UNIVERSIDAD BOLIVIANA DE INFORMATICA

TALLER DE SISTEMAS

ESTUDIANTE: YAPU FLORES CLAUDIA LORENA

**PROGRAMACION ORIENTADA A OBJETOS**

**EXTEND**

Según el [sitio oficial de PHP](https://www.php.net/manual/es/language.oop5.inheritance.php):  
  
"La herencia es un principio de programación bien establecido y PHP hace uso de él en su modelado de objetos. Este principio afectará la manera en que muchas clases y objetos se relacionan unas con otras.  
  
Por ejemplo, cuando se extiende una clase, la subclase hereda todos los métodos públicos y protegidos de la clase padre. A menos que una clase sobrescriba esos métodos, mantendrán su funcionalidad original.  
  
Esto es útil para la definición y abstracción de la funcionalidad y permite la implementación de funcionalidad adicional en objetos similares sin la necesidad de reimplementar toda la funcionalidad compartida."

Mediante el uso de la herencia vamos a poder cambiar la morfología de la clase, extendiéndola para adaptarla a nuestras necesidades concretas.  
  
Ejemplo:

Vamos a suponer en la siguiente ampliación para nuestro desarrollo de coches:

Estaba pensado para la gestión de vehículos de un parking de 2 plantas subterráneas más una de superficie.

Ahora se va a ampliar su uso a furgonetas y autobuses.

Los autobuses no pueden aparcar en las plantas subterráneas pero sí en la superficie y además debo conocer el nombre de la empresa propietaria.

Las furgonetas solo pueden aparcar en la primera planta subterránea o en superficie y además debo conocer la altura.

Los coches solo pueden aparcar en las plantas subterráneas pero no en la superficie.

Una posible solución podría pasar por realizar las siguientes acciones:

Modificar la clase Coche (y su archivo correspondiente) para que ahora se llame Vehículo.

Añadir el atributo planta y sus correspondientes get() y set() para hacer referencia a la planta en la que estaciona el vehículo (aunque no aparezca en el constructor al no saber en qué planta estará el coche).

Añadir los atributos planta y plantas con un array que incluya las diferentes plantas disponibles.

Crear las clases Autobus, Furgoneta y Coche que extiendan la clase Vehiculo e incorporen el método puedeAparcar() para comprobar si puede o no puede aparcar, en función de la planta de entrada (este método se llamará igual pero será diferente en cada clase).

Añadir a la clase Autobus el atributo empresa y sobre-escribir el constructor para que incluya dicho atributo.

La clase Vehiculo.php:

**<?php**

class Vehiculo

{

//Variables o atributos

var $marca;

var $modelo;

var $color;

var $propietario;

var $planta;

var $plantas = array("superficie","subterraneo1","subterraneo2");

function \_\_construct($miMarca,$miModelo,$miColor,$miPropietario){

$this->marca = $miMarca;

$this->modelo = $miModelo;

$this->color = $miColor;

$this->propietario = $miPropietario;

}

//Funciones o métodos

function setMarca($miMarca){

$this->marca = $miMarca;

}

function getMarca(){

return $this->marca;

}

function setModelo($miModelo){

$this->modelo = $miModelo;

}

function getModelo(){

return $this->modelo;

}

function setColor($miColor){

$this->color = $miColor;

}

function getColor(){

return $this->color;

}

function setPropietario($miPropietario){

$this->propietario = $miPropietario;

}

function getPropietario(){

return $this->propietario;

}

function setPlanta($miPlanta){

$this->planta = $miPlanta;

}

function getPlanta(){

return $this->planta;

}

}

La clase Autobus.php:

**<?php**

//Requiero el archivo de clase para extenderlo en este script

require\_once("Vehiculo.php");

class Autobus extends Vehiculo

{

var $empresa;

function \_\_construct($miMarca,$miModelo,$miColor,$miPropietario,$miEmpresa){

$this->marca = $miMarca;

$this->modelo = $miModelo;

$this->color = $miColor;

$this->propietario = $miPropietario;

$this->empresa = $miEmpresa;

}

//Funciones o métodos

function setEmpresa($miEmpresa){

$this->empresa = $miEmpresa;

}

function getEmpresa(){

return $this->empresa;

}

function puedeAparcar($miPlanta){

//True si está en el array y es superficie

return (array\_search($miPlanta,$this->plantas) !== false && array\_search($miPlanta,$this->plantas) == 0);

}

}

La clase Furgoneta.php:

**<?php**

//Requiero el archivo de clase para extenderlo en este script

require\_once("Vehiculo.php");

class Furgoneta extends Vehiculo

{

function puedeAparcar($miPlanta){

//True si está en el array y no es subterraneo2

return (array\_search($miPlanta,$this->plantas) !== false && array\_search($miPlanta,$this->plantas) < 2);

}

}  
  
La clase Coche.php:

**<?php**

//Requiero el archivo de clase para extenderlo en este script

require\_once("Vehiculo.php");

class Coche extends Vehiculo

{

function puedeAparcar($miPlanta){

//True si está en el array y no es superficie

return (array\_search($miPlanta,$this->plantas) !== false && array\_search($miPlanta,$this->plantas) > 0);

}

}  
  
El archivo index.php:

**<?php**

//Requiero los archivos de clase para incluirlos en este script

require("Autobus.php");

require("Furgoneta.php");

require("Coche.php");

**?>**

<h1>Ejemplo 4: Herencia, Extensión y Polimorfismo</h1>

**<?php**

//Instancio y configuro los vehículos

$autobus = new Autobus("Volvo","9800 2017","gris","Mario","Desfufor");

$furgoneta = new Furgoneta("Wolkswagen","Caravelle","blanco","Rebeca");

$coche = new Coche("Toyota","Corolla","verde","Marcos");

**?>**

<div>

¿Puedo aparcar el coche en la superficie?:

<strong>**<?php** echo ($coche->puedeAparcar("superficie")) ? "si" : "no" **?>**</strong>

</div>

<div>

¿Puedo aparcar el coche en el subterráneo 2?:

<strong>**<?php** echo ($coche->puedeAparcar("subterraneo2")) ? "si" : "no" **?>**</strong>

</div>

<div>

¿Puedo aparcar la furgoneta en la superficie?:

<strong>**<?php** echo ($furgoneta->puedeAparcar("superficie")) ? "si" : "no" **?>**</strong>

</div>

<div>

¿Puedo aparcar la furgoneta el subterráneo 2?:

<strong>**<?php** echo ($furgoneta->puedeAparcar("subterraneo2")) ? "si" : "no" **?>**</strong>

</div>

<div>

¿Puedo aparcar el autobús en la superficie?:

<strong>**<?php** echo ($autobus->puedeAparcar("superficie")) ? "si" : "no" **?>**</strong>

</div>

<div>

¿Puedo aparcar el autobús el subterráneo 1?:

<strong>**<?php** echo ($autobus->puedeAparcar("subterraneo1")) ? "si" : "no" **?>**</strong>

</div>

<div>

¿A qué empresa pertenece el autobús?:

<strong>**<?php** echo $autobus->getEmpresa() **?>**</strong>

</div>

Y como siempre, vamos a analizar qué conclusiones podríamos extraer del ejemplo anterior:

Para extender una clase y poder acceder a todos sus métodos y atributos, tan solo tenemos que utilizar la directiva extends después del nombre de clase y añadir el nombre de la clase principal.

Los nombres de clases, variables o funciones nunca llevan tildes, caracteres extraños ni espacios, y no pueden empezar por un número.

No todos los atributos deben estar en el constructor, solo los necesarios para inicializar correctamente el objeto de clase.

Es posible sobre-escribir un atributo o método (incluyendo el constructor) en la clase secundaria simplemente declarándolo de nuevo en ella.

Al usar varias clases que extienden de la misma, es mejor utilizar require\_once() para evitarnos la carga de la clase principal de nuevo.

A la hora de mezclar PHP con HTML, es mejor y más seguro frente a posibles inyecciones de código de tipo XSS incrustar el PHP dentro del HTML, esto es, abrir y cerrar las etiquetas de PHP en vez de hacer echos par el HTML.

Para PHP en una comparación, 0 y false es el mismo valor, lo que puede dar lugar a confusión.