PROGRAMACION ORIENTA DA A OBJETOS (POO)
Concepto igual en todos los lenguaje de programación.
La POO trata de trasladar el concepto y comportamiento de os objetos de la vida real al código de programación
El objetivo de la POO es:

- reutilizar (algo parecido a la función, pero mas complejo) el código y
- hacer la vida más de la programación , más fácil

Tenemos que tener claros tres términos:

Objeto.

- Elementos con propiedades y comportamientos.
- Ej en la vida real un coche con propiedades color, velocidad, potencia, tamaño….y comportamientos o métodos acelerar, frenar….
- Esto también se lleva a la programación

Clase

- Elementos comunes a un grupo de objetos
- Ej todos los coches tienen motor, ruedas….
- Estas características comunes son lasas que se programan en una clase

Instancia

- Ejemplar perteneciente a una clase
- De ahí el termino "instanciar una clase"
- Se trata de objetos que tienen características comunes pero se diferencia entre ellos ,

Es decir creas la clase coche para crear un primer coche que tendrá algunas características diferentes

Hay que tener en cuenta que tenemos dos tipos de lenguajes de programación

- Lenguaje orientado a objetos (JAVA, JSCRIPT , PHP)
- Lenguaje orientado a programación (antiguos)

EJERCICIO
Vamos a crear una clase en PHP y luego una instancia

1.Creamos una clase
Class Coche{}

OJO!!! Los nombres de las clases suelen tener la 1< letra en mayúscula
2.Una vez creada la clase (se trata de una clase vacía).
¿cómo creamos ahora una instancia perteneciente a esa clase Coche?
con **New**

```php
<?php
    class Coche{



    }

    new Coche();
?>
```

Con esto, habremos creado una clase y una instancia de dicha clase

1.Importante: A esa instancia le tenemos que dar un nombre, pues eso e pone delante del New, cada una con una instancia diferente : Renaul, Mazda. Seat..

```php
<?php
    class Coche{



    }

    $renault=new Coche();
    $mazda=new Coche();
    $seat=new Coche();
```

tenemos la clase coche que se encarga de las características comunes y 3 instancias (ejemplares)de tipo coche.

1.¿cómo damos propiedades y métodos a los objetos? Con variables

- propiedades, con **var**

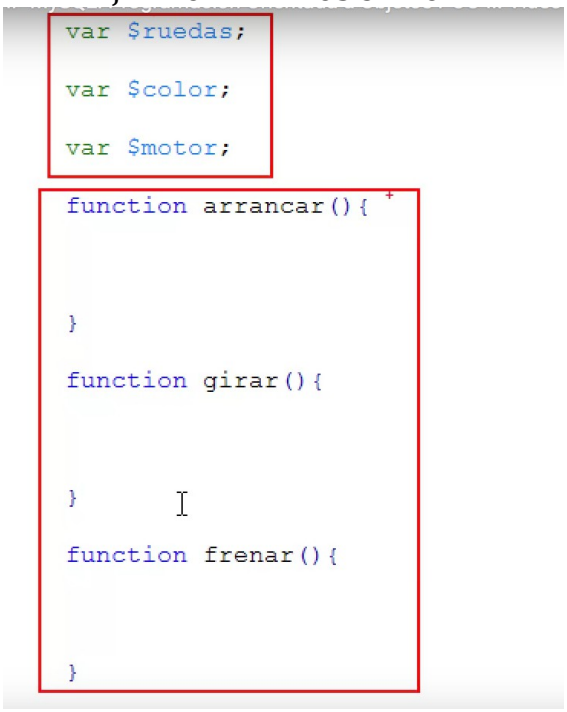
```
var $ruedas;
var $color;
var $motor
```

i

- métodos : con **funciones**

```
Function arrancar {}
Function girar{}
```

EN EL EJMPLO TENEMOS 3 PROPIEDADES Y 3 MÉTODOS



Ya tenemos programadas las propiedades y las funciones o métodos

1.Los objetos tienen un estado inicial

Ej coche tiene las siguientes características iniciales, antes de salir de la fábrica:

- Estado sin arrancar
- 0 km
- 4 ruedas
- color

1.Para crear el estado inicial en PHP creamos un constructor.
Este constructor es el que va a definir el estado inicial
¿Cómo se crea el constructor?
Funtion nombre de la clase (es a lo que se llama un método constructor)
dentro del método constructor al poner un $, sale la palabra This
esta palabra la utilizamos para referirnos a la propia clase; Es como si implícitamente pusieras Coche
2.A continuación **guion y símbolo mayor que,** que nos permite introducir una propiedad (circulo verde), o un método (cuadrado rojo)

```
class Coche{

    var $ruedas;

    var $color;

    var $motor;

    function Coche(){  //Método constructor;

        $this->ruedas=4;

        $this->color="";

        $this->motor=1600;


    }

    function arrancar(){


    }

    function girar(){


    }

    function frenar(){
```

3.
4.ojo!!! Precisamente el operado NEW llama al constructor

Cada vez eu ponemos new y el nombre del método constructor, estás ejecutando las instrucciones que tiene, por tanto le das un estado inicial al objeto.
Una vez completado el constructor , lo que quedaría es definir el resto de métodos; métodos que no son constructores; Se trata de métodos que definen el comportamiento que tiene un objeto de tipo coche

```
    function arrancar(){


    }
    function girar(){


    }
    function frenar(){


    }

}

$renault=new Coche(); //Estado inicial al objeto o instancia
$mazda=new Coche();
$seat=new Coche();
```

programamos un echo en cada uno de ellos

```
        echo "Estoy arrancando<br>";


    }
    function girar(){
        echo "Estoy girando<br>";
    }
    function frenar(){
        echo "Estoy frenando<br>";
    }
    function establece_color($color_coche){
        $this->color=$color_coche;
        echo "El color de este coche es: " . $this->color;

    }

}

$renault=new Coche(); //Estado inicial al objeto o instancia
$mazda=new Coche();
$seat=new Coche();
```

Resumen
Nuestro objeto coche tiene un estado inicial de $ ruedas, color indefinido, motor 1600; Con capacidad(método) de arrancar, girar, frenar.

1.**Ahora queremos que una de nuestras instancias gire: por ejemplo el mazda**
$mazda-girar() (Llamada a un método para que nuestro objeto de tipo coche haga esa tarea)
2.le doy una propiedad al mazda
echo $mazda-ruedas;
3.vamos a ver ahora cómo llamar a métodos pertenecientes a una clase, métodos que puedan recibir parámetros
•En el constructor creamos la propiedad color sin definir
¿como podríamos ahora crear color en cada instancia?
iCreamos un método o función que recibe el parámetro que va a ser el color y después que asigne el parámetro a la variable
iila función establece_color va a recibir un parámetro
iiidentro con la expresión This , hacemos referencia a la propiedad color y le asignamos el argumento que le pasamos al parámetro de la función, a la propiedad color
ivy creamos un echo para imprimir
vA partir de aquí damos color a cada una de las instancias; Además queremos que nos especifique el nombre, por lo que en el método debo poner un parámetro más.
viY en el echo añadimos contenido y el valor de la nueva variable

```
    function establece_color($color_coche,$nombre_coche){
        $this->color=$color_coche;
        echo "El color de " . $nombre_coche . " es: " . $this->color . "<br>";
    }
}
$renault=new Coche(); //Estado inicial al objeto o instancia
$mazda=new Coche();
$seat=new Coche();
$renault->establece_color("Rojo","Renault");
$seat->establece_color("Azul","Seat");
//$mazda->girar();
//echo $mazda->ruedas;
```

vii

Cómo reutilizar este código en un futuro

Ejercicio
1vamos a crear ahora otra clase
2la llevamos a un documento a parte y lo aguardo con el nombre de vehiculos.php
3copio y pego clase coche cambiando nombre coche por camión
4cambio también el constructor y cambio las propiedades del estilo inicial
5para reutilizar el código utilizo Include

```
<?php

    include("vehiculos.php");


?>
</body>
</html>
```

OJO!! cuando un constructor no recibe parámetros no es necesario si no quiero ponerle paréntesis
también es posible que una instancia no tenga método constructor, es decir que no le demos estilo inicial al objeto

```
class Coche{
    var $ruedas;
    var $color;
    var $motor;
    function Coche(){   //Método constructor;
        $this->ruedas=4;
        $this->color="";
        $this->motor=1600;
    }
    function arrancar(){
        echo "Estoy arrancando<br>";

    }
    function girar(){
        echo "Estoy girando<br>";
    }
```

Cuando esto ocurre es como si tuviera un constructor vacio

Ahora creamos una isntacia del camion llamado pegaso

```
<?php

    include("vehiculos.php");

    $mazda=new Coche();
    $pegaso=new Camion();
    echo "El Mazda tiene " . $mazda->ruedas . " ruedas <br>";
    echo "El Pegaso tiene " . $pegaso->ruedas . " ruedas <br>";


?>
```

Y por último quitamos del objeto la funcion o método crear color y le pongo un color predeterminado gris en el método constructor.
Comprobar que si utilizas una instancia Magda si tiene colo ry si utilizadmos pgaso no

1.
2.
3.

4.

```php
class Camion{

    var $ruedas;

    var $color;

    var $motor;

    function Camion(){  //Método constructor;

        $this->ruedas=8;

        $this->color="";

        $this->motor=2600;


    }

    function arrancar(){

        echo "Estoy arrancando<br>";
```

5.
6.
7.

1.