University of Anbar
College of Computer Sciences and
Information Technology

# Web Development

Dr. Alaa Abdalqahar Jihad

Fourth Stage – Second Course

# PHP Web Development

## Lecture 1: Introduction to Web Development and PHP

**Objectives:**

- Understand the basics of web development.
- Learn the difference between front-end and back-end development.
- Get an overview of PHP and its role in web development.
- Set up a local development environment using **XAMPP**.
- Write and run their first PHP script.

### Introduction to Web Development

- **What is Web Development?**
  - Websites are built using **HTML, CSS, and JavaScript** (Front-end).
  - Server-side programming languages like **PHP, Python, Node.js** (Back-end).
  - Databases like **MySQL, PostgreSQL, MongoDB** for data storage.
- **Front-end vs. Back-end Development:**

| Aspect | Front-end | Back-end |
|---|---|---|
| Technologies | HTML, CSS, JavaScript | PHP, Python, Node.js |
| Runs On | Browser | Server |
| Example | Page layout, animations | User authentication, data processing |

- **Why PHP for Web Development?**
  - Open-source and widely used.

    - Compatible with most web servers and databases.
    - Supports server-side scripting for dynamic websites.

## Setting Up a Local Development Environment

### Why do we need a local server?

- PHP needs a server to execute scripts (unlike HTML, which runs directly in a browser).
- A local server (XAMPP) allows running PHP on a personal computer.

### Installing XAMPP:

1. **Download XAMPP** from apachefriends.org.
2. Install and launch the control panel.
3. Start **Apache** (for PHP) and **MySQL** (for database).

### Testing PHP Installation:

1. Open the **htdocs** folder in XAMPP (C:\xampp\htdocs\).
2. Create a new file called test.php.
3. Write the following code:

```
<?php
echo "Hello, World!";
?>
```

4. Open a browser and go to http://localhost/test.php.

## Understanding PHP Basics

### What is PHP?

- PHP stands for **"PHP: Hypertext Preprocessor"**.

- A server-side scripting language used to generate dynamic web pages.
- Can be embedded inside **HTML**.

**Embedding PHP in HTML:**

```
<!DOCTYPE html>
<html>
<head>
    <title>My First PHP Page</title>
</head>
<body>
    <h1>Welcome to My Website</h1>
    <p><?php echo "This is generated using PHP!";
?></p>
</body>
</html>
```

**Key Features of PHP:**

- Simple and easy to learn.
- Can handle forms, process data, and interact with databases.
- Supports cookies, sessions, and file handling.

**PHP Syntax Rules:**
Starts with <?php and ends with ?>
Each statement ends with ;
Case-sensitive for variables, NOT case-sensitive for functions

**PHP Comments:**

```
// This is a single-line comment
# Another single-line comment
/*
   This is a multi-line comment.
   Used for documentation.
*/
```

**Understanding How PHP Works**

- The client (browser) sends a request to the server.
- The server processes the PHP file and executes PHP code.
- The server sends the generated **HTML output** back to the browser.

**Illustration of the PHP Execution Process:**
User Requests a Page → http://localhost/page.php
Server Reads the PHP File (page.php)
PHP Code is Executed on the Server
Output (HTML) is Sent Back to the Browser

**Example:**

```
<?php
echo "Welcome to my PHP page!<br>";
echo "Today's date is: " . date("Y-m-d H:i:s");
?>
```

**Run the file in the browser** → http://localhost/welcome.php

**Summary**

Web development consists of front-end and back-end.
PHP is a powerful server-side language for dynamic web applications.
XAMPP is used to run PHP locally.
PHP scripts run on the server and return HTML to the browser.

**Homework**

**Create a simple PHP script that:**

1. Displays your name.
2. Prints today's date.

## Lecture 2: PHP Basics - Variables and Data Types

**Objectives:**

- Understand how variables work in PHP.
- Learn about different data types in PHP.
- Perform basic operations using variables.
- Learn best practices for naming and using variables.

**Introduction to Variables**

**What is a Variable?**

- A **variable** is a container for storing data.
- In PHP, a variable starts with a **$** sign, followed by the variable name.

**Declaring and Assigning Variables in PHP:**

```
<?php
$name = "Alice"; // String
$age = 25; // Integer
$price = 10.5; // Float
$isStudent = true; // Boolean
echo "My name is $name and I am $age years old.";
?>
```

**Rules for Naming Variables:**
Must start with $ followed by a letter or underscore.
Can contain letters, numbers, and underscores (_).
**Case-sensitive** ($name is different from $Name).
Cannot use PHP reserved keywords (e.g., $echo, $if).

**Valid vs. Invalid Variable Names:**
✓ $user_name, $age25, $_price
x $123name, $user-name, $echo

## Data Types in PHP

**PHP has 8 primary data types:**

| Data Type | Example |
|---|---|
| **String** | "Hello, PHP!" |
| **Integer** | 25 |
| **Float (Double)** | 10.5 |
| **Boolean** | true or false |
| **Array** | ["Apple", "Banana", "Cherry"] |
| **Object** | new Car() |
| **NULL** | NULL |
| **Resource** | Database connection |

**Example Code:**

```
<?php
$var1 = "Hello, PHP!"; // String
$var2 = 100; // Integer
$var3 = 3.14; // Float
$var4 = true; // Boolean
$var5 = NULL; // Null
echo gettype($var1); // Outputs: string
?>
```

**Checking Data Types:**

- gettype($variable): Returns the type of the variable.
- var_dump($variable): Displays detailed information (type & value).

```
<?php
$val = 42;
var_dump($val); // Outputs: int(42)
?>
```

## Type Casting and Type Juggling

**PHP automatically converts data types based on context (Type Juggling).**
**You can manually convert data types (Type Casting).**

**Example of Type Juggling:**

```
<?php
$x = "5" + 10; // PHP converts "5" to integer
automatically
echo $x; // Outputs: 15
?>
```

**Example of Type Casting:**

```
<?php
$number = "42"; // String
$converted = (int) $number; // Converts to integer
echo gettype($converted); // Outputs: integer
?>
```

## Basic Arithmetic and String Operations

**Arithmetic Operators:**

| Operator | Description | Example |
|---|---|---|
| + | Addition | $x + $y |
| - | Subtraction | $x - $y |
| * | Multiplication | $x * $y |
| / | Division | $x / $y |
| % | Modulus | $x % $y |

**Example Code:**

```
<?php
$a = 10;
$b = 3;
echo "Sum: " . ($a + $b); // Outputs: Sum: 13
?>
```

**String Concatenation (. operator):**

```
<?php
$firstName = "John";
$lastName = "Doe";
$fullName = $firstName . " " . $lastName;
echo $fullName; // Outputs: John Doe
?>
```

**String Interpolation (Using Variables Inside Strings):**

```
<?php
$name = "Alice";
echo "Hello, $name!"; // Outputs: Hello, Alice!
?>
```

**Constants in PHP**

A constant is a variable whose value cannot be changed once defined. Define a constant using define() or const.

**Example using define():**

```
<?php
define("SITE_NAME", "MyWebsite");
echo SITE_NAME; // Outputs: MyWebsite
?>
```

**Example using const:**

```
<?php
const PI = 3.14;
echo PI; // Outputs: 3.14
?>
```

**Constants vs. Variables:**

| Feature | Variable | Constant |
|---|---|---|
| Starts with | $name | define("NAME", "value") |
| Value can be changed | Yes | No |
| Case-sensitive | Yes | Yes |

**Example:** Declares variables for name, age, and city and concatenates them into a sentence.. Then, displays the current year using a constant.

```php
<?php
$name = "Alice";
$age = 25;
$city = "London";
define("CURRENT_YEAR", 2025);
echo "My name is $name, I am $age years old, and I
live in $city.<br>";
echo "The current year is " . CURRENT_YEAR;
?>
```

**Run the file in the browser** → http://localhost/info.php

**Summary**

Variables store data in PHP.
PHP supports different data types like strings, integers, and arrays.
PHP automatically converts types when needed (Type Juggling).
Constants store fixed values that don't change.
Arithmetic and string operations are essential for dynamic programming.

**Homework Assignment**

**Create a PHP script that:**

1. Stores your first name, last name, and birth year in variables.
2. Calculates your age dynamically based on the current year.