

Day 8: DOM Basics & Selection

بنیادی باتیں DOM

Quote of the Day: "The DOM is your bridge between JavaScript and the visual webpage. Cross it, and your code comes alive!"

"کو صفحے کی زندگی دیتا ہے۔ JavaScript وہ پل ہے جو آپ کے DOM"

🎉 WEEK 2 STARTS TODAY! (ہفتہ 2 شروع)

Welcome back! Week 1 aapne JavaScript basics master kar liye.

Week 1 recap - you can:

- ☒ Store data (variables, arrays, objects)
- ☒ Make decisions (conditionals)
- ☒ Repeat tasks (loops)
- ☒ Organize code (functions)

Week 2 starts NOW - you'll learn to:

- 🎯 Change what users SEE (DOM manipulation)
- 🎯 Respond to user actions (events)
- 🎯 Build interactive pages (buttons, forms)
- 🎯 Create REAL websites!

Until yesterday, your code lived in the console.

From TODAY, your code changes what people see on screen! 🚀

📋 Today's Learning Goals (آج کے اہداف)

By the end of today, you will:

- ☐ Understand what the DOM is and how it works
- ☐ Select HTML elements using `querySelector`
- ☐ Change text content and HTML of elements
- ☐ Modify CSS styles with JavaScript
- ☐ Build a Dynamic Profile Card that updates via JavaScript

Time Breakdown (کل وقت: 150 منٹ)

- 🕒 7:00-7:05 PM (5min): Standup - Show your Restaurant Menu System!
- 🕒 7:05-8:05 PM (60min): Understanding DOM (3× Pomodoro)
- 🕒 8:05-8:50 PM (45min): Practice selecting and changing elements
- 🕒 8:50-9:25 PM (35min): Build Dynamic Profile Card
- 🕒 9:25-9:30 PM (5min): Quiz & reflection

What We're Building Today

Today you'll build a **Dynamic Profile Card** - an HTML page with a professional profile card that JavaScript fills with data, styles, and can even switch between light/dark themes!

Why This Matters for Your Career:

DOM manipulation is what frontend development IS:

- **Facebook:** JavaScript adds posts to your feed dynamically
- **Daraz:** Products appear without page reload
- **YouTube:** New videos load as you scroll
- **Gmail:** Emails marked read/unread without refresh

Every visual change on a website is JavaScript manipulating the DOM!

سمجھنا (Understanding): What is the DOM?

The Real-World Analogy

Scenario: A Government Office Building (سرکاری دفتر)

Imagine a government building has:

```
Building (Document)
├── Ground Floor (html)
│   ├── Reception (head)
│   │   ├── Building Name Sign (title)
│   │   └── Interior Design Rules (CSS)
│   └── Main Hall (body)
│       ├── Information Board (h1)
│       ├── Notice Boards (p, p, p)
│       └── Application Counter (form)
│           ├── Name Field (input)
│           └── Submit Button (button)
```

The DOM is this building's MAP (نقشه)!

JavaScript uses this map to:

- Find any room/floor (select elements)
- Change the sign on any board (change content)
- Repaint any room (change styles)
- Add or remove rooms (add/remove elements)

The Real Story: HTML → DOM

What you write (HTML file):

```
<body>
  <h1 id="title">My Website</h1>
  <p class="intro">Welcome to my page</p>
  <button>Click Me</button>
</body>
```

What browser creates (DOM tree):

```
document
├── html
│   └── body
│       ├── h1 (id="title") → "My Website"
│       ├── p (class="intro") → "Welcome to my page"
│       └── button → "Click Me"
```

JavaScript can grab ANY node and change it!

Why This Matters

Static webpage (without DOM):

- Shows same content for everyone
- User can't interact
- No real functionality
- Like a printed poster (پوسٹر)

Dynamic webpage (with DOM):

- Changes based on user actions
- Responds to clicks, typing, scrolling
- Different content for different users
- Like a live TV screen (ٹی وی)

The Mental Model

Think of the DOM like a **live news ticker** on GEO TV:

```
TV Screen (webpage)
├── Top Bar: "GEO News" (heading)
├── Main Content: News story (paragraph)
└── Ticker: Breaking news (scrolling text)
```

```
Remote Control (JavaScript)
├── Change channel (change content)
├── Adjust volume (change style)
└── Pause/play (add/remove elements)
```

JavaScript is your **remote control** for the webpage!

Building Block #1: Linking JavaScript to HTML

Why Linking Matters (کیوں؟)

Before JavaScript can control HTML, they must be **connected**!

Urdu Analogy: Like connecting your phone to WiFi before using internet - no connection, no communication!

How to Link JavaScript

Method 1: Internal Script (same file)

```
<!DOCTYPE html>
<html>
<head>
  <title>My Page</title>
</head>
<body>
  <h1 id="heading">Hello World</h1>

  <!-- JavaScript goes at BOTTOM of body -->
  <script>
    // Your code here
    console.log("Page loaded!");
  </script>
</body>
</html>
```

Method 2: External Script (separate .js file)

```
<!-- In your HTML file -->
<body>
  <h1 id="heading">Hello World</h1>

  <!-- Link to external file -->
  <script src="script.js"></script>
</body>
```

```
// In script.js
console.log("External file works!");
```

Why Script Goes at Bottom

```
<!-- ❌ WRONG - script before HTML elements -->
<head>
  <script>
    document.getElementById("heading");
    // FAILS! heading doesn't exist yet!
  </script>
</head>
<body>
  <h1 id="heading">Hello</h1>
</body>

<!-- ✅ RIGHT - script after HTML elements -->
<body>
  <h1 id="heading">Hello</h1>

  <script>
    document.getElementById("heading");
    // Works! heading already loaded
  </script>
</body>
```

💡 **Why:** Browser reads HTML top to bottom. If script runs before elements exist, it can't find them!

Your First HTML + JS Page

```
<!-- TODO: Complete this starter page -->
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>My First DOM Page</title>
</head>
<body>
  <h1 id="main-heading">Original Heading</h1>
  <p id="description">Original description text</p>

  <!-- TODO: Add script tag here -->
  <_____>
    console.log("JavaScript is working!");
    console.log("I can see the heading:", document.getElementById("main-
heading"));
  </_____>
</body>
</html>
```

Check Your Understanding

- ☐ Where should the script tag go in HTML?
- ☐ What's the difference between internal and external JS?
- ☐ Why can't script go in the `<head>` without special handling?
- ☐ How do you link an external JS file?

Building Block #2: Selecting Elements (عناصر تلاش کرنا)

What is Selection? (کیا ہے؟)

Urdu Analogy: Think of selecting elements like finding someone in a **university crowd**:

```
"Find the student with ID card 101"    → getElementById
"Find all students in red shirts"      → querySelectorAll(".red")
"Find the first student in blue shirt" → querySelector(".blue")
```

JavaScript needs to FIND elements before it can change them!

Method 1: getElementById

```
// THINKING: Get element by its unique ID

// HTML: <h1 id="page-title">Welcome</h1>

const title = document.getElementById("page-title");
// Returns: the h1 element

console.log(title);
// Shows: <h1 id="page-title">Welcome</h1>
```

Method 2: querySelector (most useful!)

```
// THINKING: querySelector is like CSS selectors

// HTML:
// <h1 id="main">Title</h1>
// <p class="intro">Text</p>
// <button>Click</button>

// Select by ID (use # symbol)
const heading = document.querySelector("#main");

// Select by class (use . symbol)
const intro = document.querySelector(".intro");

// Select by tag name
```

```
const btn = document.querySelector("button");

// Select nested element
const navLink = document.querySelector("nav a");
```

Method 3: querySelectorAll (multiple elements)

```
// THINKING: Get ALL matching elements

// HTML: (3 paragraphs with class "product")
// <p class="product">Laptop</p>
// <p class="product">Mouse</p>
// <p class="product">Keyboard</p>

const allProducts = document.querySelectorAll(".product");
// Returns: NodeList (like array) of all 3

console.log(allProducts.length); // 3

// Loop through them
allProducts.forEach(product => {
  console.log(product);
});
```

Selector Cheat Sheet

```
// By ID:      #idName      (unique, one element)
// By class:   .className   (multiple elements)
// By tag:     tagName      (all same tags)
// By attribute: [attr="value"] (elements with attribute)

document.querySelector("#header"); // ID: header
document.querySelector(".card");   // First .card
document.querySelectorAll(".card"); // ALL .cards
document.querySelector("h1");       // First h1
document.querySelector("input[type='text']"); // text inputs
```

Your First Example

```
<!DOCTYPE html>
<html>
<body>
  <h1 id="city-name">Lahore</h1>
  <p class="description">City of Gardens</p>
  <p class="description">Population: 13 million</p>
</body>
```

```

<script>
  // TODO: Select heading by ID
  const heading = document.querySelector("_____");
  console.log(heading);

  // TODO: Select first paragraph by class
  const firstPara = document.querySelector("_____");
  console.log(firstPara);

  // TODO: Select ALL paragraphs
  const allParas = document.querySelectorAll("_____");
  console.log("Total paragraphs:", allParas._____);
</script>
</html>

```

Common Mistakes

✗ Wrong:

```
const el = document.querySelector("main-heading"); // No # for ID!
```

✓ Right:

```
const el = document.querySelector("#main-heading"); // # for ID
// or
const el = document.getElementById("main-heading"); // No # needed
```

✗ Wrong:

```
const el = document.querySelector(".products"); // Gets only ONE
el.forEach(...) // ERROR! querySelector returns ONE element
```

✓ Right:

```
const els = document.querySelectorAll(".products"); // Gets ALL
els.forEach(...) // Works! querySelectorAll returns NodeList
```

Check Your Understanding

- ☐ What symbol do you use for ID in querySelector?
- ☐ What symbol for class?

- ☐ What's the difference between `querySelector` and `querySelectorAll`?
- ☐ How do you select elements by tag name?

Building Block #3: Changing Content

textContent vs innerHTML

Urdu Analogy: Two ways to change what's on a whiteboard (وائٹ بورڈ):

```
textContent = Write plain text only (صرف متن)
innerHTML   = Write text WITH formatting (متن + رنگ + شکل)
```

textContent - Plain Text

```
// THINKING: Change text without HTML

// HTML: <h1 id="greeting">Hello</h1>

const greeting = document.querySelector("#greeting");

// GET current text
console.log(greeting.textContent); // "Hello"

// SET new text
greeting.textContent = "السلام علیکم";
// Page now shows: السلام علیکم

// Even HTML tags shown as plain text!
greeting.textContent = "<strong>Bold</strong>";
// Shows Literally: <strong>Bold</strong> (not bold!)
```

innerHTML - HTML Content

```
// THINKING: Change content WITH HTML formatting

const greeting = document.querySelector("#greeting");

// SET HTML content
greeting.innerHTML = "<strong>جی آیاں نوں!</strong>";
// Page shows: جی آیاں نوں! (in BOLD)

greeting.innerHTML = "Welcome <span style='color:green'>to Pakistan</span>!";
// "Pakistan" appears in green color!
```

When to Use Each

```
// Use.textContent when:
// - Showing user input (safer - no HTML injection)
// - Showing plain data
element.textContent = userInput; // Safe!

// Use.innerHTML when:
// - You control the content (no user input)
// - You need HTML formatting
element.innerHTML = "<b>Important!</b>"; // Only if YOU write it
```

Other Content Properties

```
// THINKING: Different ways to update page content

// For input fields
const input = document.querySelector("#name-input");
input.value = "Hassan"; // Set input value
console.log(input.value); // Get input value

// For images
const img = document.querySelector("#profile-pic");
img.src = "new-photo.jpg"; // Change image source
img.alt = "Profile Photo"; // Change alt text

// For Links
const link = document.querySelector("a");
link.href = "https://google.com"; // Change link URL
link.textContent = "Google"; // Change link text
```

Your First Example

```
<!DOCTYPE html>
<html>
<body>
  <h1 id="student-name">Loading...</h1>
  <p id="student-info">Please wait...</p>
  <img id="student-photo" src="" alt="Student">
</body>
<script>
  // Student data (from Week 1!)
  const student = {
    name: "Hassan Ali",
    rollNo: "ADPCS-2024-101",
    program: "Computer Science"
  };
  </script>
```

```
// TODO: Update heading with student name
const nameEl = document.querySelector("#student-name");
nameEl._____ = student.name;

// TODO: Update paragraph with roll number and program
const infoEl = document.querySelector("#student-info");
infoEl._____ = student.rollNo + " | " + student.program;

// TODO: Add bold formatting to name using innerHTML
nameEl._____ = "<strong>" + student.name + "</strong>";
</script>
</html>
```

Check Your Understanding

- ☐ What's the difference between textContent and innerHTML?
- ☐ Which is safer for user input?
- ☐ How do you change an image's source?
- ☐ How do you get the current value of an input?

Building Block #4: Changing Styles

Inline Styles with JavaScript

Urdu Analogy: Think of changing clothes (لباس بدلنا):

Adding CSS class = Wearing a full outfit
style.property = Changing one accessory

Method 1: Direct Style Property

```
// THINKING: Change one style at a time

const heading = document.querySelector("h1");

// Change individual styles
heading.style.color = "green";
heading.style.fontSize = "32px";
heading.style.backgroundColor = "#f0f0f0";
heading.style.padding = "10px";
heading.style.textAlign = "center";

// Note: CSS uses kebab-case (font-size)
// JavaScript uses camelCase (fontSize)
```

CSS vs JavaScript Style Names

CSS Property	JavaScript Style
font-size	fontSize
background-color	backgroundColor
text-align	textAlign
border-radius	borderRadius
margin-top	marginTop
z-index	zIndex

Method 2: classList (PREFERRED!)

```
// THINKING: Add/remove CSS classes

// This is MUCH better than inline styles!
// Write styles in CSS, switch classes in JS

const card = document.querySelector(".card");

// Add a class
card.classList.add("highlighted");

// Remove a class
card.classList.remove("highlighted");

// Toggle (add if not there, remove if there)
card.classList.toggle("dark-mode");

// Check if class exists
if (card.classList.contains("active")) {
  console.log("Card is active!");
}
```

Real Example: Theme Switcher

```
// CSS (in stylesheet):
// .dark-mode { background: #333; color: white; }
// .light-mode { background: white; color: black; }

const body = document.querySelector("body");

function toggleTheme() {
  body.classList.toggle("dark-mode");
}
```

```
body.classList.toggle("light-mode");  
}
```

Your First Example

```
<!DOCTYPE html>  
<html>  
<head>  
  <style>  
    .highlight { background-color: yellow; padding: 5px; }  
    .large-text { font-size: 24px; font-weight: bold; }  
    .success { color: green; border: 2px solid green; }  
    .error { color: red; border: 2px solid red; }  
  </style>  
</head>  
<body>  
  <p id="message">This is a message</p>  
  <p id="status">Status: Unknown</p>  
</body>  
<script>  
  const message = document.querySelector("#message");  
  const status = document.querySelector("#status");  
  
  // TODO: Make message yellow background  
  message.classList.add("_____");  
  
  // TODO: Make status green and Large  
  status.classList.add("_____");  
  status.classList.add("_____");  
  
  // TODO: Change status text to "Connected"  
  status.textContent = "_____";  
  
  // TODO: Change message color directly  
  message.style._____ = "purple";  
</script>  
</html>
```

Check Your Understanding

- ☐ How do CSS property names differ in JavaScript?
- ☐ Why is classList preferred over style?
- ☐ What does classList.toggle() do?
- ☐ How do you check if an element has a class?

Practice Session: DOM Manipulation

Practice Goal

By the end of this section, you'll confidently select elements and change their content and styles!

Exercise 1: Guided Practice (ہم ساتھ کریں)

Scenario: Update a Pakistani news ticker dynamically

HTML (copy this first):

```
<!DOCTYPE html>
<html>
<head>
  <style>
    body { font-family: Arial, sans-serif; padding: 20px; }
    #channel-name { color: #cc0000; font-size: 28px; }
    .breaking { background: #cc0000; color: white; padding: 5px 10px; }
    .ticker-text { font-size: 18px; margin-top: 10px; }
  </style>
</head>
<body>
  <h1 id="channel-name">Loading...</h1>
  <p id="ticker-text">Loading news...</p>
  <span id="time-stamp">--:-- AM</span>
</body>
<script>
  // TODO Step 1: Select elements
  const channelName = document.querySelector("_____");
  const ticker = document.querySelector("_____");
  const timeStamp = document.querySelector("_____");

  // TODO Step 2: Update content
  channelName.textContent = "GEO News";
  ticker.textContent = "_____"; // Write a breaking news headline
  timeStamp.textContent = "9:00 PM";

  // TODO Step 3: Add styling
  channelName.style.color = "_____";
  ticker.classList.add("_____");
</script>
</html>
```

Exercise 2: Student Card Updater (اب آپ)

Problem: Create HTML page with empty student card, fill it with JavaScript

Requirements:

- ☐ HTML has placeholders (Loading... text)

- ☐ JavaScript fills in ALL details
- ☐ Card has nice styling (use classList)
- ☐ Uses student object from Week 1

Starter Code:

```
<!DOCTYPE html>
<html>
<head>
  <style>
    .student-card {
      border: 2px solid #333;
      padding: 20px;
      max-width: 400px;
      border-radius: 10px;
    }
    .active { border-color: green; background: #ffff0; }
    .student-name { font-size: 24px; color: #333; }
    .cgpa-good { color: green; }
    .cgpa-average { color: orange; }
  </style>
</head>
<body>
  <div id="student-card" class="student-card">
    <h2 id="name-display">Loading...</h2>
    <p id="roll-display">Roll: Loading...</p>
    <p id="program-display">Program: Loading...</p>
    <p id="cgpa-display">CGPA: Loading...</p>
  </div>
</body>
<script>
  // Student data
  const student = {
    name: "Hassan Ali",
    rollNo: "ADPCS-2024-101",
    program: "Computer Science",
    cgpa: 3.75
  };

  // TODO: Select all elements
  const card = document.querySelector("_____");
  const nameEl = document.querySelector("_____");
  const rollEl = document.querySelector("_____");
  const programEl = document.querySelector("_____");
  const cgpaEl = document.querySelector("_____");

  // TODO: Fill with student data
  nameEl.textContent = _____;
  rollEl.textContent = "Roll: " + _____;
  programEl.textContent = _____;
  cgpaEl.textContent = "CGPA: " + _____;
```

```

// TODO: Add 'active' class to card
card.classList.add(____);

// TODO: Color CGPA based on value
if (student.cgpa >= 3.5) {
  cgpaEl.classList.add("_____"); // green
} else {
  cgpaEl.classList.add("_____"); // orange
}
</script>
</html>

```

Don't Look Below Until You Try! 

Hints (if stuck):

► Stuck on selectors?

```

const card = document.querySelector("#student-card");
const nameEl = document.querySelector("#name-display");
// Use # for IDs!

```

► Stuck on CGPA coloring?

```

if (student.cgpa >= 3.5) {
  cgpaEl.classList.add("cgpa-good");
} else {
  cgpaEl.classList.add("cgpa-average");
}

```

Exercise 3: City Info Updater

Problem: Array of Pakistani cities, update display for each

```

<!DOCTYPE html>
<html>
<head>
  <style>
    .city-card { padding: 15px; margin: 10px; border: 1px solid #ddd; }
    .big-city { background: #e3f2fd; font-weight: bold; }
    .small-city { background: #f5f5f5; }
  </style>
</head>

```



```

<body>
  <div id="city-display" class="city-card">
    <h2 id="city-name">Loading...</h2>
    <p id="city-province">Province: Loading...</p>
    <p id="city-population">Population: Loading...</p>
  </div>
</body>
<script>
  const cities = [
    { name: "Lahore", province: "Punjab", population: 13000000, isBig:
true },
    { name: "Karachi", province: "Sindh", population: 16000000, isBig:
true },
    { name: "Gilgit", province: "GB", population: 300000, isBig: false }
  ];

  // Show first city
  const city = cities[0];

  // TODO: Select elements and fill data
  document.querySelector("#city-name")._____ = city.name;
  document.querySelector("#city-province")._____ = "Province: " +
city.province;
  document.querySelector("#city-population")._____ =
    "Population: " + city.population.toLocaleString();

  // TODO: Add appropriate class
  const card = document.querySelector("#city-display");
  if (city.isBig) {
    card.classList.add("_____");
  } else {
    card.classList.add("_____");
  }

  // Try cities[1] and cities[2] to see different results!
</script>
</html>

```

اج کا چیلنج (Today's Challenge)

Project: Dynamic Profile Card

ڈائنامک پروفائل کارڈ

The Problem:

Build a professional HTML profile card that looks like a LinkedIn-style card. Instead of writing content directly in HTML, JavaScript will fill everything in from a data object - just like how real websites work!

What You're Building:

A complete profile card with photo, name, bio, skills, and a theme toggle button!

Success Criteria:

- ☐ HTML has structure but empty/placeholder content
 - ☐ JavaScript fills ALL content from a data object
 - ☐ Card looks professional with CSS
 - ☐ Skills show as styled tags
 - ☐ Active/professional styling applied
 - ☐ No hardcoded content in HTML (all from JS!)
-

Phase 1: Planning (سوچیں پہلے)

Before coding, answer:

1. What HTML elements do I need?

```
- Profile card container (div)
- Profile photo (img)
- Name (h2)
- Title/Role (p)
- Bio (p)
- Skills list (div with spans)
- Contact info (p)
```

2. What data object for profile?

```
const profile = {
  name: "...",
  title: "...",
  bio: "...",
  skills: ["...", "..."],
  email: "...",
  photo: "..."
};
```

3. What JavaScript operations?

- Select each element
- Update textContent/innerHTML
- Loop through skills array
- Add CSS classes

Planning Checkpoint:

- ☐ I know what HTML structure to write
- ☐ I have profile data object ready
- ☐ I know which JS methods to use

- ☐ I understand querySelector selectors

Phase 2: Foundation (بنیاد)

Starter Code:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Dynamic Profile Card</title>
  <style>
    /* Reset & Base */
    * { margin: 0; padding: 0; box-sizing: border-box; }
    body {
      font-family: 'Segoe UI', Arial, sans-serif;
      background: #f0f2f5;
      display: flex;
      justify-content: center;
      align-items: center;
      min-height: 100vh;
      padding: 20px;
    }

    /* Profile Card */
    .profile-card {
      background: white;
      border-radius: 15px;
      padding: 30px;
      max-width: 380px;
      width: 100%;
      box-shadow: 0 4px 20px rgba(0,0,0,0.1);
      text-align: center;
    }

    /* Profile Photo */
    .profile-photo {
      width: 100px;
      height: 100px;
      border-radius: 50%;
      object-fit: cover;
      border: 4px solid #0077b5;
      margin-bottom: 15px;
    }

    /* Name */
    .profile-name {
      font-size: 22px;
      font-weight: bold;
      color: #333;
    }
  </style>
</head>
</html>
```

```
        margin-bottom: 5px;
    }

    /* Title */
    .profile-title {
        color: #0077b5;
        font-size: 14px;
        margin-bottom: 15px;
    }

    /* Bio */
    .profile-bio {
        color: #666;
        font-size: 14px;
        line-height: 1.6;
        margin-bottom: 20px;
    }

    /* Skills */
    .skills-container {
        display: flex;
        flex-wrap: wrap;
        justify-content: center;
        gap: 8px;
        margin-bottom: 20px;
    }
    .skill-tag {
        background: #e8f4fd;
        color: #0077b5;
        padding: 4px 12px;
        border-radius: 20px;
        font-size: 12px;
        font-weight: bold;
    }

    /* Contact */
    .profile-contact {
        color: #888;
        font-size: 13px;
        border-top: 1px solid #eee;
        padding-top: 15px;
    }

    /* Dark Mode */
    .dark-mode {
        background: #1a1a2e;
        color: white;
    }
    .dark-mode .profile-card {
        background: #16213e;
        color: white;
    }
    .dark-mode .profile-name { color: white; }
```

```

    .dark-mode .profile-bio { color: #aaa; }
    .dark-mode .profile-contact { color: #888; border-color: #333; }

    /* Theme Button */
    .theme-btn {
        margin-top: 15px;
        padding: 8px 20px;
        background: #0077b5;
        color: white;
        border: none;
        border-radius: 20px;
        cursor: pointer;
        font-size: 13px;
    }
    .theme-btn:hover { background: #005fa3; }
</style>
</head>
<body>
    <!-- NOTICE: HTML is EMPTY - JavaScript fills everything! -->

    <div class="profile-card" id="profile-card">

        <!-- TODO: Profile photo -->
        <img
            class="profile-photo"
            id="profile-photo"
            src=""
            alt="Profile Photo"
        >

        <!-- TODO: Name and title -->
        <h2 class="profile-name" id="profile-name">Loading...</h2>
        <p class="profile-title" id="profile-title">Loading...</p>

        <!-- TODO: Bio paragraph -->
        <p class="profile-bio" id="profile-bio">Loading...</p>

        <!-- TODO: Skills will be added by JavaScript -->
        <div class="skills-container" id="skills-container">
            <!-- JavaScript will insert skill tags here -->
        </div>

        <!-- TODO: Contact info -->
        <p class="profile-contact" id="profile-contact">Loading...</p>

        <!-- Theme toggle button -->
        <button class="theme-btn" id="theme-btn">🌙 Dark Mode</button>
    </div>

    <script>
        // =====
        // DYNAMIC PROFILE CARD
        // By: [Your Name]
    </script>

```

```
// =====

// ===== PROFILE DATA =====
// Change this data to customize the card!

const profileData = {
  name: "Hassan Ali",
  title: "JavaScript Developer | ADPCS Student",
  bio: "First-year CS student at Superior University, passionate
about web development. Building cool things with JavaScript!",
  skills: ["JavaScript", "HTML", "CSS", "Problem Solving",
"Arrays", "Functions"],
  email: "hassan@superior.edu.pk",
  phone: "+92-300-1234567",
  photo: "https://api.dicebear.com/7.x/avataaars/svg?seed=Hassan"
  // ^ This generates a random avatar! Change 'Hassan' to your name
};

// ===== SELECT ELEMENTS =====

// TODO Step 1: Select all elements using querySelector
const photoEl = document.querySelector("_____"); // #profile-
photo
const nameEl = document.querySelector("_____"); // #profile-
name
const titleEl = document.querySelector("_____"); // #profile-
title
const bioEl = document.querySelector("_____"); // #profile-
bio
const skillsEl = document.querySelector("_____"); // #skills-
container
const contactEl = document.querySelector("_____"); // #profile-
contact
const themeBtn = document.querySelector("_____"); // #theme-btn

// ===== FILL CONTENT =====

// TODO Step 2: Update photo
photoEl._____ = profileData.photo; // Set src attribute
photoEl._____ = profileData.name + "'s photo"; // Set alt attribute

// TODO Step 3: Update name and title
nameEl._____ = profileData.name;
titleEl._____ = profileData.title;

// TODO Step 4: Update bio
bioEl._____ = profileData.bio;

// TODO Step 5: Create skill tags for each skill
// HINT: Loop through skills array, create HTML for each
profileData.skills.forEach(skill => {
  // TODO: Create skill HTML
  // HINT: Something Like: <span class="skill-
```

```

tag">JavaScript</span>
    const skillHTML = "<span class='skill-tag'>__</span>";
    skillsEl.innerHTML ____ skillHTML; // Add to container
});

// TODO Step 6: Update contact info
contactEl.innerHTML = `📷 ${profileData.____} | 📱
${profileData.____}`;

// ===== THEME TOGGLE =====

const body = document.querySelector("body");
let isDark = false;

// TODO Step 7: Add click handler for theme toggle
themeBtn.addEventListener("click", function() {
    isDark = !isDark;

    // TODO: Toggle dark-mode class on body
    body.classList.____(____);

    // TODO: Update button text
    themeBtn.textContent = isDark ? "☀️ Light Mode" : "🌙 Dark
Mode";
});

// ===== CONSOLE CONFIRMATION =====

console.log("✅ Profile card loaded!");
console.log("Profile name:", profileData.name);
console.log("Skills count:", profileData.skills.length);

</script>
</body>
</html>

```

Phase 3: Milestones (سنگ میل)

Milestone 1: Content Loads ✅

- ☐ Photo displays correctly
- ☐ Name and title show
- ☐ "Loading..." placeholders replaced
- Test: Open browser - do you see your name?

Milestone 2: Skills Show ✅

- ☐ Each skill appears as styled tag
- ☐ All skills from array display
- ☐ Tags have blue background

- Test: Count skill tags matches array length

Milestone 3: Contact Info Shows

- ☐ Email and phone display
- ☐ Format looks good
- Test: Does contact section show?

Milestone 4: Theme Toggle Works

- ☐ Click button switches to dark mode
- ☐ Click again switches back
- ☐ Button text changes (Dark/Light)
- Test: Click button multiple times

Debugging Guide (اگر پھنس جائیں)

Problem: Elements show "Loading..."

- ☐ Check: Did you select element correctly?
- ☐ Check: Is the ID spelling exact (case-sensitive)?
- ☐ Check: Did you set `textContent/innerHTML`?
- Add: `console.log(nameEl)` - is it null?

Problem: `querySelector` returns null

- ☐ Check: Is HTML element present?
- ☐ Check: Did you use `#` for ID, `.` for class?
- ☐ Check: Is script at bottom (after HTML)?
- Common fix: `document.querySelector("#name")` not `("#name")`

Problem: Skills not showing

- ☐ Check: Are you using `+=` not `=` for `innerHTML`?
- ☐ Check: Did you close the `span` tag?
- ☐ Check: Is `skillsEl` selected correctly?

Problem: Theme not switching

- ☐ Check: Did you add event listener?
- ☐ Check: Is `toggle` spelled correctly?
- ☐ Check: Is CSS dark-mode class defined?

Extension Challenges (بونس چیلنج)

If you finish early:

🌟 Level 1: Location & Time


```
// Add current time to card
const timeEl = document.querySelector("#current-time");
const now = new Date();
timeEl.textContent = "🕒 " + now.toLocaleTimeString();

// Add "Active Now" indicator
const statusEl = document.querySelector("#status");
statusEl.textContent = "🟢 Active Now";
statusEl.style.color = "green";
```

🌟🌟 Level 2: Multiple Profiles

```
// Array of profiles
const profiles = [profile1, profile2, profile3];
let current = 0;

// Add "Next Profile" button
function showNextProfile() {
  current = (current + 1) % profiles.length;
  // Update all elements with profiles[current]
}
```

🌟🌟🌟 Level 3: Animated Entry

```
// Card fades in on load
const card = document.querySelector("#profile-card");
card.style.opacity = "0";

// After 100ms, fade in
setTimeout(() => {
  card.style.transition = "opacity 1s";
  card.style.opacity = "1";
}, 100);
```

Daily Quiz (منٹ کا ٹیسٹ 5)

Instructions: Answer WITHOUT looking at notes!

1. What does the DOM stand for?

- A) Data Object Model
- B) Document Object Model
- C) Dynamic Object Manipulation
- D) Display Output Mode

► See Answer (Try first!)

Answer: B - Document Object Model. It's a tree structure representing your HTML that JavaScript can access and modify. Browser creates it automatically from your HTML file.

2. What will `document.querySelector("#title")` select?

- A) Element with class "title"
- B) Element with ID "title"
- C) All title elements
- D) The first element

► See Answer (Try first!)

Answer: B - `#title` selects by ID. Remember: `#` = ID, `.` = class. So `querySelector("#title")` finds the element with `id="title"`.

3. What's the difference between `textContent` and `innerHTML`?

- A) No difference
- B) `textContent` is faster
- C) `innerHTML` can include HTML tags, `textContent` cannot
- D) `textContent` only works on headings

► See Answer (Try first!)

Answer: C - `innerHTML` allows HTML tags (`Bold` actually makes text bold). `textContent` treats everything as plain text (even HTML tags show as literal text). Use `textContent` for user data, `innerHTML` when YOU control the content.

4. What does `classList.toggle("dark")` do?

- A) Always adds "dark" class
- B) Always removes "dark" class
- C) Adds "dark" if absent, removes if present
- D) Checks if "dark" class exists

► See Answer (Try first!)

Answer: C - `toggle()` is like a switch. If class is there, remove it. If not there, add it. Perfect for dark mode buttons, menu toggles, etc.!

5. Why should the script tag go at the bottom of body?

- A) It's just a convention
- B) HTML elements must exist before JavaScript can find them
- C) JavaScript runs slower at the top
- D) It doesn't matter where it goes

► See Answer (Try first!)

Answer: B - Browser reads HTML top to bottom. If JavaScript runs before the elements exist, `querySelector` returns null. Script at bottom ensures all HTML is already loaded and ready to be selected.

Scoring:

- **5/5:** 🎉 DOM Master! You're ready for events!
 - **4/5:** 🏆 Excellent! Review the one you missed
 - **3/5:** 👍 Good! Practice selecting elements more
 - **❤️/5:** 🤔 Re-read the building blocks section
-

🎓 Today's Homework (گھر کا کام)

Required (لازمی):

- ☐ Complete the Dynamic Profile Card
- ☐ Customize it with YOUR information
- ☐ Add the theme toggle feature
- ☐ Show it to your team in tomorrow's standup!

Optional (اختیاری):

- ☐ Try the extension challenges
- ☐ Recreate a Daraz product card with JavaScript
- ☐ Make your Week 1 projects display in browser (not just console)
- ☐ Build a "team roster" showing all 4 students' cards

For Tomorrow:

- ☐ Think: "What happens when you click a button?"
 - ☐ Tomorrow: Event Handling - making things respond to user actions!
-

💭 Daily Reflection (روزانہ کی سوچ)

(What I Learned Today): آج میں نے کیا سیکھا

(What I Found Difficult): مشکل کیا لگا

(What I Want to Explore More): مزید کیا سیکھنا ہے

My Confidence Level (1-10): _____

Tomorrow's Preview

Tomorrow we'll learn about **Event Handling** where you'll build a **Interactive Quiz Game!**

You'll learn how to:

- Listen for user clicks, typing, mouse events
- Respond to user actions in real time
- Build interactive buttons and menus
- Handle keyboard events

Get Ready By:

- ☐ Making sure your Profile Card works
- ☐ Thinking: "How does a button know when it's clicked?"
- ☐ The answer is EVENT LISTENERS!

Resources (اگر مزید پڑھنا ہو)

Free Resources (3G-Friendly):

MDN - Introduction to the DOM

- Link: https://developer.mozilla.org/en-US/docs/Web/API/Document_Object_Model/Introduction
- Best for: Understanding DOM structure

MDN - `querySelector`

- Link: <https://developer.mozilla.org/en-US/docs/Web/API/Document/querySelector>
- Best for: Selector syntax reference

DOM Manipulation in Urdu

- Search: "JavaScript DOM manipulation Urdu tutorial"
- Best for: Visual demonstration

CodeSensei's Tip of the Day: 💡

"DOM manipulation is the MOST used skill in frontend development. Every job interview will test it.

Practice this daily: open any website, press F12, go to Console, and type

`document.querySelector('h1').textContent = 'I changed this!'` — see it change instantly!

That's the power you now have!"

"DOM manipulation والی skill سب سے زیادہ استعمال ہونے والی ہے۔ ہر website پر F12 دباؤ، اور کچھ بدل دو۔ F12، کھولو website ہے۔ ہر skill سب سے زیادہ استعمال ہونے والی ہے۔ DOM manipulation

Team Activity

Tomorrow at 7:00 PM, show your team:

1. Open your Profile Card in browser
 2. Show theme toggle working
 3. Explain: "What is the DOM?" in Urdu
 4. Share: One cool DOM trick you discovered
-

کوڈ سیکھنا ایک سفر ہے، منزل نہیں۔ ہر دن ایک قدم آگے۔

"Learning to code is a journey, not a destination. One step forward every day."

Day 8 Complete! Tomorrow your pages respond to users! 🚀

🎯! کے ساتھ Events اللہ حافظ! کل ملتے ہیں