

Week 1 Training Notes: Foundations & Programming Basics

Day 1: Introduction & Environment Setup

Morning Session (3 hours)

1. What is Programming?

Programming is the process of writing instructions that a computer can understand and execute. These instructions tell the computer how to solve problems by breaking them down into smaller, manageable steps.

- Computers follow instructions exactly.
- Programmers must think logically and sequentially.
- Programming helps automate tasks and build applications (web, mobile, desktop, embedded systems).

2. Breaking Down Problems into Steps

Before writing code, you plan the solution:

1. Identify the problem.
2. Break it into small steps (algorithm).
3. Convert the steps into code.
4. Test and debug.

Example: Making tea

- Boil water
- Add tea leaves
- Add sugar and milk
- Stir and serve

Programming works the same way: step-by-step.

3. How the Web Works

The web follows the **Client-Server Model**:

- **Client:** The browser (Chrome, Firefox). Sends requests.
- **Server:** A computer hosting files or applications. Sends responses.
- **HTTP:** The communication protocol used between client and server.

Example:

- You type `http://localhost/hello.php`
- Browser (client) sends request
- Apache (server) receives request and delivers the PHP output

4. Introduction to PHP

PHP (Hypertext Preprocessor):

- Server-side scripting language
- Used to build dynamic websites
- Works well with HTML
- Popular in CMS (WordPress, Joomla), frameworks (Laravel)

Why PHP?

- Easy to learn
- Widely used
- Fast for backend development
- Works well on shared hosting

5. Understanding the Development Environment

You need:

- A server (Apache)
- A database (MySQL)

- A scripting engine (PHP)
- A local environment like **Laragon** to run everything

Practical

1. Install **Laragon**
 2. Start Apache & MySQL
 3. Visit `http://localhost/`
 4. Create folder: `C:\laragon\www\training`
-

Afternoon Session (2 hours)

1. Installing VS Code

A lightweight code editor suitable for PHP development.

2. VS Code Extensions

- **PHP Intelephense** – code intelligence
- **Prettier** – auto-formatting
- **Bracket Pair Colorizer** – color-coded brackets

3. Browser Developer Tools

Chrome DevTools helps you:

- Inspect HTML elements
- Check console errors
- Debug frontend issues

4. Understanding File Structure

Inside `training/`:

- PHP files
- Asset folders (css, js, images)

Practical

Create `hello.php`:

```
<!DOCTYPE html>
<html>
<body>
    <?php echo "Hello World"; ?>
</body>
</html>
```

View it in browser and inspect with DevTools.

Homework

- Create 3 PHP files with different greetings
 - Experiment with HTML tags around PHP
-

Day 2: PHP Syntax & Variables

Morning Session (3 hours)

1. PHP Syntax Rules

- Files end with `.php`
- PHP code starts with `<?php` and ends with `?>`
- Every statement ends with a `;`
- PHP is **case-sensitive** for variables but not for functions

2. Comments

- Single-line: `// comment`
- Multi-line: `/* comment */`
- Documentation: `/** comment */`

3. Variables

Rules:

- Start with \$
- Must start with letter or underscore
- Cannot start with number
- Case-sensitive

Examples:

```
$name = "John";  
$age = 20;
```

4. Data Types

- String: "Hello"
- Integer: 30
- Float: 5.7
- Boolean: true/false

Debugging Tools

- `var_dump()` – shows type + value
- `print_r()` – readable format

Practical Exercises

Create variables:

- Name (string)
 - Age (integer)
 - Height (float)
 - Student status (boolean)
- Display using echo and var_dump.

Afternoon Session (2 hours)

5. Operators

- Arithmetic: + - * / %
- Assignment: = += -=
- Comparison: == === != > < <= >=
- Logical: && || !

6. String Concatenation

Use . operator:

```
$full = $first . " " . $last;
```

7. Type Juggling & Casting

```
$number = "20" + 10; // becomes 30
```

```
(int) $value
```

8. Constants

```
define("SCHOOL", "Smart School Africa");  
const PI = 3.14;
```

Practical Exercises

- Calculator program
- String combination
- Comparison operations

Homework

- Build a bio page
 - Perform 10 math operations
 - Practice comparison operators
-

Day 3: Control Structures (Conditional Statements)

Morning Session (3 hours)

1. If Statements

```
if ($score > 50) {  
    echo "Passed";  
}
```

2. If-Else

```
if ($age >= 18) {  
    echo "Adult";  
} else {  
    echo "Minor";  
}
```

3. If-Else-If Chain

Used when checking multiple conditions.

4. Nested If

An if statement inside another.

5. Ternary Operator

```
$result = ($age >= 18) ? "Adult" : "Minor";
```

Practical Exercises

- Number checker
- Grade calculator
- Age verification
- Simple login

Afternoon Session (2 hours)

6. Switch Statement

Used when comparing one value to many cases.

```
switch ($day) {  
    case 1: echo "Monday"; break;  
    default: echo "Invalid";  
}
```

7. When to Use Switch

- Clean alternative to long if-else chains.

Practical Exercises

- Day of week
- Menu system
- Season identifier

Homework

- BMI calculator
 - Five-question quiz
 - Traffic light simulator
-

Day 4: Loops & Iteration

Morning Session (3 hours)

1. While Loop

Repeats while condition is true.

2. Do-While Loop

Executes at least once.

3. For Loop

Used when number of iterations is known.

4. Loop Control

- `break` – exit loop
- `continue` – skip iteration

5. Avoiding Infinite Loops

Always update counters.

Practical Exercises

- Print 1–100
 - Print even numbers
 - Countdown
 - Multiplication table
 - Sum of 100 numbers
-

Afternoon Session (2 hours)

6. Nested Loops

Loop inside another loop.

Used for patterns and tables.

Practical Exercises

- Pyramid pattern
- Times table (1–10)
- Prime numbers 1–100

Homework

- Fibonacci (20 numbers)
 - Pattern generator
 - Factorial calculator
-

Day 5: Arrays (Indexed)

Morning Session (3 hours)

1. What Are Arrays?

Arrays store multiple values in one variable.

2. Creating Indexed Arrays

```
$fruits = ["Apple", "Banana", "Orange"];
```

3. Accessing Elements

```
echo $fruits[0];
```

4. Array Length

```
count($fruits);
```

5. Looping Arrays

```
foreach ($fruits as $fruit) {  
    echo $fruit;  
}
```

Practical Exercises

- Favorite foods
 - Student names (longest name)
 - Sum & average of numbers
 - Manual reverse
-

Afternoon Session (2 hours)

6. Array Functions

- `array_push()` – add to end
- `array_pop()` – remove from end
- `array_shift()` – remove first
- `array_unshift()` – add to start
- `in_array()` – check value
- `array_search()` – find index
- `sort() / rsort()` – sort
- `shuffle()` – randomize

- `array_merge()` – combine
- `array_slice()` – extract

Practical Exercises

- Student list manager
- Number sorter
- Search in array
- Array manipulation challenge

Homework

- Shopping cart system
- Todo list operations
- Number analysis tool (min, max, median)