

## **UNIDAD DIDÁCTICA 3:**

### **ESTRUCTURAS BÁSICAS DE CONTROL Y CÁLCULO EN PHP.**

**Módulo profesional:**  
**Desarrollo Web en Entorno Servidor**

# Índice

|   |    |
|---|----|
| RESUMEN INTRODUCTORIO.....                      | 3  |
| INTRODUCCIÓN .....                              | 3  |
| CASO INTRODUCTORIO .....                        | 3  |
| 1. TIPOS BÁSICOS DE DATOS .....                 | 4  |
| 1.1 Definir una variable .....                  | 4  |
| 1.2 Sintaxis en la definición de variables..... | 4  |
| 1.3 Tipos de variables .....                    | 6  |
| 1.4 Tipo numérico .....                         | 6  |
| 1.5 Conversión de decimal a entero .....        | 8  |
| 1.6 Tipo cadena .....                           | 9  |
| 1.7 Funciones de cadena .....                   | 10 |
| 1.8 Tipo boolean .....                          | 11 |
| 1.9 Imprimir por pantalla. Función echo .....   | 11 |
| 2. ESTRUCTURAS DE CONTROL .....                 | 13 |
| 2.1 Estructura if .....                         | 14 |
| 2.2 Estructura switch .....                     | 16 |
| 2.3 Operadores de control .....                 | 18 |
| 2.4 Estructura while.....                       | 19 |
| 2.5 Aplicación de while con código html .....   | 21 |

## RESUMEN INTRODUCTORIO

Antes de comenzar a utilizar el lenguaje de programación php, necesitamos tener las herramientas básicas para poder crear código. Esas herramientas básicas son las variables y las estructuras de control.

## INTRODUCCIÓN

En cualquier lenguaje de programación dos de las primeras nociones que debemos aprender para poder comenzar a escribir código son:

- Los tipos de variables disponibles
- Cuál es la sintaxis de escritura y uso de dichas variables.
- Las estructuras de control básicas para poder tomar decisiones dentro de nuestro código

Con estas herramientas nuestro nivel de conocimiento se incrementa muchísimo ya que podremos comenzar a resolver pequeños retos que podamos programar a través de código php.

## CASO INTRODUCTORIO

Queremos comenzar a resolver pequeños problemas y dar la solución a través de páginas php que el usuario pueda visualizar por pantalla.

Al final de esta unidad serás capaz de escribir variables y utilizando estructuras de control pintar de forma automática el contenido por la pantalla.

## 1. TIPOS BÁSICOS DE DATOS

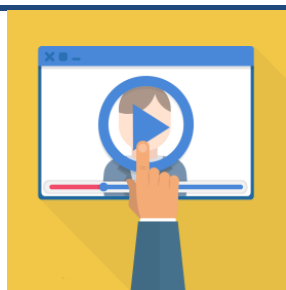
En todos los lenguajes de programación, las variables, los tipos y/o los atributos constituyen un elemento básico de programación en el que almacenamos información, modificamos la misma y donde también intercambiamos la información con otras zonas del desarrollo e incluso con otros lugares.

### 1.1 Definir una variable

¿Qué es una variable? Una variable en cualquier lenguaje de programación es el elemento que nos va a proporcionar la capacidad de almacenar, modificar y utilizar un determinado contenido. Dependerá de lo que se denomina "tipo" y que va a indicarnos si la variable es numérica, de cadena, una matriz u otro tipo más complejo.

PHP es un lenguaje muy poco "tipificado", esto es positivo y negativo a la vez, ya que nos permite una flexibilidad enorme en la definición y uso de las variables. Sin embargo también puede ser un cúmulo de posibles errores ya que podemos perder la perspectiva del tipo de contenido y como usarlo si no definimos correctamente la variable.

Dicho de esta forma suena como muy abstracto, por lo que no nos vamos a detener mucho en esta definición y lo que vamos a realizar son ejemplos específicos con cada uno de los tipos con un doble objetivo, por un lado trabajar con las variables y por otro definir esos tipos.



En el vídeo que encontrarás una introducción a los tipos y variables con php

 <https://youtu.be/oA4rpPvdlu4>

### 1.2 Sintaxis en la definición de variables

La sintaxis básica de definición de una variable según la referencia de php.net es:

**VARIABLE EN PHP**

<http://php.net/manual/es/language.variables.basics.php>

En PHP las variables se representan con un signo de dólar seguido por el nombre de la variable. El nombre de la variable es sensible a minúsculas y mayúsculas.

Los nombres de variables siguen las mismas reglas que otras etiquetas en PHP. Un nombre de variable válido tiene que empezar con una letra o un carácter de subrayado (underscore), seguido de cualquier número de letras, números y caracteres de subrayado. Como expresión regular se podría expresar como: `'[a-zA-Z_\x7f-\xff][a-zA-Z0-9_\x7f-\xff]*'`

Como va a ocurrir a lo largo del curso esta definición la podremos ver representada de una forma más clara a través de ejemplos de código:

**Ejemplo creación variables**

```
<?php
$nombre = "Paco";
$edad= 42;
$esProfesor=true;
?>
```

Aún sin todos los conocimientos necesarios, podemos observar tres variables definidas en el anterior ejemplo:

- Una primera, cuyo nombre es "nombre" y cuyo contenido es Paco
- Una segunda, cuyo nombre es "edad" y cuyo contenido es 42
- Una tercera cuyo nombre es "esProfesor" y cuyo contenido es true

## 1.3 Tipos de variables

Según la referencia de php.net, nos encontramos con 8 tipos de datos y otros dos especiales. No los vamos a ver ahora todos, aunque sí que es interesante nombrarlos.

Cuatro tipos escalares:

- boolean
- integer
- float (número de punto flotante, también conocido como double)
- string

Cuatro tipos compuestos:

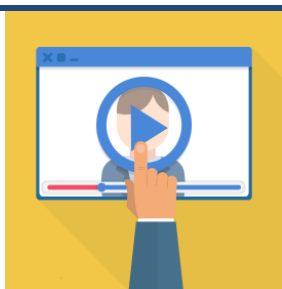
- array
- object
- callable
- iterable

Y finalmente dos tipos especiales:

- resource
- NULL

## 1.4 Tipo numérico

Dentro del grupo de tipos denominados numéricos, nos encontramos de dos tipos, los enteros y los decimales. RECORDAD QUE LOS VÍDEOS SON OBLIGATORIOS VISULIZARLOS PARA EL BUEN APRENDIZAJE Y SEGUIMIENTO DE LA ASIGNATURA:



En el vídeo que encontrarás una introducción a los tipos numéricos y varios ejemplos sobre estos tipos



<https://youtu.be/cP9hFBuI1RA>



**Para saber más sobre los números enteros**

<http://php.net/manual/es/language.types.integer.php>

Un número entero (o integer) es un número del conjunto  $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$ .

En concreto el ejemplo que aparece en el anterior ejemplo lo podéis recuperar del siguiente repositorio:

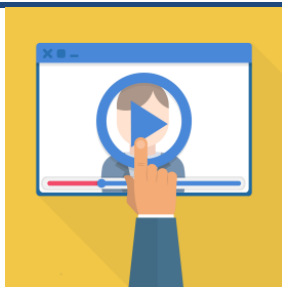


En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursPHP/blob/master/tipos/tipoInteger.php>

El segundo gran tipo numérico es el número decimal, o coma flotante el cual nos permite almacenar números mucho más grandes tanto en formato entero+decimal o en formato mantisa. Veamos un ejemplo con su código y su página de referencia:



En el vídeo que encontrarás un ejemplo sobre números decimales



<https://youtu.be/vUBrpyfB-Bg>



En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursPHP/blob/master/tipos/tipoDecimal.php>

## 1.5 Conversión de decimal a entero

En el anterior apartado y vídeo ya vimos cómo convertir un tipo decimal a entero:

```
$soloEuros=(int)$dinero;
```

Como podemos observar, se utiliza una metodología muy parecida a la de Java, ya que utilizando el *caster* (int) podremos convertir un tipo decimal a entero.

La conversión inversa, de entero a decimal, no tiene mucho sentido, ya que el intérprete PHP será el encargado de convertir automáticamente dicho número a decimal siempre y cuando se necesite:

```
<?php
$soloEuros=5;
$centimos=43;
$dinero=$soloEuros+($centimos/100);
?>
```

Como podemos ver en el anterior ejemplo, la variable \$dinero almacenará directamente un número en decimal.



### ***Para saber más sobre el tipo decimal***

<http://php.net/manual/es/language.types.float.php>

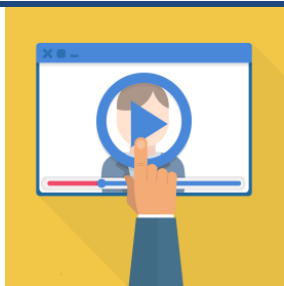
Los números de punto flotante tienen una precisión limitada. Aunque depende del sistema, PHP típicamente utiliza el formato de doble precisión IEEE 754, el cual dará un error relativo máximo por aproximación del orden de  $1.11e-16$ . Las operaciones aritméticas elementales no podrán generar grandes errores y, por supuesto, se han de considerar los errores por propagación al componer varias operaciones.



## 1.6 Tipo cadena

Como comentamos en la introducción, dos grandes apartados dentro de los tipos dentro de cualquier lenguaje de programación son los numéricos por un lado y los de carácter o textuales por otra parte. Este segundo bloque es tan importante que en muchos casos y sin darnos cuenta, ya hemos hecho uso de los mismos aunque no hayamos definido ninguna variable de este tipo, es el caso del uso de la función **echo**, la cual convierte automáticamente las variables a tipo cadena para poder mostrarla.

Dos son las formas que tenemos de representar cadenas:



En el vídeo que encontrarás un ejemplo de uso del tipo cadena

 <https://youtu.be/oAjhQMzRIOY>



En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursPHP/blob/master/tipos/tipoCaracter.php>



**Para saber más sobre el tipo string**

<http://php.net/manual/es/language.types.string.php>

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, y de ahí que no ofrezca soporte nativo para Unicode. Véanse los detalles del tipo string.

Cabe destacar una diferencia entre el doble entrecomillado y el simple tal y como se destaca en el vídeo y lo vemos en el ejemplo:

```
$tengoEuros=10;  
$fraseTrabajador="Tengo ${tengoEuros} euros";
```

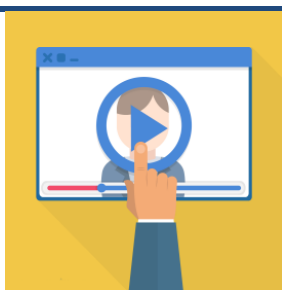
En el anterior ejemplo, el resultado al lanzar el fichero en el servidor será de:

```
Tengo 10 euros
```

Es decir, el entrecomillado doble permite evaluar variables siendo muy útil para la construcción de contenido por pantalla.

## 1.7 Funciones de cadena

Igual que en otros lenguajes de programación, el número de funciones que existen para el manejo de String es muy muy elevado, permitiendo realizar prácticamente todas las operaciones posibles con los String. La recomendación es que a partir de la documentación oficial realicéis pruebas y ejemplos.



En el vídeo que encontrarás un ejemplo de uso de diferentes funciones de trabajo con cadenas



<https://youtu.be/nN5zDLU4jgQ>



Para saber más sobre las funciones de string

<http://php.net/manual/es/ref.strings.php>

Listado de muchas funciones de string



En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursosPHP/blob/master/tipos/funcionesCaracter.php>

## 1.8 Tipo boolean

Según la documentación de php net, este es el tipo más simple ya que un boolean expresa un valor que indica verdad. Puede ser TRUE (verdadero) o FALSE (falso).

Veámoslo en un ejemplo sencillo:

```
<?php
$haceSol = true; // asigna el valor TRUE la variable haceSol
?>
```

Como podemos observar la variable *haceSol* tomaría el valor VERDADERO que en PHP sería *true*. Ahora mismo no podemos decir mucho más sobre el tipo boolean, ya que tiene mucho más sentido cuando lo utilizemos con estructuras de control.

## 1.9 Imprimir por pantalla. Función echo

Anteriormente ya hemos visto en varios ejemplos el uso de la función **echo**. Según la documentación de php.net, **echo** no es realmente una función (es una construcción del lenguaje), por lo que no se requiere el uso de paréntesis con él. **echo** (a diferencia de otras construcciones del lenguaje)

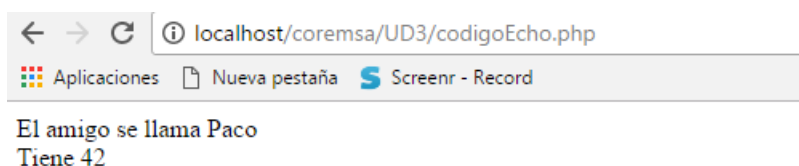
no se comporta como una función, es decir no siempre se puede usar en el contexto de una función.

Veámoslo con un ejemplo:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ejemplos echo</title>
  </head>
  <body>
    <?php
      $nombre="Paco";
      $edad=42;
      echo "El amigo se llama ".$nombre;
      echo "<br>";
      echo "Tiene ".$edad;
    ?>
  </body>
</html>
```

- En primer lugar recordar que un fichero con extensión .php será ejecutado por el servidor PHP junto a Apache.
- El código php está intercalado con el código html, será el servidor Apache el encargado de traducir el código php a un código html comprensible por el navegador.
- Todas las funciones echo van sin paréntesis, ya que no es una función.
- Echo puede imprimir tanto cadenas de texto como variables numéricas convirtiendo automáticamente a texto los parámetros
- La concatenación en php se realiza mediante el punto "."
- Para imprimir por pantalla código html también utilizamos echo, `echo "<br>"`;

El resultado por pantalla es:



Y el código html:

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>Ejemplos echo</title>
  </head>
  <body>
    El amigo se llama Paco<br>Tiene 42  </body>
</html>
```

Como se puede observar, las etiquetas **<?php** y **<?>**, no solo eso, todos los echo se han convertido en impresiones en pantalla, que en el caso de desarrollo web, es escritura en el anterior código html.

|                           |
|---------------------------|
| Tarea 1                   |
| Título: Variables y Tipos |

## 2. ESTRUCTURAS DE CONTROL

Parafraseando y haciendo referencia a la WikiPedia, en lenguajes de programación, las estructuras de control permiten modificar el flujo de ejecución de las **instrucciones de un programa**.

Con las estructuras de control se puede:

- De acuerdo con una condición, ejecutar un grupo u otro de sentencias (If-Then-Else)
- De acuerdo con el valor de una variable, ejecutar un grupo u otro de sentencias (Select-Case)
- Ejecutar un grupo de sentencias mientras se cumpla una condición (Do-While)
- Ejecutar un grupo de sentencias hasta que se cumpla una condición (Do-Until)

- **Ejecutar un grupo de sentencias un número determinado de veces (For-Next)**



**Para saber más sobre las estructuras de control en los lenguajes de programación**

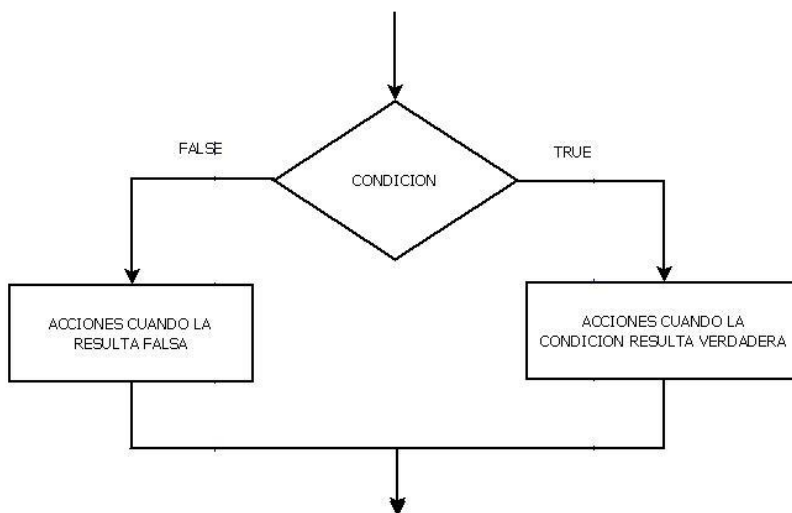
[https://es.wikipedia.org/wiki/Estructuras\\_de\\_control](https://es.wikipedia.org/wiki/Estructuras_de_control)

Todas las estructuras de control tienen un único punto de entrada. Las estructuras de control se pueden clasificar en: secuenciales, iterativas y de control avanzadas. Esta es una de las cosas que permiten que la programación se rija por los principios de la programación estructurada.

La anterior definición y clasificación la podemos aplicar a cualquier lenguaje de programación, ahora es necesario aterrizarla en php.

## 2.1 Estructura if

La estructura *if* dentro de un lenguaje de programación es una de las estructuras más básicas y más usadas para tomar una decisión ante un estado o punto del desarrollo. De forma gráfica:

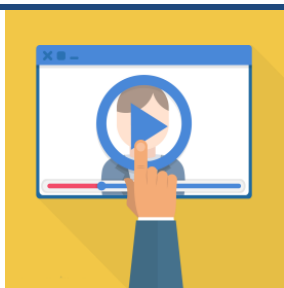


**Imagen 1: Estructura If**

Fuente de la imagen:

<https://funddeprogr.files.wordpress.com/2011/10/diagramaif-else1.jpg>

Veámoslo a través de un vídeo y un ejemplo:



En el vídeo que encontrarás un ejemplo de uso de la estructura if

 [https://youtu.be/EJnuda5Zp\\_E](https://youtu.be/EJnuda5Zp_E)



*Para saber más sobre la estructura if*

<http://php.net/manual/es/control-structures.if.php>



En el siguiente enlace tienes el código utilizado:



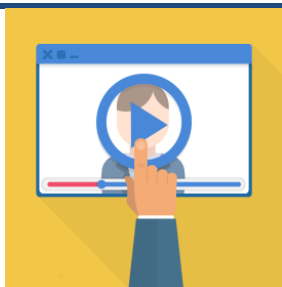
<https://github.com/pacogomezarnal/codigoEjemploCursosPHP/blob/master/estructurasControl/if.php>

Como podéis observar en la anterior referencia desde php.net, la estructura completa de la estructura if sería:

```
<?php
if ($a > $b) {
    echo "a es mayor que b";
} elseif ($a == $b) {
    echo "a es igual que b";
```

```
} else {  
    echo "a es menor que b";  
}  
?>
```

Veámoslo a través de otro ejemplo y su código:



En el vídeo que encontrarás un ejemplo de uso de la estructura if/else

 <https://youtu.be/Ns3xaBGdB1c>



*Para saber más sobre la estructura if/elseif/else*

<http://php.net/manual/es/control-structures.elseif.php>



En el siguiente enlace tienes el código utilizado:



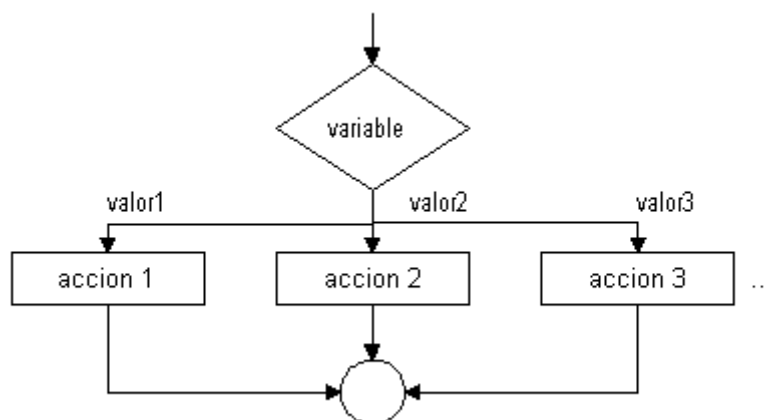
<https://github.com/pacogomezarnal/codigoEjemploCursPHP/blob/master/estructurasControl/ifelse.php>

## 2.2 Estructura switch

Tenemos una segunda estructura de control, la estructura switch, la cual nos permite realizar una comparación con una variable concreta. En este caso nos encontramos ante una estructura muy parecida a *if*, sin embargo

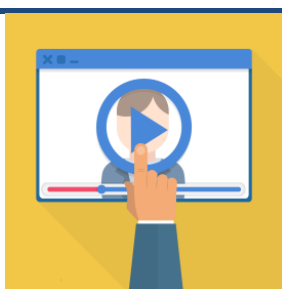


en este caso sólo compararemos una variable de referencia. De forma gráfica:



**Imagen 2: Estructura Switch**

Fuente de la imagen: [http://3.bp.blogspot.com/-oZ1VeOrc9rA/U9vI6jDCg-I/AAAAAAAAAw/BK8J4K\\_ob9k/s1600/switch.gif](http://3.bp.blogspot.com/-oZ1VeOrc9rA/U9vI6jDCg-I/AAAAAAAAAw/BK8J4K_ob9k/s1600/switch.gif)



En el vídeo que encontrarás un ejemplo de uso de la estructura switch

 <https://youtu.be/e05WyB4JXHY>



**Para saber más sobre la estructura switch**

<http://php.net/manual/es/control-structures.switch.php>



En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursoPHP/blob/master/estructurasControl/switch.php>

## 2.3 Operadores de control

Un operador de comparación es aquel que nos permite comparar dos variables bajo ciertas condiciones. Son utilizados principalmente en la estructura de control if, for, while , etc, y en ejemplos anteriores ya hemos utilizado algunos de ellos.

No es el objetivo en este punto detallar todos los operadores, ya que si visitamos la documentación de php, veremos muchos y muy variados:

- Precedencia de operadores
- Operadores aritméticos
- Operadores de asignación
- Operadores bit a bit
- Operadores de comparación
- Operadores de control de errores
- Operadores de ejecución
- Operadores de incremento/decremento
- Operadores lógicos
- Operadores para strings
- Operadores para arrays
- Operadores de tipo

**Para saber más sobre operadores**

<http://php.net/manual/es/language.operators.php>

En esta referencia tendremos todos los tipos de operadores que php nos ofrece.

Veámoslo con un par de ejemplos:



En el vídeo que encontrarás ejemplos sobre uso de operadores de comparación:

 <https://youtu.be/St3gNG5S6ys>



En el siguiente enlace tienes el código utilizado:

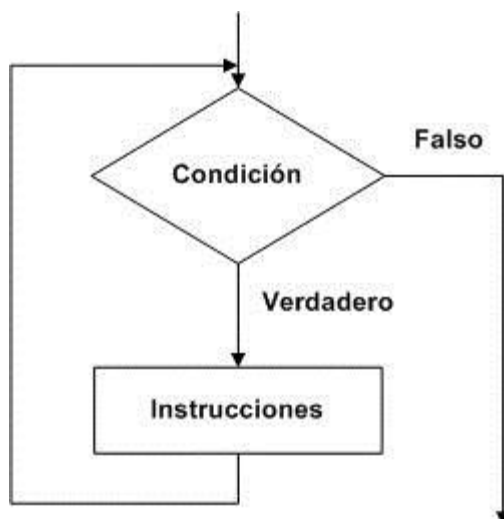


<https://github.com/pacogomezarnal/codigoEjemploCursosPHP/blob/master/estructurasControl/operadores.php>

## 2.4 Estructura while

Dentro de las estructuras de control nos encontramos con las estructuras de decisión, como es la estructura de control if o como el switch, y un segundo tipo de estructuras de repetición donde a partir de la evaluación de una expresión se repetirá el contenido de acciones definidas en su interior hasta que la expresión evaluada sea falsa.

La estructura de repetición más sencilla es while, en este caso la evaluación de la expresión se realiza a principio, y se repetirá el contenido tantas veces como dicha expresión sea verdadera. De forma gráfica:

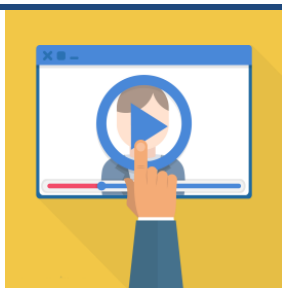


**Imagen 3: Estructura While**

Fuente de la imagen:

<https://image.jimcdn.com/app/cms/image/transf/none/path/s37cc796afdd27ec2/image/i174a5e4a83b64a45/version/1368834058/image.jpg>

Y a través de un ejemplo:



En el vídeo que encontrarás un ejemplo de uso de la estructura while

 <https://youtu.be/yj-EkePaNw4>



*Para saber más sobre la estructura while*

<http://php.net/manual/es/control-structures.while.php>



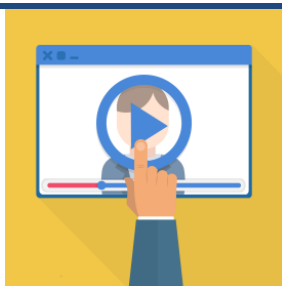
En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursosPHP/blob/master/estructurasControl2/while.php>

## 2.5 Aplicación de while con código html

Dos de los usos más frecuentes del while es bien algorítmico, bien repetición de código en html. Este último caso es el que vamos a ver en el siguiente ejemplo y en el que comenzaremos a comprobar la grandísima utilidad que significa el uso de php junto con html, css y js para automatizar la presentación de páginas web o vistas (ya veremos que significa una vista dentro del modelo MVC).



En el vídeo que encontrarás un ejemplo de uso de la estructura while junto con html



<https://youtu.be/CmfWoUKKs1Y>



En el siguiente enlace tienes el código utilizado:



<https://github.com/pacogomezarnal/codigoEjemploCursosPHP/blob/master/estructurasControl2/aplicandoWhile.php>