VARIABLES

Las variables se reconocen con el signo $

El nombre de la variable es un espacio en la memoria del servidor que sirve para referenciar a un dato que puede o no variar, pero sabemos que esta siempre en memoria.

Hay varios tipos de datos:
$variableString = “Va con comillas para formar un texto unido”;
$variableString = ‘Puede ir con comillas dobles o simples’;

$variableNumerica = 8;
$variableFloat = 6.7;

$variableBooleana = true;

$variablePuedeNoTenerDatos = null;

$variableArray = [“Son distintos”,”tipos de”, “datos entre corchetes”, 23, false];
$variableArray[] = “Agrego otro string al array poniendo los corchetes en la variable”;

```
$arrayAsociativo = [
        “primer indice” => “String”,
        “segundo indice para bool” => true,
        “3” => 3
];
```

El array asociativo tiene un indice y y un valor. Recuperamos el dato (valor), llamando en el array a traves del indice:
echo $arrayAsociativo[“primer indice”]; //devuelve “String”

Estructura de un condicional IF

```
if(  condicion para llegar al true o false ){
        //aca va el codigo que se ejecuta si es verdadera la condicion
} else {
        //el else es opcional y se ejecuta si la condicion es falsa
}
```

Estructura del bucle FOR

```
for ( $i = 0; $i < count($array) ; $i++){
        // se ejecuta en cada vuelta con la variable $i con distinto valor
}
```

Dentro de los parentesis del for, hay tres pasos:

1) inicializar la variable $i.
2) Crear la condicion que mientras de true, va a seguir dando vueltas en el bucle
3) La accion que pasa con la variable $i al final de cada vuelta

Para arrays asociativos, se puede usar el FOREACH:

```
foreach ( $paises as $indice => $valor){

}
```

$paises seria el array asociativo a analizar, y en cada vuelta recuperamos el indice con la variable $indice, y el valor con la variable $valor. Tambien conocidos como $key => $value.

Funciones

Una funcion, es una accion que creamos, para poder ejecutarla las veces que necesitemos, y en donde la necesitemos.
Por si sola, no hace nada cuando la declaramos. Va a ejecutarse al momento de llamarla.
Se reconocen las funciones por los parentesis ( )

Para declararlas, esta es la estructura:

```
function nombreDeLaFuncion($parametro1,$parametro2){
        //ejecuto, analizo y/o uso los parametros para llegar al objetivo buscado
}
```

ejemplo:

```
function inscribirAlumno($nombreAlumno,$curso = “Full Stack”){
        echo “Bienvenido” . $nombreAlumno . “ al curso de “ . $curso;
}
```

La funcion puede o no tener parametros. En el ejemplo, el nombre del alumno es un parametro obligatorio a llenar cuando llamamos a la funcion, el curso no, porque tiene un valor por defecto al inicializar el parametro.

GET Y POST

Cuando enviamos datos desde un formulario HTML, los campos a llenar por el usuario, tienen un name=” “.
Ese valor que completamos en name, es el indice del array asociativo que se genera en PHP.

El array asociativo puede ser GET o POST.

Si el formulario envia a traves del metodo GET, arma en la URL, los name y los valores, que despues toma el array asociativo $_GET[]

digitalhouse.com?nombre=Diego&curso=FullStack

$_GET[‘nombre’] //devuelve Diego

$_GET[‘curso’] //devuelve FullStack

Si el formulario envia a traves de POST, llega sin visualizacion de los datos, el array asociativo POST

$_POST[‘user’] = “Gomez Pablo”;
$_POST[‘contrasena’] = “1234”;

Tanto GET como POST, son variables SUPERGLOBALES porque puedo usarlas donde quiero y no tienen un contexto predefinido.

VALIDACIONES

Las validaciones varian dependiendo el contexto y el dato, segun lo que necesitemos.
Podemos validar campos vacios comparandolos con “”.
Podemos preguntar si el dato existe isset()
Podemos validar formatos de email filter_var()
Podemos preguntar si los datos son del tipo que necesitamos is_array() is_numeric()
Y siempre vamos a poder crear logicas propias para validar lo que necesitemos.

JSON

JSON es un lenguaje que se usa para hablar entre servidores. Puede ser con el propio servidor de un sitio, o con un servidor externo, como por ejemplo, hablar con facebook, twitter o el sitio que necesiten.

Tiene reglas de sintaxis:
Siempre usar comillas dobles.
Usar {} llaves para englobar conjuntos de datos.
Asigna datos con dos puntos :

Ejemplo de JSON:

```
{
  “alumnos” : [
        {
```

```
            "nombre" : "Pedro",
            "apellido" : "Gomez",
            "email" : "pedro@gmail.com",
            "cursos" : [ "full stack" , "android", "ux" ]
           },
           {
            "nombre" : "Luciana",
            "apellido" : "Perez",
            "email" : "luciana@gmail.com",
            "cursos" : [ "ios" , "android", "marketing" ]
           }
    ]
}
```

Desde PHP podemos crear un array asociativo, y convertirlo en JSON a traves del metodo json_encode()
Y podemos traer un JSON y convertirlo en un array asociativo con json_decode()

Creando un archivo con formato json (archivo.json), podemos escribir datos con la funcion file_put_contents("archivo.json",$arrrayAsociativo);

Y podemos tomar datos de un archivo json con la funcion
file_get_content("archivo.json")

SESSION

Crear una sesion en php, nos permite persistir datos mientras el navegador este abierto para compartir datos entre secciones del sitio propio.

$_SESSION[] es un array asociativo como GET o POST.

Empieza vacio, y atraves de indices que creamos nosotros, podemos ubicar datos que vamos a reutilizar en otra parte.

ej: $_SESSION['usuario_id'] = 635;
     $_SESSION['avatarUsuario'] = "fotopedro.jpg";

Para comenzar la session se usa la funcion session_start() y se pone como primera linea de codigo en todos los archivos donde podamos usar una sesion abierta.

Para destruir la sesion, como en el caso del "logout" de usuarios, usamos la funcion session_destroy()

COOKIES

Las cookies tienen la misma utilidad que SESSION, pero en ves de persistir en el servidor mientras el navegador este abierto, se guardan en el navegador hasta que se vence la cookie.

Podemos crear cookies con al funcion setCookie(‘indiceDeCookie’, $valorAGuardar)

Y podemos recuperar los datos con el indice creado dentro de la superglobal $_COOKIE[‘indiceDeCookie’]