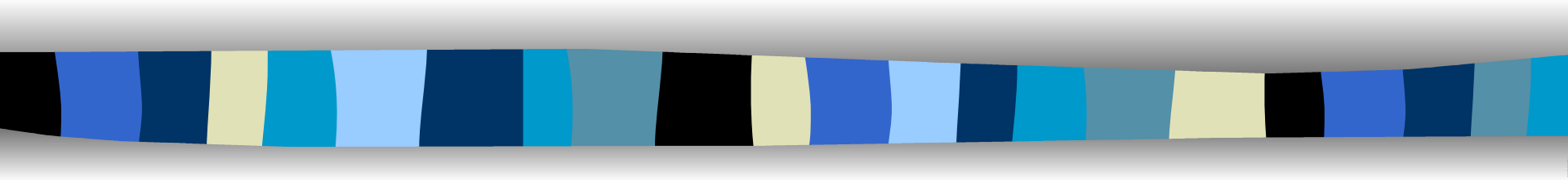
PHP – 1



Elementi del linguaggio

Che cosa è PHP – 1

- PHP è l'acronimo *ricorsivo* di "PHP HyperText Preprocessor"
  - Inizialmente era l'acronimo di "Personal Home Page"
- PHP è un linguaggio di programmazione creato da Rasmus Lerdorf nel 1994 per costruire delle estensioni in documenti HTML e migliorare così la sua home page personale
  - Porzioni del documento HTML sono generate dinamicamente
- PHP è *open source*

Che cosa è PHP – 2

- PHP convive normalmente all'interno di documenti HTML
  - È possibile creare degli script in PHP eseguiti da una shell (#!/usr/local/bin/php –q)
- PHP viene normalmente eseguito *dal server* prima che la pagina venga inviata all'utente
- L'output di PHP è normalmente codice HTML per il browser, ma ci sono molte eccezioni
  - Può generare anche immagini, documenti pdf, documenti XML, filmati Flash, …

Breve storia di PHP

- 1994
  - PHP fu pensato e sviluppato nell'autunno del 1994 da Rasmus Lerdorf (membro del team di sviluppo di Apache).
- 1995
  - Rilasciato PHP/FI (Form Interpreter) versione 2. Permetteva l'accesso a database (MySql)
- 1997
  - Rilasciato PHP versione 3 (riscrittura in C++ dell'interprete da parte di Zeev Suraski e Andi Gutmans)
- 2000
  - Rilascio di PHP 4 basato sul motore di scripting Zend www.zend.com
- 2004
  - Rilascio di PHP 5

Testi di riferimento e versione PHP

- Rasmus Lerdorf e Kevin Tatroe
  Programming PHP
  O'Reilly, ISBN 1-56592-610-2
- Manuale PHP su www.php.net
- Non usate versioni vecchie di manuali
  - fate attenzione, da PHP 4.2.0 l'insieme di default di variabili predefinite accessibili nello scope globale è cambiato
    - Ad esempio, si accede a $_GET['nome'] invece che a $nome

Vantaggi nell'uso di PHP

- La curva di apprendimento è brevissima
- Veloci tempi di sviluppo
- Alte prestazioni e stabilità
- Supporto dei sistemi operativi principali (UNIX, Linux, Windows, …)
- Supporto nativo per i database più popolari
- Molte librerie built-in
- Pre-installato nelle distribuzioni Linux

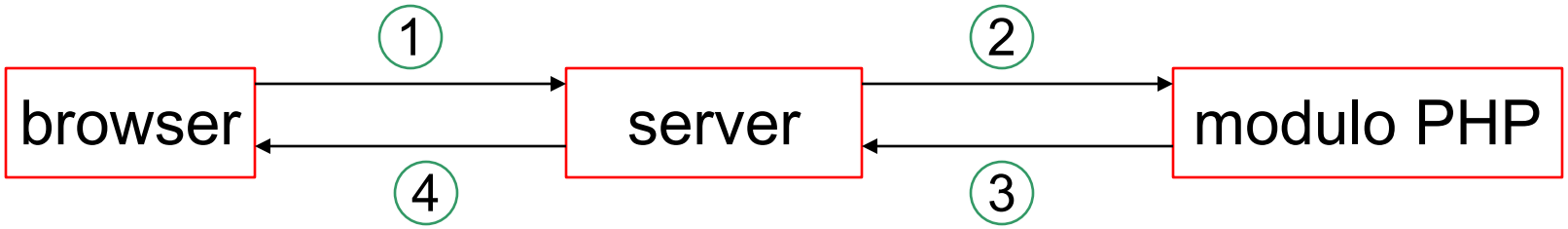
Il primo programma PHP (ciao.php)

```
<HTML>
<HEAD> <TITLE> Il classico programma
  </TITLE> </HEAD>
<BODY>
<H1>Indovinate cosa scriviamo?</H1>
<?php
   echo "<em>Ciao mondo!</em> ";
?>
</BODY>
</HTML>
```

Ecco cosa viene inviato al client

```
<HTML>
<HEAD> <TITLE> Il classico programma
  </TITLE> </HEAD>
<BODY>
<H1>Indovinate cosa scriviamo?</H1>
<em>Ciao mondo!</em>
</BODY>
</HTML>
```

Architettura



1 il browser richiede al server un documento con estensione .php

2 il server invia il documento al modulo PHP

3 il modulo PHP interpreta lo script produce un output e lo invia indietro

- tutto quello che non è PHP non viene interpretato dal modulo e viene rinviato al server

4 il server risponde alla richiesta del browser inviandogli l'output del modulo PHP

Come inserire codice PHP

- Tutto quello che è racchiuso dai tag **<?php** e **?>** viene interpretato dal modulo PHP
- Esistono altri modi per inserire del codice PHP all'interno di documenti HTML, bisogna configurarli in php.ini
  - **<?** CODICE PHP **?>** (short_open_tag)
  - **<%** CODICE PHP **%>** (asp_tags)
  - **<script language="php">**
    CODICE PHP **</script>**

Pagina di configurazione

- Per vedere quali particolari estensioni sono state installate o per vedere come il file php.ini è stato configurato è possibile eseguire il seguente script

  <?php phpinfo(); ?>

  - Inserire la riga precedente in un file (e.g., info.php) e poi invocare http://localhost/info.php
- La funzione phpinfo() crea una pagina HTML contente informazioni su come PHP è stato installato

ESEMPIO

Dove mettiamo gli script PHP? – 1

- Ovunque!
  - O quasi, dobbiamo vedere come il web server è stato configurato
- Per Apache, all'interno del file default-server.conf bisogna cercare la linea contenente la direttiva DocumentRoot, la directory ad essa associata indica il punto da dove il server andrà a cercare i documenti richiesti

Dove mettiamo gli script PHP? – 2

- La directory dove si trova default-server.conf (httpd.conf), e come DocumentRoot è configurata, cambia a seconda del sistema operativo e della distribuzione utilizzata

Esempio di httpd.conf

- In Linux
  - RedHat, il file httpd.conf si trova nella directory /etc/httpd/conf
  - SuSE, il file httpd.conf si trova nella directory /etc/apache2/
- La direttiva DocumentRoot è configurata di default come segue

  DocumentRoot "/srv/www/htdocs“
- Utilizzo di grep

PHP in Laboratorio

- Configurazione Apache 2.0.x
- Il servizio è disponibile sia da macchina windows che da macchina linux.
- le pagine php devono avere estensione .php.
- Ogni utente potrà visualizzare il suo spazio web all'indirizzo http://webstudenti.csedu-priv/home/$USERNAME/
- Le directory dove pubblicare i propri siti web saranno
- - su linux $HOME/public_html
- - su windows z:\public_html
- I path sopra corrispondono allo stesso file system (la $HOME di linux viene montata come drive z su windows).

Inviare informazioni al browser

- Si usano alcune funzioni quali:
  - print, echo, printf
- Esse inseriscono una stringa nel flusso di informazioni di HTTP dal server verso il client

Andare a capo

- Come in C, il carattere “\n” significa newline (a capo)
- Nel codice HTML l’interruzione di riga viene di solito ignorata
- Occorre utilizzare il tag
 nelle stringhe che passiamo a print o a echo
- Possiamo usare tutti i tag HTML che vogliamo

Inserire commenti in PHP

- Esistono vari modi per inserire commenti all'interno di script PHP
- Commenti su singola linea
  - // COMMENTO
  - # COMMENTO
- Commenti su più linee
  - /* COMMENTO */

Lettere maiuscole o minuscole?

- I nomi di classi e funzioni definite dall'utente così come costrutti predefiniti e parole chiave (tipo: while, class, echo, …) **sono** *case insensitive*
- Ad esempio le seguenti tre linee di codice sono equivalenti
  - echo "ciao a tutti";
  - Echo "ciao a tutti";
  - EcHo "ciao a tutti";

Le variabili in PHP – 1

- Sono rappresentate dal simbolo del dollaro ($) seguito dal nome della variabile
  - Il nome di una variabile inizia con una lettera o un *underscore* (_) seguito da un qualsiasi numero di lettere, numeri o *underscore,* e caratteri ASCII da 127 a 255
- Sono *case-sensitive*
- Non è necessario dichiarare le variabili

Le variabili in PHP – 2

- Il linguaggio è non *tipato*
- Le variabili utilizzate prima di essere assegnate hanno valori predefiniti
  - assumono il valore 0 se utilizzate in ambito numerico
  - sono la stringa vuota se utilizzate come stringhe
  - assumono il valore false se utilizzate come valore booleano
- Morale: inizializzate sempre voi le variabili, tenete sotto controllo il vostro codice!

Esempi

- L'operatore di assegnamento è l'uguale

```
$A = 1;
$B = "2";
$C = ($A + $B);        // Somma di interi
$D = $A . $B;          // Concatenazione di strighe
echo $C;               // Stampa 3
echo $D;               // Stampa "12"
$4F = "ciao";          //Errore
$città = "salerno";    //Corretto à è ASCII esteso
```

Variabili

- La funzione isset() verifica se una variabile è assegnata o meno

```
$A = 1;
if (isset($A))
        print “A è assegnata”
if (!isset($B))
        print “B non è assegnata”;
```

- Molto utile/importante

Variabili *variabili*

- Si può far riferimento al valore di una variabile il cui nome è memorizzato in un'altra variabile
  - Usare il valore di una variabile come il nome di una seconda variabile
- Ad esempio

```
$foo = 'bar';
$$foo = 'ciao';
```

dopo la seconda istruzione, la variabile $bar contiene il valore "ciao"

Variabili riferimento – 1

- Bisogna inserire il simbolo & prima del nome della variabile a cui vogliamo far riferimento
- Ad esempio

  $nero = & $bianco;

  le variabili $nero e $bianco fanno riferimento alla stessa zona di memoria
- Possiamo modificare il valore di $nero attraverso la variabile $bianco

Variabili riferimento – 2

- Attraverso la funzione unset() possiamo rimuovere completamente una variabile dall'ambiente PHP
- L'invocazione unset($bianco) non altera il contenuto della variabile $nero

EsempioVblDiRiferimento

Campo di visibilità – 1

- Una volta che una variabile è introdotta in uno script PHP essa risulta visibile in tutto lo script PHP

```
<HTML>
<HEAD><TITLE>Variabili </TITLE></HEAD>
<BODY>
<?php
    $saluto = "Ciao a tutti";  //variabile globale
?>
<H1>Pagina iniziale</H1>
<?php
    echo ("<b>Saluto: $saluto</b>");
?>
</BODY>
</HTML>
```


ESEMPIO

Errori in PHP

- Il server non comunica gli errori al client
- Possiamo modificare una direttiva in php.ini per mostrare nel browser gli eventuali errori (solo il primo) di script PHP

- Cercare la direttiva display_errors e settare
  - display_errors = On
- SOLO PER DEBUGGING

Campo di visibilità – 2

- In PHP esistono quattro campi di visibilità per le variabili
- Locale
  - Una variabile dichiarata in una funzione è visibile solo nella funzione e nelle definizioni delle funzioni annidate
- Parametri di una funzione
  - Sono locali alla funzione e non accessibili al di fuori di essa

Campo di visibilità – 3

- Globale
  - Le variabili dichiarate al di fuori di una funzione sono globali
  - Non sono visibili all'interno delle funzioni
    - Per renderle visibili le si deve dichiarare all'interno delle funzioni come globali

      global $NomeVariabile
- Statico
  - Dichiarate all'interno di una funzione tramite la parola chiave static
  - Conservano il loro valore tra chiamate successive della funzione
  - Sono visibili solo nella funzione

Tipi di dati supportati da PHP – 1

- Non è necessario dichiarare il tipo delle variabili
- Il tipo di una variabile è assegnato dal contesto
- PHP effettua una conversione di tipo automatica
- Esistono otto tipi di dato: quattro scalari, due composti e due tipi speciali

Tipi di dati supportati da PHP – 2

- Tipi scalari (assumono solo un valore)
  - interi, numeri in virgola mobile, booleani e stringhe
- Tipi composti
  - array ed oggetti
- Tipi speciali (null e risorse)
  - Il tipo null assume solo il valore null che indica che una variabile non ha valore
  - Una risorsa, ad esempio, è il riferimento ottenuto dopo l'apertura di una connessione ad un database (è un numero intero)

Tipi di dati supportati da PHP – 3

- Interi
  - Sono di 32 bit con segno
  - Hanno rappresentazione ottale ed esadecimale
  - Valori tra –2^31+1 e 2^31
  - Si usa is_int($x) per verificare se $x è numero intero
- Virgola mobile
  - Hanno 14 cifre di precisione
  - Valori tra ±1.8E-308 e ±1.8E+308
  - Si usa is_float($x) per verificare se $x è un numero in virgola mobile

Tipi di dati supportati da PHP – 4

- Valori booleani
  - Assumono due valori
    - true (vero) false (falso)
- Altri tipi utilizzati come booleani:
  - Il numero 0 ed il numero 0.0 sono false
  - La stringa vuota oppure uguale a “0” è false
  - L’array con zero elementi è false
  - L’oggetto senza dati o metodi è false
  - Il valore null è false
- Si usa is_bool($x) per verificare se $x è un booleano

Tipi di dati supportati da PHP – 5

- Il tipo stringa
  - Come al solito, arbitraria sequenza di caratteri racchiusi da virgolette semplici oppure doppie
- PHP fornisce molte funzioni per operare con le stringhe
- Per concatenare due stringhe si usa l'operatore punto (.)
  - $nome . " e " . $cognome
- Si usa is_string($x) per verificare se $x è una stringa

Nota sulle stringhe

- Le variabili all'interno di una stringa racchiusa dalle virgolette (“ --- “) vengono espanse (al loro posto verrà mostrato il loro valore)
- Esempio

```
$nome = “Ambrogio”;
Echo “Ciao $nome”; //produce Ciao Ambrogio
echo 'Ciao $nome'; //produce Ciao $nome
```

  - att.ne alla forma degli apici con gli editor

Esempio

```
<?php
   $a = 3 + 2 * 5;       // $a = 13
   $b = (3 + 2) * 5;     // $b = 25
   $c = $a + $b;         // $c = 38
   print ("Il valore è $c");
?>
```

- Output: Il valore è 38

print oppure echo? – 1

- Entrambi sono utilizzati per mandare in output la valutazione dei loro argomenti
  - In genere l'output prodotto è codice HTML che verrà inviato al browser per poter essere interpretato
- echo non è una funzione
  - È un costrutto del linguaggio, ecco perché si possono omettere le parentesi
  - Si possono specificare più elementi separati da virgole
    - echo "ciao ", "a ", "tutti!";

print oppure echo? – 2

- echo è un po' più veloce di print
- print è anch'esso un costrutto del linguaggio
  - print potrebbe fallire, **restituisce** true se la stringa passata come argomento è mostrata, false altrimenti
- Esiste anche printf simile alla analoga funzione in C
  - printf("%.2f",12.34567); stampa 12.34

printf

printf(“Il valore esadecimale di %d è %x”, 214, 214);

Il valore decimale di 214 è d6

printf(“Completo al %.2f%%”, 2.1);

Completo al 2.10%

Sequenze di escape

- Sequenze di caratteri che possiamo inserire in stringhe contenute da virgolette doppie cominciano con il backslash (\) e vengono sostituite con caratteri speciali
- Sequenze di escape più comuni
  - \n = newline (nuova riga)
  - \r = carriage return (ritorno a capo)
  - \t = carattere di tabulazione (TAB)
  - \$ = carattere dollaro ($)
  - \" = carattere doppi apici (")
  - \\ = carattere backslash (\)

Output contenente le virgolette

- Se l'output che lo script deve generare contiene delle virgolette, allora bisogna far precedere le virgolette dal backslash (\) ed includere l'output nelle virgolette doppie
- Ad esempio

```
print ("<font color=\"blue\">");
```

produce in output: <font color="blue">

Funzioni sulle stringhe

- Esistono moltissime funzioni per la manipolazione delle stringhe
  - strlen
  - strpos
  - strcmp
  - ltrim, rtrim, trim
    - Eliminano gli spazi bianchi a sinistra, a destra, sia a sx sia a dx della stringa in input
- Maggiori dettagli sul manuale PHP

Stringhe su più linee

- Per inserire una stringa che occupa più linee si può usare la seguente sintassi

```
$NomeVariabile = <<<  Identificativo
  qui inseriamo
  la stringa che
  deve andare su più
  linee
  Identificativo;
  echo $NomeVariabile;
```

Accesso al singolo carattere

- Si considera la stringa come un array di caratteri (stessa cosa del C)
- Per accedere al carattere i-esimo si fa riferimento alla posizione i-1
  - $str =“casa”;
  - $secondo = $str[1];

Manipolazione stringhe

- strtolower
  - tutta la stringa in minuscolo
- strtoupper
  - tutta la stringa in maiuscolo
- ucfirst
  - prima lettera della prima parola della stringa in maiuscolo
- ucwords
  - prima lettera di ogni parola della stringa in maiuscolo

Costanti in PHP

- È possibile definire delle costanti in PHP che possono contenere solo valori scalari
- Si usa la sintassi

  define('NomeCostante', Valore);

- Ad esempio
  - define('Nome', 'Ambrogio'); echo Nome;
  - define(‘Massimo’, 49);

Non si usa il $

PHP e caratteri speciali HTML

- A volte in una stringa ci possono essere dei caratteri che possono interferire con HTML (e.g., >, <, &, …)
- PHP fornisce la funzione

  htmlspecialchars()

  che converte tutti questi caratteri nelle relative sequenze di escape (entità);

ESEMPIO

PHP ed URL

- Esistono quattro funzioni per codificare e decodificare URL
  - rawurlencode – urlencode
  - rawurldecode – urldecode
- La versione raw serve per codificare path che contengono spazi con %20, l'altra versione serve per codificare lo spazio con +
  - Mantenuta per motivi storici (legacy)