

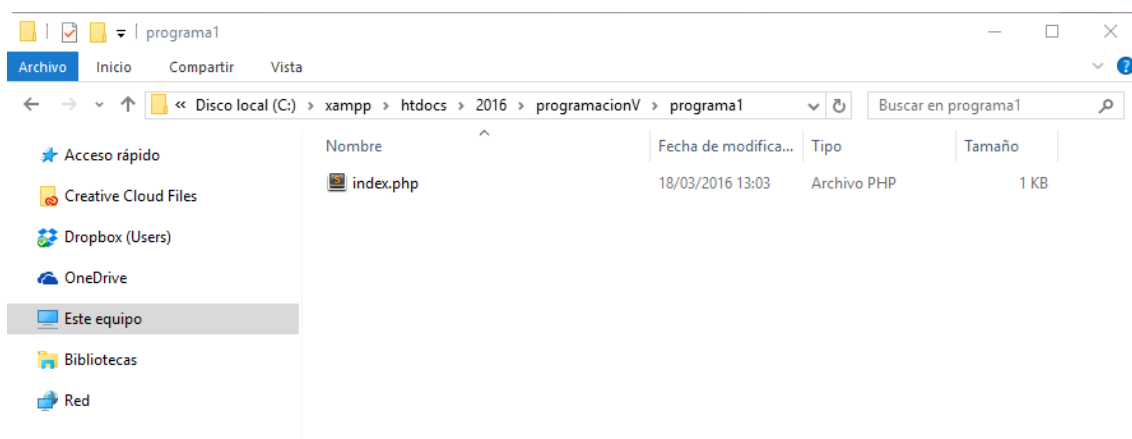
Tecnicatura Superior en Análisis de Sistemas

PROGRAMACIÓN V

Tema 2- Conociendo el lenguaje

PHP es un lenguaje bastante sencillo de aprender, tiene una similitud a C y C++, para comenzar a trabajar vamos a generar un proyecto nuevo, es decir dentro de nuestro directorio del servidor, en la carpeta **htdocs**, creamos una carpeta nueva, llamada por ejemplo “programa1” (siempre que trabajemos en web debemos crear proyectos o archivos sin espacios y sin acentos).

Dentro del directorio **programa1** creamos un archivo con la extensión .php llamado **index**. Y nos debería quedar algo parecido al siguiente directorio:



Todo archivo PHP debe comenzar con las etiquetas de apertura y cierre:

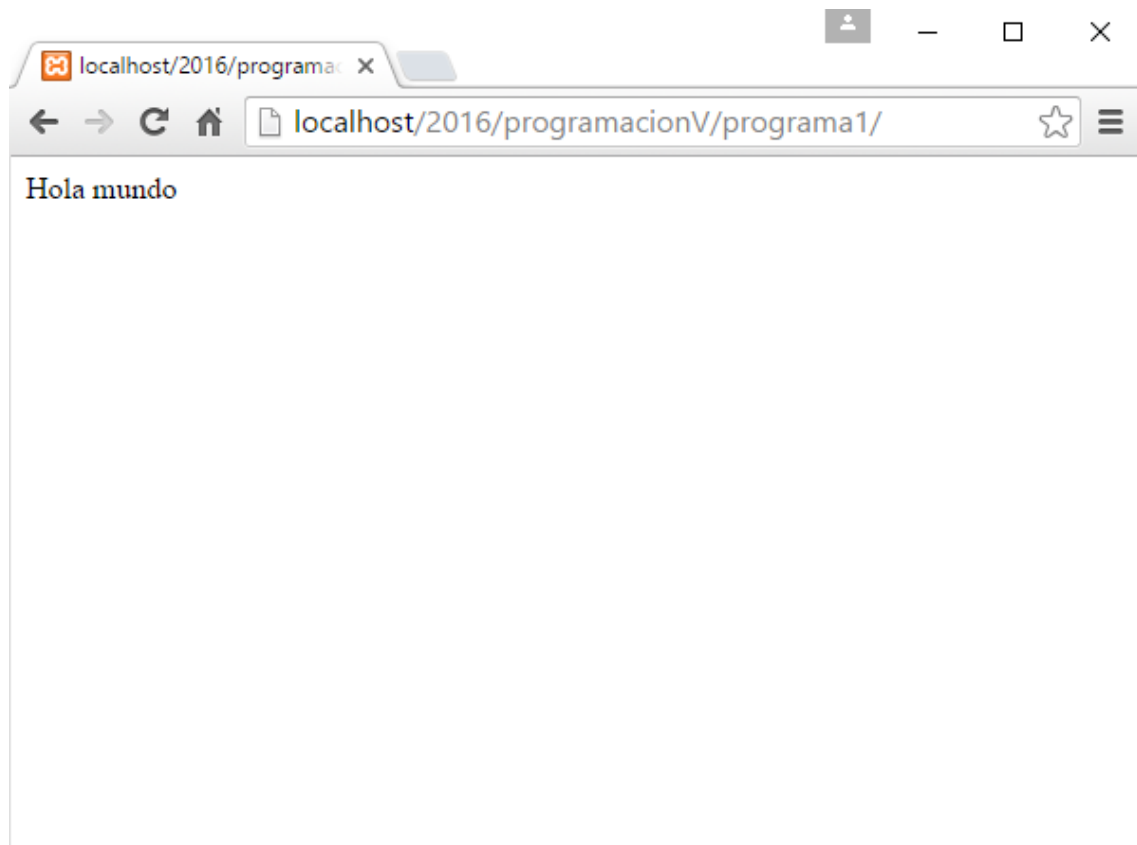
```
<?php
?>
```

De esta forma le estamos indicando al intérprete que el siguiente código corresponde al lenguaje **PHP**. Y dentro de esa apertura y cierre de las etiquetas debemos escribir nuestro código.

Para mostrar texto se utiliza la palabra reservada **echo**. El comando **echo** no es una función por lo que no es necesario utilizar paréntesis. Y el texto a mostrar debe estar entre comillas simples o dobles, es decir si comenzamos escribiendo con comillas simples deberemos terminar el texto con comillas simples, lo mismo ocurre si utilizamos comillas dobles, siempre comenzamos y terminamos con las mismas comillas. Entonces si queremos hacer un “Hola mundo” debemos escribir el siguiente código:

```
<?php
    echo "Hola mundo" ;
?>
```

Y en el navegador se vería lo siguiente:



Básicos

Para realizar comentarios de una sola línea o multilinea debemos hacerlo de la siguiente manera:

```
<?php
    //Esto es un comentario
    /*Esto es un
    comentario
    multilínea
    */
?>
```

Otro tema a considerar es que toda línea de código al finalizar debe terminar con punto y coma (;), salvo la última línea de código en la que se puede omitirlo.

Variables

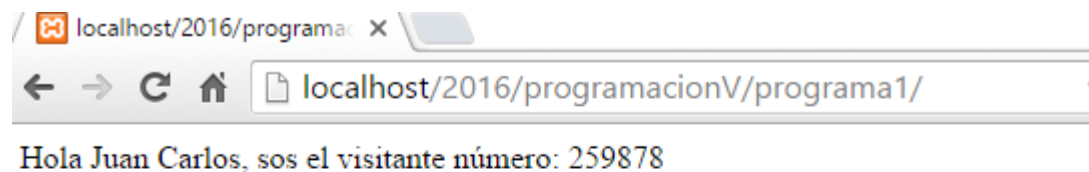
El uso de variables en PHP es bastante sencillo, se representan con un signo \$ seguido del nombre de la variable, y este es **sensible a mayúsculas y minúsculas** y deben comenzar con una letra.

A diferencia de otros lenguajes, las variables no necesitan ser asignadas a un tipo, solamente tenemos que hacer la definición del nombre de la variable. Para asignar un valor a la variable debemos utilizar el signo igual, seguido del valor a asignar y punto y coma (;). **El tipo de variable depende del valor que le asignemos**, es decir si a una variable le asignamos un número esa variable va a ser del tipo integer y si le asignamos un texto, esa variable va a ser del tipo string. Si queremos mostrar una variable sólo tenemos que utilizar la sentencia **echo** seguida del nombre de la variable. También podemos concatenar variables utilizando el punto (.) como separador.

```
<?php
    $texto = "Hola ";
    $nombre = "Juan Carlos";
    $texto2 = ", sos el visitante número: ";
    $numero = 259878;

    echo $texto.$nombre.$texto2.$numero;
?>
```

Y el resultado sería el siguiente, visto a través del navegador web:



Analizando el código anterior podemos observar, que tanto la primera variable **\$texto** como **\$nombre** y **\$texto2** son del tipo string mientras que la última variable **\$numero** es del tipo integer. Y para concatenar utilizamos el punto (.);

Tipos de datos

PHP admite ocho tipos primitivos.

Enteros y Flotantes (integer y float)

Los enteros pueden especificarse mediante notación decimal (base 10), hexadecimal (base 16), octal (base 8) o binaria (base 2), opcionalmente precedidos por un signo (- o +).

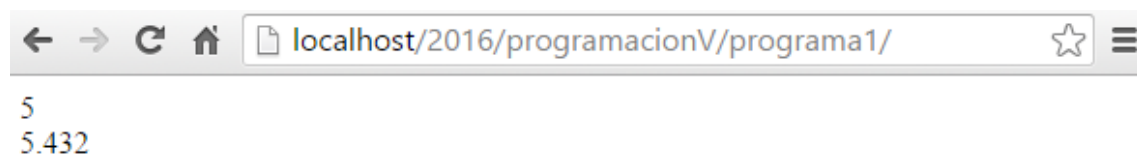
```
<?php
$a = 1234; // número decimal
$a = -123; // un número negativo
$a = 0123; // número octal (equivale a 83 decimal)
$a = 0x1A; // número hexadecimal (equivale a 26 decimal)
$a = 0b11111111; // número binario (equivale al 255 decimal)
?>
```

Los números de punto flotante (también conocidos como "float" en inglés) pueden ser especificados usando cualquiera de las siguientes sintaxis:

```
<?php
    $a = 1.234;
    $b = 1.2e3;
    $c = 7E-10;
?>
```

Ejemplo práctico:

```
<?php
    $entero = 5;
    echo $entero;
    echo "<br />";
    $decimal = 5.432;
    echo $decimal;
?>
```



Booleanos

Este es el tipo más simple. Un boolean expresa un valor que indica verdad. Puede ser **TRUE** (verdadero) o **FALSE** (falso).

Ejemplo práctico:

```
<?php
    $booleana = true;

    echo "el valor de la variable es ".$booleana;
?>
```

el valor de la variable es 1

Si es verdadero devuelve 1 si es falso 0, como pusimos que es verdadero, nos devuelve 1.

String (cadena de caracteres)

Un **string**, o cadena, es una serie de caracteres donde cada carácter es lo mismo que un byte. Esto significa que PHP solo admite un conjunto de **256** caracteres.

Ejemplo práctico:

```
<?php
    echo "Soy una cadena";
    echo "</br>";
    echo 'Soy una cadena';
    echo "</br>";
    echo "Soy una 'cadena' ";
    echo "</br>";
    echo "Soy una \"cadena\" ";
?>
```

Soy una cadena
Soy una cadena
Soy una 'cadena'
Soy una "cadena"

Estas son distintas formas de armar un texto en PHP, si en medio del mismo tenemos que escribir comillas, podemos alterar las comillas, es decir comenzar con dobles y en el medio simples o al revés. Sino con barra (\) y comillas dobles. De esta manera se entendería que lo que está adentro debe estar entre comillas.

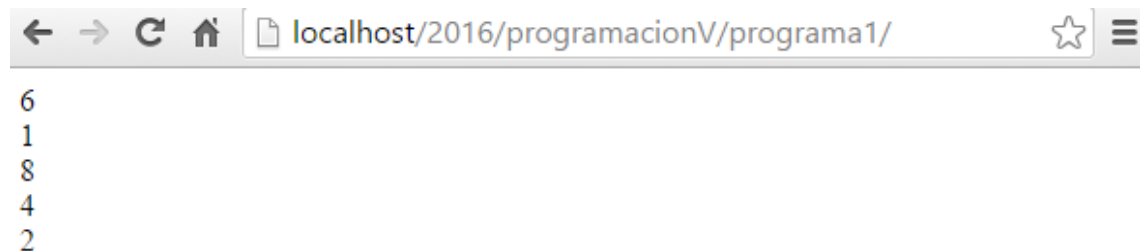
Operadores aritméticos

Un operador es algo que toma uno o más valores (o expresiones) y produce otro valor (de modo que la construcción en sí misma se convierte en una expresión).

Los operadores aritméticos nos permiten realizar operaciones aritméticas: suma, resta, multiplicación, división, etc. así como obtener el módulo o resto de una división entre dos enteros.

```
<?php
    echo 3 + 3 ; //Suma
    echo "</br>";
    echo 3 - 2; //Resta
    echo "</br>";
    echo 4 * 2; //Multiplicación
    echo "</br>";
    echo 20 / 5; //División
    echo "</br>";
    echo 20 % 6; //Resto de la división (MOD)
?>
```

Y en el navegador veríamos lo siguiente:



Pre-incremento y Post-incremento

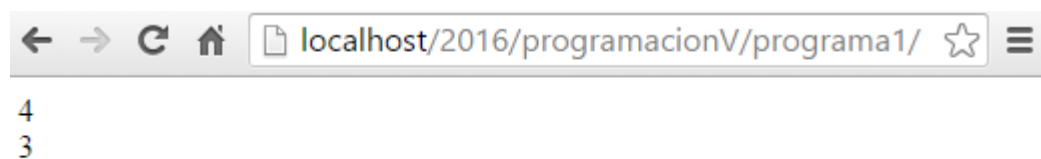
Los operadores de pre-incremento o post-incremento, incrementan o reducen el valor almacenado en una variable.

Ejemplo práctico:

```
<?php
    $mi_variable = 3;
    echo ++$mi_variable;
    echo "</br>";

    $mi_variable = 3;
    echo $mi_variable++;
?>
```

Y como resultado el navegador nos va a mostrar:



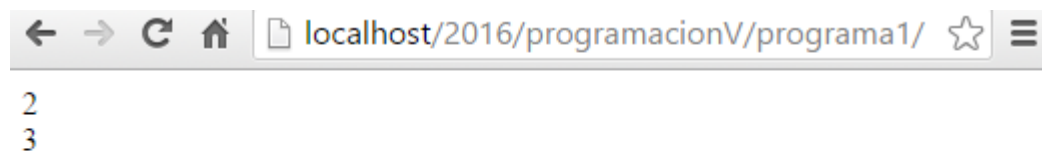
En el primer caso, incrementa y muestra la variable, entonces a 3 le suma 1, entonces muestra 4.

En el segundo caso, muestra la variable y después incrementa 1, entonces muestra 3.

Si hiciéramos **pre-decremento** y **post-decremento**; es exactamente lo mismo:

```
<?php
    $mi_variable = 3;
    echo --$mi_variable;
    echo "</br>";

    $mi_variable = 3;
    echo $mi_variable--;
?>
```



En el primer caso muestra 2 porque resta uno a la variable, y en el segundo caso muestra lo mismo 3 porque restaría 1 después de mostrar el contenido de la variable.

Asignación

El operador de asignación es el = "igual a", esto significa que el operador de la izquierda obtiene el valor de expresión de la derecha.

La asignación copia la variable original a la nueva utilizando el concepto de asignación por valor.

Ejemplo práctico:

```
<?php
    $mi_variable = 7;
    echo $mi_variable;
    echo "</br>";

    $mi_variable += 5;
    echo $mi_variable;
    echo "</br>";

    $mi_variable -= 5;
    echo $mi_variable;
```

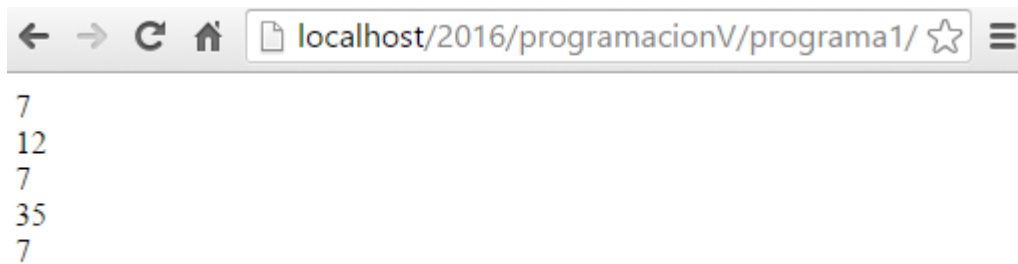
```
echo "</br>";

$mi_variable *= 5;
echo $mi_variable;
echo "</br>";

$mi_variable /= 5;
echo $mi_variable;
echo "</br>";

?>
```

Y el resultado sería el siguiente:



```
← → ↻ 🏠 📄 localhost/2016/programacionV/programa1/ ☆ ≡

7
12
7
35
7
```

En el primer caso solamente muestra el valor de la variable es decir 7, en el segundo caso a ese valor le suma 5, entonces muestra 12. Luego a esa misma variable le resta 5 ($=5$) para mostrar como resultado nuevamente 7. Después a ese valor (7) lo multiplica por 5 ($\times 5$), nos muestra 35 y por último al valor de la variable lo divide por 5 y muestra el resultado (7).

Comparación

Los operadores de comparación nos permiten comparar dos valores.

```
<?php
    echo 3 == 3;
    echo "</br>";

    echo 3 === 4;
    echo "</br>";

    echo 3 === 3;
    echo "</br>";

    echo 3 === "3";
    echo "</br>";

?>
```

El primer caso devuelve verdadero porque **3** es igual a **3**, en el segundo caso devuelve falso ya que **3** no es igual a **4**, en el tercer caso **3** es igual a **3** y no solo es igual sino

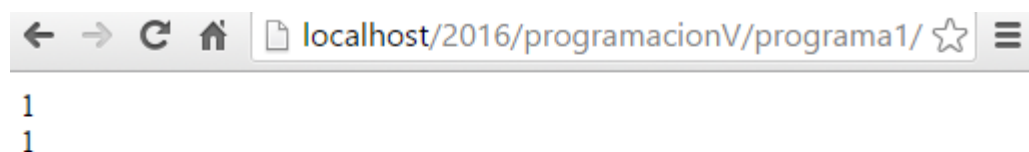
que **es idéntico**, entonces devuelve verdadero. Y en el último caso devuelve falso ya que **3** no es igual ni idéntico a **"3"**, ya que **"3"** es del tipo string y no integer dado que tiene comillas.

Diferencias

```
<?php
    echo 3 != 4;
    echo "</br>";

    echo 3 <> 4;
    echo "</br>";

?>
```



En ambos casos devuelve verdadero (1) ya que 3 no es igual a 4.

Lógicos

Los operadores lógicos nos permiten crear condiciones para las distintas estructuras condicionales como también, en estructuras repetitivas.

| | |
|-----|----|
| And | && |
| Or | |
| Not | != |

```
<?php
$uno = 1;
$dos = 2;

echo ($uno == 1 && $dos ==2);

?>
```

1

El resultado final es 1 (verdadero) ya que la variable \$uno es igual a 1 y la variable \$dos es igual a 2;