Tecnologie Web - MFN0634
PHP

Docente

Marco **Botta**[marco.botta@unito.it]

- Sviluppo server side
- Sintassi PHP di base
- Includere codice PHP nelle pagine
- Sintassi PHP avanzata

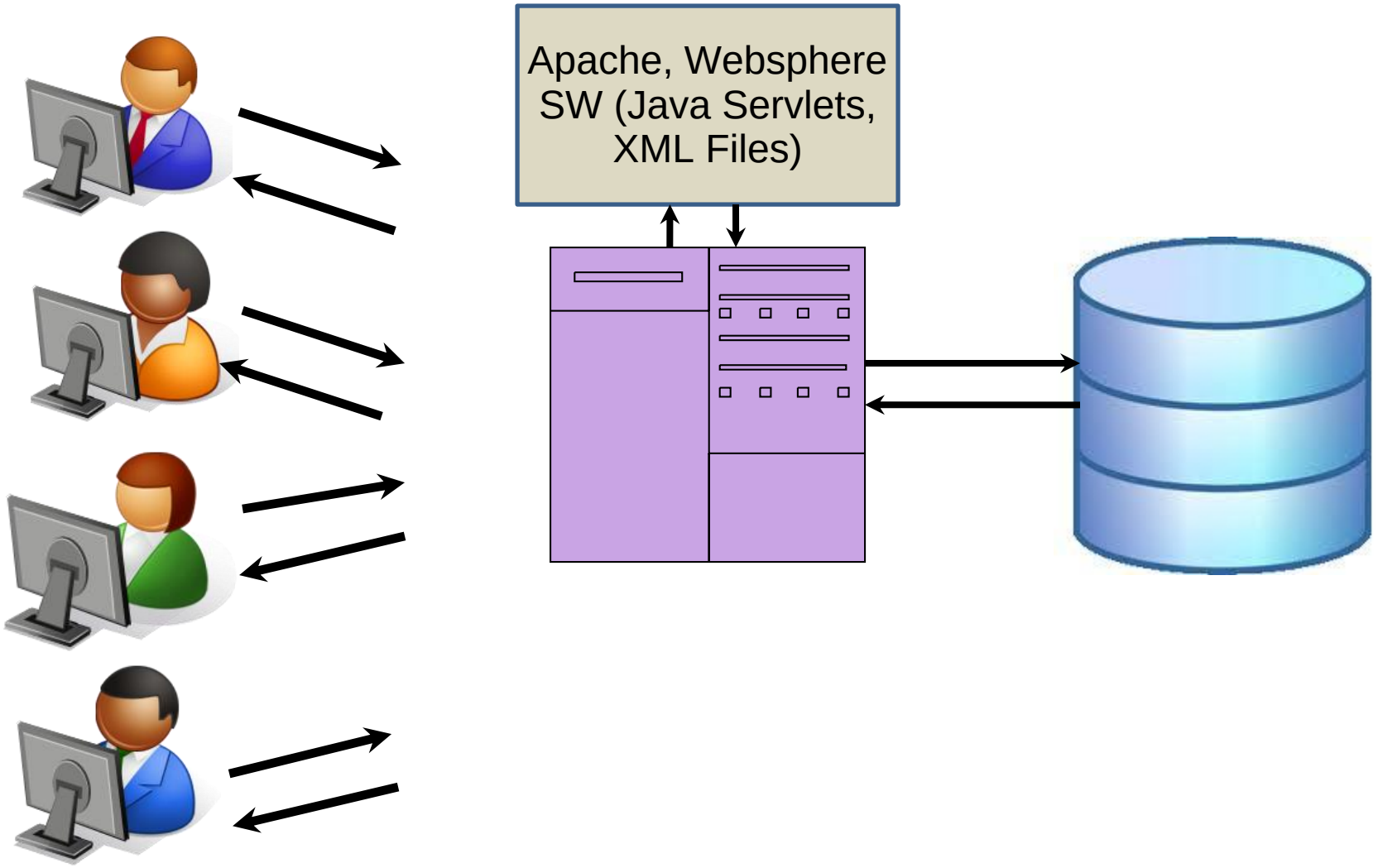
- **Sviluppo server side**
- Sintassi PHP di base
- Includere codice PHP nelle pagine
- Sintassi PHP avanzata

URL e server web

http://server/path/file

- Normalmente quando inserisci una URL nel tuo browser:
  - Il S.O. del tuo computer cerca l'indirizzo IP del server usando il DNS
  - Il tuo browser si connette a quell'indirizzo IP e richiede quel dato file
  - Il processo software del server web remoto (e.g. Apache) individua quel file dal suo file system locale
  - Il server invia ti spedisce il contenuto di quel file (come *stream* html)

URL e server web (cont.)



URL e server web (cont.)

http://**www.facebook.com/home.php**

- Qualche URL in realtà specifica dei programmi che il server dovrebbe *eseguire*. Ciò che viene restituito in questi casi è l'output di detti programmi:
  - La URL dell'esempio chiede al server **facebook.com** di eseguire il programma **home.php** e di mandare indietro il suo output

Programmazione Web Server-Side

- I programmi Server-side si scrivono usando uno dei tanti possibili linguaggi/ambienti di programmazione per il web
  - esempi: PHP, Java/JSP, Ruby on Rails, ASP.NET, Python, Perl, Node.js

Programmazione Web Server-Side (cont.)

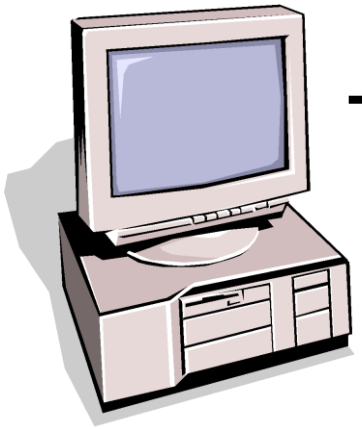
- Chiamato anche *scripting server side*, è in grado di:
  - Modificare dinamicamente, cambiare o aggiungere qualsiasi contenuto ad una pagina Web
  - Rispondere alle richieste dell'utente o ai dati inviati dai moduli (form) HTML
  - Accedere ai dati (spesso gestiti tramite db backend) e restituire il risultato al browser
  - Personalizzare una pagina Web per renderela più fruibile per gli utenti individuali
  - Fornire maggiore protezione dato che il tuo codice lato server non può essere visto da un browser

Programmazione Web Server-Side (cont.)

- Un server Web:
  - Contiene il software che ti consente di eseguire i programmi server side
  - Invia al mittente l'output generato in forma di risposta alle richieste HTTP
- Ogni linguaggio/ambiente ha i sui pro ed i suoi contro
  - Noi useremo PHP (non necessariamente la scelta migliore in ogni situazione)

Cosa è PHP?

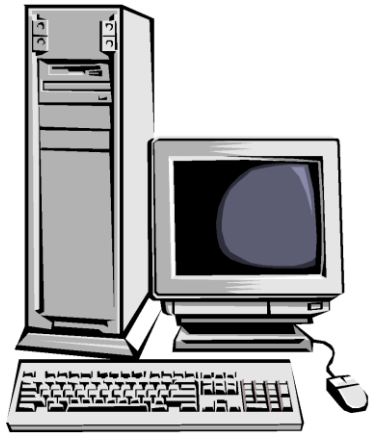
- PHP: "PHP Hypertext Preprocessor"
- Linguaggio di scripting Server-side
- Usato per rendere dinamiche le pagine web:
  - Fornisce contenuti diversi dipendenti dal contesto
  - Si interfaccia con altri servizi: database, e-mail, etc.
  - Autentica gli utenti
  - Processa l'informazione proveniente dai moduli presenti sulla parte client-side dell'applicazione
- Il codice PHP può essere inserito direttamente all'interno del codice (X)HTML



GET /b/f/file.php

200 OK

(file.php)

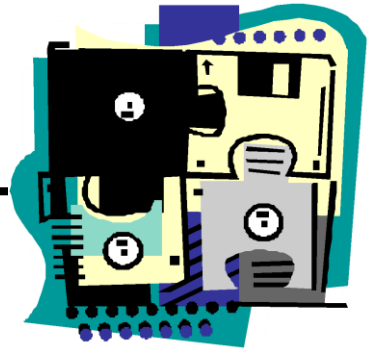


CLIENT

SERVER

file.php
(sorgente)

```
<html>
<head>
<title>Esempio 1</title>
</head>
<body>
<h1>Esempio con PHP</h1>
<p>
<?php
echo "questo &egrave; un
 semplice script";
?></p>
</body>
</html>
```



MODULO PHP

```
<html>
<head>
<title>Esempio 1</title>
</head>
<body>
<h1>Esempio con PHP</h1>
<p>
questo &egrave; un
 semplice script
</p>
</body>
</html>
```

file.php
(dopo l'esecuzione)

Perché PHP?

- Gratuito e open source
- Compatibile
  - A gennaio 2013 quasi 100.000.000 di siti attivi usavano PHP.
- Semplice

Hello World!

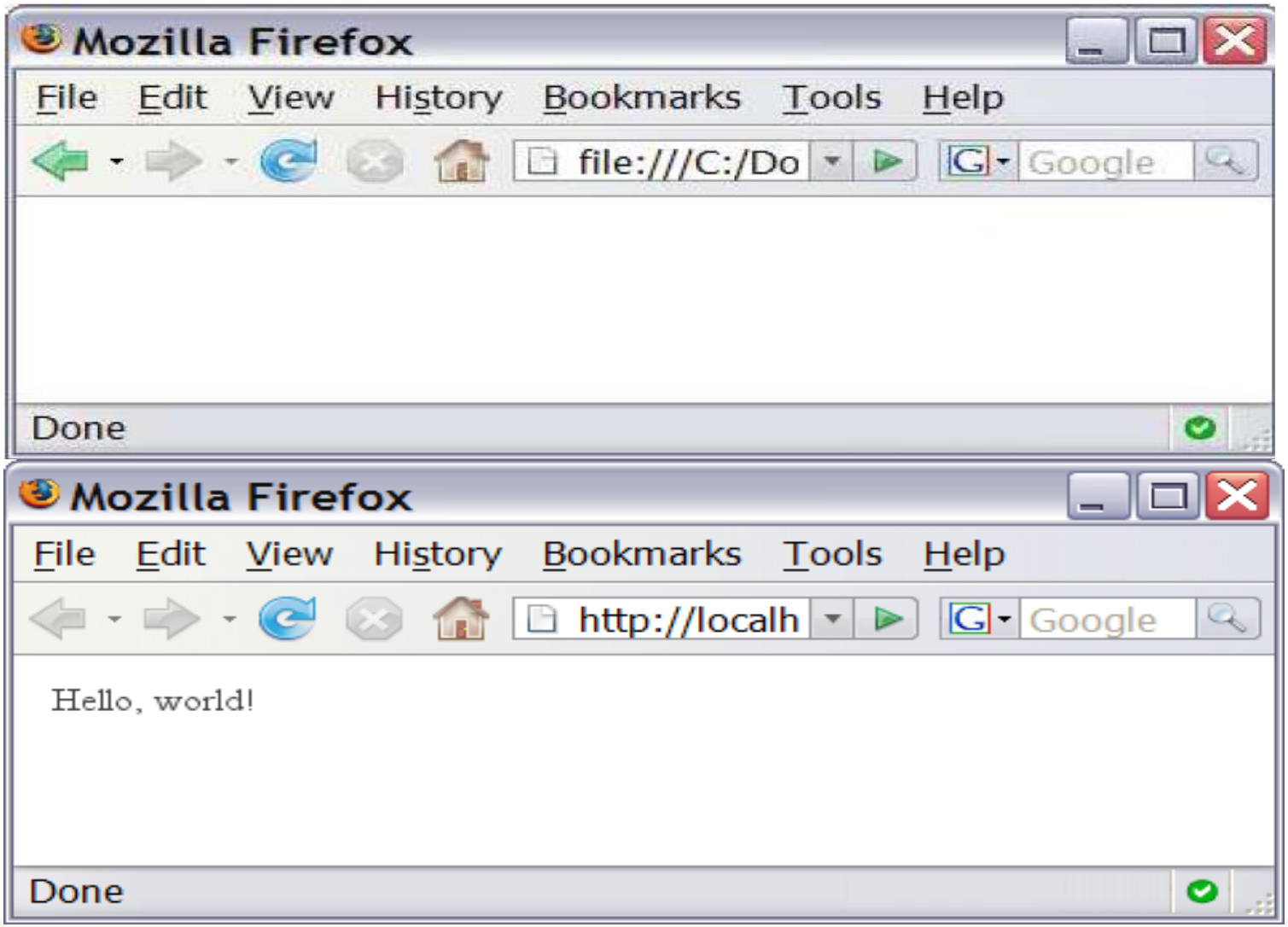
```php
<?php
print "Hello, world!";
?>
```

PHP

```
Hello world!
```

output

Visualizzare l'output PHP



PHP info

- Un altro esempio è il comando che ci dà informazioni sulla configurazione del server. Se dentro al file *.php* si inserisce la seguente linea:

<? phpinfo(); ?>

si ottengono tutte le informazioni sulla configurazione del web server e del php su cui si sta lavorando.

HTTP Server Agent

Lo script:

```
<html>
<body>
   Il server agent che stai usando &egrave;:
      <?php echo $_SERVER['HTTP_USER_AGENT']; ?>
</body>
</html>
```

HTTP Server Agent

.... produrrà (su una macchina Windows) il seguente output:

<html>
<body>
Il server agent che stai usando è:
Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)
</body>
</html>

- Sviluppo server side
- **Sintassi PHP di base**
- Includere codice PHP nelle pagine
- Sintassi PHP avanzata
- Caso di studio

Template della sintassi PHP

```
HTML content
<?php
PHP code
?>
HTML content
<?php
PHP code
?>
HTML content ...
```

PHP

- Il codice php all'interno dei tag `<?php` and `?>` è eseguito dal server
- Tutto il resto del contenuto è interpretato direttamente come HTML puro
- Possiamo passare continuamente dalla modalità PHP a quella HTML e viceversa

Console output: `print`

```
print "text";
```

PHP

```
print "Hello, World!\n";
print "Escape \"chars\" are the SAME as in Java!\n";
print "You can have
line breaks in a string.";
print 'A string can use "single-quotes". It\'s cool!';
```

PHP

```
Hello, world!  Escape "chars" are the SAME as in Java! You can have line
breaks in a string. A string can use "single-quotes". It's cool!
```

output

- Alcuni preferiscono il comando equivalente `echo` al posto di `print`

Console output: `print`

```php
print "text";
```

PHP

```php
print "Hello, World!<br>";
print "Escape \"chars\" are the SAME as in Java!<br>";
print "You can have
line breaks in a string.";
print 'A string can use "single-quotes". It\'s cool!';
```

PHP

```
Hello, world!
Escape "chars" are the SAME as in Java!
You can have line breaks in a string. A string can use "single-quotes". It's cool!
```

- Alcuni preferiscono il comando equivalente `echo` al posto di `print`

Variabili

```
$name = expression;
```
PHP

```
$user_name = "mundruid78";
$age = 16;
$drinking_age = $age + 5;
$this_class_rocks = TRUE;
```
PHP

- I nomi sono *case-sensitive*
- I nomi iniziano sempre con un $, sia in fase di dichiarazione che nell'accesso
- Sono sempre dichiarati nel momento dell'assegnazione (senza dichiarre il tipo)
- E' un linguaggio debolmente tipato (come JavaScript o Python)

Variabili

- Tipi di base: *int, float, boolean, string, array, object, NULL*
  - Verifica il tipo di una variabile con le funzioni `is_type`, e.g. `is_string`
  - La funzione `gettype` restituisce il tipo di una variabile in termini di una stringa
- PHP converte da un tipo all'altro *automaticamente in molti casi*:
  - string → int conversione automatica con +
  - int → float conversione automatica con /
- Cast di tipo esplicito con **(type):**
  - `$age = (int) "21";`

Operatori aritmetici

- + - * / % . ++ --
- = += -= *= /= %= .=
- Molti operatori convertono il tipo implicitamente: 5 + "7" è 12

Commenti

```php
# single-line comment
// single-line comment
/*
multi-line comment
*/
```

PHP

- Come in Java, ma anche con #
  - Molto codice PHP adotta commenti con # invece che con //

Tipo `String`

```php
$favorite_food = "Ethiopian";
print $favorite_food[2];
$favorite_food = $favorite_food . " cuisine";
print $favorite_food;
```

PHP

- Indicizzazione a base zero e notazione a parentesi []
- Non esiste un tipo `char`; ogni lettera è comunque una stringa
- L'operatore di concatenazione tra stringhe è . (punto), non +
  - `5 + "2 turtle doves" === 7`
  - `5 . "2 turtle doves" === "52 turtle doves"`
- Si può specificare tra "" oppure tra ''

Funzioni `String`

```php
# index  0123456789012345
$name = "Stefanie Hatcher";
$length = strlen($name);
$cmp = strcmp($name, "Brian Le");
$index = strpos($name, "e");
$first = substr($name, 9, 5);
$name = strtoupper($name);
```

PHP

Funzioni `String` (cont.)

Nome	**Equivalente in Java**
strlen	length
strpos	indexOf
substr	substring
strtolower, strtoupper	toLowerCase, toUpperCase
trim	trim
explode, implode	split, join
strcmp	compareTo

Interpretazione di stringhe

```php
$age = 16;
print "You are " . $age . " years old.\n";
print "You are $age years old.\n"; # You are 16 years old.
```
PHP

- Stringhe dentro " " vengono *interpretate*
  - Variabili che sono richiamate all'interno saranno sostituite dal loro valore
- Stringhe dentro ' ' non vengono interpretate:

```php
print 'You are $age years old.\n'; # You are $age years
old.
```
PHP

Interpretazione di stringhe (cont.)

```php
print "Today is your $ageth birthday.\n"; # $ageth not found
print "Today is your {$age}th birthday.\n";
```

PHP

- Se è necessario evitare ambiguità, usare le parentesi {} per delimitare il nome della variabile

Interpretazione di stringhe (cont.)

```php
$name = "Xenia";
$name = NULL;
if (isset($name)) {
print "This line isn't going to be reached.\n";
}
```

PHP

- Una variabile è NULL se
  - Non le è stato assegnato nessun valore (variabili non definite)
  - Le viene assegnato il valore NULL
  - Il suo valore è stato rimosso usando la funzione unset
- Puoi verificare che una variabile sia NULL usando la funzione `isset`
- NULL in fase di stampa è equivalente ad una stringa vuota (no output)

Ciclo `for` (uguale a Java)

```php
for (initialization; condition; update) {
      Statements;
}
```

PHP

```php
for ($i = 0; $i < 10; $i++) {
      print "$i squared is " . $i * $i . ".\n";
}
```

PHP

Tipo (booleano) `bool`

```php
$feels_like_summer = FALSE;
$php_is_great = TRUE;
$student_count = 7;
$nonzero = (bool) $student_count; # TRUE
```

PHP

- I valori seguenti sono considerati FALSE (tutti gli altri sono TRUE):
  - 0 e 0.0 (but NON 0.00 o 0.000)
  - "", "0", and NULL (include variabili non definite)
  - Array con 0 elementi
- FALSE in fase di stampa è equivalente ad una stringa vuota (no output); TRUE viene stampata come 1

Condizioni `if/else`

```
if (condition) {
      statements;
} elseif (condition) {
      statements;
} else {
      statements;
}
```

PHP

Cicli `while` (come in Java)

```php
while (condition) {
      statements;
}
```
PHP

```php
do {
      statements;
} while (condition);
```
PHP

Operazioni matematiche

```php
$a = 3;
$b = 4;
$c = sqrt(pow($a, 2) + pow($b, 2));
```

PHP

Funzioni matematiche

abs	ceil	cos	floor	log	log10	max
min	pow	rand	round	sin	sqrt	tan

Costanti matematiche

M_PI	M_E	M_LN2

Tipi `Int` e `Float`

```php
$a = 7 / 2; # float: 3.5
$b = (int) $a; # int: 3
$c = round($a); # float: 4.0
$d = "123"; # string: "123"
$e = (int) $d; # int: 123
```

PHP

- int per gli interi e float per i reali
- Divisione tra due valori int può restituire un float

- Sviluppo server side
- Sintassi PHP di base
- **Includere codice PHP nelle pagine**
- Sintassi PHP avanzata

Scrivere tag HTML PHP = stile errato!

```
<?php
print "<!DOCTYPE html>\n";
print "<html>\n";
print " <head>\n";
print " <title>Geneva's web page</title>\n";
...
for ($i = 1; $i <= 10; $i++) {
print  "<p> I can count to $i! </p>\n";
}
?>
```

HTML

- Lo stile PHP migliore è quello che minimizza il numero di istruzioni print/echo
- Se non usiamo print, come inseriamo il contenuto dinamico dentro la pagina?

Convivenza corretta HTML PHP

```
<!DOCTYPE html>
<html>
 <head>
   <title>Geneva's web page</title>
…
<?php
for ($i = 1; $i <= 10; $i++) {
?>
      <p> I can count to <?php print $i ?>! </p>
<?php
}
?>
```

HTML

Risultato codice html

```
<!DOCTYPE html>
<html>
 <head>
   <title>Geneva's web page</title>
…

      <p> I can count to 1! </p>
      <p> I can count to 2! </p>
      …
      <p> I can count to 10! </p>
```

HTML

Espressioni blocco PHP

```
<?= expression ?>
```
PHP

```
<h2> The answer is <?= 6 * 7 ?> </h2>
```
PHP

The answer is 42

output

- Espressione blocco PHP: un piccolo pezzo di codice php che valuta ed include automaticamente il risultato di un'espressione all'interno del codice HTML
  - `<?= expression ?>` è equivalente a: `<?php print expression ?>`

Esempio

```
<!DOCTYPE html>
<html>
<head><title>CSE 190 M: Embedded PHP</title></head>
<body>
<?php
for ($i = 99; $i >= 1; $i--) {
?>
<p> <?= $i ?> bottles of beer on the wall, <br>
<?= $i ?> bottles of beer. <br>
Take one down, pass it around, <br>
<?= $i - 1 ?> bottles of beer on the wall. </p>
<?php
}
?>
</body>
</html>
```

PHP

Errori comuni: parentesi non chiuse, simbolo = mancante

```
...
<body>
<p>Watch how high I can count:
<?php
for ($i = 1; $i <= 10; $i++) {
?>
      <? $i ?>
</p>
</body>
</html>
```

PHP

- Se dimentichi di chiudere le parentesi, genererai un errore: 'unexpected `$end`'
- Se dimentichi = in `<?=`, l'espressione non produrrà alcun output

Espressioni blocco complesse

```
...
<body>
<?php
for ($i = 1; $i <= 3; $i++) {
?>
      <h<?= $i ?>>This is a level <?= $i ?>
heading.</h<?= $i ?>>
<?php
}
?>
</body>
```

PHP

Espressioni blocco complesse

```
...
<body>

      <h1>This is a level 1 heading.</h1>
      <h2>This is a level 2 heading.</h2>
      <h3>This is a level 3 heading.</h3>
</body>
```

HTML

This is a level 1 heading.

This is a level 2 heading.

This is a level 3 heading.

output

- Sviluppo server side
- Sintassi PHP di base
- Includere codice PHP nelle pagine
- **Sintassi PHP avanzata**

Funzioni

```php
function name(parameterName, ..., parameterName) {
   statements;
}
```

PHP

```php
function quadratic($a, $b, $c) {
   return -$b + sqrt($b * $b - 4 * $a * $c) / (2 * $a);
}
```

PHP

- Non si dichiarano i tipi dei parametri e del valore di ritorno
- Una funziona senza l'istruzione di `return` restituirà implicitamente NULL

Chiamata di funzioni

```
name(expression, ..., expression);
```

```
$w = 163;   # pounds
$h = 70;    # inches
$my_bmi = bmi($w, $h);
```

- Se passate il numero di parametri sbagliati, generate un errore

Valori di default dei parametri

```php
function print_separated($str, $separator = ", ") {
      if (strlen($str) > 0) {
            print $str[0];
            for ($i = 1; $i < strlen($str); $i++) {
                  print $separator . $str[$i];
            }
      }
}
```

PHP

```php
print_separated("hello"); # h, e, l, l, o
print_separated("hello", "-"); # h-e-l-l-o
```

PHP

- Se non passi un valore, quello di default sarà utilizzato

Scope delle variabili: locali e globali

- Variabili dichiarate dentro una funzione sono **locali**; le altre sono **globali**.
- Se una funzione vuole usare una variabile globale, deve usare l'istruzione `global`
  - Non abusarne; usa piuttosto i parametri passati alla funzione stessa.

```
$school = "UW";                    # global
...

function downgrade() {
  global $school;
  $suffix = "(Wisconsin)";         # local

  $school = "$school $suffix";
  print "$school\n";
}
```

Array

```
$name = array();                          # create
$name = array(value0, value1, ..., valueN);
$name[index]                              # get element value
$name[index] = value;                     # set element value
$name[] = value;                          # append
```

PHP

```
$a = array();       # empty array (length 0)
$a[0] = 23;         # stores 23 at index 0 (length 1)
$a2 = array("some", "strings", "in", "an", "array");
$a2[] = "Ooh!";     # add string to end (at index 5)
```

PHP

- Append: usa la notazione a parentesi [] senza un indice all'interno
- Il tipo dell'elemento non è specificato: un array può contenere elementi di tipi diversi

Funzioni per Array

function name(s)	description
count	number of elements in the array
print_r	print array's contents
array_pop, array_push, array_shift, array_unshift	using array as a stack/queue
in_array, array_search, array_reverse, sort, rsort, shuffle	searching and reordering
array_fill, array_merge, array_intersect, array_diff, array_slice, range	creating, filling, filtering
array_sum, array_product, array_unique, array_filter, array_reduce	processing elements

Esempio

```php
$tas = array("MD", "BH", "KK", "HM", "JP");
for ($i = 0; $i < count($tas); $i++) {
      $tas[$i] = strtolower($tas[$i]);
}
$morgan = array_shift($tas);
array_pop($tas);
array_push($tas, "ms");
array_reverse($tas);
sort($tas);
$best = array_slice($tas, 1, 2);
```

PHP

- Un array in PHP sostituisce molti altri tipi di collezioni in Java
  - list, stack, queue, set, map, ...

Ciclo `foreach`

```php
foreach ($array as $variableName) {
      ...
}
```

PHP

```php
$fellowship = array("Frodo", "Sam", "Gandalf",
"Strider", "Gimli", "Legolas", "Boromir");
print "The fellowship of the ring members are: \n";
for ($i = 0; $i < count($fellowship); $i++) {
      print "{$fellowship[$i]}\n";
}
```

PHP

Ciclo `foreach`

```php
foreach ($array as $variableName) {
      ...
}
```

PHP

```php
$fellowship = array("Frodo", "Sam", "Gandalf",
"Strider", "Gimli", "Legolas", "Boromir");
print "The fellowship of the ring members are: \n";
for ($i = 0; $i < count($fellowship); $i++) {
      print "{$fellowship[$i]}\n";
}
print "The fellowship of the ring members are: \n";

foreach ($fellowship as $fellow) {
      print "$fellow\n";
}
```

PHP

Array multidimensionali

```php
<?php $AmazonProducts = array( array("BOOK", "Books", 50),
                               array("DVDs", "Movies", 15),
                               array("CDs", "Music", 20)
                             );
for ($row = 0; $row < 3; $row++) {
       for ($column = 0; $column < 3; $column++) { ?>
              <p> | <?= $AmazonProducts[$row][$column] ?>
       <?php } ?>
      </p>
<?php } ?>
```

PHP

Array multidimensionali (cont.)

```php
<?php
$AmazonProducts = array(
array("Code" =>"BOOK", "Description" => "Books", "Price" => 50),
array("Code" => "DVDs", "Description" => "Movies", "Price" => 15),
array("Code" => "CDs", "Description" => "Music", "Price" => 20)
);
for ($row = 0; $row < 3; $row++) { ?>
      <p> | <?= $AmazonProducts[$row]["Code"] ?> |
            <?= $AmazonProducts[$row]["Description"] ?> |
            <?= $AmazonProducts[$row]["Price"] ?>
     </p>
<?php } ?>
```

PHP

Funzioni di confronto per `String`

Name	**Function**
strcmp	compareTo
strstr, strchr	find string/char within a string
strpos	find numerical position of string
str_replace, substr_replace	replace string

Esempi su confronto di tipi `String`

```php
$offensive = array( offensive word1, offensive word2);
$feedback = str_replace($offensive, "%!@*", $feedback);
```

PHP

```php
$test = "Hello World! \n";
print strpos($test, "o");
print strpos($test, "o", 5);
```

PHP

```php
$toaddress = "feedback@example.com";
if(strstr($feedback, "shop")
      $toaddress = "shop@example.com";
else if(strstr($feedback, "delivery")
      $toaddress = "fulfillment@example.com";
```

PHP

- Sviluppo server side
- Sintassi PHP di base
- Includere codice PHP nelle pagine
- **Sintassi PHP avanzata (cont.): gestione file**

Inclusione file in PHP

- Inserisci il contenuto di un un file PHP in un altro file PHP, prima che il server lo esegua, con queste funzioni:
  - `include(f)` copia a runtime il contenuto del file f all'interno del file che chiama la funzione. Se il file non è trovato, genera un warning, ma l'esecuzione non sarà interrotta
  - `require(f)` come include, ma se il file non è trovato, genera un errore, che blocca l'esecuzione
- Preferite però le funzioni seguenti:
  - `include_once(f)` come include, ma evita copia ed incolla a cascata se lo stesso file è incluso più volte
  - `require_once(f)` come require, ma evita copia ed incolla a cascata se lo stesso file è incluso più volte

Esempio `include()`

```php
<a href="/default.php">Home</a>
<a href="/tutorials.php">Tutorials</a>
<a href="/references.php">References</a>
<a href="/examples.php">Examples</a>
<a href="/contact.php">Contact Us</a>
```

PHP

```php
<html>
<body>

<div class="leftmenu">
<?php include("menu.php"); ?>
</div>

<h1>Welcome to my home page.</h1>
<p>I have a great menu here.</p>

</body>
</html>
```

PHP

Funzioni I/O con file in PHP

funzioni	categoria
file, file_get_contents, file_put_contents	leggere/scrivere file interi
basename, file_exists, filesize, fileperms, filemtime, is_dir, is_readable, is_writable, disk_free_space	Recupera informazioni sul file
copy, rename, unlink, chmod, chgrp, chown, mkdir, rmdir	Manipolazione file e directory
glob, scandir	Leggere directory

Leggere da/scrivere su file

contents of foo.txt	file("foo.txt")	file_get_contents("foo.txt")
Hello how are you? I'm fine	array("Hello\n", #0 "how are\n", #1 "you?\n", #2 "\n", #3 "I'm fine\n" #4)	"Hello\n how are\n you?\n \n I'm fine\n"

- `file` restituisce le righe di un file dentro un array
- `file_get_contents` restituisce l'intero contenuto di un file in una stringa

Leggere/scrivere un file intero

```php
# reverse a file
$text = file_get_contents("poem.txt");
$text = strrev($text);
file_put_contents("poem.txt", $text);
```

PHP

- `file_get_contents` restituisce l'intero contenuto di un file in una stringa
- `file_put_contents` sostituisce l'intero contenuto precedente di un file con quello contenuto in una stringa

Opzione append

```php
# add a line to a file
$new_text = "P.S. ILY, GTG TTYL!~";
file_put_contents("poem.txt", $new_text,
FILE_APPEND);
```

PHP

Contenuto precedente

Roses are red,
Violets are blue.
All my base,
Are belong to you.

Nuovo contenuto

Roses are red,
Violets are blue.
All my base,
Are belong to you.
P.S. ILY, GTG TTYL!~

La funzione `file`

```php
# display lines of file as a bulleted list
$lines = file("todolist.txt");
<?php
foreach ($lines as $line) {
      ?>
      <li> <?= $line ?> </li>
<?php
}
?>
```

PHP

- Restituisce le righe di un file sotto forma di un array di stringhe
  - Ultimo carattere di ogni stringa: \n
  - Per togliere il \n, usa questo parametro opzionale:

```php
$lines = file("todolist.txt",FILE_IGNORE_NEW_LINES);
```

PHP

Estrarre i valori dall'array: `list`

```php
list($var1, ..., $varN) = array;
```

```php
$values = array("mundruid", "18", "f", "96");
...
list($username, $age, $gender, $iq) = $values;
```

- La funzione list accetta una lista di nomi di variabili separate da virgola come parametri
- Usa questa funzione per "spacchettare" il contenuto dell'array in diverse variabili

File di lunghezza fissa, `file` e `list`

```
Xenia Mountrouidou
(919)685-2181
570-86-7326
```

contents of file personal.txt

```
list($name, $phone, $ssn) = file("personal.txt");
```

PHP

- Legge il file in reads con un array e ne memorizza subito il contenuto di ogni linea in una variabile separata
- Devi conoscere ovviamente la lunghezza esatta del file in anticipo (oltre che il formato)

Dividere/unire stringhe

```php
$array = explode(delimiter, string);
$string = implode(delimiter, array);
```
PHP

```php
$class = "CS 380 01";
$class1 = explode(" ", $class); # ("CS", "380", "01")
$class2 = implode("...", $class1); # "CSE...380...01"
```
PHP

- `explode` e `implode` convertono stringhe in array

Esempio `explode`

```
Harry Potter, J.K. Rowling
The Lord of the Rings, J.R.R. Tolkien
Dune, Frank Herbert
```

contents of input file books.txt

```php
<?php foreach (file("books.txt") as $book) {
      list($title, $author) = explode(", ", $book);
      ?>
      <p> Book title: <?= $title ?>, Author: <?=
$author ?> </p>
<?php
}
?>
```

PHP

Leggere il contenuto delle directory

funzione	**descrizione**
scandir	Restituisce un array contenente tutti i nomi dei file della cartella (restituisce solo i nomi, come "`myfile.txt`")
glob	Restituisce un array contenente tutti i nomi dei file corrispondenti ad un pattern (restituisce i cammini, come "`foo/bar/myfile.txt`")

Esempio di `scandir`

```
<ul>
<?php
$folder = "taxes/old";
foreach (scandir($folder) as $filename) {
      ?>
      <li> <?= $filename ?> </li>
<?php
}
?>
</ul>
```

PHP

- .
- ..
- 2009_w2.pdf
- 2007_1099.doc

output

Esempio di `glob`

```
# reverse all poems in the poetry directory
$poems = glob("poetry/poem*.dat");
foreach ($poems as $poemfile) {
      $text = file_get_contents($poemfile);
      file_put_contents($poemfile, strrev($text));
      print "I just reversed " .
basename($poemfile);
}
```

PHP

- `glob` può usare le "wildcard" con il carattere *
- La funzione `basename` recupera solo il nome dal cammino di un file