OOP PHP

PiTechnologies

Agenda

PHP

- Introduction to OOP
- Features of OOP
- Structuring a Class
- Using Constructors and Destructors

Agenda

PHP

- Introduction to OOP
- Features of OOP
- Structuring a Class
- Using Constructors and Destructors

Introduction to OOP

- **What is OOP?**
  - Object-oriented programming is a style of coding that allows developers to group similar tasks into classes.
  - This helps keep code following the tenet “don’t repeat yourself” (DRY) and easy-to-maintain.

Introduction to OOP

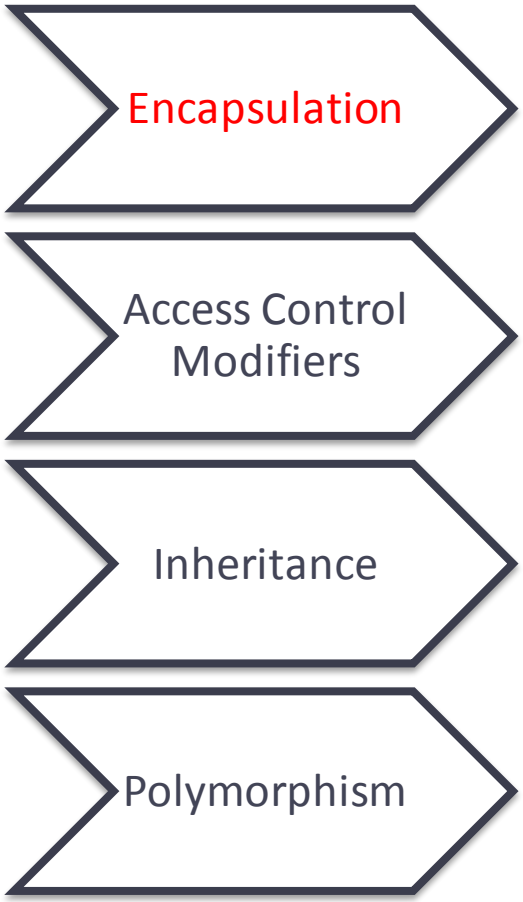
- **What does OOP aim to achieve?**
  - Allow compartmentalized refactoring of code.
  - Promote code re-use.
  - Promote extensibility, flexibility and adaptability.
  - Better for team development.
  - Many patterns are designed for OOP.
  - Some patterns lead to much more efficient code.

Agenda

PHP

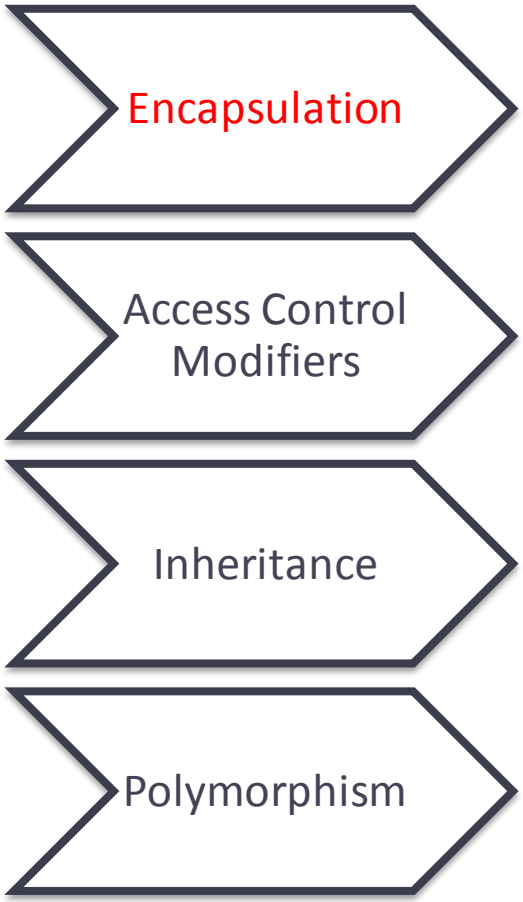
- Introduction to OOP
- Features of OOP
- Structuring a Class
- Using Constructors and Destructors

Features of OOP



- Encapsulation is about grouping of functionality (operations) and related data (attributes) together into a coherent data structure (classes).
- Classes represent complex data types and the operations that act on them. An object is a particular instance of a class.

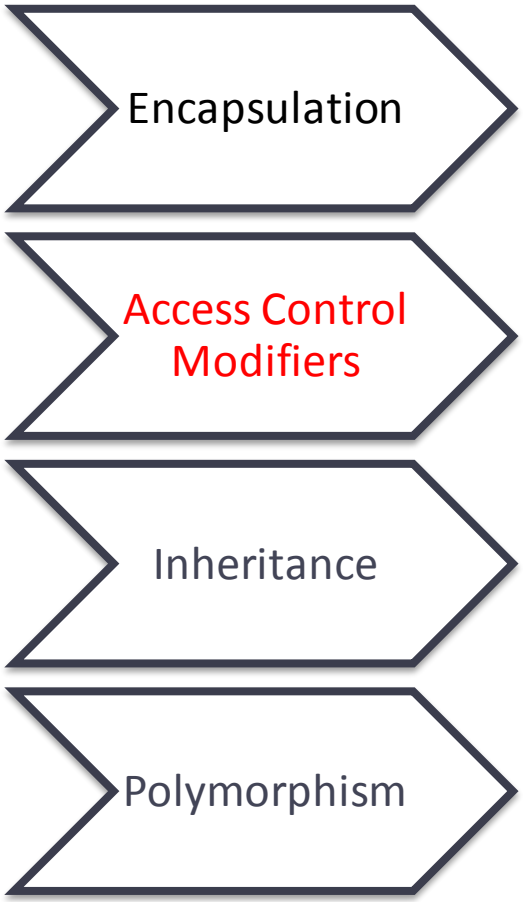
Features of OOP




```
class Humans{
        public function __construct($name){
                /*...*/
        }
        public function eat() { /*...*/}
        public function sleep() { /*...*/}
        public function snore() { /*...*/}
        public function wakeup() { /*...*/}
}
```

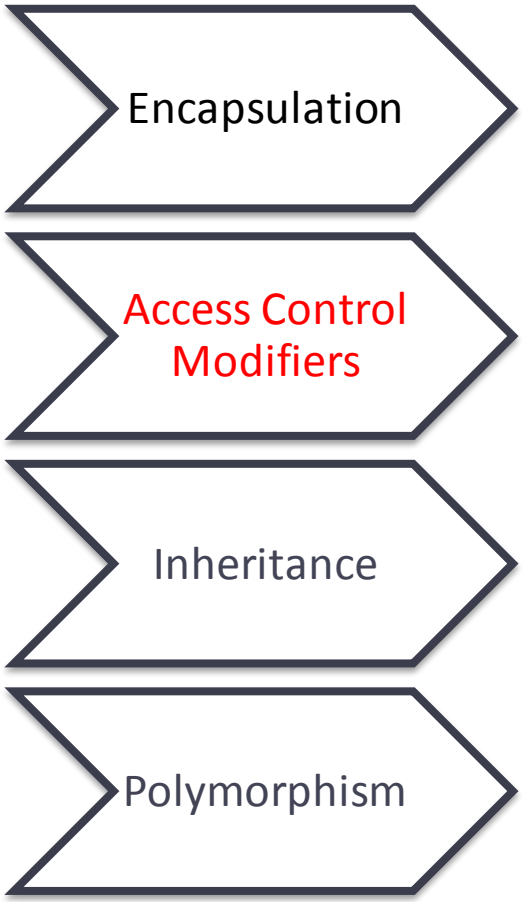
Technologies Group

Features of OOP



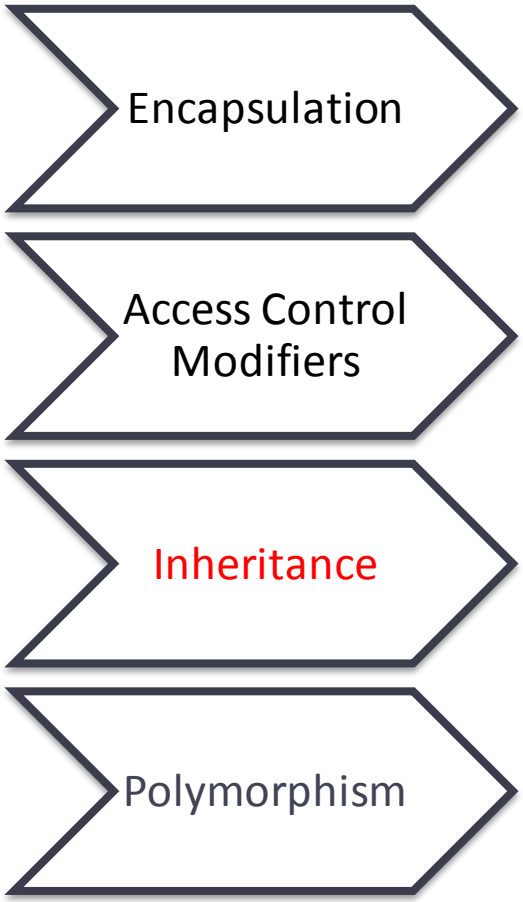
- **Public**: This property or method can be used from anywhere in the script
- **Private**: This property or method can be used only by the class or object it is part of; it cannot be accessed elsewhere
- **Protected**: This property or method can be used only by code in the class it is part of, or by descendants of that class

Features of OOP



- **Final**: This property, method, or class cannot be overridden in subclasses
- **Abstract**: This method or class cannot be used directly you have to subclass this

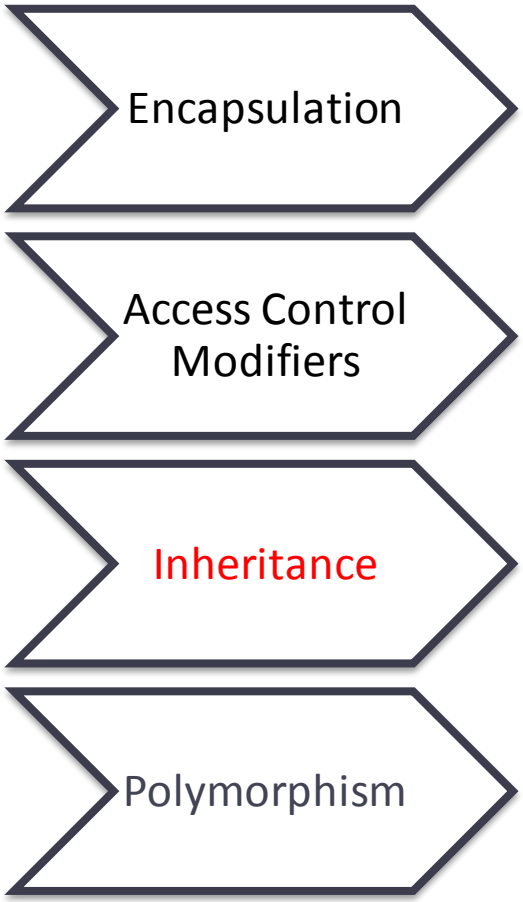
Features of OOP



- Inheritance allows a class to specialize (or extend) another class and inherit all its methods, properties and behaviors.

- **This promotes**
  - Extensibility
  - Reusability
  - Code Consolidation
  - Abstraction
  - Responsibility

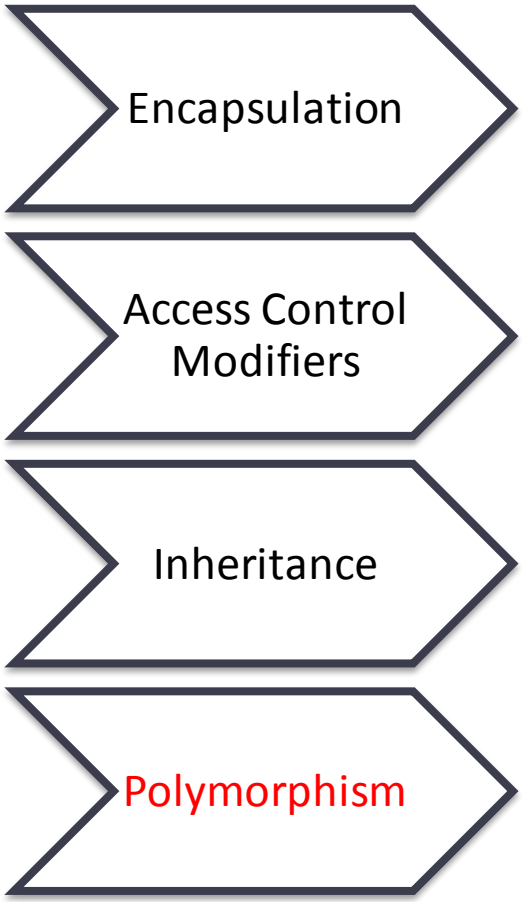
Features of OOP




```
class Humans{
        public function __construct($name){
                /*...*/
        }
        public function eat() { /*...*/}
        public function sleep() { /*...*/}
}
class Women extends Humans{
        public function giveBirth() { /*...*/ }
}
```

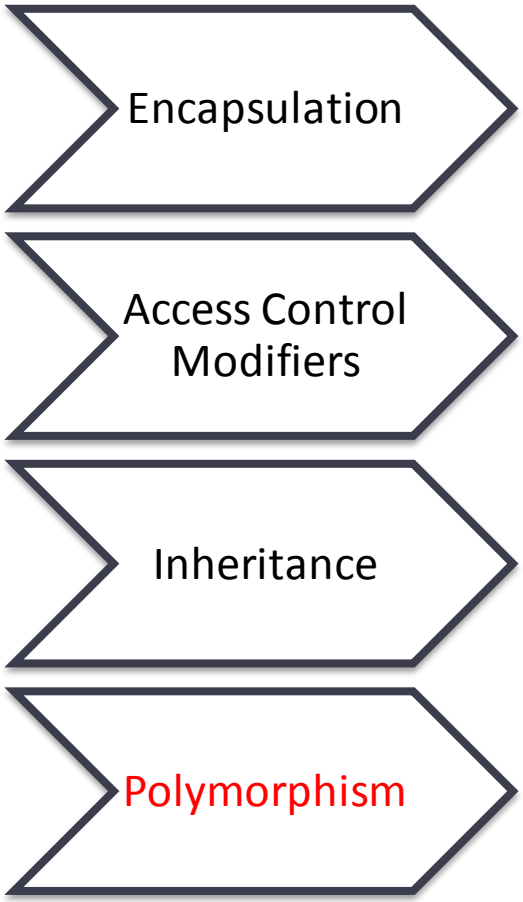
Technologies Group

Features of OOP



- Polymorphism describes a pattern in object oriented programming in which classes have different functionality while sharing a common interface.
- Polymorphism is used to make applications more modular and extensible.
- Instead of conditional statements describing different courses of action, you create interchangeable objects that you select based on your needs.

Features of OOP




```
class Humans{
        public function __construct($name){ /*...*/}
        public function eat() { /*...*/ }
        public function sleep() { /*...*/ }
        public function snore() { /*...*/}
        public function wakeup() { /*...*/ }
}
class Women extends Humans{
        public function giveBirth() { /*...*/ }
}
class Men extends Humans{
        public function snore() { /*...*/}
}
```

Technologies Group

Agenda

PHP

- Introduction to OOP
- Features of OOP
- Structuring a Class
- Using Constructors and Destructors

Structuring a Class

- The syntax to create a class is pretty straightforward: declare a class using the class keyword, followed by the name of the class and a set of curly braces ({}):

```
class MyClass{
        // Class properties and methods go here
}
```

Structuring a Class

- After creating the class, a new class can be instantiated and stored in a variable using the new keyword:

```
$obj = new MyClass;
```

Structuring a Class

- **Defining Class Properties**
  - To add data to a class, properties, or class-specific variables, are used.
  - These work exactly like regular variables, except they're bound to the object and therefore can only be accessed using the object.

```
<?php
class MyClass
{
    public $prop1 = "I'm a class property!";
}
$obj = new MyClass;
?>
```

Structuring a Class

- **Defining Class Properties**
  - The keyword public determines the visibility of the property
  - To read this property and output it to the browser, reference the object from which to read and the property to be read:

```
<?php
class MyClass
{
    public $prop1 = "I'm a class property!";
}
$obj = new MyClass;
echo $obj->prop1;
?>
```

Structuring a Class

- **Defining Class Methods**
  - Methods are class-specific functions. Individual actions that an object will be able to perform are defined within the class as methods.

```
class MyClass
{
    public $prop1 = "I'm a class property!";
    public function setProperty($newval)
    {
        $this->prop1 = $newval;
    }
    public function getProperty()
    {
        return $this->prop1 . "<br />";
    }
}
```

Structuring a Class

- **Defining Class Methods**
  - To use these methods, call them just like regular functions, but first, reference the object they belong to.

```
$obj = new MyClass;
echo $obj->getProperty(); // Get the property value
$obj->setProperty("I'm a new property value!"); // Set a new one
echo $obj->getProperty(); // Read it out again to show the change
```

Agenda

PHP

- Introduction to OOP
- Features of OOP
- Structuring a Class
- Using Constructors and Destructors

Using Constructors and Destructors

- PHP provides the magic method __construct(), which is called automatically whenever a new object is created.

```php
<?php
class MyClass
{
    public $prop1 = "I'm a class property!";
    public function __construct()
    {
        echo 'The class was initiated!<br />';
    }
}
// Create a new object
$obj = new MyClass;
// Get the value of $prop1
echo $obj->getProperty();
?>
```

Using Constructors and Destructors

- To call a function when the object is destroyed, the __destruct() magic method is available.

```php
<?php
    class MyClass
    {
        public function __construct()
        {
            echo 'The class was initiated!<br />';
        }
        public function __destruct()
        {
            echo 'The class was destroyed.<br />';
        }
    }
    $obj = new MyClass;
    unset($obj);
?>
```

Questions