JQuery

JQUERY

Definition:
jQuery is a **fast, small, and feature-rich JavaScript library** that simplifies:

- HTML **DOM traversal and manipulation**
- Event handling
- Animations
- Ajax calls

It allows writing **less code** compared to vanilla JS.

Why jQuery Is Needed

- Simplifies **cross-browser compatibility**
- Easy to **select elements and manipulate DOM**
- Supports **animations and effects**
- Makes **Ajax calls** simpler
- Speeds up development for small/medium projects

Key Features

1. **DOM Manipulation** → Select, modify, or remove elements easily
2. **Event Handling** → Attach events like click, hover
3. **Ajax Support** → Simplifies HTTP requests
4. **Animations** → Fade, slide, show/hide effects
5. **Plugins** → Extend functionality with prebuilt plugins

Syntax & Example

```
<script src="https://code.jquery.com/jquery-3.7.0.min.js"></script>
<script>
  $(document).ready(function(){
    $("#btn").click(function(){
      $("#text").text("Hello, jQuery!");
      $("#text").css("color", "blue");
    });
  });
</script>

<button id="btn">Click Me</button>
<p id="text">Original Text</p>
```

- $() → jQuery selector, similar to document.querySelectorAll
- .click() → attach click event
- .text() → change text content
- .css() → modify style

Real-Life Example

- Form validation
- Toggle menus & modals
- Ajax-based content loading
- Animations like slideshows

Interview-Ready Lines

- jQuery = **JS library** for easy DOM, events, Ajax, animations
- $() → universal selector
- Reduces code complexity & **cross-browser issues**

- Popular in legacy projects; modern projects often use **React/Angular/Vue**

JQUERY SELECTORS

Definition:
jQuery selectors are used to **select HTML elements** so that you can **manipulate, style, or attach events** to them.

Why Selectors Are Needed

- Access elements quickly
- Perform **DOM manipulation**
- Attach **events** or apply **effects**

Types of jQuery Selectors
1. Basic Selectors

Selector	Description	Example
$("p")	Select all <p> elements	$("p").hide();
$("#id")	Select element by ID	$("#btn").click();
$(".class")	Select elements by class	$(".card").css("color","red");
$("*")	Select all elements	$("*").hide();

---

2. Hierarchy / Descendant Selectors

Selector	Description	Example
$("div p")	All <p> inside <div>	$("div p").css("color","blue");
$("div > p")	Direct children <p> of <div>	$("div > p").hide();

---

3. Attribute Selectors

Selector	Description	Example
$("[href]")	Elements with href attribute	$("[href]").css("color","green");
$("[type='text']")	Elements with type="text"	$("[type='text']").val("Hello");
$("[name^='user']")	Name starts with "user"	$("[name^='user']").css("border","1px solid red");

---

4. Filter Selectors

Selector	Description	Example
:first	First element	$("li:first").css("color","red");
:last	Last element	$("li:last").hide();
:even	Even elements	$("li:even").css("background","#eee");
:odd	Odd elements	$("li:odd").css("background","#ccc");

Real-Life Example

```
<ul>
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ul>
```

```
<script>
  $("li:first").css("color","blue"); // First item blue
  $("li:odd").css("background","#eee"); // Odd items grey
</script>
```

- Highlight menu items
- Style first/last row in table
- Apply effects selectively

Interview-Ready Lines

- $() → universal selector in jQuery
- Supports **ID, class, element, attribute, hierarchical & filter selectors**
- Powerful for **DOM manipulation and event handling**

JQUERY: DOM MANIPULATION & EVENTS

Definition:

- **DOM Manipulation** → Changing HTML elements, attributes, content, or structure dynamically using jQuery.
- **Events** → Responding to user actions like click, hover, keypress, or form submission.

Why It's Needed

- Create **interactive web pages**
- Dynamically **update content** without reloading the page
- Handle **user inputs and actions** efficiently

---

jQuery DOM Manipulation

1. Changing Text & HTML

```
$("#text").text("Hello World");      // Change text
$("#text").html("<b>Hello World</b>"); // Change HTML content
```

2. Changing Attributes & CSS

```
$("#img").attr("src", "new.jpg"); // Change image source
$("#btn").css("background-color", "blue"); // Change CSS
```

3. Adding / Removing Elements

```
$("#list").append("<li>New Item</li>");  // Add to end
$("#list").prepend("<li>First Item</li>"); // Add to start
$("#item").remove();                      // Remove element
```

4. Show / Hide / Toggle

```
$("#box").hide();   // Hide element
$("#box").show();   // Show element
$("#box").toggle(); // Toggle visibility
```

---

jQuery Events

1. Common Events

Event	**Description**	**Example**
click	User clicks element	$("#btn").click()
dblclick	Double click	$("#btn").dblclick()
hover	Mouse enters/leaves	$("#box").hover()
keypress	Key pressed	$("#input").keypress()
submit	Form submitted	$("#form").submit()

2. Event Binding with .on()

```
$("#btn").on("click", function() {
  alert("Button clicked!");
});
```

- Preferred over direct methods (.click())
- Can bind **multiple events** or **dynamic elements**

Real-Life Examples

- **DOM Manipulation** → Add/remove products in a shopping cart dynamically
- **Events** → Show modal on button click, validate form on submit, highlight menu on hover

---

Interview-Ready Lines

- $().text() / $().html() → change content
- $().attr() → change attributes
- $().css() → modify styles
- $().append() / prepend() / remove() → dynamic content changes
- $().on() → attach event handlers efficiently
- jQuery **simplifies DOM & events**, reduces cross-browser issues

---

JQUERY EFFECTS & ANIMATIONS

Definition:
jQuery Effects are built-in methods to **animate HTML elements**, **show/hide content**, or **create visual feedback** for users.

Why Effects Are Needed

- Improve **user experience**
- Make UI **interactive and dynamic**
- Highlight **changes on the page**
- Save **manual CSS/JS animation coding**

Common jQuery Effects
1. Show / Hide / Toggle

```
$("#btnShow").click(function(){ $("#box").show(); });
$("#btnHide").click(function(){ $("#box").hide(); });
$("#btnToggle").click(function(){ $("#box").toggle(); });
```

- .show() → display element
- .hide() → hide element
- .toggle() → toggle visibility

---

2. Fade Effects

```
$("#fadeIn").click(function(){ $("#box").fadeIn(); });
$("#fadeOut").click(function(){ $("#box").fadeOut(); });
$("#fadeToggle").click(function(){ $("#box").fadeToggle(); });
$("#fadeTo").click(function(){ $("#box").fadeTo("slow", 0.5); }); // opacity 0.5
```

- .fadeIn() / .fadeOut() → smooth visibility change
- .fadeToggle() → toggle with fade
- .fadeTo(duration, opacity) → set specific opacity

---

3. Slide Effects

```
$("#slideDown").click(function(){ $("#box").slideDown(); });
```

```
$("#slideUp").click(function(){ $("#box").slideUp(); });
$("#slideToggle").click(function(){ $("#box").slideToggle(); });
```
- .slideDown() → show element sliding down
- .slideUp() → hide element sliding up
- .slideToggle() → toggle slide animation

---

4. Animate Custom Properties
```
$("#animateBtn").click(function(){
  $("#box").animate({
    left: '250px',
    height: '150px',
    opacity: 0.5
  }, 1000); // duration in ms
});
```
- Animate **CSS numeric properties** like width, height, opacity, position

---

Real-Life Examples
- Fade in/out **images in a slider**
- Slide toggle **accordion menus**
- Animate **progress bars or notifications**
- Highlight **added or removed items** in a shopping cart

Interview-Ready Points
- jQuery provides **ready-to-use visual effects**
- Common methods: .show(), .hide(), .toggle(), .fadeIn(), .fadeOut(), .slideUp(), .slideDown(), .animate()
- Easy to **enhance UI interactivity** with minimal code
- **Chainable** → multiple effects in a single line

JQUERY AJAX

Definition:
AJAX (Asynchronous JavaScript and XML) allows web pages to **load or send data** to a server **without reloading the page.**
jQuery simplifies AJAX calls with methods like $.ajax(), $.get(), and $.post().

Why AJAX Is Needed
- Improves **user experience** (no full page reload)
- Fetch data **dynamically** (JSON, HTML, or text)
- Submit forms **without refresh**
- Power **modern interactive web apps**

Basic jQuery AJAX Methods
1. $.ajax() (Most Flexible)
```
$.ajax({
  url: "data.json",      // Server URL or file
  type: "GET",           // Method: GET or POST
  dataType: "json",      // Expected data type
  success: function(response) {
    console.log(response); // Handle success
    $("#result").text(response.name);
  },
  error: function(xhr, status, error) {
    console.log(error); // Handle error
  }
});
```

---

2. $.get() (Simpler GET Request)

```
$.get("data.json", function(response){
  $("#result").text(response.name);
});
```

---

3. $.post() (Simpler POST Request)

```
$.post("submit.php", { name: "John", age: 25 }, function(response){
  $("#result").text(response);
});
```

Real-Life Example

```
<button id="loadBtn">Load Data</button>
<div id="result"></div>

<script>
$("#loadBtn").click(function(){
  $.get("data.json", function(data){
    $("#result").html("Name: " + data.name + "<br>Age: " + data.age);
  });
});
</script>
```

- Click button → load JSON data → display in <div> without page reload

---

Interview-Ready Points

- AJAX = **asynchronous server communication**
- jQuery methods: $.ajax(), $.get(), $.post()
- Works with **JSON, HTML, or plain text**
- Improves **user experience** and **performance**
- Success and error handlers handle server responses
