

Introduzione a PHP



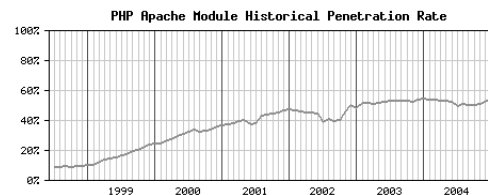
Moreno Marzolla
Dipartimento di Informatica
Università Ca' Foscari di Venezia
marzolla@dsi.unive.it
<http://www.dsi.unive.it/~marzolla>

Testo di riferimento

- Bulger Greenspan, *Sviluppare applicazioni per database con MySQL/PHP*, Apogeo, 2001. ISBN 88-7303-866-2
- <http://www.php.net/>
- <http://conf.php.net/>

PHP Hypertext Processor Un po' di storia

- Inizio sviluppo nel 1994
- Ver 1.0: Personal Home Page Tools 1995
- Ver 2.0: PHP/FI 1995-1997
- Ver 3.0: PHP 1997-2000
- Ver 4.0: PHP mid-2000
- Ver 4.1: 10 Dec 2001
- Ver 4.2: 22 Apr 2002



Fonte: SecuritySpace.com

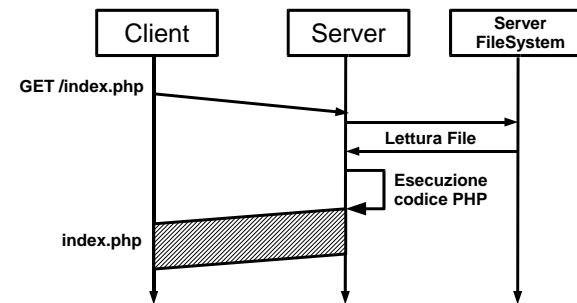
Portabilità

- Piattaforme
 - UNIX (all variants)
 - Win32 (NT/W95/W98/W2000)
 - QNX
 - MacOS (WebTen)
 - OSX
 - OS/2
 - BeOS
- Web Server
 - Apache module (UNIX, Win32)
 - CGI/FastCGI
 - thttpd
 - fhttpd
 - phttpd
 - ISAPI (IIS, Zeus)
 - NSAPI (Netscape iPlanet)
 - Java servlet
 - AOLServer
 - Roxen/Caudium module

L'idea di base / 1

- L'utente scrive una pagina (es index.php) che contiene frammenti di codice PHP
- Il server WEB riceve la richiesta per index.php
 - Legge la pagina ed esegue tutte le istruzioni PHP incluse
 - Spedisce la pagina processata al client
- Gli utenti finali ricevono codice HTML puro
 - Non vedranno mai le istruzioni php incluse nelle pagine, perché il server spedisce loro i risultati, non il codice

L'idea di base / 2



L'idea di base / 3

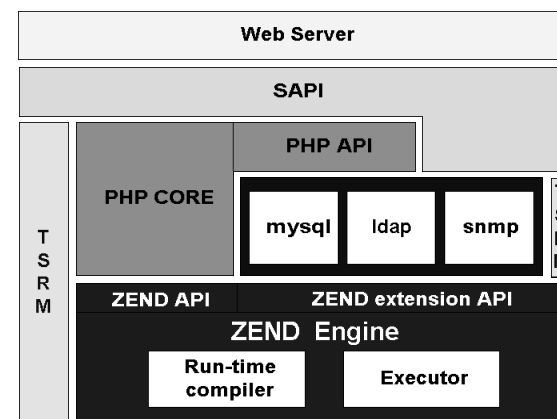
- Esempio: index.php

```
<html>
<head><title>Prova</title></head>
<body>
<?php echo "Hello, world"; ?>
</body>
```

- Ciò che l'utente riceve

```
<html>
<head><title>Prova</title></head>
<body>
Hello, world
</body>
```

Architettura di PHP



Un esempio un po' più utile

- Il seguente codice (PHP >= 4.2.0):

```
Your browser is:
<?php echo $_SERVER['HTTP_USER_AGENT']
/* echo $HTTP_USER_AGENT se register_globals=off */ ?>
```

produce:

```
Your browser is: Mozilla/5.0 (X11; U; Linux i686; en-
US; rv:1.7.3) Gecko/20041007 Galeon/1.3.18 (Debian
package 1.3.18-1.1)
```

Embedding

- SGML style: `<? code ?>`
- XML style: `<?php code ?>`
- ASP style: `<% code %>`
- Javascript style:
`<script language="php">`
`code`
`</script>`

```
<html><head><title>
Search results for
"<?php print $query; ?>"
</title></head>
<body>
...
</body>
```

Switching mode

```
<? if(strstr($_SERVER['HTTP_USER_AGENT'], "MSIE")) { ?>
<b>You are using Internet Explorer</b>
<? } else { ?>
<b>You are not using Internet Explorer</b>
<? } ?>
```

Sintassi

```
<?php
for ($loop = -5; $loop < 5; $loop++) {
    if ($loop < 0) {
        echo "-";
    } elseif ($loop > 0) {
        echo "+";
    }
    echo "$loop<BR>\n";
}
while ( --$loop ) {
    switch($loop % 2) {
        case 0:
            echo "Even<BR>\n";
            break;
        case 1:
            echo "Odd<BR>\n";
            break;
    }
}
do {
    echo "$loop<BR>";
} while (++$loop < 10);
?>
```

```
--5
--4
--3
--2
--1
0
+1
+2
+3
+4
Even
Odd
Even
Odd
0
1
2
3
4
5
6
7
8
9
```

Tipi di dato

- Tipi di dato fondamentali
 - numeri (integers e real)
 - Decimal 1234, Octal 0777, Hex 0xff
 - stringhe
 - Double-quoted "abc", single-quoted 'abc'
 - boolean
 - true,false
- Dynamic typing
 - Non occorre dichiarare il tipo delle variabili
 - Eventuali conversioni di tipo automatiche
- `<? echo 5 + "1.5" + "10e2"; ?>`
 - Output: 1006.5

Array

- Ordinati (C-style)

```
<?php
$a[0] = 1;
$a[1] = "foo";
?>
```

- Associativi

```
<?php
$catch_it['cat'] = "mouse";
$catch_it['dog'] = "cat";
?>
```

Manipolare gli array

```
<?php
$arr = array(1, 'foo', 1.57, 'cat' => 'mouse', 'dog' => 'mailman');
?>
```

```
<?php
PHP 3: while(list($k,$v)=each($arr)){
PHP 4: foreach( $arr as $k=>$v ) {
    echo "\$arr[$k] = $v<br>\n";
}
?>
```

Output:

```
$arr[0] = 1
$arr[1] = foo
$arr[2] = 1.57
$arr[cat] = mouse
$arr[dog] = mailman
```

Funzioni

```
<?php
function header($title="Default Title")
{
    echo "<HTML><HEAD><TITLE>";
    echo $title;
    echo "</TITLE></HEAD><BODY>";
}
?>
```

Gestione dei form

```
<form action="action.php" method="POST">
Your name: <input type="text" name="name"><br/>
You age: <input type="text" name="age"><br/>
<input type="submit"/>
</form>
```

action.php

```
Hi <?php echo $HTTP_POST_VARS["name"] ?>.
You are <?php echo $HTTP_POST_VARS["age"] ?>
years old.
```

Manipolazione di stringhe

```
<?php
$str = "Fast String Manipulation";
echo substr($str,0,4) . substr($str,-9);
?>
```

Output:

Fastipation

```
<?php
$a = explode(":", "This:string:has:delimiters.");
while (list($value) = each($a)) {
    if (strcmp($value, "has") == 0) {
        echo "had ";
    } else echo $value." ";
}
?>
```

Output

This string had delimiters.

Encoding di stringhe

```
<?php
echo htmlspecialchars("This & that are <problems>.");
?>
```

Output

This & that are <problems>.

```
<?php
echo ($encoded = base64_encode(
    "A rose by any other name..."));
echo ($decoded = base64_decode($encoded));
?>
```

Output

QSBYb3NlIGJ5IGFueSBvdGhlciBuYW11Li4u
A rose by any other name...

Semplice guestbook

```
<html><head><title>My Guestbook</title></head>
<body>
<h1>Welcome to my Guestbook</h1>
<h2>Please write me a little note below</h2>
<form action="<?php echo $PHP_SELF?>" method="POST">
<textarea cols="40" rows="5" name="note"
wrap="virtual"></textarea>
<input type="submit" value=" Send it ">
</form>
<?php if(isset($note)) {
    $fp = fopen("/tmp/notes.txt","a");
    fputs($fp,nl2br($note).'\n');
    fclose($fp);
}
?><h2>The entries so far:</h2>
<?php @ReadFile("/tmp/notes.txt") ?>
</body></html>
```

Gestione delle sessioni

- PHP gestisce nativamente le *sessioni*
 - Una *sessione* è un meccanismo per preservare informazioni tra una pagina ed un'altra visitata dall'utente
 - Le sessioni possono anche essere gestite tramite cookies, PHP mette a disposizione un meccanismo diretto
- La funzione `start_session()` registra una nuova sessione
 - Di solito tramite cookie, ma anche tramite querystring
- Una volta creata una sessione è possibile registrare variabili PHP il cui valore viene preservato da una pagina all'altra

Esempio

```
<?php
session_start();
session_register("my_var");
$my_var = "hello world";
?>
```

- Registra la variabile `$my_var`
- In ogni pagina in cui si esegue `session_start()` e `session_register("my_var")` ci si ritrova lo stesso valore di questa variabile

Esempio 2 (PHP < 4.2)

```
<?php
session_start();
session_register("your_name");
//check to see if $your_name contains anything
if(!empty($your_name))
{
    echo "I already know your name, $your_name";
}
//this portion will probaby run the first time to
//this page.
elseif(empty($your_name) && !isset($submit))
{
    echo "<form name=\"myform\" method=\"post\" action=\"\$_PHP_SELF\">
    <input type=\"text\" name=\"first_name\"> first name<br>
    <input type=\"text\" name=\"last_name\"> last name<br>
    <input type=\"submit\" name=\"submit\" value=\"submit\">
    </form>";
    //if the form has been submitted, this portion will
    //run and make an assignment to $your_name.
} elseif (isset($submit) && empty($your_name))
{
    $your_name = $first_name . " " . $last_name;
    echo "Thank you, $your_name";
}
```

Implementazione

- PHP usa i cookies per memorizzare informazioni sulle sessioni
- Che si fa se l'utente disabilita i cookies?
 - Il nome della sessione si può memorizzare nella querystring
 - `<?=SID?>` visualizza qualcosa come `PHPSESSID=07d696c4fd787cd6c78b734fb4855520`

```
<a href="mypage.php?<?=SID?>">click to page</a>
```

Gestire i cookies

- Sintassi

```
setcookie( name, value, time_to_expire, path,  
          domain, security_setting );
```

- NB: Deve essere usata *prima* di stampare qualunque cosa. Idealmente, va messa come prima istruzione di una pagina PHP

Esempio

```
setcookie("mycookie",  
         "my_id",time()+(60*60*24*30),"/",".mydomain.com", 0)
```

- Il nome del cookie è "mycookie"
 - Potrà essere letto come
`$HTTP_COOKIE_VARS["mycookie"]`
- Il suo valore è "my_id"
- Scade fra 30 giorni
- Il cookie è valido su tutte le pagine del dominio indicato ("/")
- Il cookie vale per il dominio ".mydomain.com"
- Nessun livello di sicurezza particolare