Laboratorio Progettazione Web
Le funzioni in PHP

Andrea Marchetti IIT-CNR

2017/2018

Overview

- Introduzione
- Definizione Vs Invocazione
- Parametri formali e reali
- Visibilità delle variabili
- Passaggio parametri per valore e per reference
- Librerie di funzioni

Funzioni

- Il PHP ha molte funzioni già definite (built-in) che possiamo usare

  `print(), echo(), random(),…`

- Il nome di una funzione è case insensitive

```
print(), Print(), PrInT() // sono la stessa funzione
```

- **Possiamo definire nuove funzioni**

Motivazioni

- Una funzione raccoglie una **sequenza di istruzioni** che svolgono una "funzione" particolare
- Nella programmazione possiamo incontrare più volte la stessa **sequenza di istruzioni**
- Conviene estrarre questa sequenza ed inserirla in una funzione che **invocheremo** invece di riscrivere la stessa sequenza

Effetti

- Programmazione più rapida
  - riuso del codice
- Codice più chiaro
- Manutenzione migliore

Definizione Vs Invocazione

Definizione di funzione

```
function pippo($a){
   $a++;
   print($a);
}
```

Invocazione di **funzione**

```
$a=3;
pippo($a); // stampa 4
```

Parametri

- Informazione può essere passata ad una funzione con i parametri
- Un parametro è una **variabile locale** per cui può essere assegnata nel corpo della funzione

```
function name ($p1,…,$pn){
    codice da eseguire
}
```

Parametri

- I parametri di una funzione all'interno del corpo di una funzione sono equivalenti a delle variabili
- Servono a rendere il codice riutilizzabili in contesti differenti
- In fase di **definizione** della funzione i parametri sono detti **parametri formali** (formal par.)
- In fase di **invocazione** della funzione i paramteri sono detti **parametri reali** (actual par.) o argomenti

Esempio passaggio di parametri

```
<?php
function dichiarazione($nome,$citta,$data,$residenza){
    print("Il sottoscritto $nome nato a $citta il $data e
    residente in $residenza");
}

dichiarazione("Paolo Rossi","Vicenza","25/10/1992","via
Giuseppe Verdi 43, Pisa");

dichiarazione("Vito Bianchi", "Caltanissetta",
"5/03/1996","via XX Settembre 12, Pisa");

?>
```

Valore di default di un parametro

- Possiamo definire dei valori di default per parametri
- Se nella definizione imposto dei valori di default, al momento dell'invocazione posso omettere di passare l'argomento
- Attenzione se abbiamo più parametri con valore di default

Valore di default di un parametro

```
<?php
function
dichiarazione($nome,$citta,$data,$residenza="Pisa"){
    print("Il sottoscritto $nome nato a $citta il $data e
    residente in $residenza");
}

dichiarazione("Paolo Rossi","Vicenza","25/10/1992");

dichiarazione("Vito Bianchi","Caltanissetta","5/03/1996");

?>
```

Variabili Globali

```
<?php
$residenza = "Pisa";

function dichiarazione($nome,$citta,$data){
   global $residenza;
    print("Il sottoscritto $nome nato a $citta il $data e
    residente in $residenza");
}

dichiarazione("Paolo Rossi","Vicenza","25/10/1992");

dichiarazione("Vito Bianchi","Caltanissetta","5/03/1996");

?>
```

Visibilità/utilizzabilità di una variabile

- Parametri (variabili locali)
  - sono visibili solo all'interno della funzione e in generale dove sono definite
- Variabili globali
  - sono visibili in tutto il codice
- Variabili statiche
  - come le variabili locali ma mantengono sempre lo stesso valore

Restituzione di valore

```php
<?php
function somma($x, $y) {
    $z = $x + $y;
    return $z;
}

echo "5 + 10 = " . somma(5, 10) . "\n";
echo "7 + 13 = " . somma(7, 13) . "\n";
echo "2 + 4  = " . somma(2, 4);

?>
```

Restituzione di valore

Una funzione può restituire qualsiasi tipo di valore compreso gli array

```
function getTemperature(){
        $temperature = array(24,25,17,18);
        return ($temperature);
}

$valori = getTemperature();
foreach($valori as $valore) print($valore);
```

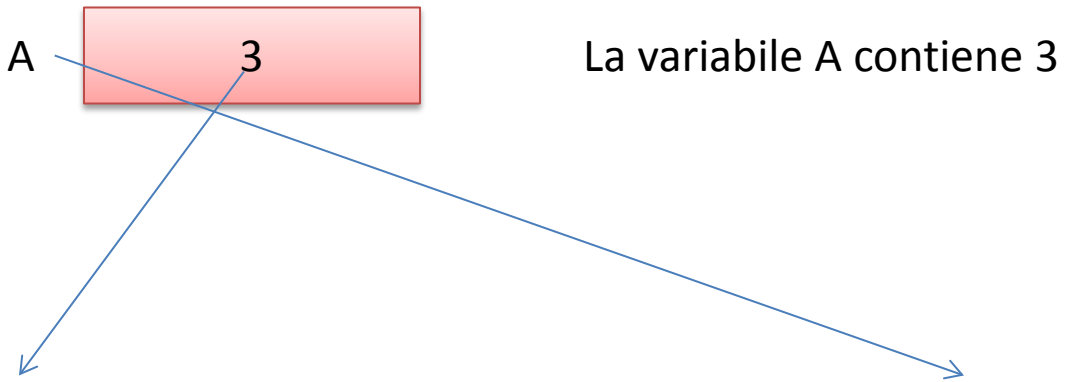
Passaggio dei parametri per valore

Qualsiasi modifica ai **parametri formali** (quelli cioè che compaiono nella definizione, non si riflette (per quanto visto finora) automaticamente sui **parametri reali** (quelli effettivamente usati in una chiamata della funzione).

```
function f1($a) {$a++;} // definizione di f1

$b=3;

f1($b);     // invocazione di f1, incrementa $b di 1

print($b);  // stampa 3
```

Passaggio dei parametri per reference

Affinchè le modifiche sul parametro reale fatte dalla funzione rimangano persistenti dopo l'invocazione devo passare i parametri per reference



Passaggio per **valore**
passo 3

Passaggio per **reference**
passo l'indirizzo

Passaggio dei parametri per reference

- In fase di dichiarazione posso stabilire che la funziona accetta i parametri per reference.
- In fase di invocazione non cambia nulla

Esempio passaggio per reference

```php
<?php

// Funzione con passaggio parametro per valore
function f1($a) {$a++;}

// Funzione con passaggio parametro per riferimento
function f2(&$a){$a++;}

$a=3;

f1($a);
print($a); // stampa 3

f2($a);
print($a); // stampa 4
```

Librerie di funzioni

- Una libreria di funzione è un insieme di funzioni che hanno qualcosa in comune
- Per importare la libreria di funzioni utilizzare la funzione built-in **include**() o **require**() o **require_once**()

Librerie di funzioni

- Ad esempio uno potrebbe crearsi una libreria
  - per la connessione ad un DB
  - per gestire il codice HTML
  - in generale per raggruppare funzioni dello stesso tipo e riutilizzabili in script differenti

Libreria di funzioni per MySql

- Motivazioni
  - raggruppare in un unico file le operazioni classiche connessione, select, comando generico sql, chiusura
  - gestire in modo estensivo i possibili errori

Connessione

Uso i parametri di default per i parametri che cambiano raramente. Attenzione all'ordine

```
/*******************************
 * Open a Connection to MySQL *
 *******************************/

function openDB($database="lpw",  $password=NULL,
                $username="root", $servername="localhost"){

   /* Create connection */
   $conn = mysqli_connect($servername, $username, $password, $database);
   if (!$conn) die("Connection failed: ".mysqli_connect_error());

   /* change character set to utf8 */
   if (!mysqli_set_charset($conn, "utf8"))
         printf("Error set utf8: ". mysqli_error($conn));


   return $conn;
}
```

Istruzione sql (insert, delete, …)

```
/*******************************
 * Istruzione generica SQL    *
 *******************************/

function sql($conn,$sql){
   // Esecuzione query
   $resultSet = mysqli_query($conn, $sql);
   if(!$resultSet)
         print("Errore esecuzione $sql:" . mysqli_error($conn));
}
```

Select

```
/*******************************
 * Lettura dei records         *
 *******************************/

function select($conn,$sql){
   // Esecuzione query
   $resultSet = mysqli_query($conn, $sql);
   if(!$resultSet) print("Errore esecuzione $sql:" . mysqli_error());

   // Copio i records in un array associativo
   $records = array();
   while ($record = mysqli_fetch_assoc($resultSet)) $records[]=$record;

   // Liberazione della memoria impegnata dal result set
   mysqli_free_result($resultSet);

   return $records;
}
```

Chiusura connessione

```
/*********************************
 * Close the Connection to MySQL *
 *********************************/
 function closeDB ($conn){
   mysqli_close($conn);
}
```

Esempio di utilizzo libreria

```
<?PHP
include("dbLibrary.php");

$db   = openDB();
$recs = select($db,"SELECT * FROM capitali");

foreach($recs as $rec)
   print($rec['nazione']." con capitale "$rec['capitale'];

closeDB($db);
?>
```