# Comparativa JS PHP

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Conceptos | JS | | | PHP | | |
| variables | let a;  var a; | | | $a; | | |
| const a;  const a = “pato”; | | | define (“a”, “pato”);  const a=”pato”; | | |
| When using the const keyword, only scalar (bool, int, float and string) expressions and constant arrays containing only scalar expressions are accepted | | |
| Las constantes han de ser inicializadas al declararse | | | | | |
| a = b; | copia el contenido de la  variable b en a.  Excepción: b es un array, en cuyo caso a se iguala por referencia y ambos apuntan a lo mismo | | $a=&$b; | Asignación por referencia, las dos variables apuntan al mismo objeto. | |
| strings | a = “pato”;  a=’pato’;  a=`pato`;  // las `` permiten el uso de variables | los string que se puedan interpretar como números pueden ser usados como string o como int/float dependiendo de la operación. | | $a = “pato”; se pueden usar variables para dar valor;  $a = ‘pato’; | Las variables que se puedan interpretar como números serán convertidas a int/float. **Ir con cuidado con notación científica** | |
| a=”pato”; b=loco; c=a+b; 🡪 c es patoloco | | | $a=”pato”; $b=”loco”  $c=$a.$b; 🡪 c es “patoloco” | | |
| int | a = 5 | | | $a = 5; | | |
| float | a = 1.3; | | | $a=1.3; || $a=”1.3”; | | |
| arrays declaración | a = [“pato”, 5 , “hola”, 4] | | | $a = [“pato”, “hola”];  $a = [5, 4];  $array = array("foo" => "bar",  "bar" => "foo",);  $array = ["foo" => "bar",  "bar" => "foo",]; | | |
| a = { "foo": "bar", "bar" : "foo"}  Lo más parecido a los arrays (asociativos) de php son los objetos. | | | Si no se usan kewords explícitos se asignan int como key, estos enteros no tienen por qué ser consecutivos si se asigna una key de tipo int se continuará la cuenta a partir de este número.  Si se declaran elementos con la misma keyword estos son sobrescritos. **Cuidado con la transformación a int de los strings!!!** | | |
| arrays acceso | a[0] 🡪 pato | | | a[0] 🡪 string(4) pato **php 8.0**  a{0} 🡪 string(4) pato **php 7.4 deprecated**. | | |
| booleans | a = true; // a=false; | | | $a=True; // $a=False; | | |
| funciones | function foo(arg1, arg2, …, argn){{  console.log("Example function.\n";)  return ;} | | Case Sensitive | function foo($arg\_1, $arg\_2, $arg\_n){  echo "Example function.\n";  return ;} | | Case InsensiTive |
| bucles | var\_array.foreach(function a aplicar) | | | foreach(iterable) | | |
|  |  | | |  | |  |

# PHP-MySQL connexion

El PDO se gestiona siempre en dos pasos:

* Preparación
* Ejecución

# PHP info

Para consultar los valores de una array podemos usar var\_dump() o print\_r().

Para acceder a los métodos y atributos se utiliza -> en lugar del punto(.)

header() : permite la recarga de los ficheros