**1. ECMAScript**

* **O que é:** É o **padrão** oficial da linguagem, definido pela **ECMA International**.
* **Função:** Serve como "manual de regras" para que qualquer implementação (JavaScript, JScript, etc.) siga a mesma base.
* **Importante:** ECMAScript **não é um programa ou navegador**, é apenas a especificação.
* **Exemplo:** ECMAScript diz “funções devem ser declaradas assim”, e cada navegador implementa isso na sua engine.

**2. JavaScript (de hoje)**

* **O que é:** A implementação mais famosa do ECMAScript, criada inicialmente pela **Netscape** e hoje usada em todos os navegadores modernos.
* **Inclui mais que o padrão:** Além do que o ECMAScript define, o JavaScript moderno vem com APIs extras (como DOM, Fetch API, WebSockets) que **não** estão no padrão ECMAScript.
* **Onde roda:** Navegadores, servidores (Node.js), apps híbridos etc.

**3. JScript**

* **O que é:** Uma implementação da Microsoft do ECMAScript.
* **Onde foi usado:** Principalmente no Internet Explorer antigo (anos 90 e 2000).
* **Diferença:** Seguia o padrão ECMAScript, mas tinha recursos exclusivos da Microsoft, e nem sempre era 100% compatível com JavaScript da Netscape.
* **Situação atual:** Obsoleto, já não é usado.

💡 **Resumindo numa frase:**  
ECMAScript é o **manual oficial**, JavaScript é a **versão mais popular** seguindo esse manual (com extras), e JScript foi a **versão da Microsoft**, que já morreu junto com o IE.

Se quiser, posso te montar um **quadro comparativo rápido** para visualizar essas diferenças lado a lado. Isso facilita muito a memorização. Quer que eu faça?

**You said:**

e como o chrome tem o javascript?

**ChatGPT said:**

O Google Chrome “tem” JavaScript porque ele vem com um **motor de execução** embutido, chamado **V8**.

Funciona assim:

**1. O padrão vem do ECMAScript**

* O **ECMAScript** define como a linguagem deve funcionar.
* O Chrome precisa de algo que **entenda esse padrão e execute o código**.

**2. O motor V8 no Chrome**

* **O que é:** É o mecanismo de execução de JavaScript criado pelo Google.
* **Função:** Pega o código JavaScript que você escreve e transforma em código de máquina que o processador entende, **diretamente** (sem máquina virtual intermediária, o que o torna muito rápido).
* **Localização:** Está embutido dentro do Chrome (e também no Node.js).
* **Extras:** O V8 também dá acesso a APIs do navegador, como manipular o DOM, ouvir eventos de clique, fazer requisições HTTP etc.

**3. O “JavaScript do Chrome” na prática**

* Não é um JavaScript “diferente” — é uma **implementação do ECMAScript feita pelo V8**, com APIs do navegador.
* Por isso, se você escreve alert("Oi") no Chrome, o V8 executa o código e a parte do navegador exibe a janelinha.

📌 **Resumo visual:**

scss

CopyEdit

ECMAScript (padrão) → Implementação (V8) → APIs do Navegador (DOM, Fetch, etc.)

Se quiser, posso te fazer um **diagrama mostrando como o JavaScript flui dentro do Chrome**, desde que você digita o código até ele aparecer na tela. Isso deixaria super claro. Quer que eu monte?

**You said:**

a diferneça de - jquery - angular.js - react.js - Veue - electron - ionic - cordova - e oq são e pra que servem e como foram feito por quem , difernça etc .

**ChatGPT said:**

Certo — vamos organizar isso para não virar um “macarrão de frameworks”.  
Vou explicar **o que cada um é**, **pra que serve**, **quem criou**, **como foi feito** e **a principal diferença entre eles**.

**1. jQuery**

* **O que é:** Biblioteca JavaScript (não é framework) focada em simplificar manipulação do DOM, eventos e AJAX.
* **Pra que serve:** Facilitar tarefas que, na época (2006), eram complicadas devido a diferenças entre navegadores.
* **Quem criou:** John Resig.
* **Como foi feito:** Escrito em JavaScript puro, abstraindo diferenças de APIs do navegador.
* **Situação hoje:** Pouco usado em projetos novos — JavaScript moderno e APIs nativas já fazem quase tudo que o jQuery fazia.
* **Palavra-chave:** **Facilidade e compatibilidade**.

**2. AngularJS (Angular 1.x)**

* **O que é:** Framework JavaScript para criar aplicações web **SPA** (*Single Page Applications*).
* **Pra que serve:** Construir front-ends complexos com **data binding** (ligação entre dados e interface) e injeção de dependência.
* **Quem criou:** Google (2009).
* **Como foi feito:** JavaScript + arquitetura MVC.
* **Situação hoje:** Substituído pelo **Angular** moderno (reescrito em TypeScript).
* **Palavra-chave:** **Estrutura completa para apps web**.

**3. React.js**

* **O que é:** Biblioteca JavaScript para criar interfaces de usuário.
* **Pra que serve:** Componentizar a UI, atualizando-a de forma eficiente com o **Virtual DOM**.
* **Quem criou:** Facebook (2013).
* **Como foi feito:** JavaScript + JSX (sintaxe parecida com HTML dentro do JS).
* **Situação hoje:** Muito usado, enorme ecossistema.
* **Palavra-chave:** **Componentes e velocidade**.

**4. Vue.js**

* **O que é:** Framework progressivo para criar interfaces.
* **Pra que serve:** Unir o melhor do AngularJS (data binding) e do React (componentização) com mais simplicidade.
* **Quem criou:** Evan You (ex-Google) em 2014.
* **Como foi feito:** JavaScript + reatividade declarativa.
* **Situação hoje:** Popular em projetos novos, fácil de aprender.
* **Palavra-chave:** **Simplicidade e flexibilidade**.

**5. Electron**

* **O que é:** Framework para criar **aplicações desktop** usando JavaScript, HTML e CSS.
* **Pra que serve:** Permitir que devs web criem programas para Windows, macOS e Linux.
* **Quem criou:** GitHub (2013).
* **Como foi feito:** Combina o **Chromium** (para renderizar a interface) e **Node.js** (para acessar o sistema).
* **Exemplos famosos:** VS Code, Discord, Slack.
* **Palavra-chave:** **Web virando desktop**.

**6. Ionic**

* **O que é:** Framework para criar **apps móveis híbridos** usando tecnologias web.
* **Pra que serve:** Criar aplicativos para Android e iOS com uma base de código HTML, CSS, JS.
* **Quem criou:** Drifty Co. (2013).
* **Como foi feito:** Baseado em Angular (versões antigas) e hoje suporta React e Vue também.
* **Palavra-chave:** **App mobile via web**.

**7. Apache Cordova**

* **O que é:** Plataforma que empacota aplicações web em **apps móveis** nativos.
* **Pra que serve:** Dá acesso a APIs nativas (GPS, câmera) via JavaScript.
* **Quem criou:** Nitobi (2011), depois comprado pela Adobe e doado à Apache Foundation.
* **Como foi feito:** WebView + ponte JavaScript ↔ código nativo.
* **Palavra-chave:** **Empacotador de apps web**.

**📊 Diferença resumida:**

| **Tecnologia** | **Tipo** | **Plataforma alvo** | **Criador** | **Ano** | **Palavra-chave** |
| --- | --- | --- | --- | --- | --- |
| jQuery | Biblioteca JS | Web | John Resig | 2006 | Facilidade DOM |
| AngularJS | Framework JS | Web (SPA) | Google | 2009 | Estrutura completa |
| React.js | Biblioteca JS | Web | Facebook | 2013 | Componentes rápidos |
| Vue.js | Framework JS | Web | Evan You | 2014 | Simples e progressivo |
| Electron | Framework | Desktop | GitHub | 2013 | Web → Desktop |
| Ionic | Framework híbrido | Mobile | Drifty Co. | 2013 | Web → Mobile híbrido |
| Cordova | Plataforma híbrida | Mobile | Nitobi/Adobe | 2011 | Empacotar apps web |