**1. Classes and Objects**

A **class** is a blueprint that defines the structure and behavior of something, while an **object** is an actual instance created from that blueprint.

**Example:**

class Car {

public $brand;

public $color;

public function drive() {

echo "The $this->color $this->brand is driving.";

}

}

$car1 = new Car();

$car1->brand = "Toyota";

$car1->color = "Red";

$car1->drive();

**2. Properties and Methods**

* **Properties** are variables inside a class that hold data.
* **Methods** are functions inside a class that define behavior.
* Access them using the **arrow operator (->)**.

**Example:**

class Student {

public $name;

public function sayHello() {

echo "Hello, my name is $this->name.";

}

}

$student = new Student();

$student->name = "Jerico";

$student->sayHello(); // Output: Hello, my name is Jerico.

**3. Access Modifiers (Visibility)**

Access modifiers control where properties and methods can be used.

| **Modifier** | **Inside Class** | **Subclass** | **Outside Class** |
| --- | --- | --- | --- |
| public | ✅ Yes | ✅ Yes | ✅ Yes |
| protected | ✅ Yes | ✅ Yes | ❌ No |
| private | ✅ Yes | ❌ No | ❌ No |

**Example:**

php

CopyEdit

class BankAccount {

private $balance = 1000;

public function getBalance() {

return $this->balance;

}

}

$account = new BankAccount();

echo $account->getBalance(); // ✅ Works

echo $account->balance; // ❌ Error

**4. Static Properties and Methods**

Static members belong to the class itself, not to objects.

* Accessed using ClassName::$property or ClassName::method().

**Example:**

php

CopyEdit

class MathHelper {

public static $pi = 3.14;

public static function square($num) {

return $num \* $num;

}

}

echo MathHelper::$pi; // 3.14

echo MathHelper::square(5); // 25

**5. $this vs self**

* $this → Refers to the **current object instance**.
* self → Refers to the **class itself** (used for static properties/methods).

**Example:**

php

CopyEdit

class Example {

public $message = "Hello";

public function showMessage() {

echo $this->message;

}

public static function showStaticMessage() {

echo self::class;

}

}

$obj = new Example();

$obj->showMessage(); // Hello

Example::showStaticMessage(); // Example

If you want, I can now prepare the **next lesson on the Four Pillars** so it connects smoothly after this foundation.