🧩 Unit 4 – AJAX & jQuery Concepts

---

1. Introduction to JavaScript

What is JavaScript?

- JavaScript is a programming language used in web development.
- It runs inside the browser (and also on servers using Node.js).
- It makes web pages *interactive* — like dropdown menus, slideshows, form validation, and live updates without refreshing the page.

Where to Write JavaScript?

- It is written inside <script> tags in an HTML page.

Example:

```
<script>
  alert("Hello, JavaScript!");
</script>
```

Basic Concepts:

- **Variables:** Used to store data.
  Example:
- var name = "John";
- var age = 25;
- **Data Types:**
  - String → "Hello"
  - Number → 42 or 3.14
  - Boolean → true / false
  - Array → ["apple", "banana", "cherry"]
  - Object → { name: "John", age: 25 }
  - Null → no value
  - Undefined → not assigned

Operators: Used for calculations or comparisons.
Example: +, -, *, /, ==, >, <

Conditions: Used to make decisions.
Example:

```
if (age > 18) {
```

```
  console.log("Adult");
}
```

Loops: Used to repeat actions.
Example:

```
for (let i = 0; i < 5; i++) {
  console.log(i);
}
```

Functions: Reusable blocks of code.
Example:

```
function greet() {
  alert("Hello!");
}
```

Events: Respond to user actions like clicks or keypresses.

---

2. Introduction to jQuery

What is jQuery?

- jQuery is a **JavaScript library** that makes coding easier and shorter.
- It helps you:
    - Select and modify HTML elements.
    - Handle user events (click, hover, etc.).
    - Create animations.
    - Work with AJAX easily.

How to Use jQuery?

- You must include jQuery in your HTML file using a CDN link:

```
<script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
```

Basic Syntax:

```
$(selector).action();
```

- $ → jQuery function.
- selector → HTML element to target.
- action() → What to do with that element.

Examples:

```
$("p").hide();      // hides all paragraph tags
```

```
$("#test").hide();   // hides element with id="test"
$(".du").hide();     // hides elements with class="du"
$(this).hide();      // hides the current element
```

---

3. Document Ready Event

- jQuery waits for the webpage to fully load before running the code.
- This prevents errors (for example, trying to hide an element that hasn't loaded yet).

Syntax:

```
$(document).ready(function() {
  // code here runs after the document is ready
});
```

Shortcut version:

```
$(function() {
  // code here
});
```

4. jQuery Selectors

Selectors help you find (or "select") HTML elements to work with.

Examples:

```
$("p")       // selects all paragraphs
$("#test")   // selects element with id="test"
$(".du")     // selects all elements with class="du"
$(this)      // selects the current element
```

---

5. jQuery Events

What are Events?

- Events are actions that happen on a webpage (like a user clicking, typing, or moving the mouse).
- jQuery makes it easy to handle these events.

Syntax:

```
$(selector).event(function() {
  // action when event occurs
```

});

Common jQuery Event Methods:

1. **click()** – when user clicks
2. $("p").click(function() {
3. $(this).hide();
4. });
5. **dblclick()** – when user double-clicks
6. **mouseenter()** – when mouse pointer enters
7. **mouseleave()** – when mouse pointer leaves
8. **hover()** – runs two functions (mouseenter + mouseleave)

Form Events:

- Example: change(), submit(), focus(), etc.

on() Method:
Used to attach one or more events to selected elements.

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

---

6. Introduction to AJAX

Traditional Websites:
Earlier, if users wanted to update something on a page (like email inbox), they had to **reload the whole page**.
This was slow and inefficient.

What is AJAX?

- AJAX stands for **Asynchronous JavaScript and XML**.
- It allows web pages to **send or receive data from the server without reloading the page**.
- This makes websites faster and more interactive.

Examples of AJAX:

- Gmail – loads new mails without refreshing.
- Google Maps – loads map parts dynamically.

In ASP.NET Core:
AJAX is used to make asynchronous (non-blocking) calls between browser and server.
You can use jQuery, Fetch API, or async/await for AJAX requests.

---

7. How AJAX Works

AJAX Request Process:

1. JavaScript sends a request to a server (URL).
2. Server processes it and sends back data.
3. The browser updates only part of the webpage (without reloading).

Basic AJAX Syntax:

```
$.ajax({
  url: "your-endpoint-url",
  type: "GET" or "POST",
  data: { key: value },
  success: function(response) {
    // what to do on success
  },
  error: function(request, status, error) {
    // what to do on error
  },
  async: true
});
```

Main Properties:

- url: Server location to send request.
- type: Method type – GET, POST, PUT, DELETE.
- data: Data you want to send.
- success: Runs when request succeeds (HTTP 200).
- error: Runs when something goes wrong.
- async: True = non-blocking call.

---

8. Types of AJAX Calls

1. Using XMLHttpRequest Object (Old Way):

```
let xhttp = new XMLHttpRequest();
xhttp.open("GET", "data.txt", true);
```

```
xhttp.send();
```

2. Using jQuery (Common Way):

```
$.ajax({
  url: "data.txt",
  success: function(result) {
    console.log(result);
  }
});
```

3. Using Fetch API (Modern Way):

```
fetch("data.txt")
  .then(response => response.text())
  .then(data => console.log(data))
  .catch(error => console.log(error));
```

---

9. jQuery AJAX Methods

Method	Description
.ajaxComplete()	Called when all AJAX requests are done
.ajaxError()	Called if an AJAX request fails
.ajaxSend()	Runs before an AJAX request is sent
.ajaxStart()	Runs when first AJAX request begins
.ajaxStop()	Runs when all AJAX requests have completed
.ajaxSuccess()	Runs when an AJAX request succeeds
.load()	Loads data from the server into selected elements
.serialize()	Converts form data into a query string
.serializeArray()	Converts form data into an array of objects

---

10. jQuery vs AJAX

Feature	jQuery	AJAX
Meaning	A JavaScript library	A technology/method

Feature	jQuery	AJAX
Purpose	Simplifies JS coding and DOM manipulation	Used to exchange data with a server
Usage	Handles events, animations, AJAX, etc.	Sends/receives data asynchronously
Example	$("p").hide()	$.ajax({ url: "data" })