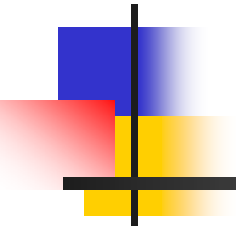
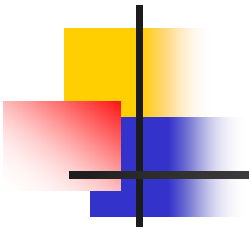


Il linguaggio PHP

Elementi base



Il Linguaggio PHP

- PHP è un acronimo ricorsivo
  che sta per PHP Hypertext Preprocessor
- Sintassi di base:
  un blocco di scripting PHP
  può stare ovunque in un documento,
  inizia con <?php e finisce con ?>
- Blocchi successivi al primo possono anche
  essere nella forma <? ... ?>
  nei server abilitati

Sintassi di base

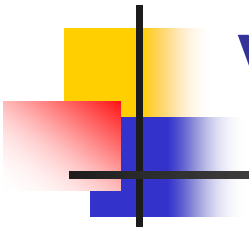
- Un file PHP può contenere codice HTML e blocchi di scripting, es:

```
<html>
<body>
<?php
   echo "Hello World";
?>
</body>
</html>
```

- Esercizio: salvare come file hello.php nella cartella www del server Apache e caricare col browser come http://localhost/hello.php

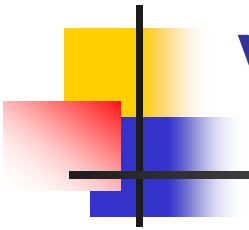
Sintassi di base (2)

- Ogni linea di codice PHP deve terminare con ; (punto e virgola) che funge da separatore fra le istruzioni
- I commenti si possono inserire come:
  // singola linea
  oppure:
  /* blocco di più linee
  di commenti */
- Ci sono due istruzioni di stampa: echo e print



Variabili

- Si possono dichiarare variabili per memorizzare valori (numeri e stringhe):
  $nome_var = valore ;
- Tutte le variabili iniziano con $, seguito da una lettera o underscore, seguito da un numero qualsiasi di caratteri alfanumerici e underscore, es: $my_var, $myVar, $Var01, $_var_01



Variabili (2)

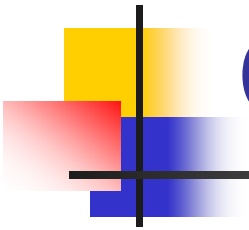
- PHP è un linguaggio *loosely typed* (come altri linguaggi di scripting, es. Javascript e Perl):
  - Le variabili sono dichiarate automaticamente qundo vengono usate
  - Non è necessario dichiarare il tipo delle variabili
  - PHP converte automaticamente i valori delle variabili al tipo corretto dei dati

Stringhe

- Es. uso di stringhe
  (. è operatore di concatenazione):

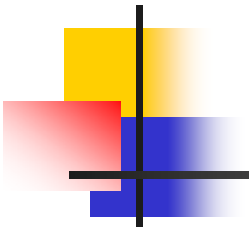
- ```
  <?php
  $txt="Hello World";
  echo $txt;
  ?>
  ```

- ```
  <?php
  $txt1="Hello";
  $txt2="World";
  echo $txt1 . " " . $txt2;
  ?>
  ```



Operatori

- Non differiscono da quelli normalmente in uso nei linguaggi di programmazione:
- Aritmetici: + - * / % ++ --
- Assegnamento: = += -= *= /= .= %=
- Confronto: == != < > <= >=
- Logici: && || !



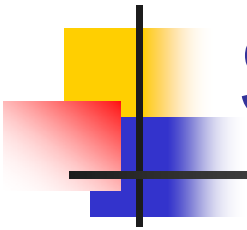
Istruzioni condizionali

- Istruzione if ... else:
- if (condizione)
  codice eseguito se condizione è true;
  else
  codice eseguito se condizione è false;
- Esiste anche l'istruzione elseif per testare condizioni multiple

Istruzioni condizionali (2)

- Esempio:

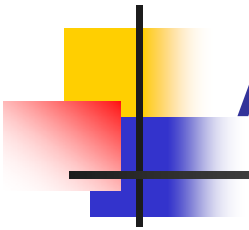
```
<html>
<body> <?php
$d=date("D");
if ($d=="Fri")
  echo "Have a nice weekend!";
elseif ($d=="Sun")
  echo "Have a nice Sunday!";
else
  echo "Have a nice day!";
?> </body>
</html>
```



Selezione multipla

- Istruzione switch:

```
switch (espressione)
{
case label1:
  codice eseguito se espressione = label1;
  break;
case label2:
  codice eseguito se espressione = label2;
  break;
default:
  codice eseguito se espressione è diversa
  sia da label1 che da label2;
}
```

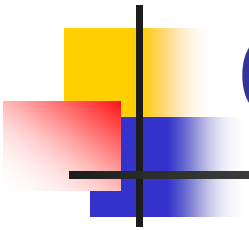


Arrays

- Esistono array con indici numerici, con chiave associativa e multimensionali
- $names = array("Pippo","Pluto","Minnie");
- equivale a:
- $names[0] = "Pippo";
  $names[1] = "Pluto";
  $names[2] = "Minnie";

Arrays (2)

- Esempio con chiave associativa:
- $ages = array("Pippo"=>32, "Pluto"=>30, "Minnie"=>34);
- oppure:
- $ages['Pippo'] = "32";
  $ages['Pluto'] = "30";
  $ages['Minnie'] = "34";



Cicli

- Esistono diversi tipi di ciclo:
- while - test condizione all'inizio
- do ... while - test condizione alla fine
- for – esegue il ciclo un numero prefissato di volte
- foreach – esegue il ciclo per tutti gli elementi di un array

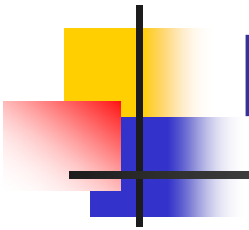
Cicli - esempi

```
<?php
$i=1;
while($i<=5)
  {
  echo "Il numero è " . $i . "<br />";
  $i++;
  }
?>
```

```
<?php
for ($i=1; $i<=5; $i++)
{
  echo "Hello World n. " . $i . "<br />";
}
?>
```

```
<?php
$i=0;
do
  {
  $i++;
  echo "Il numero è " . $i . "<br />";
  }
while ($i<5);
?>
```

```
<?php
$arr=array("uno", "due", "tre");
foreach ($arr as $value)
{
  echo "Valore: " . $value . "<br />";
}
?>
```



Funzioni

- Dichiarazione:

```
function nome_funzione(parametri)
{
    blocco di codice costituente
    il corpo della funzione
}
```

- Invocazione:

```
nome_funzione(valori_attuali);
```

Funzioni - esempi

```
<html>
<body><?php
function writeMyName()
  {
  echo "Pinco Pallino";
  }
echo "Ciao a tutti!<br />";
echo "Il mio nome è "; writeMyName();
echo ".<br />Esatto, "; writeMyName();
echo " è proprio il mio nome.";
?></body>
</html>
```

L'output prodotto da tale codice è:

Ciao a tutti!
Il mio nome è Pinco Pallino.
Esatto, Pinco Pallino è proprio il mio nome.

Funzioni – esempi (2)

Con passaggio di paranetri:

```
<html>
<body><?php
function writeMyName($fname,$punctuation)
  {
  echo $fname . " Pallino" . $punctuation . "<br />";
  }
echo "Il mio nome è "; writeMyName("Tizio",".");
echo "Il mio nome è "; writeMyName("Caio","!");
echo "Il mio nome è "; writeMyName("Sempronio","...");
?></body>
</html>
```

Produce come output:

Il mio nome è Tizio Pallino.
Il mio nome è Caio Pallino!
Il mio nome è Sempronio Pallino...

Funzioni – esempi (3)

- Le funzioni possono essere usate per restituire valori
- Esempio:

```
<html>
<body><?php
function add($x,$y)
  {
  $total = $x + $y;
  return $total;
  } echo "1 + 16 = " . add(1,16);
?></body>
</html>
```

- Il codice produce il seguente output:

1 + 16 = 17

Gestione di form HTML

- Gli elementi form HTML di inserimento sono sempre accessibili agli script PHP
- Esempio di form:

```
<html>
<body>
<form action="welcome.php" method="get">
Nome: <input type="text" name="name" />
Età: <input type="text" name="age" />
<input type="submit" value ="Invia" />
</form>
</body>
</html>
```

- Cliccando sul bottone submit (Invia) i dati sono inviati sul server al file welcome.php che li può processare

Gestione di form HTML (2)

- Se il file welcome.php ha questo contenuto:

- ```
  <html>
  <body>Benvenuto <?php echo $_GET["name"]; ?>.<br />
  La tua età è di <?php echo $_GET["age"]; ?> anni.</body>
  </html>
  ```

- Un esempio di esecuzione dello script è la seguente:

- Form:

  Nome: Tizio Età: 25 Invia

- Output dopo click su bottone Invia:
  Benvenuto Tizio.
  La tua età è di 25 anni.
  ```

Gestione di form HTML (3)

- Quando il browser visualizza il form:
  ```
  <form action="welcome.php" method="get">
  Nome: <input type="text" name="name" />
  Età: <input type="text" name="age" />
  <input type="submit" value ="Invia" />
  </form>
  ```
- Supponiamo l'utente abbia digitato nei campi Nome ed Età rispettivamente i valori Tizio e 25
- Cliccando sul bottone Invia, il browser invia al server Web la richiesta dell'URL (metodo get):
  http://localhost/welcome.php?name=Tizio&age=25

Gestione di form HTML (4)

- Alla richiesta dell'URL
  http://localhost/welcome.php?name=Tizio&age=25

- Il server Web processa il file welcome.php, inzializzando (metodo get) un array di nome $_GET in questo modo:
  $_GET['name'] = "Tizio";
  $_GET['age'] = "25";

- Gli script all'interno del file welcome.php possono pertanto accedere al contenuto di tali variabili

Gestione di form HTML (5)

- Quindi il file welcome.php:

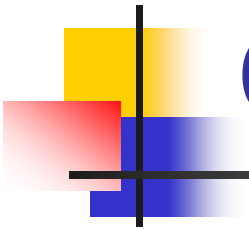
```
<html>
<body>Benvenuto <?php echo $_GET["name"]; ?>.<br />
La tua età è di <?php echo $_GET["age"]; ?> anni.</body>
</html>
```

- Viene trasformato in:

```
<html>
<body>Benvenuto Tizio <br />
La tua età è di 25 anni.</body>
</html>
```

prima di essere consegnato al browser che ha processato il form

Gestione di form HTML (6)



- In alternativa al metodo get può essere usato il metodo post
- Le variabili non sono passate tramite l'URL ma tramite il campo body della richiesta HTTP
- Il file PHP può accedere ad esse tramite l'array $_POST che viene inizializzato automaticamente

Gestione di form HTML (7)

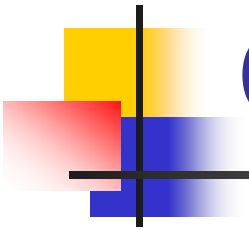
- Nell'esempio, il form diventa:

```
<form action="welcome.php" method="post">
Nome: <input type="text" name="name" />
Età: <input type="text" name="age" />
<input type="submit" value ="Invia" />
</form>
```

- E il file welcome.php diventa:

```
<html>
<body>Benvenuto <?php echo $_POST["name"]; ?>.<br />
La tua età è di <?php echo $_POST["age"]; ?> anni.</body>
</html>
```

Gestione di form HTML (8)



- Con il metodo get:
  - Il contenuto delle variabili è visibile in chiaro all'interno dell'URL
  - La lunghezza dei valori è limitata a 100 byte
  - E' però possibile creare nel browser un bookmark con l'URL completa di parametri
- In alternativa alle variabili $_GET e $_POST, negli script PHP è possibile usare l'array $_REQUEST (valido con entrambi i metodi)