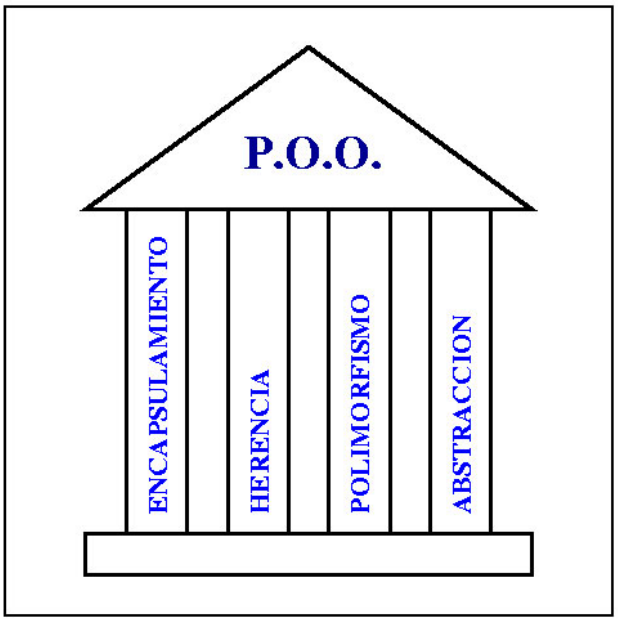
Universidad Politécnica de Victoria
Ingeniería en Tecnologías de la Información
Tecnología y Aplicaciones Web

Tarea 1
Programación Orientada a Objetos en PHP
(Resumen)

Héctor Alán De La Fuente Anaya

La POO es un paradigma de programación (o técnica de programación) que utiliza objetos e interacciones en el diseño de un sistema. Básicamente, este paradigma se compone de 4 pilares y diferentes características que veremos a continuación.



Abstracción: Aislación de un elemento de su contexto. Define las características esenciales de un objeto.

Encapsulamiento: Reúne al mismo nivel de abstracción, a todos los elementos que puedan considerarse pertenecientes a una misma entidad.

Polimorfismo: Es la capacidad que da a diferentes objetos, la posibilidad de contar con métodos, propiedades y atributos de igual nombre, sin que los de un objeto interfieran con el de otro.

Herencia: Es la relación existente entre dos o más clases, donde una es la principal (padre) y otras son secundarias y dependen (heredan) de ellas (clases “hijas”), donde a la vez, los objetos heredan las características de los objetos de los cuales heredan.

Además pueden ser encontradas otras características que pueden ser usadas para distinguir a la programación orientada a objetos de los demás lenguajes de programación, como las siguientes.

Recolección de basura: Es la técnica que consiste en destruir aquellos objetos cuando ya no son necesarios, liberándolos de la memoria.

Modularidad: Característica que permite dividir una aplicación en varias partes más pequeñas (denominadas módulos), independientes unas de otras.

Ocultación: Los objetos están aislados del exterior, protegiendo a sus propiedades para no ser modificadas por aquellos que no tengan derecho a acceder a las mismas.

CLASES O CLASES CONCRETAS

Una clase es un modelo que se utiliza para crear objetos que comparten un mismo comportamiento, estado e identidad.

Objeto

Es una entidad provista de métodos o mensajes a los cuales responde (comportamiento); atributos

con valores concretos (estado); y propiedades (identidad).

Método

Es el algoritmo asociado a un objeto que indica la capacidad de lo que éste puede hacer.

Propiedades y atributos

Las propiedades y atributos, son variables que contienen datos asociados a un objeto.

Como ya se ha reflejado antes, toda clase consta de la palabra clave class seguido del nombre de la clase y un bloque de código entre llaves.
Dentro del bloque de código se pueden crear tres tipos de bloques básicos:

· Constantes
· Variables
· Métodos

Una vez creadas dentro de las llaves, tanto la constante, como la variable, como la función pertenecen a la clase, y para ser utilizadas hay que acceder a través de la clase.

Reglas de Estilo sugeridas

Utilizar CamelCase para el nombre de las clases. La llave de apertura en la misma línea que el nombre de la clase, permite una mejor legibilidad del código.

Herencia de Clases

Los objetos pueden heredar propiedades y métodos de otros objetos. Para ello, PHP permite la "extensión" (herencia) de clases, cuya característica representa la relación existente entre diferentes objetos. Para definir una clase como extensión de una clase "padre" se utiliza la palabra clave extends.

Declaración de clases abstractas

Las clases abstractas son aquellas que no necesitan ser instanciadas pero sin embargo, serán heredadas en algún momento. Se definen anteponiendo la palabra clave abstract. Este tipo de clases, será la que contenga métodos abstractos (que veremos más adelante) y generalmente, su finalidad, es la de declarar clases "genéricas" que necesitan ser declaradas pero a las cuales, no se puede otorgar una definición precisa (No se pueden instanciar), de eso, se encargarán las clases que la hereden).

Declaración de Clases finales En PHP

PHP desde su versión 5.1 incorpora clases finales que no pueden ser heredadas por otra. Se definen anteponiendo la palabra clave final.

OBJETOS E INSTANCIAS

Una vez que las clases han sido declaradas, será necesario crear los objetos y utilizarlos, aunque hemos visto que algunas clases, como las clases abstractas son solo modelos para otras, y por lo tanto no necesitan instanciar al objeto.

Instanciar una clase

Para instanciar una clase, solo es necesario utilizar la palabra clave new. El objeto será creado, asignando esta instancia a una variable (la cual, adoptará la forma de objeto).
Lógicamente, la clase debe haber sido declarada antes de ser instanciada.

Reglas para la instanciación de los objetos

Para una mejor legibilidad y manejo de las clases, se recomienda utilizar nombres de variables (objetos) descriptivos, siempre con guion bajo al comenzar, la primera letra debe ser en minúscula, y

la siguiente palabra en mayúscula. Por ejemplo si el nombre de la clase es nombreClasecomo variable utilizar $_nombreClase. Esto permitirá una mayor legibilidad del código.

Definición de atributos o propiedades en PHP
Las propiedades representan ciertas características del objeto en sí mismo. Se definen anteponiendo la palabra clave var al nombre de la variable (propiedad). No es necesario utilizar la palabra reservada varpara la definición de la variable, pues PHP la reconoce por defecto.

Las propiedades pueden gozar de diferentes características, como por ejemplo, la visibilidad: pueden ser públicas, privadas o protegidas. Como veremos más adelante, la visibilidad de las propiedades, es aplicable también a la visibilidad de los métodos.

Niveles de acceso

1. **Propiedades públicas:** Las propiedades públicas se definen anteponiendo la palabra clave public al nombre de la variable. Éstas, pueden ser accedidas desde cualquier parte de la aplicación, sin restricción.
2. **Propiedades privadas:** Las propiedades privadas se definen anteponiendo la palabra clave private al nombre de la variable. Éstas solo pueden ser accedidas por la clase que las definió.
3. **Propiedades protegidas:** Las propiedades protegidas pueden ser accedidas por la propia clase que la definió, así como por las clases que la heredan, pero no, desde otras partes de la aplicación. Éstas, se definen anteponiendo la palabra clave protected al nombre de la variable.

Propiedades estáticas
Las propiedades estáticas representan una característica de “variabilidad” de sus datos, de gran importancia en PHP. Una propiedad declarada como estática, puede ser accedida sin necesidad de instanciar un objeto y su valor es estático (es decir, no puede ser modificada para cada objeto, es como una variable global para todas las instancias que se crean de ese objeto). Ésta, se define anteponiendo la palabra clave static al nombre de la variable.

ACCEDIENDO A LAS PROPIEDADES DE UN OBJETO
Para acceder a la propiedad de un objeto, existen varias maneras de hacerlo. Todas ellas, dependerán del ámbito desde el cual se las invoque así como de su condición y visibilidad.

- **Acceso a variables desde el ámbito de la clase** Se accede a una propiedad no estática dentro de la clase, utilizando la pseudo-variable $this siendo esta pseudo-variable una referencia al objeto mismo, se debe tener en cuenta que la variable que se llamara no llevara adelante el $. Cuando la variable es estática, se accede a ella mediante el operador de resolución de ámbito, doble dos-puntos :: anteponiendo la palabra clave self o parent según si trata de una variable de la misma clase o de otra de la cual se ha heredado, respectivamente.
- **Acceso a variables desde el exterior de la clase** Se accede a una propiedad no estática con la siguiente sintaxis: $objeto→variable Nótese además, que este acceso dependerá de la visibilidad de la variable. Por lo tanto, solo variables públicas pueden ser accedidas desde cualquier ámbito fuera de la clase o clases heredadas. Para acceder a una propiedad pública y estática el objeto no necesita ser instanciado, permitiendo así, el acceso a dicha variable mediante la siguiente sintaxis: Clase::$variable_estática

CONSTANTES
Otro tipo de “propiedad” de una clase, son las constantes, aquellas que mantienen su valor de forma permanente y sin cambios. A diferencia de las propiedades estáticas que pueden ser declaradas

dentro de una clase, las constantes solo pueden tener una visibilidad pública y no deben ser creadas dentro de las clases. El acceso a constantes es exactamente igual que al de otras propiedades.

MÉTODOS PHP

Cabe recordar, para quienes vienen de la programación estructurada, que el método de una clase, es un algoritmo igual al de una función. La única diferencia entre método y función, es que llamamos método a las funciones de una clase (en la POO), mientras que llamamos
funciones, a los algoritmos de la programación estructurada.
La forma de declarar un método es anteponiendo la palabra clave function al nombre del método, seguido por un par paréntesis de apertura y cierre y llaves que encierren el algoritmo. Al igual que cualquier otra función en PHP, los métodos recibirán los parámetros necesarios indicando aquellos requeridos, dentro de los paréntesis.

Métodos públicos, privados, protegidos y estáticos

Los métodos, al igual que las propiedades, pueden ser públicos, privados, protegidos o estáticos. La forma de declarar su visibilidad tanto como las características de ésta, es exactamente la misma que para las propiedades. Los métodos abstractos por el contrario, solo pueden ser creados en las clases cuya declaración haya sido abstracta. De lo contrario mostrará un error al ejecutar el código.

Métodos mágicos en PHP

PHP, nos trae una gran cantidad de auto-denominados “métodos mágicos”. Estos métodos, otorgan una funcionalidad pre-definida por PHP, que pueden aportar valor a nuestras clases y ahorrarnos grandes cantidades de código. Lo que muchos programadores consideramos, ayuda a convertir a PHP en un lenguaje orientado a objetos, cada vez más robusto. Entre los métodos mágicos, podemos encontrar los siguientes:

El Método Mágico __construct()

El método *__construct()* es aquel que será invocado de manera automática, al instanciar un objeto. Su función es la de ejecutar cualquier inicialización que el objeto necesite antes de ser utilizado.

El método mágico __destruct()

El método __destruct() es el encargado de liberar de la memoria, al objeto cuando ya no es referenciado. Se puede aprovechar este método, para realizar otras tareas que se estimen necesarias al momento de destruir un objeto.

Otros métodos mágicos

PHP nos ofrece otros métodos mágicos tales como __call, __callStatic, __get, __set, __unset, __sleep, __wakeup, __toString, __invoke, __set_state y __clone.

INTERFACES

Una interfaz es un conjunto de métodos abstractos y de constantes cuya funcionalidad es la de determinar el funcionamiento de una clase, es decir, funciona como un molde o como una plantilla. Al ser sus métodos abstractos estos no tiene funcionalidad alguna, sólo se definen su tipo, argumento y tipo de retorno.
Para implementar una interface es necesario que la clase que quiera hacer uso de sus métodos utilice la palabra reservada implements. La clase que la implemente, de igual modo debe sobrescribir los métodos y añadir su funcionalidad.

AGREGACIÓN Y COMPOSICIÓN EN PHP

En PHP también podemos mapear las diferentes relaciones que realizamos en los diagramas UML, como lo son la agregación y composición. Para este tema, contaremos con una breve explicación e implementación.

- **Agregación:** Según lo estudiado en UML, sabemos que una agregación es un tipo de relación dinámica, en donde el tiempo de vida del objeto incluido es independiente del que lo incluye (el objeto base utiliza al incluido para su funcionamiento).
- **Composición:** Según UML, una composición es un tipo de relación estática, en donde el tiempo de vida del objeto incluido está condicionado por el tiempo de vida del que lo incluye (el Objeto base se construye a partir del objeto incluido, es decir, es "parte/todo").