

OOP

Overview

The aim is to understand the fundamental principles of the OOP

It's a tool that detects errors and won't let you move forward until you correct it

Table of contents

Phase I - Project Initiation	2
Project requirements	2
Project Specifications	2
Phase II - Project Planning	2
Reasoning	
Implementaciones3	2
Task planning	4
Tracking the project schedule5	
Git	6 Workflow
Tools	6
Phase III - Project Execution	7
Conceptos7	
Argumentos7	
Incidents	7
Lecciones7	
Phase IV - Closure Project	8
General Comments	8

Phase I - Project Initiation

Project requirements

- We will use PHP to do the practice of the OOP project
 - We have to install whether it's a development dependency or not.
 - Checking the new directory called Vendor

Project specifications

- ✚ The project must be developed in PHP or j.
- ✚ Create a Git repository
- ✚ Don't raise dependence
- ✚ The directory structure of the project must be well defined and organized.
- ✚ The code must be documented correctly using the English language.
- ✚ Your code must use a *camelCase style*.
- ✚ If you use HTML do not use inline styles
- ✚ The project must not contain unused files.
- ✚ The project must be developed using *git*, using explicit and concise deconfirmation messages.
- ✚ Delete files that are not necessary to evaluate the project
- ✚ The project must contain a *README* file written in *Markdown* that shows a brief description and the steps for runé.

Phase II - Project Planning

Reasoning

This pill will split 3 part implementation a case study about PHP performing the tests, explaining the presentation how it develops.

- Class, object and instance
- Encapsulation
- Abstraction
- Inheritance
- Static Class
- Polymorphism
- Overload
- Override

We will also conduct research earlier on the concepts to be clarified.

- What is **object-oriented programming** in general?
- What is a **class**?
- What is an **object**?
- What is an **instance**?
- What is a **property**?
- What is a **method**?
- What is the difference between a **function** and a **method**?
- What is meant by **builder**?
- What is the difference between a **class**, an **object** and an **instance**?
- What do we mean by the concept of **encapsulation**?
- What do we mean by the concept of **abstraction**?
- What do we mean by the concept of **inheritance**?
- What do we mean by the concept of **polymorphism**?
- What do we mean by the concept of **Overload**?
- What do we mean by the concept of **Override**?
- What are the differences between the concept of **Overload** and **Override**?
- What is a **static class**?
- Look for 3 advantages over object-oriented programming over other programming paradigms
- Look for disadvantages of this paradigm.

Arguments and examples.

Concepts for understanding the theoretic part of object-oriented programming

- Explains when you consider the use of **object-oriented programming** as necessary against other programming paradigms
- Explains the differences between a **class**, an **object** and an **instance**.
- Explain in which cases you consider it appropriate to make use of **encapsulation**.
- Explain in which cases you consider it appropriate to make use of **abstraction**.
- Explain in which cases you consider it appropriate to make use of **inheritance**.
- Explain in which cases you consider it appropriate to use a **static class**.
- Explain when you consider it appropriate to make use of **polymorphism**.

Implementation of the theory

Of the concepts we need to clarify

General terms object-oriented programming

It is a programming paradigm, dividing everything into classes, objects, entities, allows us to realize

an structured and organized program, comes to innovate the way we get results. Objects manipulate input data for obtaining specific output data, where each object offers special functionality.

Classes

Classes are used to represent entities or concepts, such as nouns in the language. Each class is a model that defines a set of variables - the state, and appropriate methods for operating on that data - the behavior. Each object created from the class is called an instance of the class.

Objects

An **object** is an object-oriented computer program thing that consists of a state and behavior, which in turn consist of stored data and tasks that can be performed at run time respectively.

Instance

It is an object created from a class

Method

It is a set of statements associated with a class, which perform a certain task and can be invoked by a name

Property

They are like 'variables' that can take a single or multiple value and can be of a different type or

Method

It is a set of statements associated with a class, which perform a certain task and can be invoked by a name

Difference between methods and functions

Functions are defined outside classes. Methods are defined within and are part of the classes.

What is meant by **builder**?

a **constructor** is a subroutine whose mission is to initialize an object of a class. The constructor assigns the initial values of the new object.

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

The set of messages that an **object** can respond to is called the **object** protocol. **Instance:** Any object that derives from some other **object** is called. In this way, all **objects** are **instances** of some other, except the Object class that is the mother of all.

What do we mean by the concept of **encapsulation**?

It is the process of storing data as methods, in which you can manipulate or change this data. It consists of separating the external aspects of an object (can be accessed from other objects)

What do we mean by the concept of **Abstraction**?

It is a class that contains at least one abstract method. Abstract method is the one that is declared but not implemented. It is possible to create objects of abstract classes, its goal is to define formatting of methods for the subclasses that should over-establish them. It is oriented to the

Heritage Class Estática

What do we mean by the concept of **inheritance**?

Inheritance makes it easy to create objects from existing ones and implies that a child subclass gets all the behavior of the methods of another parent class. You can inherit from another parent class

What do we mean by the concept of **polymorphism**?

It allows the same method name to represent different code, as a result it can express many different behaviors

What do we mean by the concept of **Overload**?

Php overload provides the means to dynamically create properties and methods. These dynamic entities are processed by magic methods that can be set in a class for various actions.

What do we mean by the concept of **Override**?

It is a feature that allows a subclass or child class to provide a specific implementation of a method that is already provided by one of its parent classes or superclasses.

What are the differences between the concept of **Overload** and **Override**?

Overload is the overload of methods, is that in the same class you can have two methods that are called in the same way but that are differentiated through the amount of parameters. Override is method overwrite is used when inheriting the methods of a class and an m all redefines in the daughter class this matters more when you combine it with polymorphism

What is a **static class**?

It allows the same method name to render different code, as a result it can express many different behaviors.

Advantages of using OOP

The first advantage of the concept of objects is that all the code that has something to do with spacecraft is in one place.

Another advantage is that objects can have inherent attributes of the class to which they belong, for example, spacecraft and asteroids could both have an XY position because all objects belonging to the class of moving objects have a XY position.

Another advantage is that POO makes large programs more manageable. If all windows belong to a hierarchy of window classes and all code that refers to a particular window is within that window, all window manipulations can be written as a simple message transfer.

Disadvantages

If you want to read some data, make it simple and write again, you don't need to define classes and objects

Arguments

It is necessary to use **object-oriented programming**, as we expressed things in real life

Differences between a class an object and an instance.

Class is the generealization of features (attributes and behaviors)

Object instance of a class distinguishable by its specific characteristics

Encapsulacion

it will be needed depends on the access we want, example

If we need to access the color of a vehicle and we want to modify it from any shape or by methods of the same class

In protected you can access not only by vehicle, but also now also by buses, as we know a bus is a type of vehicle, then you must also have access to color.

Private needs access only to vehicle

Abstraction.

When an object can act and react in different ways, a set of operations that a client can perform on an object is called a protocol.

Abstraction allows us to expose limited data and object functionality publicly and hide the actual implementation. It is the most important pillar in UNRWA. In our example of mobile class and objects like Nokia, Samsung, iPhone.

Some features of mobile phones,

1. Dialing a number calls an internal method that concatenates the numbers and displays it on the screen, but we don't know what it's doing.
2. Clicking the green button sends real signals to the caller's mobile, but we don't know how they're doing.

Inheritance

Inheritance is the ability to extend the functionality of the base entity into a new entity that belongs to the same group. This will help us reuse the functionality that is already defined above and extend it to a new entity.

Therefore, mobile device brands are using this basic functionality by extending mobile class functionality and adding their own new features to their respective brands.

Static Class

When we don't need to instantiate an object, when we call a method in a simpler way, we want to quickly access class access

Polymorphism

Polymorphism can be defined as the ability to use the same name to do different things. More precisely, we call it "many forms of a single entity". This plays a vital role in the concept of UNRWA.

Let's say Samsung mobile has a 5MP camera available, that is, it has a CameraClick() functionality. Now the same mobile has Panorama mode available on the camera, so the functionality would be the same but with the mode. This type is said to be static polymorphism or compile-time polymorphism

Complete the following sentence by filling in the gaps by using one of the following terms["instance","class","object"]:

A ___ Object is the ___ Instance_ of an ___ Class___

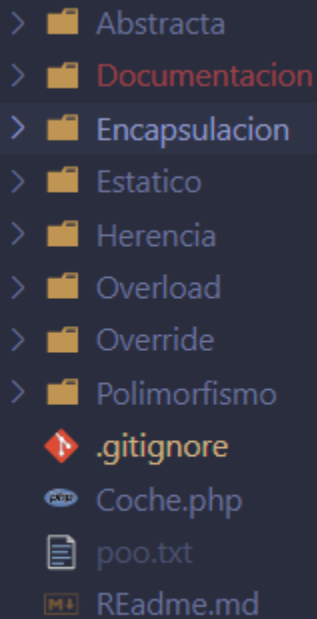
Developing an presentacido

We'll explain how the pill has been organized

- How pill tasks have been organized
- Knowledge learned
- What difficulties have arisen during the pill
- What conclusions do you draw from OOP?
- In what cases you recommend the use of the OOP paradigm
- Define the following concepts:
 - Class, object and instance
 - Encapsulation
 - Abstraction
 - Inheritance
 - Static class
 - Polymorphism
 - Overload
 - Override

Organize the code

In this small project we have focused on testing **PHP**, so we have not taken into account the organization of our code. It is very important that you organize properly. Create the following directories:

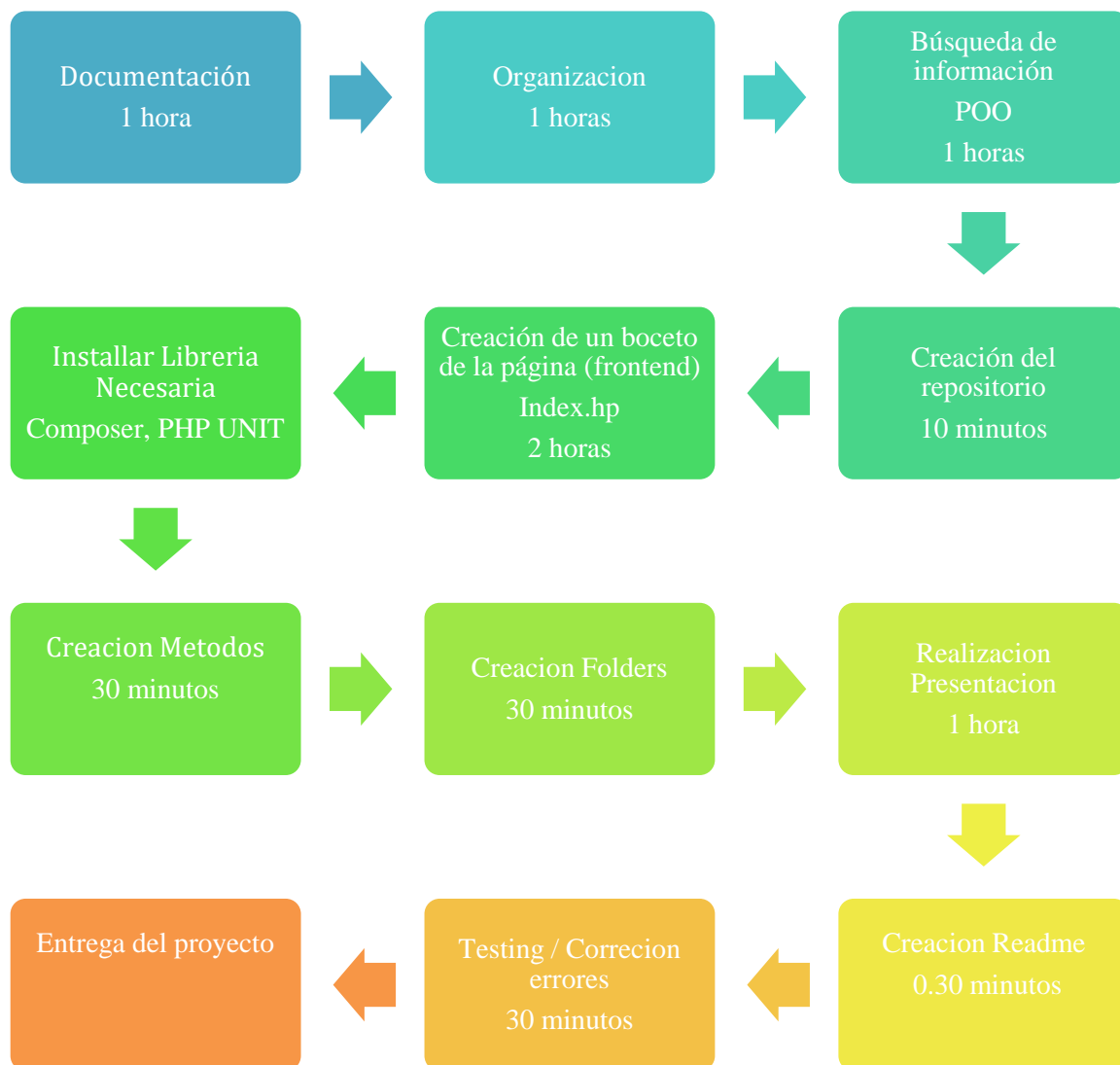


- > Abstracta
- > Documentacion
- > Encapsulacion
- > Estatico
- > Herencia
- > Overload
- > Override
- > Polimorfismo
- .gitignore
- Coche.php
- poo.txt
- REadme.md

Task planning

LISTA DE TAREAS A REALIZAR

Task	Priority	Hours	Difficulty	ID
Documentation	High	1,00	High	1
Organization	High	1,00	High	2
Pre-search for information	Normal	1,00	Normal	3
Repository creation	Low	0,15	Low	4
Index php structure	Low	0.30	Normal	5
Folders creation	Normal	0.30	Normal	6
Creation of Poo Methods	High	2.00	High	7
Presentation making	High	1.00	High	8
CREATION README	Low	0,30	Low	9
Testing / Correction Errors	High	0,30	Normal	10
Project delivery	High	0.20	High	11

Project Calendar Tracking

GIT WORKFLOW documentation

- Creating Git Hub <https://github.com/robertfox11/PooPills.git>
- We make commits of the structure of the main page.
- Chance of it occurring 80%
- Project impact 60%
- Possible alternative (mitigation) Ask colleagues for help
- Chance of it occurring 30%
- Project impact 60%
- Possible alternative (mitigation) Ask colleagues for help
- Not easily finding information related to the project
- Chance of it occurring 30%
- Project impact 60%
- Alternative alternative (mitigation)
- Ask colleagues for help

From the realization of the structure, work continued only on the "master" branch, through the Workflow "Gitflow".

But information --> <https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow>



Tools

Different tools were used in the development of the project. They are as follows:

- ***git***: *A powerful version control system* that helps track changes in the work tree.
- ***Visual Studio Code***: *A code editor* optimized for creating and debugging modern web applications.
- ***WampServer***, *comes integrated Apache Web server, openssl for SSL support, MySQL database and PHP language*
- ***Google Chrome Developer Tools***: Used to debug JavaScript code and to test design settings.
- ***Google Docs***: Used to write project documentation.
- **W3C Validator**– Used to validate HTML and CSS code.
- **ESLint**– Used to validate JavaScript code.
- ***nano***: *A basic text editor that uses the* command-line interface.
- ***curl***: *A* command-line tool used to transfer data using various network protocols.
- ***Google Docs***: Used to write project documentation.

Phase III - Project execution

Concepts

Which is a class

Incidents

None, luckily!

Lessons

All tasks were completed without having to face any major obstacles.

Phase IV - Project closure

General comments

The pill was successfully completed in the time interval that was predicted in task planning.