# OOP

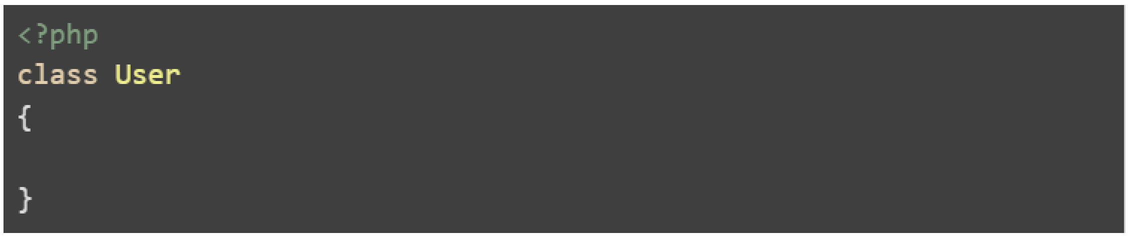
- **What is object-oriented programming in general terms**?

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data and code. The data is in the form of fields (often known as attributes or *properties*), and the code is in the form of procedures (often known as *methods*).

The OOP principles are Inheritance, Abstraction,Encapsulation and Polymorphism

- **What is a class?**

Basic class definitions begin with thekeyword class, followed by a class name, followed by a pair of curly braces which enclose the definitions of the properties and methods belonging to the class. The class name can be any valid label, provided it is not a PHP reserved word.



```php
<?php
class User
{

}
```

- **What is an object?**

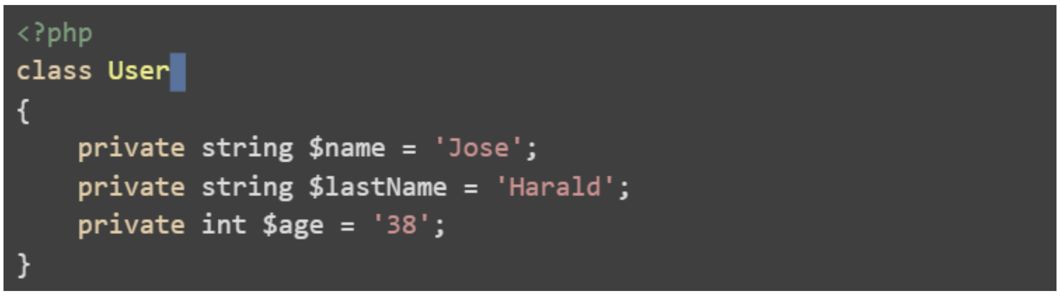
Objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process. Each object is an instance of a particular class or subclass with the class's own methods or procedures and data variables.

- **What is an instance?**

An instance is **a specific realization of any object**. An object may be different in several ways, and each realized variation of that object is an instance. The creation of a realized instance is called instantiation.

- **What is a property?**

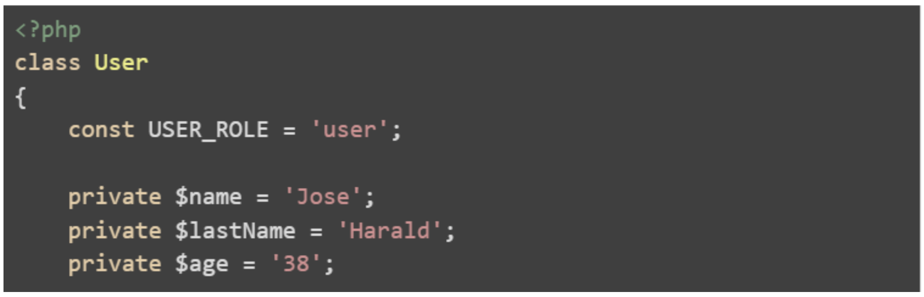
They are defined by using one of the keywords public, protected, or private, optionally followed by a type declaration, followed by a normal variable declaration.



```php
<?php
class User
{
    private string $name = 'Jose';
    private string $lastName = 'Harald';
    private int $age = '38';
}
```

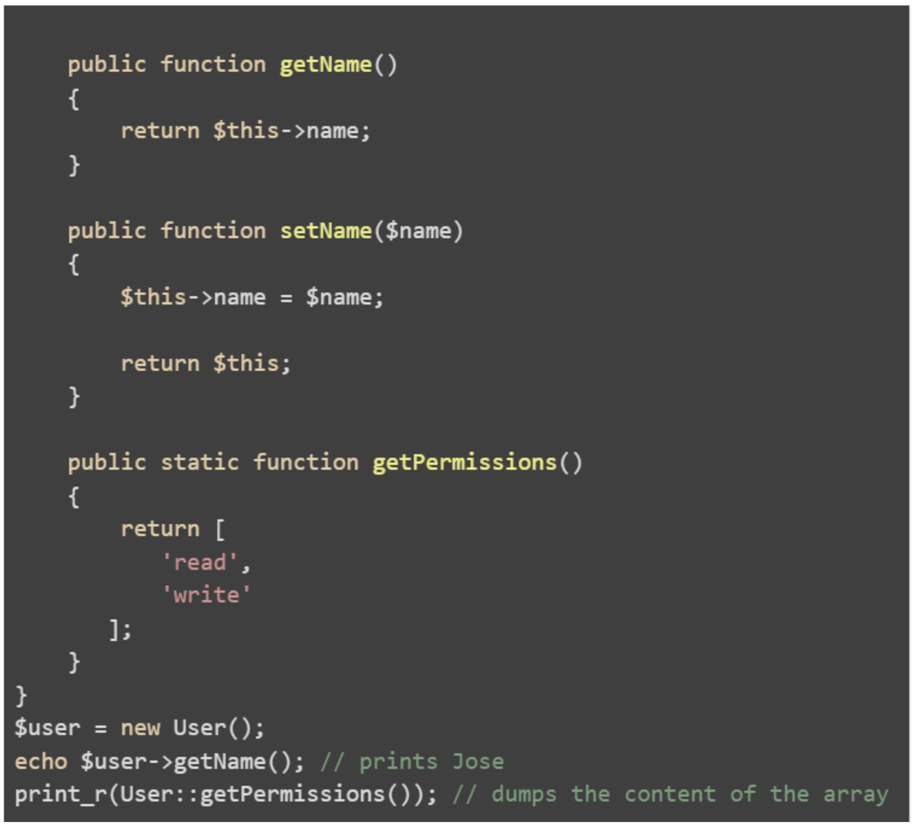
- **What is a method?**

Methods are functions inside classes. Their declaration and behavior are almost similar to normal functions, except for their special declaration and use inside classes.



```php
<?php
class User
{
    const USER_ROLE = 'user';

    private $name = 'Jose';
    private $lastName = 'Harald';
    private $age = '38';
```



```php

    public function getName()
    {
        return $this->name;
    }

    public function setName($name)
    {
        $this->name = $name;

        return $this;
    }

    public static function getPermissions()
    {
        return [
            'read',
            'write'
        ];
    }
}
$user = new User();
echo $user->getName(); // prints Jose
print_r(User::getPermissions()); // dumps the content of the array
```

- **What is the difference between a function and a method?**

A function is a set of instructions or procedures to perform a specific task, and a method is a set of instructions that are associated with an object.

- **What is a constructor?**

Since PHP 5, developers are allowed to declare constructor methods for classes. Classes with a constructor method call this method on each newly-created object, so it is suitable for any initialization that the object may need before it is used.

```php
<?php
class User
{
    const USER_FILES_PATH = './files/users/';

    private $name = 'Jose';
    private $lastName = 'Harald';
    private $age = '38';

    /**
     * User constructor.
     * @param string $name
     * @param string $lastName
     * @param string $age
     */
    public function __construct(string $name, string $lastName,
string $age)
    {
        $this->name = $name;
        $this->lastName = $lastName;
        $this->age = $age;
    }

}
```

- **What is the difference between a class, an object and an instance?**

An object is a software bundle of related state and behavior. A class is a blueprint or prototype from which objects are created. An instance is a single and unique unit of a class. Example, we have a blueprint (class) represents student (object) with fields like name, age, course (class member).

- **What do we understand about the concept of encapsulation**?

The visibility of a property, a method or a constant can be defined by prefixing the declaration with the keywords public, protected or private.

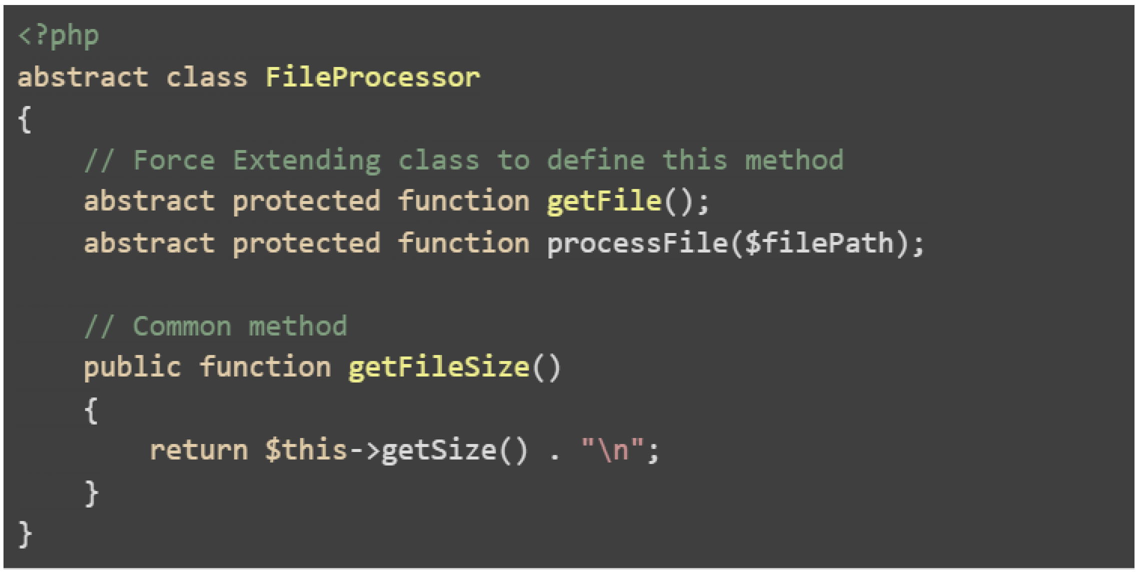
- **Public:** can be accessed everywhere.

- **Private**: may only be accessed by  the class that defines the member.

- **Protected:** can be accessed only within the class itself or by inheriting and parent classes.

- **What do we understand about the concept of abstraction?**

An abstract class is one that cannot be instantiated, only inherited. You declare an abstract class with the keyword abstract. When inheriting from an abstract class, all methods marked abstract in the parent's class declaration must be defined by the child.

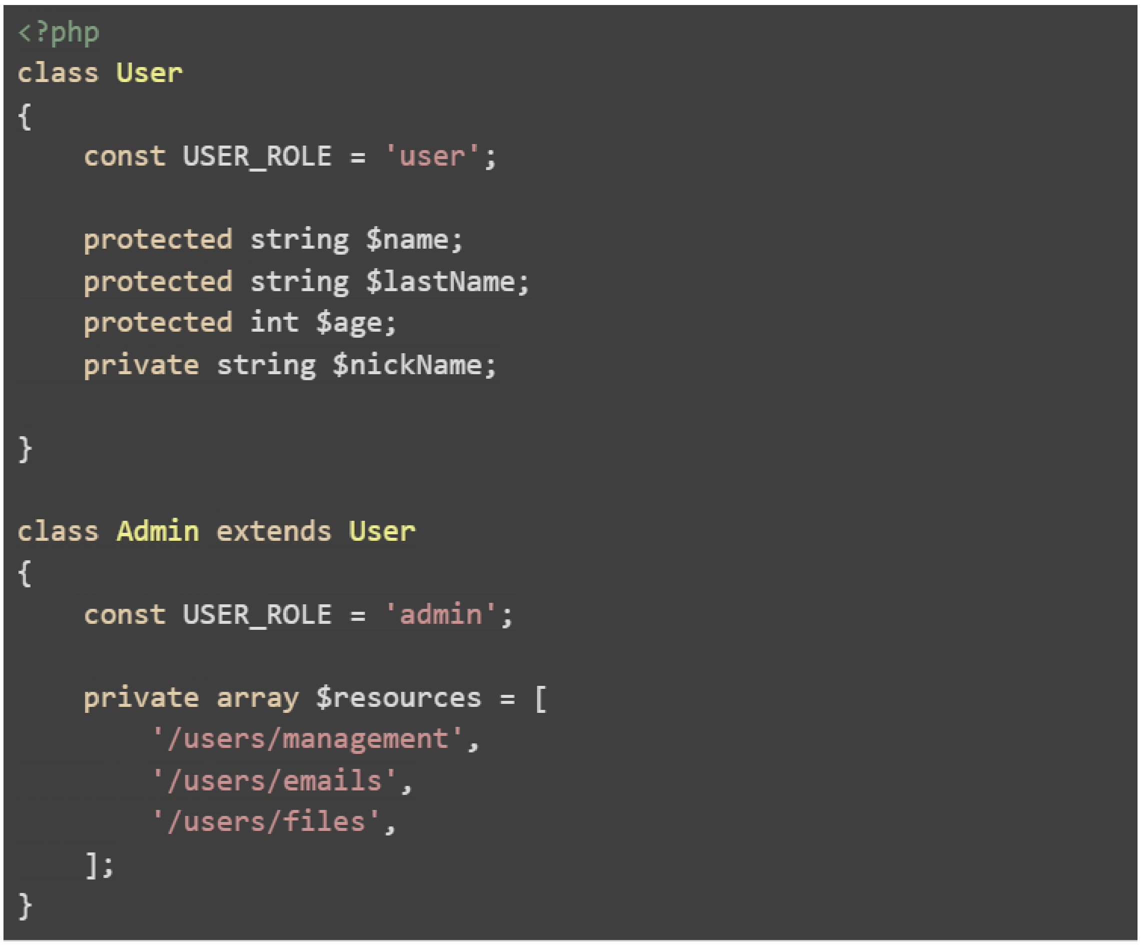


```php
<?php
abstract class FileProcessor
{
    // Force Extending class to define this method
    abstract protected function getFile();
    abstract protected function processFile($filePath);

    // Common method
    public function getFileSize()
    {
        return $this->getSize() . "\n";
    }
}
```

- **What do we understand about the concept of inheritance?**

When you extend a class, the subclass inherits all of the public and protected methods from the parent class. This allows the implementation of additional functionalities in similar objects without the need to reimplement and repeat code of all of that shared functionalities.



```php
<?php
class User
{
    const USER_ROLE = 'user';

    protected string $name;
    protected string $lastName;
    protected int $age;
    private string $nickName;

}

class Admin extends User
{
    const USER_ROLE = 'admin';

    private array $resources = [
        '/users/management',
        '/users/emails',
        '/users/files',
    ];
}
```

- **What do we understand about the concept of polymorphism**?

Polymorphism describes a pattern in Object Oriented Programming in which a class has varying functionality while sharing a common interfaces.

```php
class Manager extends User
{
    const USER_ROLE = 'manager';

    private array $resources = [
        '/users/management'
    ];
}

class Role
{
    public function addUser(User $user): void
    {
        array_push($roles, $user->getName());
    }
}

$roles = new Role();

$user = new User();
$roles->addUser($user);

$manager = new Manager();
$roles->addUser($manager);

$admin = new Admin();
$roles->addUser($admin);
```

```php
<?php
class User
{
    const USER_ROLE = 'user';

    public string $name = 'Jose';
    private $lastName = 'Harald';
    private $age = '38';

    public function getName()
    {
        return $this->name;
    }
}

class Admin extends User
{
    const USER_ROLE = 'admin';

    private array $resources = [
        '/users/management',
        '/users/emails',
        '/users/files',
    ];
}
```

- **What do we understand about the concept of Overload?**

Is the ability to create multiple functions of the same name with different implementations. Function/method overloading contains same function name and that function performs different task according to number of arguments.

Like other OOP languages function overloading can not be done by native

approach. In PHP function overloading is done with the help of magic function

__call().

This function takes function name and arguments.

- **What do we understand about the concept of Override?**

It is used to replace parent method in child class. The purpose of overriding is to change the behavior of parent class method. In function/method overriding, both parent and child classes should have same function name with and number of arguments. It is used to replace parent method in child class. The purpose of overriding is to change the behavior of parent class method. The two methods with the same name and same parameter is called overriding.

- **What differences exist between the concept of Overload and Override?**

Overriding occurs when the method signature is the same in the superclass and the child class. Overloading occurs when two or more methods in the same class have the same name but different parameters.

- **What is a static class?**

Declaring class properties or methods as static makes them accesible without an instantiation of the class. A property or method declared as static cannot be accessed with arrow operator. Double colon is the way to access them. For accessing a static element inside the class $this won’t be available, we need to use self.

- **Look for 3 advantages over object-oriented programming compared to other programming paradigms**

- Improved productivity: we can reuse code in an easy way and code faster

- High abstraction level: focused on the functionalities not in the code

- Rich encapsulation: makes secure and easy to maintain code

- Lower cost of development

Easy to read and understand code

- **Look for disadvantages of this paradigm.**

- Steep learning curve: may take time to get used to it

- Larger program size

- Slower programs