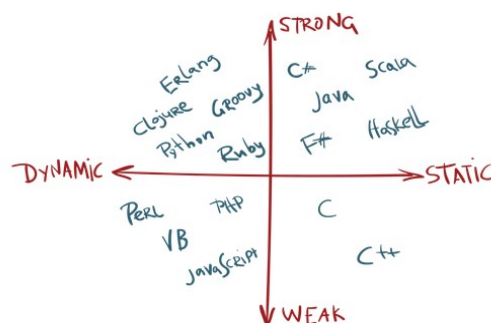


## **Tipus de dades i variables**

a) Llenguatges de programació Strongly typed, weakly typed, statically typed i dynamically typed:

- Lectura recomanda: <https://android.jlelse.eu/magic-lies-here-statically-typed-vs-dynamically-typed-languages-d151c7f95e2b>
- Es considera un llenguatge de programació amb assignació de tipus fort o fortament tipat (en anglès strongly typed) a aquell que obliga al desenvolupador a seguir unes regles molt estrictes d'assignació de tipus a les variables.
- Utilitzant un llenguatge fortament tipat, és força probable que es produeixen errors de compilació o interpretació per aquest motiu. L'assignació de tipus afecta sobre tot a la declaració de variables, funcions, el retorn de dades d'una funció i la crida a funcions.
- Els llenguatges fortament tipat tenen regles estrictes de conversió de tipus.
- Es considera un llenguatge de programació amb assignació de tipus feble o feblement tipat (en anglès weakly typed o loosely typed) a aquell a on pràcticament no existeixen unes regles d'assignació de tipus a les variables que sigui necessari seguir.
- Utilitzant un llenguatge d'assignació de tipus feble és força improbable que es produeixen errors de compilació o interpretació per aquest motiu.
- Els llenguatges de programació feblement tipats són més flexibles, ràpids i fàcils d'utilitzar però a vegades es poden produir errors d'execució inesperats per aquest motiu. Els fortament tipats són més estructurats, eviten errors d'execució més fàcilment i els tipus ajuden a fer el codi més manegable quan es treballa en equip.
- Un llenguatge d'assignació estàtica de variables, demana que es declari el tipus de variable abans d'utilitzar-la. Quan una variable té un tipus assignat no es pot assignar a la variable una dada d'un tipus diferent al seu. Si es fa, es produirà un error en temps de compilació. Aquests llenguatges permeten trobar més errors durant la fase de desenvolupament i en general donen lloc a executables més ràpids i que requereixen menys memòria.
- Un llenguatge d'assignació dinàmica de variables:
  - No demana que es declari el tipus de variable abans d'utilitzar-la. El tipus es comprova en temps d'execució.
  - En temps d'execució una variable pot treballar amb diferents tipus de dades.
  - Aquests llenguatges són més flexibles, generalment són interpretats (no cal compilació) de manera que el cicle de desenvolupament és més curt, el codi és habitualment menys òptim i poden produir errors durant l'execució.
- PHP o javascript són llenguatges feblement tipats. Java o C# són llenguatges fortament tipats.
- C o Java són llenguatges amb assignació estàtica de variables. PHP o Python treballen amb assignació dinàmica de variables.
- Representació de la situació dels llenguatges en funció del seu nivell dins de les escales d'assignació dinàmica-estàtica i fortament-feblement tipats



#### b) Tipus de les dades

b) PHP és un llenguatge feblement dinàmicament tipat però té tipus. Per PHP 7, els tipus són:

- **int**: Número no decimal entre -2.147.483.648 and 2.147.483.647. Es poden utilitzar les bases decimals, hexadecimal, octal i binària.
- **float o double**: Números amb decimals (14) o en forma exponencial (fins a  $1,8 \times 10^{308}$ ).
- **string**: Cadena de caràcters entre " o '.
- **bool**: Una variable que pot valer **true** o **false**.
- **array**: Pot emmagatzemar múltiples valors diferenciats. dins d'una variable. Ja han estat tractats a classe.
- **Objecte**: Un tipus de variable que pot emmagatzemar dades i els mètodes (o funcions) per tractar les seves dades.
- **NULL**: Una variable de tipus NULL només pot tenir el valor **null** i això vol dir que no té res assignat, que no té cap valor.

Lectura recomanada: [https://www.w3schools.com/php/php\\_datatypes.asp](https://www.w3schools.com/php/php_datatypes.asp)

#### c) Abast de l'existència de les variable:

- Una variable és d'**abast local** si es declara dins d'una funció. Només existeix mentre s'executa la funció, i després deixa d'existir. Es pot repetir el nom d'una variable local dins de diferents funcions.
- Una variable és d'**abast global** si es declara fora de qualsevol funció. Existeix mentre s'executa el codi del script PHP. Si una variable local té el mateix nom que una global, mentre s'executa la funció el valor que s'utilitza és el de la variable local.
- S'utilitza la **paraula clau global** per fer que una variable dins d'una funció sigui no local. Exemple:

```
<?php
    $x = 5;
    function myTest() {
        global $x;
        $x = $x + 2;
    }
    myTest();
    echo $x; // resultat --> 7
?>
```

- S'utilitza la **paraula clau static** per fer que una variable local no sigui esborrada un cop la funció on va ser declarada finalitza. La propera vegada que es cridi a la funció es podrà utilitzar el valor desat a la variable. Exemple:

```
<?php
function myTest() {
    static $x = 0;
    echo $x."<br>";
    $x++;
}
myTest(); // resultat --> 0
myTest(); // resultat --> 1
myTest(); // resultat --> 2
echo $x."<br>"; //resultat --> Undefined variable
?>
```

#### d) Regles de definició de noms de variables:

- Les variables comencen amb \$
- Una variable comença amb una lletra o \_
- Una variable no comença mai amb un número
- Una variable només pot tenir els caràcters [0..9], [a..z], [A..Z] i \_
- Les variables són consideren diferents les majúscules i les mnúscules. \$temp és diferent a \$Temp.

e) Declaració de tipus en el pas de paràmetres a funcions. Declaració estricta i no estricta:

- PHP7 permet declarar el tipus dels arguments que passem a una funció quan declarem la funció . En cas de no complir-se el tipus, l'interpret fa un canvi de tipus. Així doncs, en PHP7 es pot definir una funció de la següent manera:

```
<?php
function myAdd(int $a, int $b, int $c) {
    return $a + $b + $c;
}
echo myAdd(4.2,3,-2.7); // resultat --> 5 (no 4.5)
?>
```

- Es pot obligar a que les dades que es passin a la funció siguin obligatoriament del mateix tipus que les demanades a la declaració indicant al principi del script PHP que les declaracions seran **estrictes**. En aquest cas, durant l'execució es produirà un error si una variable no és del tipus demana. Això es pot fer de la següent manera:

```
<?php
declare(strict_types=1);
function myAdd(int $a, int $b, int $c) {
    return $a + $b + $c;
}
echo myAdd(4,3,-2); // resultat 5
# echo myAdd(4.2,3,-2); // resultat --> Error. Pàgina en blanc
?>
```

Lectura recomanada: <https://blog.teamtreehouse.com/5-new-features-php-7>

f) Retorn de funcions amb declaració de tipus:

- PHP7 permet declarar el tipus de la variable retornada:

```
<?php
function myAdd(float $a, float $b, float $c) : int {
    return $a + $b + $c;
}
echo myAdd(7.1,3.2,-2.7); // resultat --> 7 (no 7.6 )
?>
```

Lectura recomanada: <https://blog.teamtreehouse.com/5-new-features-php-7>

g) Operadors amb variables:

- Operadors aritmètics: + \* - / \*\* %
- Operadors de comparació: ==, !=, <=, >=, <, >, ===, !==, <=>
- Operadors d'assignació: =, +=, -=, \*=, %=
- Operadors lògics: && (and), || (or), xor, !,
- Operadors incrementals i decrementals: ++ i --
- Operadors amb strings: . (concatena) .= (\$a .= \$b ----> \$a=\$a.\$b)
- Operadors amb arrays: +, ==, ===, !=, <>, !==

Lectura recomanda: [https://www.w3schools.com/php/php\\_operators.asp](https://www.w3schools.com/php/php_operators.asp)

#### h) Operadors ternaris:

- L'operador ternari és un operador condicional que disminueix la longitud del codi mentre realitza comparacions i condicionals.
- Aquest mètode és una alternativa a utilitzar sentències if-else i if-else niutats.
- L'ordre d'execució d'aquest operador és d'esquerra a dreta.

```
<?php
$nota = 6.6;
$missatge = ($nota >= 5) ? "Aprovat" : "Suspès";
// Si $nota >=5 $missatge="Aprovat" sino $missatge="Suspès";
echo $missatge // resultat --> Aprovat
?>
```

#### i) Casting (Conversió) de tipus variables:

- Podem forçar una variable a utilitzar un determinat tipus utilitzant el casting (forçat) de tipus. Per exemple:

```
<?php
$b = 4.2;
$a = (int) $b; // El tipus de $b serà integer o sencer
gettype($a)."<br>"; // Resultat: int (integer)
gettype($b)."<br>"; // Resultat: float o double (real)
?>
```

- Tipus bàsics de casting: (int), (float) o (double), (string), (bool), (array) i (object)
- Es pot obtenir el tipus d'una variable amb la funció [gettype\(\)](#) i es pot canviar el tipus d'una variable amb la funció [settype\(\)](#).
- També es pot comprovar el tipus d'una variable amb les funcions [is\\_int\(\\$var\)](#), [is\\_string\(\\$var\)](#), [is\\_bool\(\\$var\)](#), [is\\_numeric\(\\$var\)](#), [is\\_null\(\\$var\)](#), [is\\_string\(\\$var\)](#) i [is\\_array\(\\$var\)](#)

#### j) Variables predefinides o reservades:

- PHP 7 té una sèrie de variables predefinides (i per tant reservades) i els desenvolupadors no poden definir variables que tinguin el mateix nom que una variable predefinida.
- Llista de variables predefinides: <http://php.net/manual/en/reserved.variables.php>

#### k) Taules de comparació de tipus: <http://php.net/manual/en/types.comparisons.php>

#### l) Funcions per visualitzar els continguts i les estructures dels tipus:

- Es convenient saber utilitzar les funcions [echo](#), [print](#), [var\\_dump](#) i [print\\_r](#)
- Documentació: [http://cybmeta.com/php-diferencias-entre-echo-print-print\\_r-y-var\\_dump](http://cybmeta.com/php-diferencias-entre-echo-print-print_r-y-var_dump)

#### m) Arrays:

- Definició i creació de arrays: [https://www.w3schools.com/php/php\\_arrays.asp](https://www.w3schools.com/php/php_arrays.asp)
- Array indexat: [https://www.w3schools.com/php/php\\_arrays\\_indexed.asp](https://www.w3schools.com/php/php_arrays_indexed.asp)
- Arrays associatiu: [https://www.w3schools.com/php/php\\_arrays\\_associative.asp](https://www.w3schools.com/php/php_arrays_associative.asp)
- Array multidimensional: [https://www.w3schools.com/php/php\\_arrays\\_multidimensional.asp](https://www.w3schools.com/php/php_arrays_multidimensional.asp)
- Exemples: <http://www.collados.org/daw2/m07/uf1/exemples/arrays.tar.gz>