**Arrays**

$mi\_array[1]=23; //Asignación directa a la posición 1  
$mi\_array[]=23;//Asignación directa, empieza en el indice 0

$mi\_array[]= array(23,45,76)//Asignación a través de la función array()

$mi\_array[]=array(0 => 23, 1=>45, 2=>76)// Asignación a través de función array con indices  
o  
$mi\_array[]=>array("cero"=>23, "uno"=>45, 2=>76)

Para recuperar cualquier valor

echo $mi\_array["uno"] (dara 45 como output)

Otras funciones devuelven arrays, por ejemplo la funcion range() devuelve un array con valores numericos que van desde el valor de incio hasta el final, por ejemplo range

$mi\_array = range(120,130)

Ejemplo de como crear un array multidimensional

$colores = array ("fuertes", "suaves")  
$colores ["fuertes"]=array("rojo" =>"FF0000","verde"=>"00FF00","azul"=>"0000FF");  
$colores ["suaves"]=array("rosa"=>"FE9ABC","amarillo"=>"FDF189","malva"=>"9A2F68");  
echo $colores["fuertes"] ["rojo"]

Para recuperar los valores se utilizan un indice del primer array (fuertes o suaves) y un indice de alguno de los arrays secundarios

Otra forma de definir arrays multidimensionales pag. 137

Propiedades de los arrays

count($colores);  
count($colores["fuertes"]);

cuenta el numero de elementos, sizeof() hace lo mismo

in\_array("rojo",$colores);

busca dentro de un array un valor pasado como parametro. Booleano. Toma dos argumentos el valor a buscar y el array donde buscar.

unset ($colores[2]);

borra un elemento

**Contrucción Foreach**

$ciudades = array ("Badajoz","Mérida","Cáceres","Plasencia");  
foreach ($ciudades as $valor) {  
 echo ("El valor es $valor<br>")

foreach toma el array a recorrer y sus valores los va almacenando en la variable $valor a medida que el bucle se ejecuta

alternativamente

$ciudades = array ("Badajoz","Mérida","Cáceres","Plasencia");  
foreach ($ciudades as $indice => $valor) {  
 echo ("El índice $indice tiene el valor: $valor<br>")

**Funciones current() y next()**

$ciudades = array ("Badajoz","Mérida","Cáceres","Plasencia")  
$ciudades[España]="Madrid";  
$ciudades[Portugal]="Lisboa";  
$ciudades[Francia]="Paris";  
function recorre($ciudades) {  
 do {  
 $valor=current($ciudades);  
 echo ("El valor es: $valor<br>");  
 while (next($ciudades));

}  
recorre($ciudades);

El output sera la lista de cudades en orden

Para volver al principio se pude utilizar la función reset(), que envía el puntero al principio de la lista

function recorre ($ciudades) {  
 if (!current($ciudades)){  
 reset ($ciudades);  
}  
do{  
$valor =current($ciudades);  
echo ("El valor es: $valor<br>");  
while(next($ciudades));

*Las funciones prev() retrocede una posición el puntero, y end() se coloca al final de la lista de valores. Para obtener el valor del indice puede usar la función key()*

**Intercambio de Valores, Inversion de Contenido, Mezcla de Valores**

*la funcion array\_flip() intercambia los valores de indices y datos. array\_reverse() orderna a la inversa una lista de datos. suffle() mezcla todos los valores cada vez que se ejecuta.*

**Pilas: array\_push() y array\_pop()**

$pila=array();  
array\_push($pila,"uno","dos","tres");  
array\_push($pila,"cuatro","cinco");  
while ($valor = array\_pop($pila)){  
 echo "valor extraido es $valor<br>";  
}

El resultado por pantalla muestra que la extraccion de los valores se hace en orden inverso en el que fueron introducidos LIFO (last in, first out)

Ordenación de valores

Finalmente PHP ofrece una gran variedad de funciones para ordenar arrays

asort() ordena el array en forma ascendente  
arsort() ordena el array en forma descendente  
ksort()ordena el array en forma ascendente atendiendo esta vez al indice  
krsort() ordena el array en forma descendente atendiendo esta vez al indice  
sort() ordena de forma ascendente el array pasado como argumento, se pierde el valor asociativo entre indice y valor  
rsort() lo mismo en reverso