En PHP, las variables se especifican anteponiendo un signo **$** delante delidentificador que las define.

Como su propio nombre indica, estas pueden contener diferentes valores a lo largo de la ejecución del código PHP.

Algunos ejemplos de variables válidas:

```
$a
$mi_valor
$ColumnFactory
$nombre3
```

Intentar leer el valor de una variable que no ha sido anteriormente establecida provocará un error de tipo NOTICE, devolviendo un valor NULL en su lugar.

Las constantes deben ser declaradas inicialmente mediante la instrucción**define(NOMBRE_CTE,'valor')**, para luego ser utilizadas (sin el signo $ a diferencia de las variables) directamente con el nombre definido,**NOMBRE_CTE**. Es habitual definir y utilizar las contantes con todas las letras mayúsculas, aunque no es obligatorio.

Es posible definir una constante sin establecer ningún valor. De esta manera, si más adelante se comprobase si la constante está definida, nos serviría para detectar que se ejecutó el código donde se realizaba la definición.

Algunos ejemplos de defincón de constantes son:

```
define('COLOR_FONDO','#c8c8c8');
define('DEBUG_MODE');
```

Existe en PHP multitud de variables y constantes predefinidas, cuyo contenido podemos leer, para averiguar por ejemplo la IP del cliente conectado, o la versión de PHP que se está ejecutando. Puedes consultar la lista completa en estos enlaces:
* http://php.net/manual/en/reserved.variables.php*http://php.net/manual/es/reserved.constants.php

Un par de variables predefinidas interesantes son $_GET y $_POST. Estas contienen los valores de los campos de formulario que hayan sido enviados por el navegador (en función de si su método de envio ha sido **get** o **post**), y serán una forma muy importante de recibir información desde el usuario.

No queremos que en este curso sea preciso mucho conocimiento de HTML para aprender PHP, pero pongamos como ejemplo que se ha enviado un formulario en HTML con método GET que contenía un cuadro de texto definido como:

```
<input type="text" name="nombre"/>
```

Cuyo valor era "Manuel". Esto provocará que al leer la variable $_GET['nombre'], ésta devuelva la cadena "Manuel". Las variables definidas en las URLs del navegador con ? var1=valor1&var2=valor2 también aparecerán definidas en $_GET.

Otra variable predefinida interesante es $_SESSION. Esta permite que definamos y almacenemos cualquier valor en ella, de manera que si el usuario tiene habilitadas las cookies en su navegador, podamos en subsecuentes ejecuciones de nuestro código leer los valores introducidos. Esto es importante, ya que nuestro programa carece de estado entre llamadas, olvidando todo lo que se hubiera definido en ejecuciones anteriores. Solo es preciso almacenar en el navegador del visitante una cookie de sesión que lo identifique, luego todos los datos almacenados en $_SESSION se guardan en unos ficheros de textos asociados al valor de dicha cookie. Este método es útil para almacenar pequeñas cantidades de datos, pero para cantidades grandes es más óptimos utilizar un motor de base de datos como MySQL.

Hemois visto que siempre tenemos que dar un valor a una variable antes de intentar leer un valor de ella. La función isset nos permite comprobar si una variable existe, sin provocar ningún error en caso negativo. La funciónempty es similar, solo que devolverá verdadero si la variable no existe, o es la cadena vacia, o es el entero 0, y falso en todos los demás casos.

```
//$a no está todavía definida
echo isset($a); //No devuelve nada, es decir, devuelve falso
echo empty($a); //Devuelve "1", es decir, verdadero

$a='';
echo isset($a); //Devuelve "1", es decir, verdadero
echo empty($a); //Devuelve "1", es decir, verdadero (tiene cadena vacía)

$a='A';
echo isset($a); //Devuelve "1", es decir, verdadero
echo empty($a); //No devuelve nada, es decir, devuelve falso
```