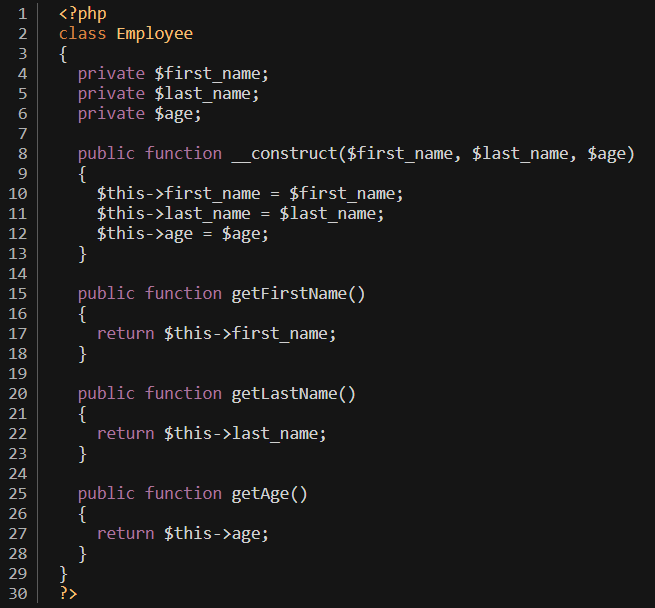
**Object-Oriented PHP With Classes and Objects Article brief**

**OOP** is a paradigm (approach) which helps you to develop complex applications in a way that's easily maintainable and scalable over the long term.

In object-oriented programming, you interact with your application by using objects.

**Objects or entity** is like person, Car, or Animal .It is an instance of a class, and you can create multiple instances of the same class. For example, there is a single**person** class, but many person objects can be instances of this class Maryam , Amr.

**Class which is used to model or map a real-world entity to a template of data (properties) and functionality (methods).**

****

**Class Properties in PHP**

class properties like variables that are used to hold information about the object.

**Constructors for PHP Classes**

**A constructor** is a special class method which is called automatically when you instantiate an object.it is used to initialize object properties when the object is being created.

You can define a constructor by defining the method. \_\_**construct**

**Methods for PHP Classes**

class methods in PHP as functions that perform specific actions associated with objects. In most cases, they are used to access and manipulate object properties and perform related operations.

**To create object in php:**

**Text

Description automatically generated**

* To instantiate an object of any class we use **new** keyword as we see in line 2
* If a class has defined the method and it requires arguments, we need to pass those arguments when we instantiate an object.

As we see above the constructor has required 3 parameters so we should pass 3 arguments

* we've called class methods on the object to print the information which was initialized during object creation.
* we can call multiple objects in the same class.

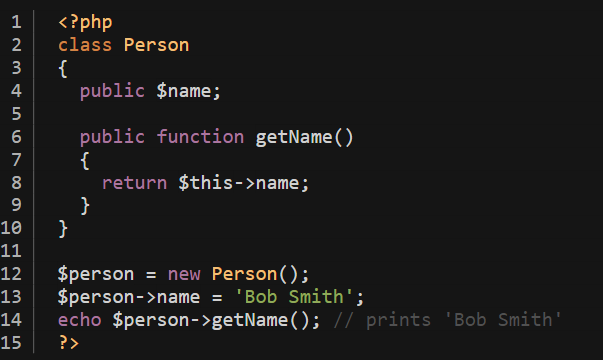
**Fundamental concepts of OOP**

1.**Encapsulation**: means that the object itself wraps together properties and methods of the class. In other words, it hides those details from the rest of the program. It is an important aspect of OOP that allows you to restrict access to certain properties or methods of the object.

**Access levels**

When you define a property or a method in a class, you can declare it to have one of these three access levels (Public, protected, private)

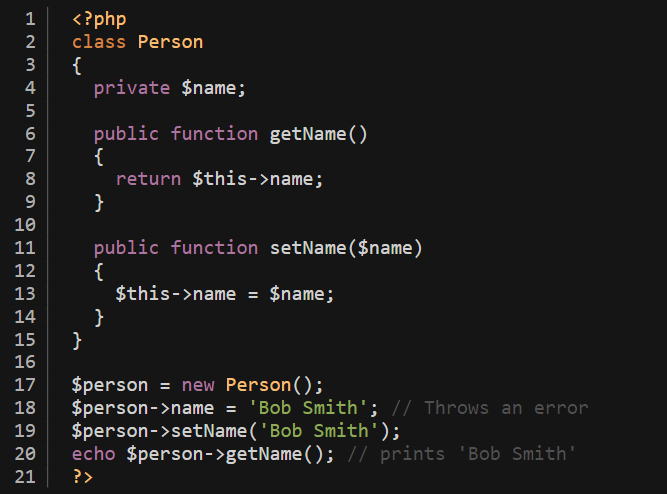
*Public Access* it can be accessed from anywhere outside the class. The value of a public property can be modified from anywhere in your code.



*Private Access* it can only be accessed from within the class. This means that you need to define getter and setter methods to get and set the value of that property.

If you try accessing a private property from outside the class, it'll throw the fatal error . Thus, you need to set the value of the private property using the setter method, as we did use the method.

There are good reasons why you might want to make a property private. For example, perhaps some action should be taken (updating a database, say, or re-rendering a template) if that property changes. In that case, you can define a setter method and handle any special logic when the property is changed.

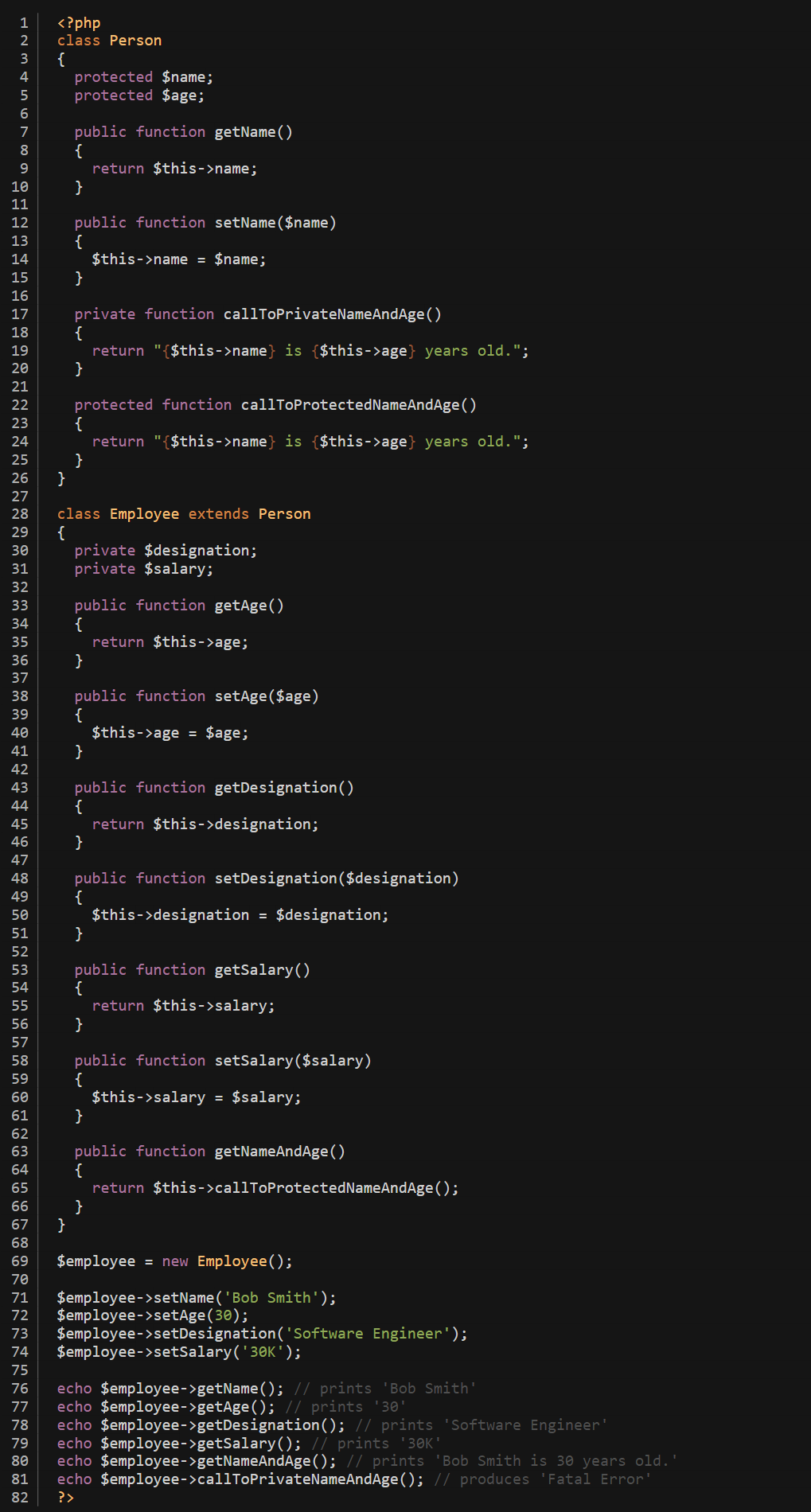


*Protected Access* it can be accessed by the same class that has defined it and classes that inherit the class in question.

2**. Inheritance**

Inheritance is an important aspect of the object-oriented programming paradigm which allows you to inherit properties and methods of other classes by extending them. The class which is being inherited is called the **parent** class, and the class which inherits the other class is called the **child** class. When you instantiate an object of the child class, it inherits the properties and methods of the parent class as well.

**In this example bellow the employee class inherit person class**

****

* When a class inherits another class using the keyword "extends", it means that the new class can access all the public and protected properties and methods of the parent class. This allows the new class to reuse code from the parent class and avoid duplicating code. However, the new class cannot access any private members of the parent class. Private members are only accessible within the parent class and cannot be accessed by any other classes, including inherited classes.

**3.polymorphism** is another important concept in the world of object-oriented programming which refers to the ability to process objects differently based on their data types.

For example, in the context of inheritance, if the child class wants to change the behavior of the parent class method, it can override that method. This is called method overriding. Let's quickly go through a real-world example to understand the concept of method overriding.

we've changed the behavior of the method by overriding it in the class. The important thing is that a message is formatted differently based on the object type, whether it's an instance of the parent class or the child class.

