ScopeTreasure Courses

PHP COURSES · LESSON PH.2 · CODE SCOPE SERIES

Syntax and Variables

Every PHP line starts with <?php and every variable starts with $ — learn the tags, echo, comments, and how to mix PHP into HTML.

Introduction

Every language has a grammar, and PHP's grammar is small. A PHP file opens with `<?php`, closes with `?>`, and every instruction ends with a semicolon `;`. That is almost the whole skeleton — the rest is vocabulary.

A **variable** is a named box that holds a value. In PHP every variable name starts with a dollar sign: `$name`, `$price`, `$age`. The dollar sign is not decoration — it is how PHP knows "this is a variable". Forget it once and PHP thinks you mean plain text.

PHP is loosely typed: you do not declare a variable's type. Put text in the box and it holds text; put a number in and it holds a number. The same box can even change what it holds later — useful, but also a reason to name things clearly.

(Roman Urdu: Har variable `$` se shuru hota hai — `$name` matlab "naam wala dabba". Har line ke aakhir mein `;` lazmi hai.)

In this lesson you will write your first variables, print them, join them into sentences, and learn the three small rules that prevent most beginner errors: the dollar sign, the semicolon, and the quotes.

Key points

- **A PHP block opens with `<?php` and closes with `?>`.** Code outside the tags is sent to the browser as plain HTML.
- **Every variable starts with `$`.** `$name = "Ali";` stores the text Ali in the box called name.
- **Every statement ends with `;`.** Forgetting it is the single most common PHP error.
- **`=` stores a value.** `$age = 20;` puts 20 into the box; it does not ask a question.
- **`echo` prints one or more things.** `echo $name;` prints the value inside the box.
- **Double quotes read variables inside them; single quotes do not.** `"Hello $name"` prints Hello Ali; `'Hello $name'` prints Hello $name.
- **Variable names are case-sensitive.** `$Name` and `$name` are two different boxes.
- **A variable name must start with a letter or underscore**, then letters, numbers or underscores. `$2price` is an error.
- **Comments explain code to humans.** `//` for one line, `/* ... */` for many lines; PHP skips both.
- **`print` also prints, but `echo` is the usual choice** — it is slightly faster and can print several items separated by commas.

Examples

Example 1 — Your first variables

```
<?php
$name = "Ali";
$age = 20;
$city = "Sahiwal";
echo $name;
echo " is ";
echo $age;
echo " years old.";
?>
```

What you will get:

```
Ali is 20 years old.
```

Three boxes, three values. `echo` prints each one in turn, with no spaces or new lines added by itself — the spaces come from the strings you printed.

Example 2 — Double quotes read the variable, single quotes do not

```
<?php
$name = "Ali";
echo "Hello, $name!";
echo "<br>";
echo 'Hello, $name!';
?>
```

What you will get:

```
Hello, Ali!
Hello, $name!
```

`<br>` is an HTML line break — PHP prints it and the browser breaks the line. Inside double quotes PHP replaces `$name` with its value; inside single quotes it prints the dollar sign and the name exactly as written.

Example 3 — One box, new value

```
<?php
$price = 250;
echo $price;
echo "<br>";
$price = 300;
echo $price;
?>
```

What you will get:

```
250
300
```

A variable is a box you can refill. The old value is gone the moment the new one is stored.

Example 4 — Maths with variables

```
<?php
$price = 250;
$qty = 3;
$total = $price * $qty;
echo "Total: $total";
?>
```

What you will get:

```
Total: 750
```

PHP works out `$price * $qty` first, stores 750 in `$total`, and then the double-quoted string prints the value.

Example 5 — Comments keep code readable

```
<?php
// Price of one cup of chai
$price = 250;
// Three cups ordered
$qty = 3;
/* Total bill:
   price times quantity */
$total = $price * $qty;
echo "Total: $total";
?>
```

What you will get:

```
Total: 750
```

The comments change nothing on the page. They are there for the next person who reads the code — which is often you, three weeks later.

Quick recap

- PHP blocks run between `<?php` and `?>`; every statement ends with `;`.
- Variables start with `$` and are case-sensitive.
- `=` stores a value; `echo` prints it.
- Double quotes replace variables with their values; single quotes do not.
- Comments (`//` and `/* */`) are for humans, never shown on the page.

Common mistakes

1. **Forgetting the `$`** — `name = "Ali";` is an error. Fix: every variable starts with `$`.
2. **Forgetting the semicolon** — the next line is glued to the previous one and PHP reports a parse error. Fix: end every statement with `;`.
3. **Using single quotes and expecting the value** — `'Hello, $name!'` prints the dollar sign. Fix: use double quotes when a variable must be replaced.
4. **Mixing up `$Name` and `$name`** — PHP treats them as different boxes, so one looks empty. Fix: pick one spelling and use it everywhere.
5. **Starting a name with a number** — `$2price` is an error. Fix: start with a letter or underscore, like `$price2`.
6. **Writing `echo` outside the PHP tags** — plain `echo` in the HTML part is printed as the word echo. Fix: `echo` only works inside `<?php ... ?>`.

Practice

1. Make a page with three variables — your name, your age, your city — and print one sentence from them.
2. Print the same sentence twice: once with double quotes, once with single quotes. Write down the difference.
3. Store a price and a quantity, work out the total, and print "Total: " followed by the total.
4. Change the price variable after printing it once, and print it again — confirm the new value shows.
5. Deliberately leave out a semicolon, read the error, fix it, and write down in one line what the error taught you.
6. Add a `//` comment above each variable explaining what it holds, then a `/* */` comment explaining the whole page.
7. Try `$Name = "Sara";` and `$name = "Ali";` on one page and print both — confirm they are two different boxes.

Practice quiz

1. How does every PHP variable start?

- With a # sign
- With a $ sign
- With a @ sign
- With a capital letter

Answer: With a $ sign

2. What must end every PHP statement?

- A full stop .
- A comma ,
- A semicolon ;
- A colon :

Answer: A semicolon ;

3. What is the difference between double quotes and single quotes?

- There is none
- Double quotes replace variables with their values; single quotes print them as written
- Single quotes replace variables, double quotes do not
- Double quotes are for numbers only

Answer: Double quotes replace variables with their values; single quotes print them as written

Watch the Code Scope channel: https://www.youtube.com/@CodeScopePK — this lesson's own video joins the channel when it is recorded.

Read online: https://scopetreasure.com/notes/ph-syntax-variables · More free tools and notes at https://scopetreasure.com