Advanced Research Paper

jQuery

Date**: March 10 2024**
Revision**: v16**

jQuery:

From Fundamentals to Advanced Applications in Web Development

1. Introduction: The Enduring Relevance of jQuery

jQuery, a widely adopted JavaScript library, was designed to streamline interactions with the Document Object Model (DOM), simplify event handling, facilitate CSS animations, and enhance AJAX implementations.[1] Its core tenet, "write less, do more" [2], resonated deeply with developers, leading to its rapid integration into countless web projects. Upon its introduction, jQuery addressed the prevalent complexities of cross-browser compatibility and the verbosity associated with standard JavaScript DOM manipulation.[4] By providing an intuitive and efficient API, it significantly lowered the barrier to entry for creating dynamic and interactive web pages. This ease of use, coupled with its ability to abstract away the inconsistencies between different browsers, quickly established jQuery as a fundamental tool in the web development landscape.

Despite the emergence of numerous contemporary JavaScript frameworks and libraries, jQuery continues to underpin a substantial portion of the internet.[2] This enduring presence speaks to its strong foundational design and the vast ecosystem of websites and plugins that rely on its functionality. While newer technologies offer compelling advantages for specific use cases, jQuery's legacy and its ongoing utility warrant a comprehensive exploration that extends beyond basic tutorials. This white paper aims to provide an in-depth analysis of jQuery, delving into its advanced features, historical context, performance considerations, and its sustained relevance in the ever-evolving domain of web development. By examining these aspects, this report seeks to offer a sophisticated understanding of jQuery's capabilities for

intermediate to advanced web developers and technical professionals.

2. **A Look Back: The History and Evolution of jQuery**

Before the advent of jQuery, web development was often characterized by significant challenges, particularly in ensuring consistent functionality across different web browsers.[4] Developers frequently encountered discrepancies in how JavaScript was interpreted and how the DOM was structured, leading to complex and often duplicated code to achieve cross-browser compatibility. While other JavaScript libraries existed, such as Prototype, jQuery emerged with the ambition of providing an even more refined and powerful solution.[6]

The genesis of jQuery can be traced back to January 2006, when John Resig unveiled the library at BarCamp NYC.[1] Resig's primary motivation was to alleviate the frustrations associated with writing cross-browser JavaScript and to simplify common web development tasks.[5] His vision was to make JavaScript coding more enjoyable by abstracting away the underlying complexities and providing an ergonomic API for manipulating web pages, especially concerning DOM element selection and handling the variations among browsers.[4] The development of jQuery was also influenced by Dean Edwards' earlier cssQuery library.[1]

Over the years, jQuery has undergone significant evolution, marked by several key milestones and releases. The initial stable version, 1.0, was released in August 2006.[1] A crucial development was the introduction of the highly efficient Sizzle selector engine into the core with version 1.3.[1] Recognizing the growing importance of the library, the jQuery Foundation was formally established in 2012 [4] to oversee its governance and promote open-source web technologies. In 2013, jQuery 2.0 was released, making a significant shift by dropping support for older Internet Explorer versions (6-8) to enhance performance and reduce file size.[4] Further modernization of the library occurred with the release of jQuery 3.0 in 2016.[8] The project continues to be actively maintained, with the latest stable version, 3.7.1, released in August 2023.[1]

The impact of jQuery extends far beyond its own codebase. Its innovative approach to DOM selection and manipulation, particularly through the Sizzle engine, significantly influenced the architecture of other JavaScript frameworks like YUI v3 and Dojo.[1] Furthermore, jQuery's success played a role in stimulating the creation of the standard Selectors API, which is now a fundamental part of web browsers.[1] In essence, jQuery laid the groundwork and demonstrated the power of abstraction and community-driven development, ultimately paving the way for many of the advanced

browser functionalities that developers rely on today.[9]

Year	Milestone	Description	Relevant Snippet IDs
2006	Initial Release	John Resig released jQuery at BarCamp NYC.	1
2007	jQuery UI Launched	Introduction of the user interface library for jQuery.	11
2009	Sizzle Integration	The Sizzle Selector Engine was introduced into the core.	1
2011	jQuery Foundation Formed	Formal creation of the jQuery Board and later the jQuery Foundation.	4
2013	jQuery 2.0 Released	Dropped support for IE 6-8 for performance improvements.	4
2016	jQuery 3.0 Released	Modernized codebase and dropped support for older browsers.	8
2019	OpenJS Foundation Formed	Merger of JS Foundation and Node.js Foundation, with jQuery as an impact project.	4
2023	jQuery 3.7.1 Released	Latest stable release.	1

3. **Beyond the Basics: Mastering Advanced jQuery Selectors**

jQuery's selector engine provides a powerful means of targeting specific HTML elements within a document, extending the capabilities of basic CSS selectors.[13] Mastering these advanced selectors is crucial for efficient DOM manipulation and event handling in complex web applications.

Attribute selectors allow for the selection of elements based on the presence or value of their attributes.[14] The [attribute] selector targets all elements that possess the specified attribute, regardless of its value.[14] To select elements with a particular attribute value, the syntax [attribute="value"] is used.[14] Conversely, [attribute!="value"] selects elements that either do not have the specified attribute or have it with a different value.[14] For more refined matching, selectors like [attribute^="value"] identify elements where the attribute value begins with a given string [14], while [attribute$="value"] targets those where the value ends with a specific string.[14] The [attribute*="value"] selector is even more flexible, selecting elements whose attribute value contains the specified substring anywhere within it.[14] To match elements containing a specific word within a space-separated attribute value, [attribute~="value"] is employed [14], and [attribute|="value"] selects elements where the attribute value is exactly the given string or starts with it followed by a hyphen.[14] Furthermore, multiple attribute conditions can be combined to target elements that satisfy all specified criteria, such as [attribute="value"][attribute2="value2"].[14]

Hierarchy selectors enable navigation and selection based on the relationships between elements in the DOM tree.[14] The child selector (parent > child) targets elements that are direct children of a specified parent [14], while the descendant selector (ancestor descendant) selects all elements that are descendants of a given ancestor, regardless of the depth of nesting.[14] To select the element that immediately follows another sibling, the next adjacent selector (prev + next) is used [14], and the next siblings selector (prev ~ siblings) targets all sibling elements that come after a specified element and share the same parent.[14]

jQuery also provides selectors based on content, visibility, and form properties.[14] Content filters such as :contains(text) select elements that contain the specified text [14], while :empty targets elements with no children, including text nodes.[14] The :has(selector) filter selects elements that contain at least one element matching the provided selector [14], and :parent selects elements that have at least one child node.[14] Visibility filters allow targeting of elements based on their display status, with :visible selecting elements that consume space in the document [14] and :hidden selecting those that are not visible.[14] Form selectors provide a convenient way to target specific

form controls based on their type and state, such as :input, :text, :password, :radio, :checkbox, :button, :submit, :reset, :image, :file, :enabled, :disabled, :selected, :checked, and :focus.[14]

For highly specific selection needs, jQuery allows the creation of custom selectors by extending the jQuery.expr[':'] or jQuery.expr.pseudos objects.[35] This extensibility enables developers to define selectors based on custom logic or element properties, enhancing the flexibility of jQuery's selection capabilities.

When utilizing jQuery selectors, performance is an important consideration.[40] Generally, the most efficient selectors are those that target elements directly using their unique IDs (#id).[40] Class selectors (.class) and HTML type selectors (element) are the next most efficient. When constructing more complex selectors, it is advisable to be as specific as possible on the right-hand side of the selector, which is the part that the selector engine evaluates first.[44] Overly complex selectors and the universal selector (*) should be used sparingly as they can lead to performance degradation, especially in large documents.[42] For selectors that are used multiple times, it is a best practice to cache the results in variables to avoid redundant DOM traversals.[42]

4. **Dynamic Interactions: Advanced DOM Manipulation with jQuery**

jQuery provides a comprehensive suite of methods for dynamically manipulating the structure and content of the DOM.[51] These capabilities are essential for creating interactive web applications that respond to user actions and update content without requiring full page reloads.

Adding content to the DOM can be achieved through several methods. The .append() method inserts content at the end of the selected elements, while .prepend() adds it to the beginning.[51] To insert content after or before existing elements, .after() and .before() are used, respectively.[51] Correspondingly, .appendTo() and .prependTo() perform similar actions but with a reversed syntax, inserting the selected elements into a target element.[54]

Removing elements and their content is facilitated by methods like .remove(), which removes the selected elements themselves as well as their child nodes.[51] The .empty() method, on the other hand, removes all child nodes from the selected elements, leaving the elements themselves intact.[51] The .detach() method is similar to .remove() but retains all jQuery data and event handlers associated with the removed elements, allowing for potential reinsertion later.[54]

Replacing elements and content can be done using .replaceWith(), which replaces the

selected elements with new content or elements [53], and .replaceAll(), which reverses the operation, replacing all target elements with the selected elements.[54]

Modifying the attributes of DOM elements is straightforward with jQuery. The .attr() method can be used to get or set the value of an attribute.[53] To remove an attribute, .removeAttr() is used.[56] For properties, which can sometimes differ from attributes, jQuery provides .prop() to get or set property values and .removeProp() to remove them.[56] CSS classes can be dynamically managed using .addClass(), .removeClass(), .toggleClass(), and .hasClass().[55] The HTML content of an element can be retrieved or set using .html(), while .text() is used to get or set the plain text content.[53]

jQuery also allows for the creation of deep copies of selected elements using the .clone() method [56], which duplicates the elements along with their descendants and text nodes. For manipulating the structural hierarchy of the DOM, jQuery offers methods like .wrap(), which wraps an HTML structure around each selected element; .unwrap(), which removes the parent of each selected element; .wrapInner(), which wraps an HTML structure around the content of each selected element; and .wrapAll(), which wraps a single HTML structure around all the selected elements as a group.[77]

A common advanced DOM manipulation scenario involves dynamic content loading and manipulation using AJAX.[70] jQuery's $.ajax(), $.get(), and $.post() methods simplify the process of fetching data from servers and updating DOM elements without requiring a full page reload. However, when dealing with dynamically loaded content, it is crucial to use event delegation via the .on() method to ensure that event handlers are correctly attached to elements that are added to the DOM after the initial page load.[85]

jQuery also provides tools for handling interactive lists [51] and dynamically generated form elements [101], allowing developers to create complex and user-friendly interfaces.

To optimize performance when performing DOM manipulations, it is essential to minimize the number of direct DOM operations.[42] This can be achieved by caching frequently accessed jQuery objects in variables, performing multiple DOM changes in a single operation, and using method chaining to write cleaner and more efficient code.[108]

5. **Event Handling Expertise: Advanced Techniques in jQuery**

Efficient event handling is paramount in creating interactive and responsive web applications. jQuery offers advanced techniques that go beyond basic event binding, providing developers with greater control and flexibility.

Event delegation is a powerful pattern for handling events on dynamically generated content or large numbers of elements.[85] Instead of attaching event listeners directly to each individual element, a single listener is attached to a parent element that exists when the page initially loads. This parent element then listens for events that bubble up from its descendants. By using the .on() method with a selector as its second argument, jQuery can efficiently determine if the event originated from a target element matching the selector, even if that element was added to the DOM dynamically.[110] This approach significantly reduces memory consumption and improves performance, especially in scenarios with numerous interactive elements.

For more specialized application logic, jQuery allows the creation and triggering of custom events.[13] Custom events can be defined and bound to elements using the .on() method, just like native DOM events. These custom events can then be triggered using .trigger() or .triggerHandler(). This mechanism enables different parts of an application to communicate and react to specific application states or user interactions in a decoupled manner.

Namespaced events provide a way to manage event handlers more effectively, particularly when multiple handlers are attached to the same event type on an element.[13] By appending a namespace to the event type when binding an event (e.g., click.myNamespace), developers can later unbind or trigger only those specific handlers using the namespace with the .off() or .trigger() methods (e.g., .off('click.myNamespace')). This helps prevent unintended side effects and makes event management in complex applications more manageable.

The jQuery event object [13] is a fundamental aspect of event handling. When an event occurs, jQuery creates an event object containing various properties and methods that provide information about the event. Key properties include target, which refers to the element that triggered the event, and currentTarget, which refers to the element to which the event listener is attached. Methods like preventDefault() can be used to stop the default action of an event (e.g., preventing a link from navigating), and stopPropagation() can be used to prevent the event from bubbling up the DOM tree to parent elements. Understanding and utilizing these properties and methods allows for fine-grained control over event flow and behavior.

6. **Asynchronous Communication: Mastering AJAX with jQuery**

jQuery's AJAX capabilities provide a powerful and simplified way to perform asynchronous HTTP requests, enabling web applications to communicate with servers in the background without disrupting the user experience.[4] The core of jQuery's AJAX functionality lies in the $.ajax() method [87], which offers a wide range of configuration options to control every aspect of the request. These options include url to specify the target endpoint, type to define the HTTP method (GET, POST, etc.), data to send to the server, dataType to indicate the expected response format, and callback functions like success, error, and complete to handle the different stages of the request. Other options such as async to control asynchronous behavior, cache to manage browser caching, and headers to set custom HTTP headers provide further flexibility.

For common AJAX operations, jQuery offers shorthand methods that simplify the syntax. $.get() is used to perform GET requests [92], $.post() for POST requests [92], and $.getJSON() for GET requests where the expected response is in JSON format.[92]

Once data is received from an AJAX request, jQuery provides mechanisms for manipulating this data.[70] The success callback function receives the response data, which can be in various formats such as JSON, XML, HTML, or plain text. Developers can then use jQuery's DOM manipulation methods to update the web page based on this data. Handling different data types often involves parsing the response (e.g., using JSON.parse() for JSON data or jQuery's XML manipulation methods for XML) before updating the DOM.

Effective error handling is crucial for robust AJAX implementations.[87] The error callback function of $.ajax() is executed if the request fails, providing information about the error. Alternatively, the .fail() method of the Deferred object returned by $.ajax() can be used for error handling.

jQuery's AJAX methods return a Deferred object [1], which represents the eventual outcome of an asynchronous operation. Deferred objects provide a structured way to manage asynchronous tasks through methods like .promise() to get a read-only version of the Deferred, .done() for success callbacks, .fail() for error callbacks, .then() for more flexible callback chaining, and .when() to handle multiple Deferred objects. This approach helps to avoid the complexities of nested callbacks and makes asynchronous code more organized and maintainable.

7. **Creating Rich Interfaces: Advanced Effects and Animations in jQuery**

jQuery's effects and animation capabilities allow developers to add visual enhancements and dynamic interactions to web pages. The .animate() method [129] is a cornerstone of custom animations, enabling the manipulation of virtually any numeric CSS property over a specified duration. This method offers granular control over the animation process, allowing for the creation of a wide range of visual effects beyond the basic show, hide, and fade methods.

For more intricate animation sequences, the .animate() method provides step and progress callbacks.[129] The step callback function is executed after each step of the animation for every animated property, providing an opportunity to modify the animation in real-time or perform intermediate actions. The progress callback, introduced in later jQuery versions, is called after each step of the animation but only once per animated element, regardless of the number of animated properties.

Managing complex animation sequences can be achieved using jQuery's queue mechanism.[141] The .queue() method allows functions to be added to an animation queue associated with an element, ensuring that animations and other functions are executed sequentially. The .dequeue() method is used to move to the next item in the queue, and .clearQueue() can be used to remove any remaining queued items. Deferred objects also play a significant role in managing sequential animations [120], providing a promise-based approach to orchestrate complex animation sequences and handle their completion.

For simpler animation sequences, jQuery allows for the chaining of effects [4], where multiple animation methods are called one after another on the same set of elements, resulting in a sequential execution of the effects.

Easing functions [129] control the rate of change of the animated properties over time, allowing for the creation of different animation styles. jQuery core provides two easing functions: linear for a constant speed and swing (the default) for an effect that accelerates at the beginning and decelerates towards the end. Additional easing functions can be incorporated using plugins like jQuery UI.[155]

In some cases, integrating jQuery animations with CSS transitions [159] can offer performance benefits, as CSS transitions can be hardware-accelerated by the browser, especially for properties like transforms and opacity. jQuery also provides methods to control animations, such as .stop() to immediately halt an animation [135], .finish() to complete all queued animations immediately [170], and .delay() to pause the execution of subsequent animations in the queue.[173]

8. **Performance Matters: Optimizing jQuery for Scalability**

In large-scale web applications, optimizing jQuery performance is crucial for ensuring a smooth and responsive user experience.[42] Several best practices can be employed to enhance the efficiency of jQuery code.

Writing efficient jQuery selectors is a primary area for optimization.[42] Prioritizing the use of ID selectors whenever possible is recommended due to their superior performance.[42] When using class or tag selectors, being specific on the rightmost part of the selector can help the selector engine narrow down the search more quickly.[44] Complex selectors should be avoided when simpler alternatives exist, and the universal selector (*) should be used sparingly as it can be computationally expensive.[42] Providing a context to selectors using the optional second argument can also improve performance by limiting the scope of the search.[42]

Minimizing DOM manipulations is another key strategy for improving performance.[42] Batching DOM changes together rather than performing them individually can reduce the number of browser reflows and repaints, leading to better responsiveness.[43] When adding multiple elements to the DOM, it is often more efficient to build the HTML string using concatenation or template literals and then insert it into the DOM in a single operation.[42] For intensive manipulations on an element, detaching it from the DOM, performing the changes, and then reattaching it can also yield performance benefits.[45]

Caching frequently used jQuery objects in variables is a fundamental optimization technique to avoid redundant DOM traversals.[42] By storing the result of a selector in a variable, subsequent operations on the same elements can be performed without querying the DOM again.

Leveraging event delegation is an efficient way to handle events, especially for large lists or dynamically added elements.[43] Attaching a single event listener to a parent element instead of attaching individual listeners to many child elements can significantly reduce the number of event handlers in the DOM, improving performance and memory usage.

In some cases, using native JavaScript methods for simple DOM operations can be more performant than relying solely on jQuery's abstractions.[43] For tasks like selecting elements by ID or class, document.getElementById() or document.querySelector()

might offer a slight performance advantage due to the reduced overhead.

Finally, for complex or performance-critical animations, considering the use of more specialized animation libraries like Velocity.js [174] or GSAP [175] can be beneficial. These libraries are often optimized for high-performance animations and may offer more advanced features and better performance than jQuery's core animation methods, especially for hardware-accelerated effects.

9. **jQuery in the Modern Era: Its Role and Integration**

The JavaScript ecosystem has undergone significant transformation since the introduction of jQuery, with the rise of modern frameworks like React [119], Angular [177], and Vue.js.[177] These frameworks often promote a declarative programming paradigm [4] and provide comprehensive solutions for building complex single-page applications (SPAs) with features like component-based architecture, virtual DOM, and state management.

In the context of this modern landscape, jQuery presents both strengths and weaknesses.[4] Its strengths include its ease of use, which makes it an excellent choice for developers who need to quickly add interactivity to web pages.[9] jQuery also boasts a vast ecosystem of plugins that provide solutions for a wide array of common web development tasks.[2] Furthermore, it continues to effectively handle browser inconsistencies, simplifying cross-browser development.[4] However, its imperative nature, where developers explicitly instruct the browser how to manipulate the DOM, contrasts with the declarative approach of modern frameworks.[4] For complex SPAs, jQuery's lack of built-in support for robust state management can be a limitation.[177] In modern browsers, the performance overhead of jQuery's abstractions might also be a consideration for highly performant applications.[43]

While integrating jQuery with frameworks like React, Angular, and Vue.js is technically possible, it is generally discouraged.[177] These frameworks have their own efficient mechanisms for DOM manipulation and state management, and introducing jQuery can lead to conflicts and potential performance issues due to the differing paradigms. However, jQuery might still be encountered within these frameworks when working with legacy components or when a specific jQuery plugin offers unique functionality that is not readily available within the framework's ecosystem.

Despite the dominance of modern frameworks for new, large-scale applications, jQuery remains a valuable and efficient tool in several scenarios in 2024.[4] For simpler websites or when adding interactive elements to static HTML pages, jQuery's ease of use and concise syntax can lead to faster development times compared to writing

verbose vanilla JavaScript.[9] It is also an indispensable tool for maintaining and updating legacy codebases that were built using jQuery. The vast library of jQuery plugins can still be beneficial for quickly implementing specific features without the need for a full framework.[186] In situations where a full-fledged framework might be overkill for the project's complexity, jQuery provides a lightweight and effective alternative. Indeed, jQuery often simplifies many common JavaScript tasks, requiring significantly less code to achieve the same results as with plain JavaScript.[9]

10. **Conclusion: The Enduring Power and Flexibility of jQuery**

jQuery stands as a testament to the power of well-designed JavaScript libraries, having significantly shaped the landscape of web development since its inception.[2] Its key features, including a simplified approach to DOM manipulation, robust cross-browser compatibility, a powerful selector engine, an extensive range of effects and animations, and versatile AJAX capabilities, have empowered developers to create more interactive and dynamic websites with greater ease.[1] By lowering the barrier to entry for front-end development and abstracting away many of the underlying complexities of browser inconsistencies, jQuery has made significant contributions to the evolution of the web.[4]

While the web development ecosystem continues to evolve with the emergence of modern frameworks and libraries, jQuery's legacy and its core strengths ensure its continued relevance in 2024 for a wide range of projects.[9] Its influence is undeniable, with many of its fundamental principles and patterns having been adopted by newer technologies.[8] For developers working on simpler websites, maintaining legacy code, or seeking to leverage its vast plugin ecosystem, jQuery remains a powerful and flexible tool. While it may not be the default choice for all new, complex applications, its enduring presence and the vast number of websites it powers underscore its lasting impact and ongoing utility in the diverse landscape of web development.

Works cited

1. jQuery - Wikipedia, accessed May 2, 2025, https://en.wikipedia.org/wiki/JQuery
2. What Is jQuery? A Look At the Web's Most-Used JavaScript Library - Kinsta, accessed May 2, 2025, https://kinsta.com/knowledgebase/what-is-jquery/
3. jQuery: Web Development Explained - Netguru, accessed May 2, 2025, https://www.netguru.com/glossary/j-query
4. The history and legacy of jQuery - LogRocket Blog, accessed May 2, 2025, https://blog.logrocket.com/the-history-and-legacy-of-jquery/
5. John Resig on building jQuery (article) - Khan Academy, accessed May 2, 2025,

https://www.khanacademy.org/computing/computer-programming/html-js-jquery/jquery-dom-access/a/history-of-jquery
6. John Resig: Building JQuery - IEEE Computer Society, accessed May 2, 2025, https://www.computer.org/csdl/magazine/co/2015/05/mco2015050007/13rRUxASu4u
7. en.wikipedia.org, accessed May 2, 2025, https://en.wikipedia.org/wiki/JQuery#:~:text=as%20a%20plugin.-,History,being%20led%20by%20Richard%20Gibson).
8. Software Innovation: JQuery - The Tiny Library That Transformed The Web | Build5Nines, accessed May 2, 2025, https://build5nines.com/software-innovation-jquery-the-tiny-library-that-transformed-the-web/
9. jQuery's Role in Modern Web Development: Beginnings, 2024, and Beyond, accessed May 2, 2025, https://dev.to/wendyver/jquerys-role-in-modern-web-development-beginnings-2024-and-beyond-1223
10. jQuery - Simple English Wikipedia, the free encyclopedia, accessed May 2, 2025, https://simple.wikipedia.org/wiki/JQuery
11. History | jQuery Foundation, accessed May 2, 2025, https://jquery.org/history/
12. The History of jQuery: Transforming Web Development - Anirban Das, accessed May 2, 2025, https://www.anirbandas.in/blog/the-history-of-jquery-transforming-web-development
13. jQuery - Learn to Code Advanced HTML & CSS, accessed May 2, 2025, https://learn.shayhowe.com/advanced-html-css/jquery/
14. Selectors | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/category/selectors/
15. Complete List of jQuery Selectors - Tutorial Republic, accessed May 2, 2025, https://www.tutorialrepublic.com/jquery-reference/jquery-selectors.php
16. jQuery [attribute] Selector - GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/jquery-attribute-selector/
17. Attribute Equals Selector [name="value"] - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/attribute-equals-selector/
18. Category: Attribute - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/category/selectors/attribute-selectors/
19. Attribute Contains Selector [name*="value"] - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/attribute-contains-selector/
20. Attribute selectors - Learn web development | MDN, accessed May 2, 2025, https://developer.mozilla.org/en-US/docs/Learn_web_development/Core/Styling_basics/Attribute_selectors
21. A Comprehensive Look at jQuery Selectors - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/comprehensive-jquery-selectors/
22. What is the difference direct descendent (>) vs. descendant in jQuery selectors?, accessed May 2, 2025, https://stackoverflow.com/questions/10223143/what-is-the-difference-direct-des

cendent-vs-descendant-in-jquery-selectors
23. Child Selector (“parent > child”) - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/child-selector/
24. Jquery Selectors Use Cases - Scaler Topics, accessed May 2, 2025, https://www.scaler.com/topics/jquery/selector-in-jquery/
25. Descendant Selector (“ancestor descendant”) - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/descendant-selector/
26. .children() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/children/
27. jQuery Descendant ('ancestor descendant') Selector, accessed May 2, 2025, https://learnjavascript.co.uk/jq/reference/selectors/descendant.html
28. Explain advance selector used in Jquery? - GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/explain-advance-selector-used-in-jquery/
29. Category: Content Filter - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/category/selectors/content-filter-selector/
30. jQuery Filters: Selecting, Manipulating, and Filtering DOM Elements - C# Corner, accessed May 2, 2025, https://www.c-sharpcorner.com/article/jquery-filters-selecting-manipulating-and-filtering-dom-elements/
31. Category: Visibility Filter - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/category/selectors/visibility-filter-selectors/
32. :visible Selector | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/visible-selector/
33. Visibility Filter - jQuery API Documentation, accessed May 2, 2025, https://api.jquery123.com/category/selectors/visibility-filter-selectors/
34. How to Select All Visible or Hidden Elements in a Page Using jQuery - Tutorial Republic, accessed May 2, 2025, https://www.tutorialrepublic.com/faq/how-to-select-all-visible-or-hidden-elements-in-a-page-using-jquery.php
35. Make Your Own Custom jQuery Selector - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/make-your-own-custom-jquery-selector/
36. jQuery Selectors - yahoo baba, accessed May 2, 2025, https://www.yahubaba.com/jquery/jquery-selectors
37. Writing custom JQuery Selectors - Code Ranch, accessed May 2, 2025, https://coderanch.com/t/121354/languages/Writing-custom-JQuery-Selectors
38. Custom jQuery selectors - Jonas Arnklint, accessed May 2, 2025, https://jonas.arnklint.com/custom-jquery-selectors
39. jQuery selectors on custom data attributes using HTML5 - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/4146502/jquery-selectors-on-custom-data-attributes-using-html5
40. Advanced Selectors used in jQuery - Universal Class, accessed May 2, 2025, https://www.universalclass.com/articles/computers/advanced-selectors-used-in-j

query.htm
41. jQuery Selector Optimizer - WalkMe Help Center, accessed May 2, 2025, https://support.walkme.com/knowledge-base/jquery-selector-optimizer/
42. 5 Tips for More Efficient jQuery Selectors - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/efficient-jquery-selectors/
43. Optimizing Performance with jQuery: Best Practices and Techniques - CloudDevs, accessed May 2, 2025, https://clouddevs.com/jquery/optimizing-performance/
44. Selector for best performance in jQuery? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/34100018/selector-for-best-performance-in-jquery
45. jQuery Performance Tips: Reducing Load Time and Enhancing Speed | Reintech media, accessed May 2, 2025, https://reintech.io/blog/jquery-performance-tips-enhancing-speed
46. Optimize Selectors - jQuery Learning Center, accessed May 2, 2025, https://learn.jquery.com/performance/optimize-selectors/
47. Good ways to improve jQuery selector performance? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/46214/good-ways-to-improve-jquery-selector-performance
48. How to Optimize jQuery Code Performance for Large-Scale Apps - Telerik.com, accessed May 2, 2025, https://www.telerik.com/blogs/how-why-optimize-jquery-code-performance-large-scale-application-development
49. jQuery Selector Performance Testing - Sparkbox, accessed May 2, 2025, https://sparkbox.com/foundry/jquery_selector_performance_testing
50. Optimize Your jQuery Selectors for Best Performance, accessed May 2, 2025, https://learningjquery.com/2017/05/optimize-your-jquery-selectors-for-best-performance-3
51. How to Efficiently Manipulate the DOM with jQuery | Savvy, accessed May 2, 2025, https://savvy.co.il/en/blog/complete-javascript-guide/efficient-dom-manipulation-with-jquery/
52. jQuery Performance Improvement - Tizen Docs, accessed May 2, 2025, https://docs.tizen.org/application/web/guides/w3c/perf-opt/jquery-performance-improvement/
53. jQuery DOM Manipulation - Tutorialspoint, accessed May 2, 2025, https://www.tutorialspoint.com/jquery/jquery-dom.htm
54. DOM manipulation using jQuery - Scaler, accessed May 2, 2025, https://www.scaler.com/topics/jquery/jquery-dom-manipulation/
55. Category: Manipulation - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/category/manipulation/
56. Manipulating Elements - jQuery Learning Center, accessed May 2, 2025, https://learn.jquery.com/using-jquery-core/manipulating-elements/

57. .html() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/html/
58. jQuery replaceWith() Method - GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/jquery-replacewith-method/
59. jQuery .empty() DOM Removal Method, accessed May 2, 2025, https://learnjavascript.co.uk/jq/reference/manipulation/empty.html
60. replaceWith() - jQuery Mobile Demos, accessed May 2, 2025, https://demos.jquerymobile.com/1.0a2/experiments/api-viewer/docs/replaceWith/index.html
61. jQuery .replaceWith() DOM Insertion, Inside Method, accessed May 2, 2025, https://learnjavascript.co.uk/jq/reference/manipulation/replacewith.html
62. JavaScript and jQuery: jQuery HTML Manipulation - Little Web Hut, accessed May 2, 2025, https://www.littlewebhut.com/javascript/jquery_html/
63. jQuery removeAttr() Method - GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/jquery-removeattr-method/
64. jQuery .removeAttr() method - server2client.com, accessed May 2, 2025, https://server2client.com/jquery1ref/removeattr.html
65. jQuery Remove Attribute Guide: Learn to Use jQuery RemoveAttr - BitDegree, accessed May 2, 2025, https://www.bitdegree.org/learn/jquery-remove-attribute
66. .attr() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/attr/
67. .removeAttr() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/removeAttr/
68. Remove Disabled Attribute Using jQuery - Tutorialspoint, accessed May 2, 2025, https://www.tutorialspoint.com/How-to-remove-disabled-attribute-using-jQuery
69. jQuery Attribute Manipulation - ScholarHat, accessed May 2, 2025, https://www.scholarhat.com/tutorial/jquery/attribute-manipulation-example
70. jQuery - Manipulating content - Culttt, accessed May 2, 2025, https://culttt.com/2012/11/21/jquery-manipulating-content
71. Cloning | Manipulating DOM Elements in jQuery - Peachpit, accessed May 2, 2025, https://www.peachpit.com/articles/article.aspx?p=2010420&seqNum=5
72. .clone() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/clone/
73. jQuery attr() and clone() Methods - Tutorialspoint, accessed May 2, 2025, https://www.tutorialspoint.com/jquery/attr-clone.htm
74. When Is jQuery's Clone Function Useful? - Tom McFarlin, accessed May 2, 2025, https://tommcfarlin.com/jquerys-clone-function/
75. jQuery clone() example - Mkyong.com, accessed May 2, 2025, https://mkyong.com/jquery/jquery-clone-example/
76. jQuery Clone Method Example - Tutorialspoint, accessed May 2, 2025, https://www.tutorialspoint.com/jqueryexamples/dom-clone.htm
77. Learn to Use jQuery .wrap(), .wrapAll() and .wrapInner() Methods - BitDegree, accessed May 2, 2025, https://www.bitdegree.org/learn/jquery-wrap

78. .wrap() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/wrap/
79. .wrapInner() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/wrapInner/
80. Difference between jQuery wrap and wrapAll - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/11946379/difference-between-jquery-wrap-and-wrapall
81. Just jQuery The Core UI - Creating Objects - I Programmer, accessed May 2, 2025, https://www.i-programmer.info/programming/207-jquery/15973-just-jquery-the-core-ui-creating-objects.html?start=3
82. Building Dynamic Content Loaders with jQuery AJAX - CloudDevs, accessed May 2, 2025, https://clouddevs.com/jquery/building-dynamic-content-loaders/
83. Dynamic Content Load using jQuery AJAX - Phppot, accessed May 2, 2025, https://phppot.com/jquery/dynamic-content-load-using-jquery-ajax/
84. Dynamic Content Loading with jQuery and AJAX | Reintech media, accessed May 2, 2025, https://reintech.io/blog/dynamic-content-loading-with-jquery-ajax
85. Why jQuery Code Doesn't Work on Dynamic Content Loaded with AJAX and How to Fix It?, accessed May 2, 2025, https://www.geeksforgeeks.org/why-jquery-code-doesnt-work-on-dynamic-content-loaded-with-ajax-and-how-to-fix-it/
86. Dynamic Content Loading with JavaScript - DEV Community, accessed May 2, 2025, https://dev.to/iamcymentho/dynamic-content-loading-with-javascript-15hh
87. jQuery.ajax(), accessed May 2, 2025, https://api.jquery.com/jQuery.ajax/
88. Manipulate ajax.dataSrc after the fact - DataTables, accessed May 2, 2025, https://datatables.net/forums/discussion/78380/manipulate-ajax-datasrc-after-the-fact
89. Element Manipulation & AJAX - jQuery 3.1.1 - BEGINNER - Skillsoft, accessed May 2, 2025, https://www.skillsoft.com/course/element-manipulation-ajax-7dbabc6b-52ce-11e7-ae20-ceb650f9130b
90. Manipulate data in jQuery AJAX success: function - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/42485129/manipulate-data-in-jquery-ajax-success-function
91. How can you manipulate the data returned by the .ajax method ? - jQuery | Forums, accessed May 2, 2025, https://forum.jquery.com/topic/how-can-you-manipulate-the-data-returned-by-the-ajax-method
92. AJAX with jQuery: A Comprehensive Guide for Dynamic Web Development - GUVI, accessed May 2, 2025, https://www.guvi.com/blog/ajax-with-jquery/
93. Master jQuery AJAX: Complete Guide to Asynchronous Requests - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/use-jquerys-ajax-function/
94. jQuery ajax() Method - GeeksforGeeks, accessed May 2, 2025,

https://www.geeksforgeeks.org/jquery-ajax-method/
95. Getting Started With jQuery - Advanced Ajax - I Programmer, accessed May 2, 2025, https://www.i-programmer.info/programming/207-jquery/8941-getting-started-with-jquery-advanced-ajax.html?start=4
96. jQuery 3.5 Tutorials - Ajax Low-Level Interface - Server Client, accessed May 2, 2025, https://server2client.com/jquery1adv/ajaxinterface.html
97. Getting Started With jQuery - Advanced Ajax - I Programmer, accessed May 2, 2025, https://www.i-programmer.info/programming/207-jquery/8941-getting-started-with-jquery-advanced-ajax.html
98. Select List Manipulation - jquery - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/23228996/select-list-manipulation
99. Controlling Lists with jQuery - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/controlling-lists-with-jquery/
100. Which alternative jQuery offers similar to List manipulation with javascript? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/8069789/which-alternative-jquery-offers-similar-to-list-manipulation-with-javascript
101. Building Dynamic Forms with jQuery Form Plugin - CloudDevs, accessed May 2, 2025, https://clouddevs.com/jquery/form-plugin/
102. Dynamically Create HTML Form Fields - jQuery Script, accessed May 2, 2025, https://www.jqueryscript.net/form/dynamic-forms-fields.html
103. Dynamically Adding Form Elements in JQuery - UDig, accessed May 2, 2025, https://www.udig.com/insights/blog/dynamically-adding-form-elements-in-jquery
104. Submit dynamically created form with jQuery - Coderanch, accessed May 2, 2025, https://coderanch.com/t/529988/languages/Submit-dynamically-created-form-jQuery
105. Create a form dynamically with jquery and submit - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/17431760/create-a-form-dynamically-with-jquery-and-submit
106. jQuery formBuilder | Drag & Drop Form Creation, accessed May 2, 2025, https://formbuilder.online/
107. Create a Form Dynamically using Dform and jQuery | GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/create-a-form-dynamically-using-dform-and-jquery/
108. Achieving Advanced DOM Manipulation with jQuery Chaining Methods - mycode.blog, accessed May 2, 2025, https://mycode.blog/navya/jquery-chaining-methods

109. Mastering jQuery Chaining for Cleaner Code | Reintech media, accessed May 2, 2025, https://reintech.io/blog/mastering-jquery-chaining-for-cleaner-code
110. A Practical Guide to jQuery Event Delegation - CloudDevs, accessed May 2, 2025, https://clouddevs.com/jquery/event-delegation/
111. Event Delegation with jQuery - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/event-delegation-with-jquery/
112. Handle Events in Dynamically Created Elements in jQuery - Tutorialspoint, accessed May 2, 2025, https://www.tutorialspoint.com/how-to-handle-events-in-dynamically-created-elements-in-jquery
113. Delegate jQuery event handlers when working with dynamic content - Phil Kurth, accessed May 2, 2025, https://philkurth.com.au/delegate-jquery-event-handlers-when-working-with-dynamic-content/
114. Event binding on dynamically created elements? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/203198/event-binding-on-dynamically-created-elements
115. jquery - Event handler not working on dynamic content - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/15090942/event-handler-not-working-on-dynamic-content
116. Mastering jQuery Events: A Comprehensive Guide | GUVI-Blogs, accessed May 2, 2025, https://www.guvi.com/blog/guide-for-jquery-event-handling/
117. Definitive way to trigger keypress events with jQuery - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/832059/definitive-way-to-trigger-keypress-events-with-jquery
118. Trigger a keypress/keydown/keyup event in JS/jQuery? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/3368578/trigger-a-keypress-keydown-keyup-event-in-js-jquery
119. Building Single-Page Applications with jQuery and AJAX - CloudDevs, accessed May 2, 2025, https://clouddevs.com/jquery/building-single-page-applications/
120. How is the deferred method in jquery important in relation to animate method ? | GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/how-is-the-deferred-method-in-jquery-important-in-relation-to-animate-method/
121. An Introduction to jQuery's Deferred Objects - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/introduction-jquery-deferred-objects/
122. Category: Deferred Object - jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/category/deferred-object/
123. jQuery.Deferred(), accessed May 2, 2025, https://api.jquery.com/jQuery.Deferred/

124. jQuery Deferred Objects - Jenkov.com, accessed May 2, 2025, https://jenkov.com/tutorials/jquery/deferred-objects.html
125. One Div animation after another using deferred object - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/35113816/one-div-animation-after-another-using-deferred-object
126. Converting between jQuery Deferred and Rx Observable - Scott Logic Blog, accessed May 2, 2025, https://blog.scottlogic.com/2011/05/10/converting-between-jquery-deferred-and-rx-observable.html
127. Make Your Own jQuery Deferreds and Promises - Rob Dodson, accessed May 2, 2025, https://robdodson.me/posts/make-your-own-jquery-deferreds-and-promises/
128. katowulf/jquery-sequence: Create sequences of asynchrous/synchronous/anything calls that will run in order using promises - GitHub, accessed May 2, 2025, https://github.com/katowulf/jquery-sequence
129. jQuery .animate() Custom Effects Method, accessed May 2, 2025, https://learnjavascript.co.uk/jq/reference/effects/animate.html
130. Mastering jQuery Animation: Elevate Web Design with Dynamic Effects - upGrad, accessed May 2, 2025, https://www.upgrad.com/tutorials/software-engineering/jquery-tutorial/jquery-animation/
131. jQuery Effect animate() Method - Tutorialspoint, accessed May 2, 2025, https://www.tutorialspoint.com/jquery/effect-animate.htm
132. Just jQuery The Core UI - Animation - I Programmer, accessed May 2, 2025, https://www.i-programmer.info/programming/207-jquery/10461-jquery-3-animation.html?start=1
133. Custom Effects with .animate() - jQuery Learning Center, accessed May 2, 2025, https://learn.jquery.com/effects/custom-effects/
134. A Guide to the jQuery animate() Method - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/guide-jquery-animate-method/
135. .animate() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/animate/
136. Using jQuery's Animate() Step Callback Function To Create Custom Animations - Ben Nadel, accessed May 2, 2025, https://www.bennadel.com/blog/1856-using-jquerys-animate-step-callback-function-to-create-custom-animations.htm
137. The jQuery animate() step callback function - Cameron McKay, accessed May 2, 2025, https://cdmckay.org/the-jquery-animate-step-callback-function
138. animate(properties, [duration], [easing], [callback]) Returns: jQuery, accessed May 2, 2025, https://demos.jquerymobile.com/1.0a1/experiments/api-viewer/docs/animate/index.html

139. step in Jquery animate function - javascript - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/27918744/step-in-jquery-animate-function
140. Using jQuery's Animate() Step Callback Function To Create Custom Animations · GitHub, accessed May 2, 2025, https://gist.github.com/bennadel/9759644
141. Animating Elements with jQuery: A Step-by-Step Tutorial | Savvy, accessed May 2, 2025, https://savvy.co.il/en/blog/complete-javascript-guide/jquery-animate-elements-tutorial/
142. jQuery 3 - Function Queues - I Programmer, accessed May 2, 2025, https://www.i-programmer.info/programming/207-jquery/10443-jquery-3-function-queues.html?start=2
143. How to run two animations simultaneously in jQuery ? | GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/how-to-run-two-animations-simultaneously-in-jquery/
144. JQuery chaining animations with different elements - ZAN KAVTASKIN, accessed May 2, 2025, http://www.zankavtaskin.com/2016/05/jquery-chaining-animations-with.html
145. Experimenting With jQuery's Queue() And Dequeue() Methods - Ben Nadel, accessed May 2, 2025, https://www.bennadel.com/blog/1864-experimenting-with-jquerys-queue-and-dequeue-methods.htm
146. How to create animations for your website using jQuery.animate() - IONOS, accessed May 2, 2025, https://www.ionos.com/digitalguide/websites/web-development/jquery-animate/
147. Animation callback functions - Dynamic Web Development with jQuery DOM and AJAX, accessed May 2, 2025, https://app.studyraid.com/en/read/12368/399285/animation-callback-functions
148. How can I animate multiple elements sequentially using jQuery? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/1218152/how-can-i-animate-multiple-elements-sequentially-using-jquery
149. Using jQuery.queue with multiple element animations? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/18499000/using-jquery-queue-with-multiple-element-animations
150. Animation queue management for jQuery - Blog - Toby Mackenzie, accessed May 2, 2025, https://www.tobymackenzie.com/blog/2011/08/02/animation-queue-management-for-jquery/
151. Queue & Dequeue Explained | jQuery Learning Center, accessed May 2, 2025, https://learn.jquery.com/effects/queue-and-dequeue-explained/
152. 6 Reasons We Still Use jQuery in 2021 - Atypic craft, accessed May 2, 2025,

https://atypiccraft.com/insights/reasons-why-we-still-use-jquery
153. What are downside and advantage of method chaining in jQuery? [closed] - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/15240221/what-are-downside-and-advantage-of-method-chaining-in-jquery
154. jquery Chaining part - 135 - YouTube, accessed May 2, 2025, https://www.youtube.com/watch?v=8ID6IHA2M3w
155. Easings - jQuery UI API Documentation, accessed May 2, 2025, https://api.jqueryui.com/easings/
156. Easings | jQuery UI 1.12 Documentation, accessed May 2, 2025, https://api.jqueryui.com/1.12/easings/
157. jQuery Easing | List of Easing Functions with Programming Example - EDUCBA, accessed May 2, 2025, https://www.educba.com/jquery-easing/
158. jQuery easing functions without using a plugin - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/5207301/jquery-easing-functions-without-using-a-plugin
159. Transitions - jQuery Mobile Demos, accessed May 2, 2025, https://demos.jquerymobile.com/1.4.5/transitions/
160. Transit - CSS transitions and transformations for jQuery - Rico Sta. Cruz, accessed May 2, 2025, https://ricostacruz.com/jquery.transit/
161. How to use jQuery to wait for the end of CSS3 transitions? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/9255279/how-to-use-jquery-to-wait-for-the-end-of-css3-transitions
162. Controlling CSS Animations and Transitions with JavaScript, accessed May 2, 2025, https://css-tricks.com/controlling-css-animations-transitions-javascript/
163. Super-smooth CSS3 transformations and transitions for jQuery - GitHub, accessed May 2, 2025, https://github.com/rstacruz/jquery.transit
164. Animated Transition Effects in CSS and jQuery - CodyHouse, accessed May 2, 2025, https://codyhouse.co/gem/animated-transition-effects
165. jQuery CSS transition animation - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/28387702/jquery-css-transition-animation
166. Animate.css | A cross-browser library of CSS animations., accessed May 2, 2025, https://animate.style/
167. 10 Awesome Techniques and Examples of Animation with jQuery - WebFX, accessed May 2, 2025, https://www.webfx.com/blog/web-design/10-awesome-techniques-and-examples-of-animation-with-jquery/
168. Using both CSS transitions and jQuery animations - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/13177648/using-both-css-transitions-and-jquery-animations

169. Weaning off of jQuery animations with CSS Transitions - Ignored By Dinosaurs, accessed May 2, 2025, https://www.ignoredbydinosaurs.com/posts/weaning-jquery-animations-css-transitions
170. .stop() | jQuery API Documentation, accessed May 2, 2025, https://api.jquery.com/stop/
171. Using jQuery Stop - CSS-Tricks, accessed May 2, 2025, https://css-tricks.com/examples/jQueryStop/
172. Using jQuery stop Method With and Without Defined Parameters - BitDegree, accessed May 2, 2025, https://www.bitdegree.org/learn/jquery-stop
173. jQuery delay() - how to stop it? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/7929266/jquery-delay-how-to-stop-it
174. Easily Improving jQuery Animations - SitePoint, accessed May 2, 2025, https://www.sitepoint.com/easily-improving-jquery-animations/
175. Best Practices for jQuery Animation Timing and Easing | Reintech media, accessed May 2, 2025, https://reintech.io/blog/jquery-animation-timing-easing-best-practices
176. Questions About Performance / Best Practices - GSAP - GreenSock, accessed May 2, 2025, https://gsap.com/community/forums/topic/23062-questions-about-performance-best-practices/
177. The Evolution of Front-End Development: jQuery vs. React - DhiWise, accessed May 2, 2025, https://www.dhiwise.com/post/jquery-vs-react-understanding-the-evolution-in-development
178. Story of jQuery, Write Less and Rest in Peace - DEV Community, accessed May 2, 2025, https://dev.to/adnanbabakan/story-of-jquery-write-less-and-rest-in-peace-1h6d
179. jQuery: A Historical Overview of the Library That Changed Web Development, accessed May 2, 2025, https://muteeb.hashnode.dev/jquery-a-historical-overview-of-the-library-that-changed-web-development
180. What is a correct way to use jQuery plugins which manipulate DOM on every page onload? (Ember 2.8.0), accessed May 2, 2025, https://discuss.emberjs.com/t/what-is-a-correct-way-to-use-jquery-plugins-which-manipulate-dom-on-every-page-onload-ember-2-8-0/11837
181. How to update DOM in Single Page Application with jQuery? - Stack Overflow, accessed May 2, 2025, https://stackoverflow.com/questions/26049849/how-to-update-dom-in-single-page-application-with-jquery
182. #44 JQuery DOM Manipulations Part 1 - YouTube, accessed May 2, 2025, https://www.youtube.com/watch?v=JFnlvhG0OgY
183. Single-Page Application with ASP.NET & jQuery Hands-On - YouTube, accessed May 2, 2025, https://m.youtube.com/watch?v=H5fGvOK9nIc&t=0s
184. Binding events to DOM for Single Page Applications with JQuery and

JavaScript, accessed May 2, 2025, https://stackoverflow.com/questions/17305003/binding-events-to-dom-for-single-page-applications-with-jquery-and-javascript
185. What Is JQuery & Why's It The Top JS Library In 2024? - Learn Enough, accessed May 2, 2025, https://www.learnenough.com/blog/what-is-jquery
186. Is Jquery relevant? : r/Frontend - Reddit, accessed May 2, 2025, https://www.reddit.com/r/Frontend/comments/10i5c2s/is_jquery_relevant/
187. Why we use jQuery in our web application ? | GeeksforGeeks, accessed May 2, 2025, https://www.geeksforgeeks.org/why-we-use-jquery-in-our-web-application/
188. No motivation to learn Jquery, is it really worth it? I love Vanilla JS - Reddit, accessed May 2, 2025, https://www.reddit.com/r/javascript/comments/8kxxo3/no_motivation_to_learn_jquery_is_it_really_worth/
189. Basic jQuery concepts for everyday development - Zipy.ai, accessed May 2, 2025, https://www.zipy.ai/blog/basic-jquery-concepts
190. Why and How we moved away from jQuery in our Web Application - DEV Community, accessed May 2, 2025, https://dev.to/devcer/why-and-how-i-moved-away-from-jquery-in-my-web-application-2of2
191. Why I'm still using jQuery in 2019 - arp242.net, accessed May 2, 2025, https://www.arp242.net/jquery.html