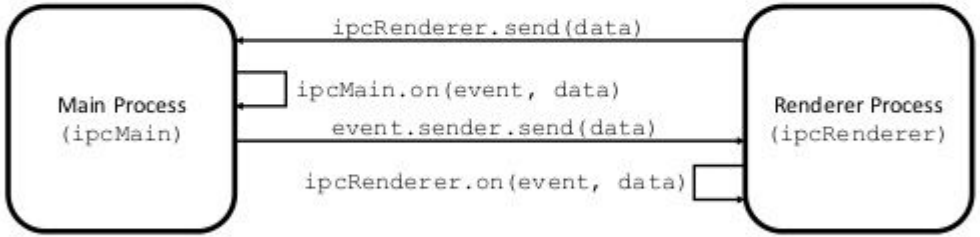
Não é um IDE - porém um excelente ponto de partida

Alt + Tab

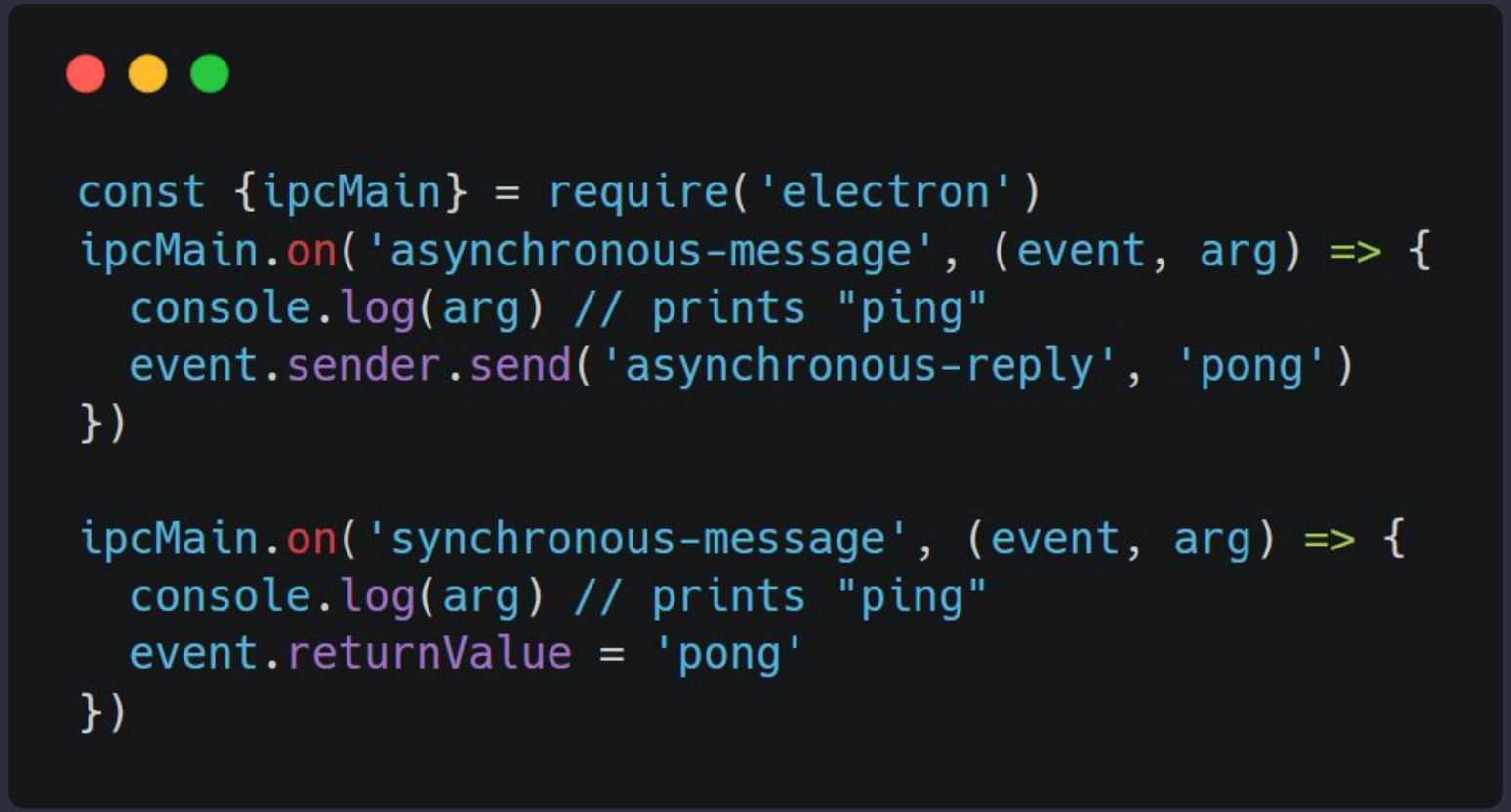
ipcMain & ipcRenderer

main.js

index.html



Troca de mensagens entre os processos



```
const {ipcMain} = require('electron')
ipcMain.on('asynchronous-message', (event, arg) => {
  console.log(arg) // prints "ping"
  event.sender.send('asynchronous-reply', 'pong')
})

ipcMain.on('synchronous-message', (event, arg) => {
  console.log(arg) // prints "ping"
  event.returnValue = 'pong'
})
```

No processo principal (Backend)

```
const {ipcRenderer} = require('electron')
console.log(ipcRenderer.sendSync('synchronous-message', 'ping')) // prints "pong"

ipcRenderer.on('asynchronous-reply', (event, arg) => {
  console.log(arg) // prints "pong"
})
ipcRenderer.send('asynchronous-message', 'ping')
```

No processo filho (Frontend)

Alt + Tab

Aplicações famosas feitas com Electron

Atom (O primeiro)

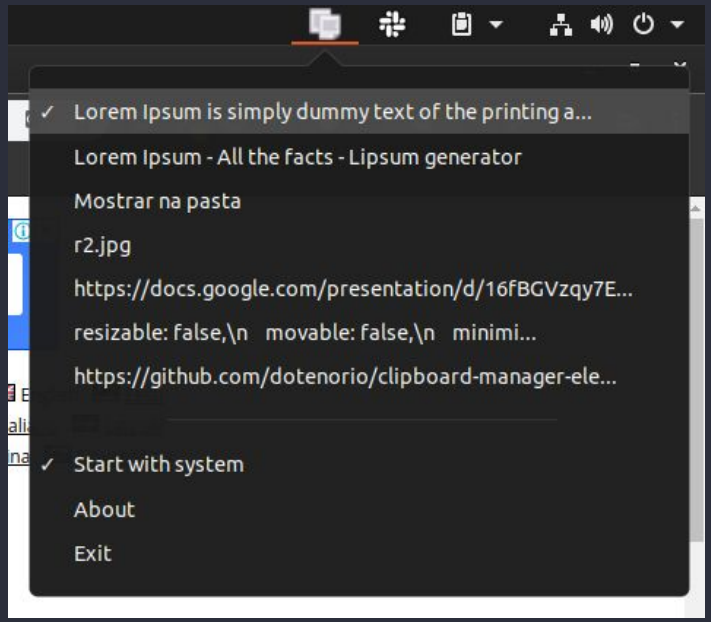
Skype

discord

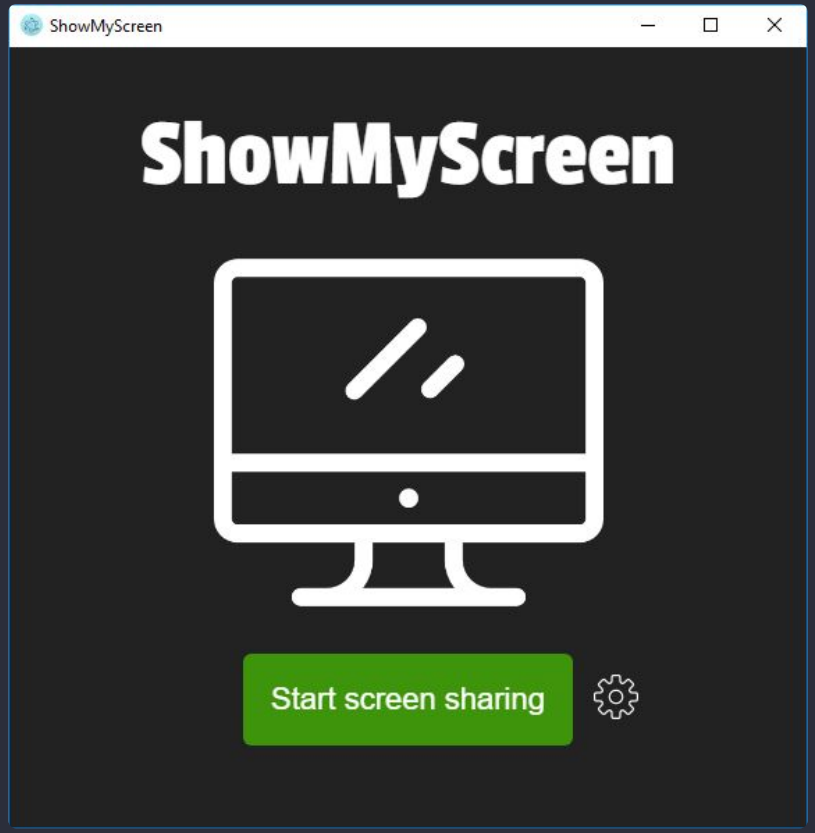
Visual Studio Code

Slack

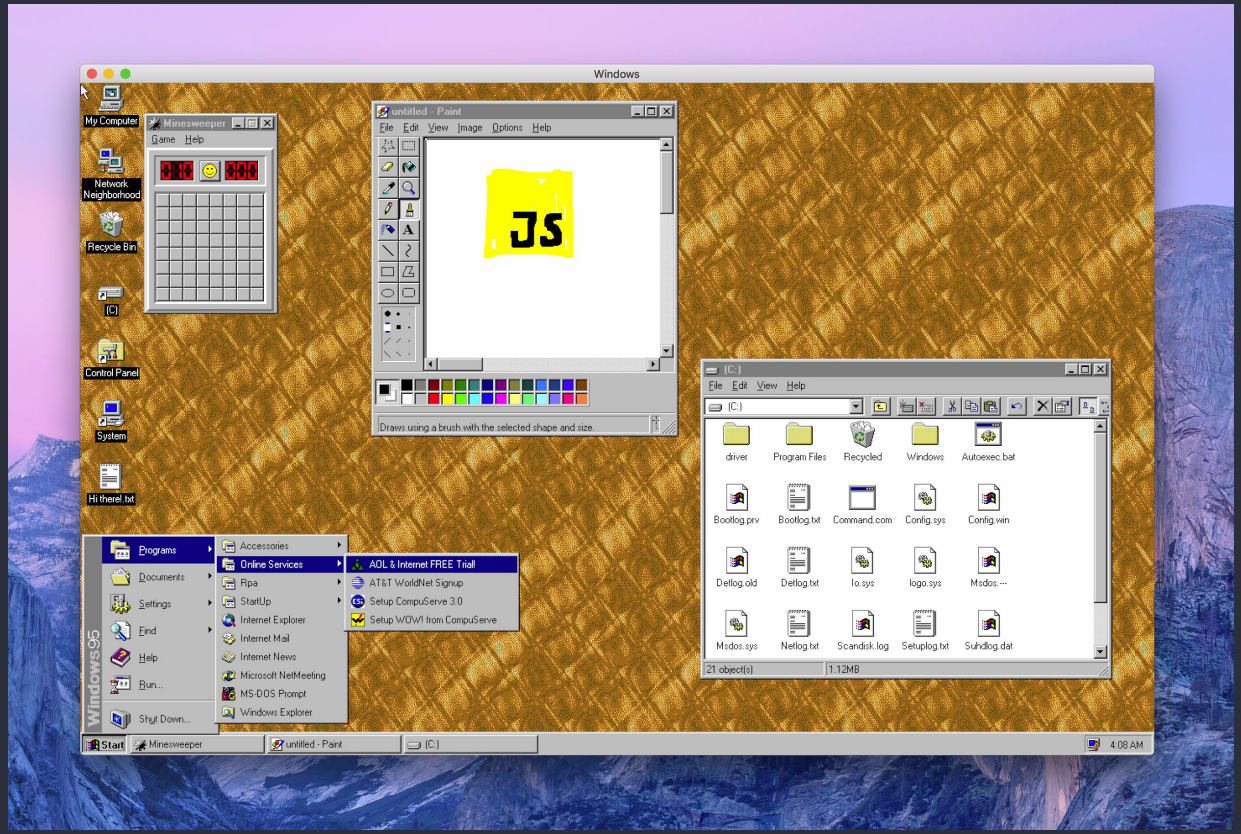
Aplicações nem tão famosas assim..



Clipboard Manager Electron
http://bit.ly/clipboard-manager-electron



ShowMyScreen
http://bit.ly/showmyscreen-electron



Windows 95
http://bit.ly/windows95-electron

Distribuição de Aplicativos

Ferramentas de empacotamento

- electron-forge
- electron-builder
- electron-packager

electron-packager

```
$ electron-packager . clipboard-manager-electron
--overwrite
--platform=win32
--arch=ia32
--prune=true
--out=dist
--version-string.CompanyName=dotenorio
--version-string.FileDescription='clipboard-manager-electron'
--version-string.ProductName='clipboard-manager-electron'
--icon=\"icons/icon.ico\"
--asar
```

Ferramentas para criar instaladores

- electron-installer-windows
- electron-installer-dmg
- electron-installer-snap
- etc..

Problemas

Desculpa, eu não queria falar sobre isso :(

- Demora para abrir
- Muito grande
- Quer uma versão para Linux? Faça o build no Linux. (O mesmo serve para Windows e macOS)
- Falta suporte a atualizações Delta
- Segurança (arquivos .asar)

Leia mais:
https://hackernoon.com/electron-the-bad-parts-2b710c491547
https://medium.com/commitlog/electron-is-cancer-b066108e6c32

Minha opinião!

Não é bom para aplicações pequenas!!

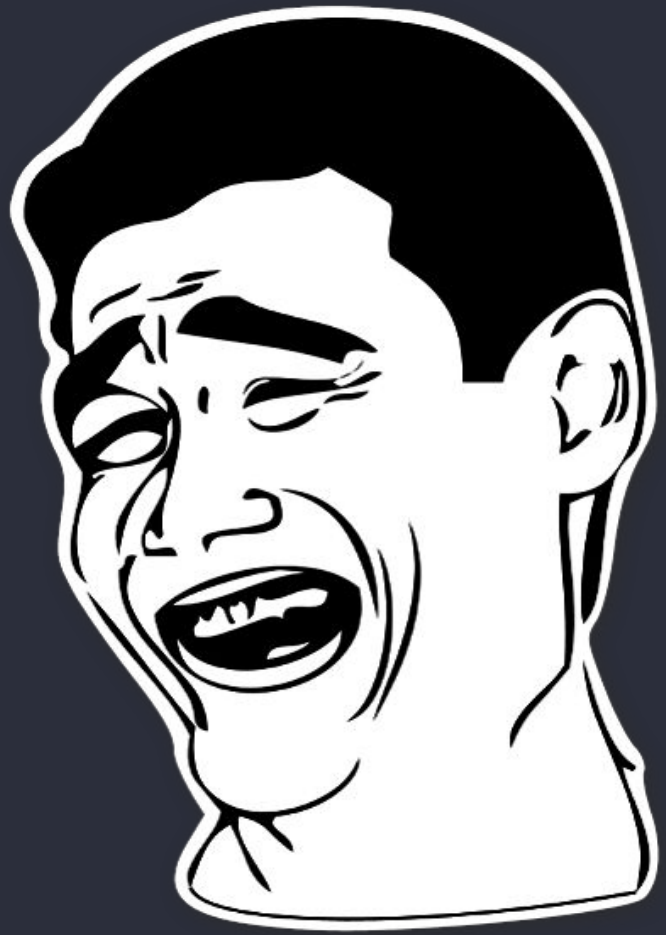
Cuidado com o ipc..

Aproveite bem as APIs para criar uma boa experiência para seu usuário.

Faça Build (CI/CD) e Update automáticos!

“Geralmente não precisamos do Electron, a gente usa porque é legal” (Zonetti)

Use, é legal mesmo...



Obrigado!
Perguntas?