PHP-6

Funciones

Introducción a Funciones

- Las funciones „permiten encapsular código por funcionalidad y su reutilización.
- Las variables dentro de las funciones tienen ámbito local, para utilizar variables globales es necesario usar la palabra reservada global.
- Pueden :

–Recibir valores (parámetros o argumentos).

– Devolver valores (return).

- Sintaxis:

```
function nombreFunción (param1,param2)
{
  instrucción1;
  instrucción2;
  return valor_de_retorno;
}
```

Ejemplo:

```
function suma ($x, $y)
{
 $s = $x + $y;
 return $s;
}
$a=1;
$b=2;
$c=suma ($a, $b);
print $c;
```

Definición

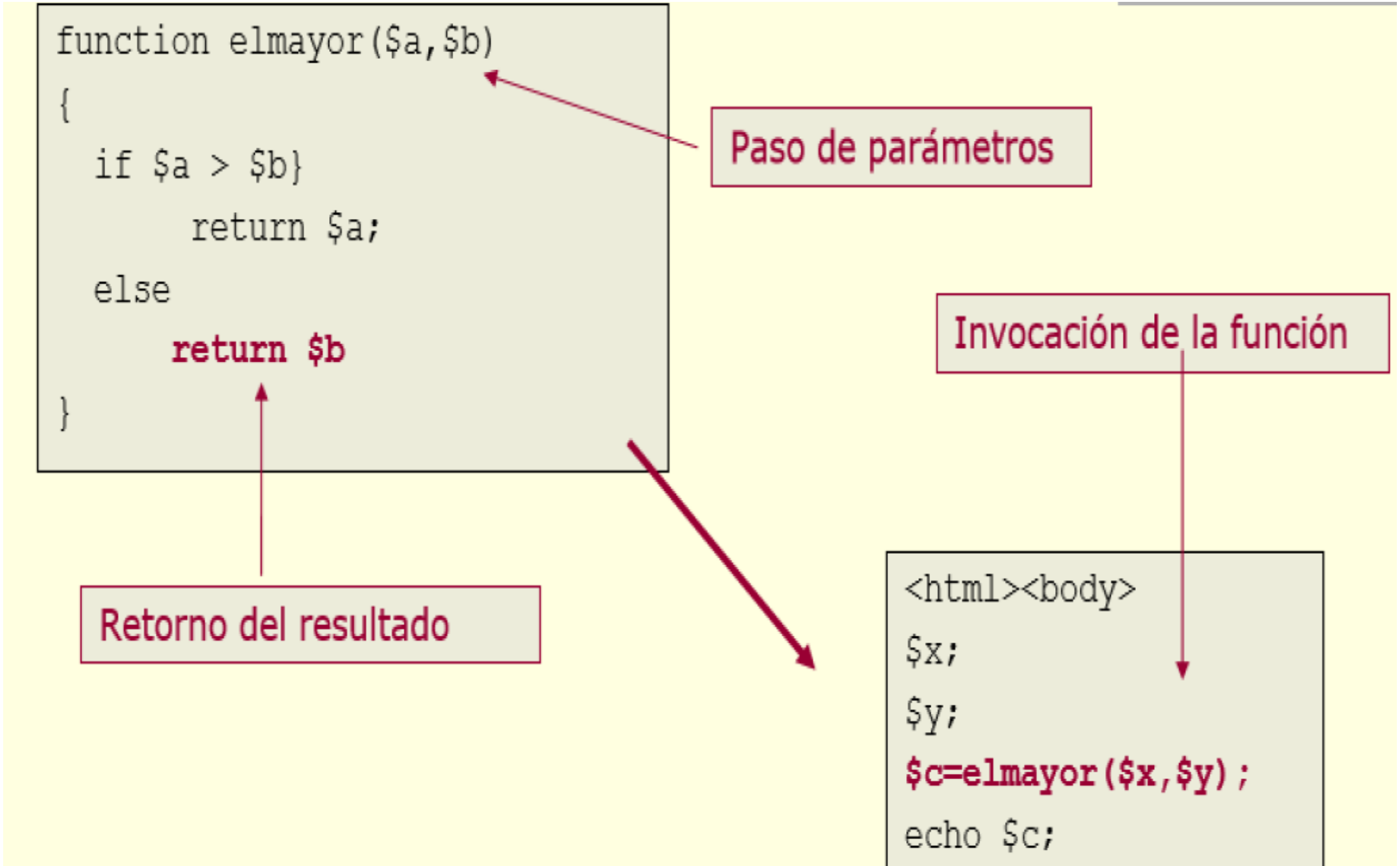
Invocación

- La primera línea es la cabecera de la función y consta de:

– La palabra clave function.

–El nombre de la función, que no debe llevar acentos, espacios en blanco, ni caracteres especiales.

–La lista de parámetros encerrada entre paréntesis, separados por comas ',‘.


function elmayor($a,$b)
{
if $a > $b}
return $a;
else
return $b
}
Paso de parámetros
Invocación de la función
Retorno del resultado
<html><body>
$x;
$y;
$c=elmayor($x,$y);
echo $c;

Paso de parámetros

- Por defecto los parámetros se pasan por valor

–Cambiar el valor dentro de la función pero no lo cambia fuera (defecto) .

```php
function sumar_uno($x) {
    $x++;
    echo $x;
}

$a = 2;

sumar_uno($a); // 3

echo $a; // 2
```

Paso de parámetros

- Paso por referencia:

–Cambia el valor dentro y fuera de la función.

–Usar &

```php
function sumar_uno(&$x) {
    $x++;
    echo $x;
}

$a = 2;

sumar_uno($a); // 3

// $a se pasó por referencia a suma_uno()
// El cambio dentro de la función
// se refleja en la referencia original
echo $a; // 3
```

Parámetros por defecto

- Sirven para que los parámetros contengan un dato predefinido, con el que se inicializarán si no se le pasa ningún valor en la llamada de la función. Los valores por defecto se definen asignando un dato al parámetro al declararlo en la función.

```
function muestranombre ($titulo = "Sr.")
{
    print "Estimado $titulo:\n";
}
muestranombre ();
muestranombre ("Prof.");
```

Salida →

```
Estimado Sr.:
Estimado Prof.:
```

Parámetros por defecto

- Los argumentos con valores por defecto se declaran al final en la lista de parámetros de la cabecera de la función.

```
function muestranombre ($apel, $titulo= "Sr.")
{
   print "Estimado $titulo $apel:\n";
}
muestranombre ("Fernández");
muestranombre ("Fernández", "Prof.");
```

Salida

```
Estimado Sr. Fernández:
Estimado Prof. Fernández:
```

Ámbito de las variables

- Hay tres posibles ámbitos de una variable son: „

–Variables locales

- Definidas dentro de una función y sólo pueden ser accedidas desde dentro de esta función.
- Si hay una variable local y otra global con el mismo nombre , dentro de la función se usa la local.
- Pierden su valor al salir de la función

Ámbito de las variables

- Las **funciones** en php son **bloques independientes**, las variables que se definen dentro de una función tienen **ámbito local** (***local scope***) y no afectan a otras en el **script global**, de la misma forma que las **variables globales** no están disponibles dentro de las funciones:

```php
function saludar()
{
    $saludo = "Hola";
    print $saludo;
}

$saludo = "Buenos dias";
saludar(); // Devuelve: Hola
print $saludo; // Devuelve: Buenos días
```

Ámbito de las variables

- La ejecución del script empieza en *$saludo = "Buenos días"*, y después llama a la función *saludar()*. Esta función establece *$saludo* a *"Hola"* y lo imprime, y después devuelve el control al script principal donde se imprime *$saludo*. Como véis sus valores son diferentes y no se afecta el uno al otro.

Ámbito de las variables

- Variables locales estáticas

– Existen sólo en el ámbito local de la función

–No pierden su valor cuando el programa abandona la función

–Anteponer la palabra reservada static.

–Las variables estáticas **no pueden tener un resultado derivado** de una expresión a la hora de definirlas

```
function Test ()
{
static $a = 0;
echo $a;
$a++;
}
```


Cada vez que se llame a la función Test(), se representará el valor de $a y se incrementará.
La primera vez que se llame a Test escribirá un 0, luego 1,2,…

Ámbito de las variables

- Variables globales

–Se definen fuera del cuerpo de una función

–Accesibles desde cualquier punto del código.

–Deben ser declaradas globales dentro de la función para usarlas o bien usar el array $GLOBALS(ojo: No $_GLOBALS)

Ámbito de las variables

```php
// --- Ejemplo con la palabra global
$x = 2;
$y = 20;

function suma()
{
    global $x, $y;
    return $x + $y;
}

echo suma(); // Devuelve: 22
// --- Ejemplo con $GLOBALS
$x = 4;
$y = 21;

function otraSuma()
{
    return $GLOBALS['x'] + $GLOBALS['y'];
}

echo otraSuma(); // Devuelve 25
```

Devolución de valores

- Para devolver valores en funciones en PHP se utiliza la estructura de control return, que representa el valor de respuesta de la función
- Se puede devolver cualquier variable que se quiera, ya sea un integer, un string, un array, …
- Si se quiere devolver más de un valor, es necesario utilizar un **array**:

```php
function numeros(){
    return array (1, 2, 3);
}

list($uno, $dos, $tres) = numeros();
```

Devolución de valores

- El valor devuelto desde una función

-Puede ser asignado a una variable

–Puede ser utilizado dentro de una expresión.

- Devuelve sólo un único valor„

Devolución de valores

- Usar la palabra reservada return

–Cuando aparece return la función deja de ejecutarse

–Si después de return hay más líneas de código, no se ejecutarán nunca

```
function prueba()
{
    print "Esta es mi función";
    return 1;
    print "Ahora saldré de la función";
}

print prueba(); //  Devolverá: Esta es mi función1
```

Funciones con un número variable de parámetros

- PHP permite una lista de valores de longitud variable como parámetro
- Funciones a usar:
- func_num_args()=>número de argumentos pasados a la función.
- func_get_args() =>array con los argumentos pasados a la función
- func_get_arg(num) =>el argumento que está en la posición num en la lista de argumentos. La primera posición es la 0

```php
function prueba()
{
 $num_args = func_num_args();
 echo "Numero de argumentos:$num_args<br/>\n";
 if ($num_args >= 2) {
 echo "El 2ºargumento es:".func_get_arg(1)."<br/>\n";}
 $parametros=func_get_args();
 echo "Array con todos los argumentos:<br />\n";
  print_r($parametros);
}
prueba (1, 2, 3);
```

Numero de argumentos: 3
El 2º argumento es: 2
Array con todos los argumentos:

```
Array
(
    [0] => 1
    [1] => 2
    [2] => 3
)
```

Funciones recursivas

- Una función se llama recursiva cuando en algún punto de su cuerpo se llama a sí misma.
- Cuidado! puede llamarse a sí misma indefinidamente.
- Es muy importante la condición de salida.

Funciones recursivas

- Calcula el factorial de un número:

```php
<?php
function factorial($n)
{ if($n==1)
return 1;
else
return $n * factorial($n-1); }

echo "El factorial de 5 es ".factorial(5);
?>
```

El factorial de 5 es 120