JQuery

jQuery is a fast, small, and feature-rich JavaScript library. It makes things like HTML document traversal and manipulation, event handling, animation, and Ajax much simpler with an easy-to-use API that works across a multitude of browsers. With a combination of versatility and extensibility, jQuery has changed the way that millions of people write JavaScript.

What is the status of JQuery now?

While jQuery is still widely used and considered a relevant library in web development, it is generally considered a "legacy" framework, meaning its active development has slowed down significantly as newer, more advanced JavaScript frameworks like React, Angular, and Vue.js have gained popularity; however, it remains a valuable tool for simpler projects or when working with older codebases due to its established functionality and large community support.

Key points about jQuery's current status:

Active development:

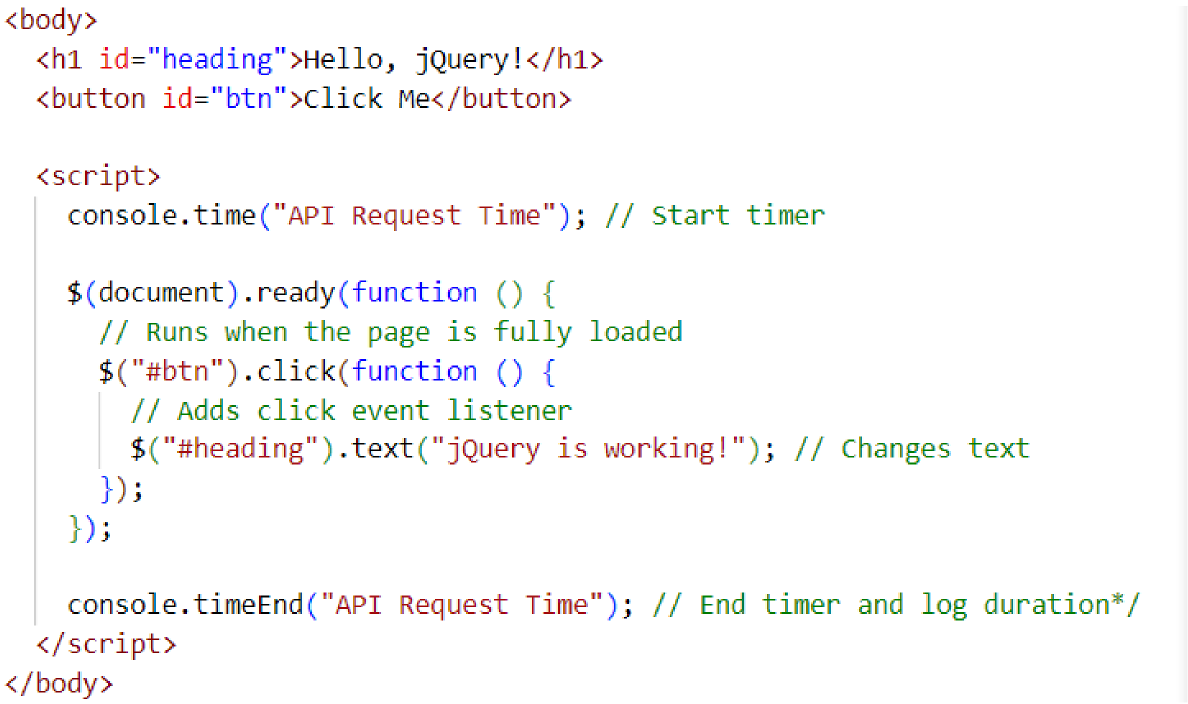
While still maintained, new feature development in jQuery is much slower compared to newer frameworks.

Community usage:

A large portion of developers still use jQuery, especially for older projects, due to its familiarity and extensive documentation.

Modern alternatives:

Developers often choose newer frameworks like React or Vue.js for complex web applications, which offer more advanced features and a different approach to building user interfaces.



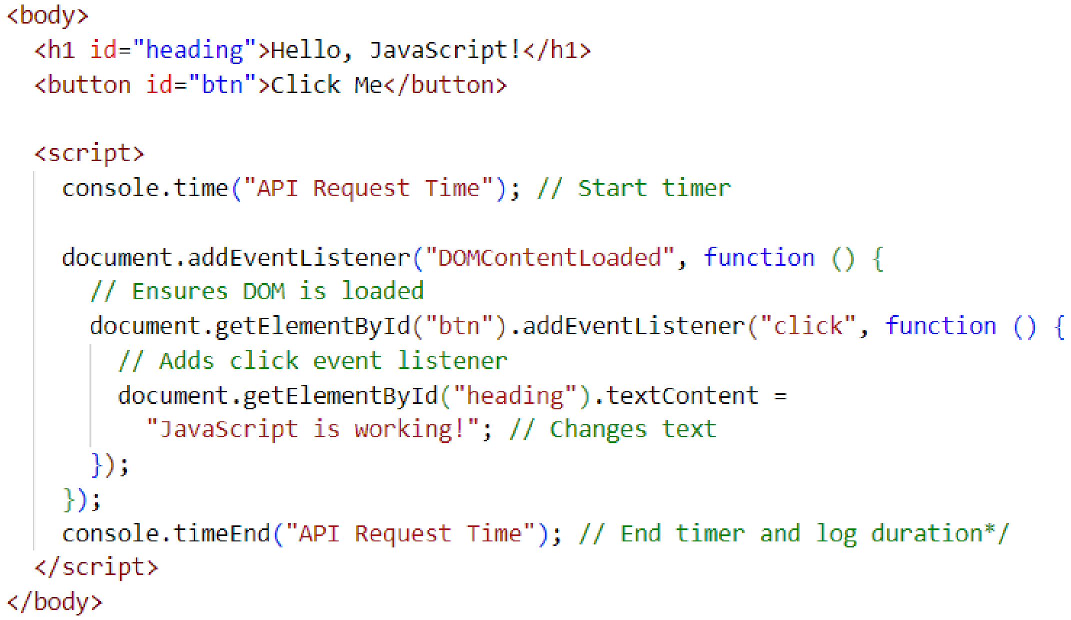
```
<body>
  <h1 id="heading">Hello, jQuery!</h1>
  <button id="btn">Click Me</button>

  <script>
    console.time("API Request Time"); // Start timer

    $(document).ready(function () {
      // Runs when the page is fully loaded
      $("#btn").click(function () {
        // Adds click event listener
        $("#heading").text("jQuery is working!"); // Changes text
      });
    });

    console.timeEnd("API Request Time"); // End timer and log duration*/
  </script>
</body>
```

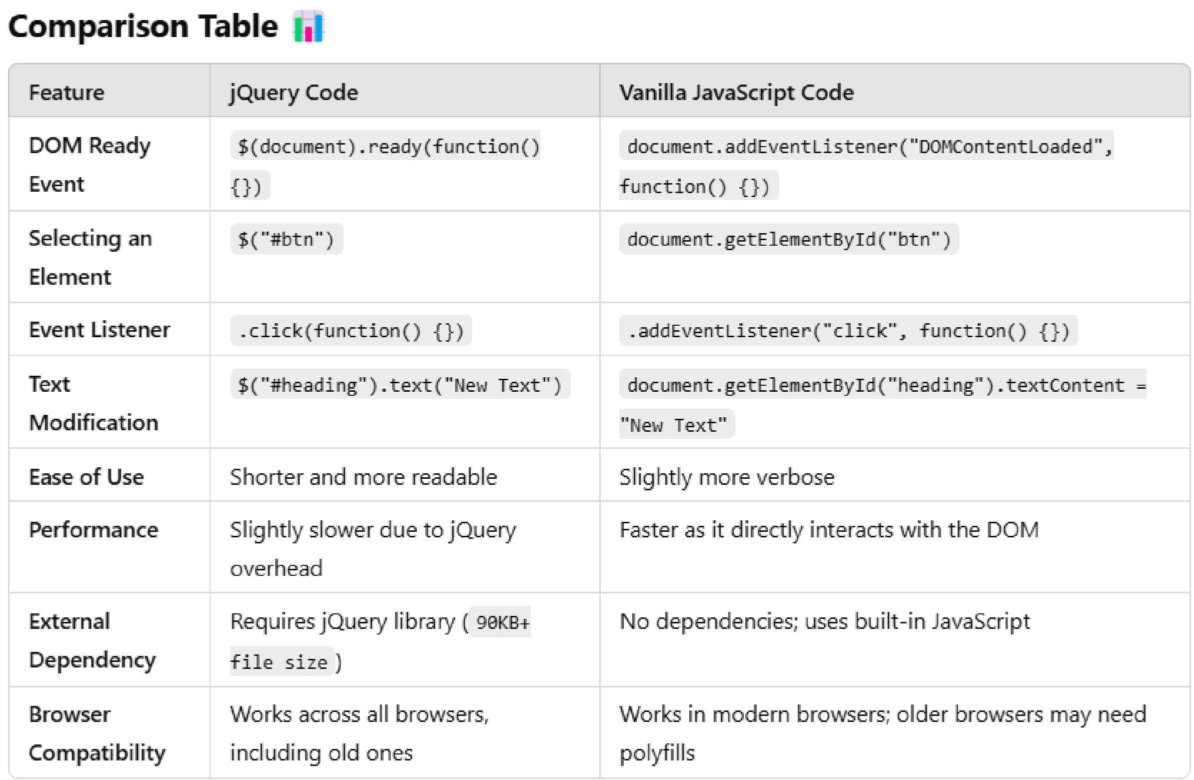
Vanilla JS



```
<body>
  <h1 id="heading">Hello, JavaScript!</h1>
  <button id="btn">Click Me</button>

  <script>
    console.time("API Request Time"); // Start timer

    document.addEventListener("DOMContentLoaded", function () {
      // Ensures DOM is loaded
      document.getElementById("btn").addEventListener("click", function () {
        // Adds click event listener
        document.getElementById("heading").textContent =
          "JavaScript is working!"; // Changes text
      });
    });
    console.timeEnd("API Request Time"); // End timer and log duration*/
  </script>
</body>
```



Comparison Table 📊

Feature	jQuery Code	Vanilla JavaScript Code
DOM Ready Event	`$(document).ready(function() {})`	`document.addEventListener("DOMContentLoaded", function() {})`
Selecting an Element	`$("#btn")`	`document.getElementById("btn")`
Event Listener	`.click(function() {})`	`.addEventListener("click", function() {})`
Text Modification	`$("#heading").text("New Text")`	`document.getElementById("heading").textContent = "New Text"`
Ease of Use	Shorter and more readable	Slightly more verbose
Performance	Slightly slower due to jQuery overhead	Faster as it directly interacts with the DOM
External Dependency	Requires jQuery library (`90KB+ file size`)	No dependencies; uses built-in JavaScript
Browser Compatibility	Works across all browsers, including old ones	Works in modern browsers; older browsers may need polyfills

Key Takeaways:

jQuery is easier & more concise, especially for beginners or older projects.

Vanilla JavaScript is faster & lightweight, ideal for modern web apps.

If you don't need jQuery for other reasons, using pure JavaScript is recommended for performance.

🚀 Modern best practice: Use Vanilla JS unless you're working with an existing jQuery-based project.

ReactJS

What is ReactJS?

ReactJS is a JavaScript library for building fast, interactive, and scalable user interfaces, mainly for single-page applications (SPAs). It was developed by Facebook (now Meta) and is widely used by companies like Instagram, Netflix, and Airbnb.

Why Use React?

✅ **Component-Based Architecture** → Build reusable UI components.
✅ **Virtual DOM** → Faster updates, better performance.

✅ **Declarative UI** → Write less code, make UI updates easier.
✅ **Huge Ecosystem** → Tons of libraries, tools, and community support.
✅ **Used by Industry Giants** → Facebook, Instagram, Netflix, Airbnb, etc.

Getting started with React

1. Install React
2. Create React Project
3. Run the Project
4. Work with Component

Key React Concepts

Components → Small, reusable UI blocks.
SX (JavaScript XML) → Write HTML-like syntax inside JavaScript.

Props → Pass data between components.
State → Manage dynamic data inside components.
Hooks (useState, useEffect) → Handle state and side effects.

Why React is Popular?

Fast rendering with Virtual DOM.

Modular & Reusable components.

Great for building interactive UIs.

Strong community & ecosystem (Next.js, Redux, Tailwind, etc.).