PHP

Introduzione alla programmazione Web

INTRODUZIONE

PHP è un linguaggio
- ❑ di scripting
- ❑ interpretato
- ❑ open source
- ❑ lato server.

PHP

PHP è un acronimo ricorsivo per:
PHP: Hypertext Preprocessor

Noi faremo riferimento alla versione 5.4

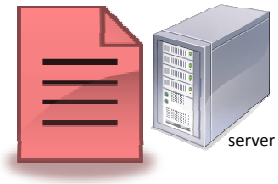
http://www.php.net

ALESSANDRO URSOMANDO

PROGRAMMAZIONE LATO CLIENT

Il browser chiede una pagina.

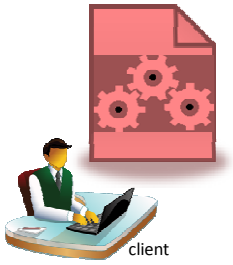
PROGRAMMAZIONE LATO CLIENT



Il browser chiede una pagina.

Il server spedisce quella pagina.

PROGRAMMAZIONE LATO CLIENT

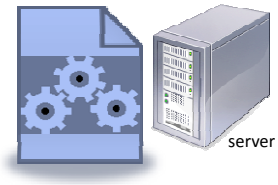


- Il browser chiede una pagina.
- Il server spedisce quella pagina.
- Il **browser esegue** quella pagina.

PROGRAMMAZIONE LATO SERVER

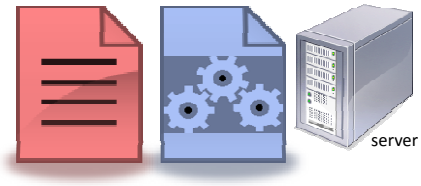
- Il browser chiede una pagina.

PROGRAMMAZIONE LATO SERVER



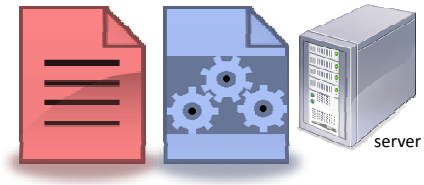
- Il browser chiede una pagina.
- Il **server esegue** quella pagina..

PROGRAMMAZIONE LATO SERVER



- Il browser chiede una pagina.
- Il **server esegue** quella pagina..
- ..producendo un'altra pagina..

PROGRAMMAZIONE LATO SERVER



- Il browser chiede una pagina.
- Il **server esegue** quella pagina..
- ..producendo un'altra pagina..
- ..che spedisce al client.

TIPICO SCENARIO DI PROGRAMMAZIONE WEB

- Il browser chiede una pagina php.
  La rete indirizza la richiesta al server.

TIPICO SCENARIO DI PROGRAMMAZIONE WEB

pagina web

webserver

dati

server

Il browser chiede una pagina php. La rete indirizza la richiesta al server.

Sul server ci sono le pagine, un **webserver** e un **gestore di database**.

Il webserver interpreta i comandi **php** (con i quali si interroga il **db**) e produce una pagina html che viene spedita al client.

client

LA NOSTRA PIATTAFORMA

XAMPP

BROWSER

NOTEPAD++

LA NOSTRA PIATTAFORMA

XAMPP è un pacchetto portable (versione ZIP da **copiare** su pendrive) che simula un server remoto.

XAMPP

Il nostro server sarà localhost

Metteremo i file .php in HTDOCS.

`http://www.apachefriends.org/it/xampp.html`

PHP

Le basi del linguaggio

Variabili, costanti, espressioni, output e random

TAG DI APERTURA E CHIUSURA SCRIPT

```
<!doctype html>
<html>
    <head>
        <title>Un documento PHP</title>
        <meta charset="UTF-8"/>
    </head>
</head>
<body>
    <?php
        // questo qui sopra è uno dei possibili tag di apertura

        // questo qui sotto è uno dei possibili tag di chiusura
    ?>
</body>
</html>
```


I COMMENTI IN PHP

```
<!doctype html>
<html>
    <head>
        <title>Un documento PHP</title>
        <meta charset="UTF-8"/>
    </head>
</head>
<body>
    <?php
        // questo qui sopra è uno dei possibili tag di apertura

        // questo qui sotto è uno dei possibili tag di chiusura
    ?>
</body>
</html>
```

```
<?php
    /*
        Ci sono diversi modi per introdurre dei commenti.
        Quello precedente (//) va fino a fine riga.
        Questo apre e chiude un commento multiriga.
    */
?>
```

LE VARIABILI

- ❑ Cominciano con il $
- ❑ Sono case-sensitive
- ❑ Non si dichiarano
  - ❑ Il tipo viene definito nel momento in cui prendono un valore
  - ❑ I tipi primitivi sono booleano, intero, virgola mobile, stringa, array e oggetto
  - ❑ Ci sono funzioni per manipolare il tipo di una variabile
  - ❑ Ci sono i cast

VARIABILI

```php
<?php
  $boolean = true;              // o false

  $intero1 = 10000;             // numero decimale
  $intero2 = 0123;              // numero ottale
  $intero3 = 0x1A;              // numero esadecimale

  $float1 = 1.234;
  $float2 = 1.2e3;              // 1.2 * (10^3)
  $float3 = 1.2E3;              // 1.2 * (10^3)
  $float4 = 1.2e-3;             // 1.2 * (10^-3)
?>
```


LE STRINGHE

```php
<?php
  // delimitiamo le stringhe con apici singoli o doppi
  $nomeLui = "Luca";
  $nomeLei = 'Anna';

  // gli apici doppi interpretano le variabili
  echo "Mi chiamo $nomeLui";    // Mi chiamo Luca
  echo 'Mi chiamo $nomeLei';    // Mi chiamo $nomeLei

  // per stampare gli apici usati per delimitare la
  //  stringa si usa il backslash
  echo "Un'altra volta";    // Un'altra volta
  echo 'Un\'altra volta';   // Un'altra volta
  echo "Le ho detto \"ciao\""; // Le ho detto "ciao"
  echo 'Le ho detto "ciao"';    // Le ho detto "ciao"
?>
```

I CAST

```php
<?php
  // cast
  $num1 = 123.45;
  $num2 = (int)$num1;                  // 123
  $bool = (boolean)$num1;              // true
  $str  = "$num1";                     // "123.45"
?>
```


QUALCHE OPERATORE

```php
<?php
  // concatenazione di stringhe
  $str1 = "Ciao";
  $str2 = "Mondo";
  $str3 = $str1." ".$str2;             // Ciao Mondo

  // operatori algebrici
  $a = 3+7;   // addizione
  $a = 3-7;   // sottrazione
  $a = 3*7;   // prodotto
  $a = 3/7;   // divisione
  $a = 3%7;   // modulo
?>
```

QUALCHE OPERATORE

```php
<?php
  // operatori di confronto

  //  ==  // uguale
  //  !=  // diverso
  //  === // uguale valore e tipo
  //  > // maggiore
  //  >=  // maggiore o uguale
  //  < // minore
  //  <=  // minore o uguale
?>
```


QUALCHE OPERATORE

```php
<?php
  // operatori logici

  //  or  // oppure
  //  ||  // oppure
  //  and // e
  //  &&  // e
  //  ! // non
  //  xor // or esclusivo
?>
```

QUALCHE OPERATORE

```php
<?php
  // operatori di assegnamento

  $a = 10;              // in a c'è 10
  $a += 7;              // adesso in a c'è 17
  $a -= 7;              // adesso in a c'è 10
  $a *= 7;              // adesso in a c'è 70
  $a /= 7;              // adesso in a c'è 10
  $a %= 7;              // adesso in a c'è 3
  $a /= 7;              // adesso in a c'è 0,428..

  $a = "Alessandro";    // adesso in a c'è la stringa Alessandro
  $a .= "Ursomando";    // adesso c'è “AlessandroUrsomando”

?>
```


FUNZIONI SULLE VARIABILI

isset($x)	restituisce true se è già stato assegnato un valore
unset($x)	dealloca la variabile
empty($x)	restituisce true se la var è vuota
Settype($x, <tipo>)	imposta il tipo della variabile
gettype($x)	restituisce il tipo della variabile
is_numeric($x)	restituisce true se la variabile è di tipo numerico
is_int($x)	restituisce true se la variabile è di tipo intero
is_string($x)	restituisce true se la variabile è di tipo stringa
is_array($x)	restituisce true se la variabile è di tipo array
is_object($x)	restituisce true se la variabile è di tipo oggetto
is_null($x)	restituisce true se la variabile vale null
print_r($x)	stampa il valore della variabile in un formato leggibile dall’essere umano

COSTANTI

```php
<?php
  // per convenzione il nome di una costante è sempre in maiuscolo

  define ("PIGRECO", 3.14);
  define ("NOME", "Alessandro");
?>
```

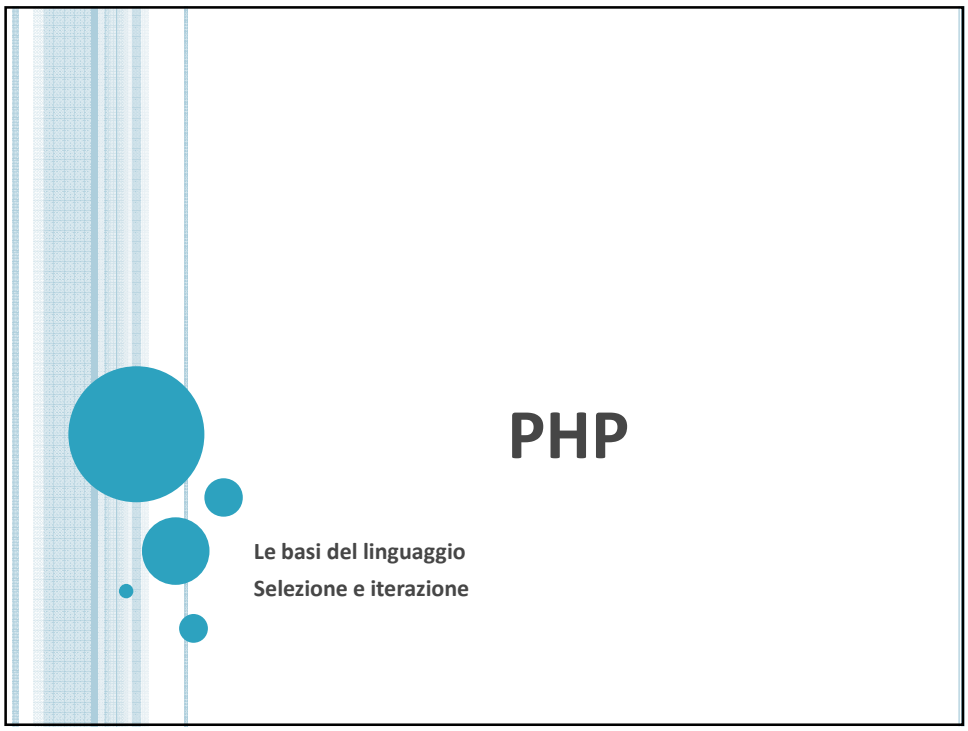

RANDOM

```php
<?php
  srand();            // inizializza il generatore
                      // di numeri pseudocasuali
  $x = rand(1,1000);  // prende un numero tra 1 e 1000
?>
```

OUTPUT

```php
<?php
    srand();
    $sentinella = rand(1,100);
    echo "<p>\n";
    echo "\t\t\til numero sentinella è $sentinella.\n";
    echo "\t</p>\n";
?>
```


ESERCIZI



SELEZIONE A UNA VIA: IF

```php
<?php
  if ($nome == "Alessandro")
    echo "Ciao Alessandro!";
?>
```

```php
<?php
  if ($nome == "Alessandro") {
    echo "Ciao Alessandro! <br/>";
    echo "Che ci fai di nuovo al PC?";
  }
?>
```


ALESSANDRO URSOMANDO

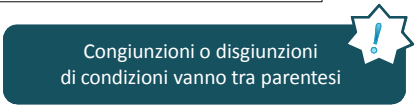
SELEZIONE A DUE VIE : IF..ELSE

```php
<?php
  if ($sesso == "m")
    echo "Ciao bello";
  else
    echo "Ciao bella";
?>
```

```php
<?php
  if ($sesso == "m") {
    echo "Ciao bello <br>";
    echo "Ti vedo in forma!";
  } else {
    echo "Ciao bella";
    echo "Questo vestito ti sta benissimo";
  }
?>
```


SELEZIONE AD N VIE: IF..ELSEIF..ELSE

```php
<?php
  srand();
  $x = rand(1,1000);

  if (($x>=0) and ($x<=10))
    echo "Da zero a dieci";
  elseif (($x>=10) and ($x<=20))
    echo "Da dieci a venti";
  elseif (($x>=20) and ($x<=30))
    echo "Da venti a trenta";
  else
    echo "non lo so";
?>
```



SELEZIONE AD N VIE: SWITCH

```php
<?php
  switch ($nome) {
    case 'Luca':
    case 'Giorgio':
    case 'Enzo':
      echo "Ciao vecchio!";
      break;
    case 'Paolo':
      echo "Ciao Paoletto!";
      break;
    default:
      echo "Ciao straniero";
  }
?>
```


IF COMPATTO

```php
<?php
  $a = 10;
  $b = 20;
  $maggiore = ($a>$b)?$a:$b;

  // equivale a scrivere
  if ($a>$b)
    $maggiore = $a;
  else
    $maggiore = $b;
?>
```

ITERAZIONE SU CONTATORE: FOR

```php
<?php
  $somma = 0;
  for ($i=1; $i<=10; $i++) {
    $somma += $i;
    echo "<p>Sto sommando $i</p>";
  }
  echo "<p>La somma dei primi dieci interi è $somma.</p>";
?>
```

Anche qui ci sono **break** e **continue**.

ITERAZIONE SU CONDIZIONE: WHILE

```php
<?php
  srand();
  while (rand(0,1)==0) {
    echo "<p>è uscito testa, ritiro!</p>";
  }
  echo "<p>è uscito croce!</p>"
?>
```

Anche qui ci sono **break** e **continue**.

ITERAZIONE SU CONDIZIONE: DO..WHILE

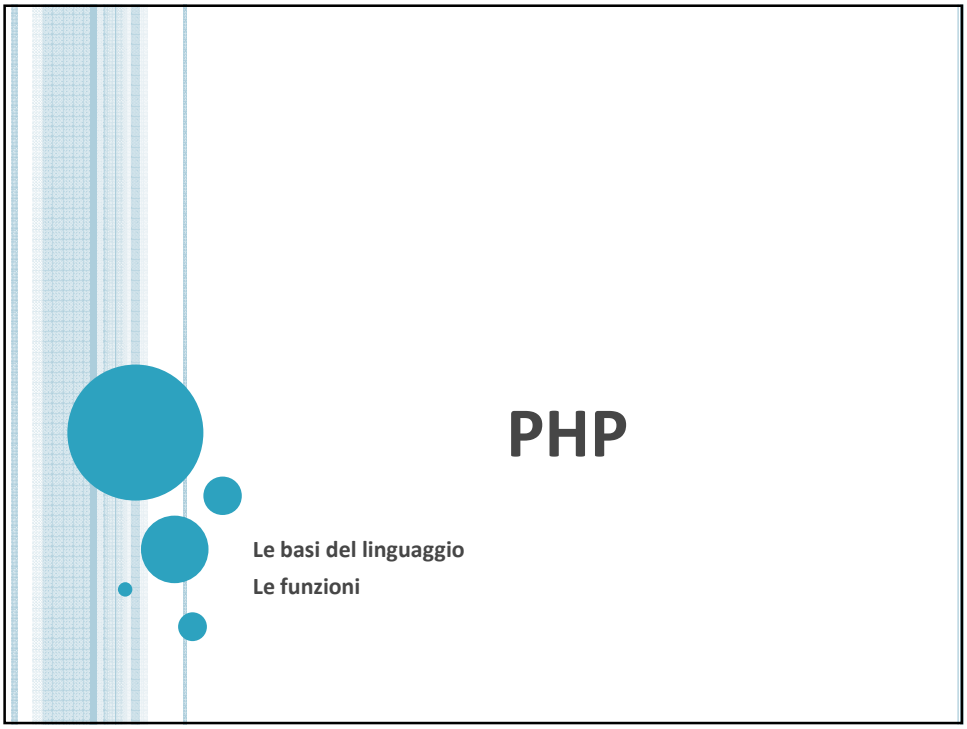
```php
<?php
  srand();
  $obiettivo = rand(1,6);
  do {
    $esito = rand(1,6);
    echo "<p>è uscito $esito!</p>";
  } while ($esito != $obiettivo);
  echo "<p>proprio quello che volevo!!!</p>";
?>
```

Anche qui ci sono **break** e **continue**.

ITERAZIONE SUGLI ELEMENTI DEL VETTORE: FOREACH

Il costrutto **foreach** è un costrutto avanzato che vedremo perciò nella sezione relativa.

ESERCIZI



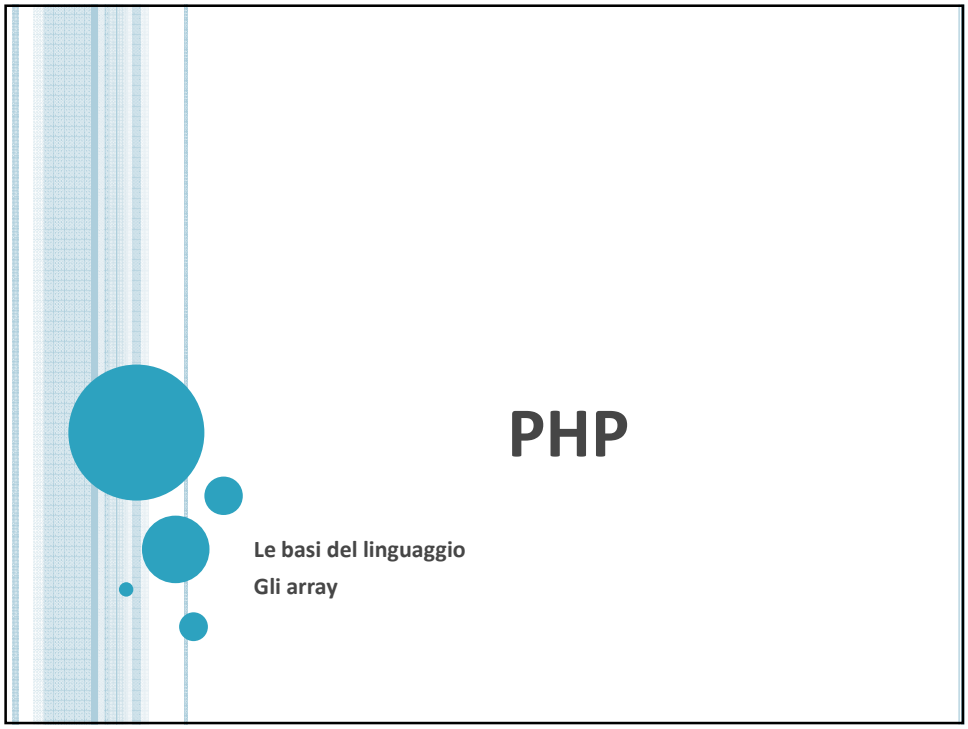
FUNZIONI DELL'UTENTE

```php
<?php
  // definizione della funzione
  function Maggiore($a, $b) {
    if (($a<=0) or ($b<=0))
      return false;
    if ($a > $b)
      return $a;
    else
      return $b;
  }

  // uso della funzione
  $x = 12;
  $y = 34;
  $z = Maggiore($x, $y);
  echo $z;
?>
```


FUNZIONI DELL'UTENTE: PASSAGGIO PER RIFERIMENTO

```php
<?php
  // definizione della funzione
  function Scambia(&$a, &$b) {
    $temp = $a;
    $a = $b;
    $b = $temp;
  }

  // uso della funzione
  $x = 12;
  $y = 34;
  Scambia($x, $y);
?>
```

ESERCIZI



ARRAY

```php
<?php
  // creazione esplicita
  $primi = array(2,3,5,7,11);

  // creazione implicita
  $pari[0] = 2;
  $pari[1] = 4;
  $pari[2] = 6;
?>
```

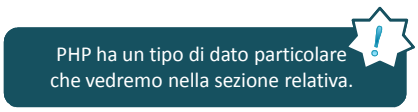

ARRAY

```php
<?php
  // un array può inoltre contenere valori non omogenei
  $mix = array(
    1,
    "ciao",
    3.14,
    array(1,2,3,4)
  );
?>
```

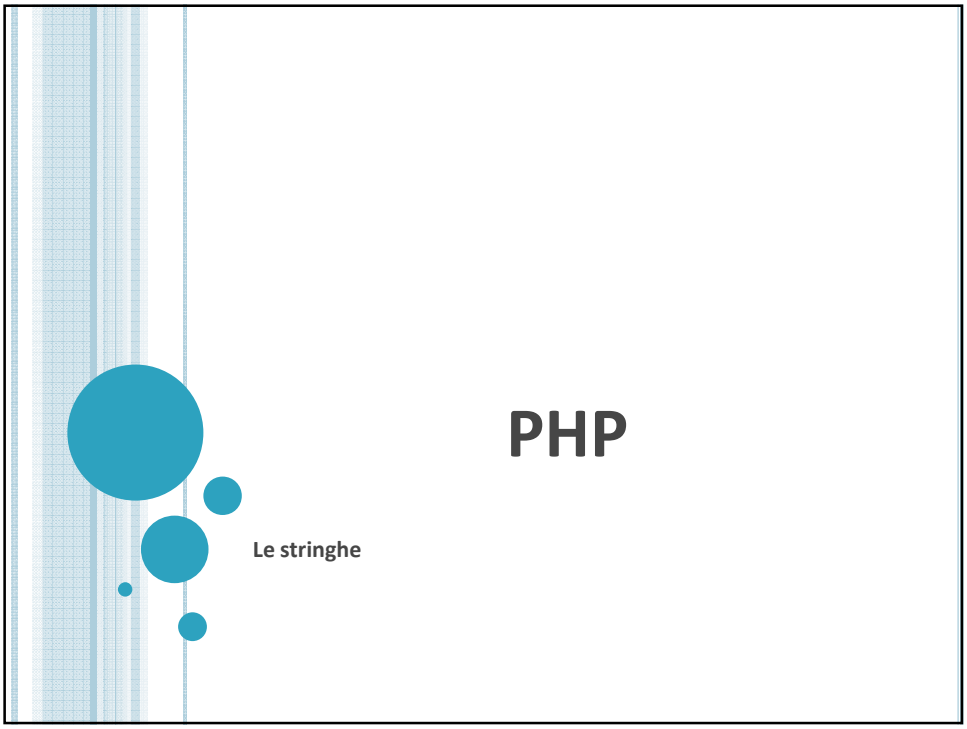
FUNZIONI SUGLI ARRAY

count($a)	restituisce quanti elementi contiene l'array
array_push($a, $valore)	inserisce il nuovo valore in coda all'array
array_pop($a)	toglie e restituisce l'ultimo valore dell'array
in_array($valore, $a)	restituisce true se trova il valore nell'array
sort($a)	ordina il vettore

ARRAY ASSOCIATIVI



ESERCIZI



FUNZIONI SULLE STRINGHE (1)

strlen($s)	restituisce la lunghezza della stringa
trim($s)	restituisce la stringa passata alla quale sono stati tolti gli spazi all'inizio e alla fine
ltrim($s)	solo gli spazi iniziali
rtrim($s)	solo gli spazi finali
substr($s,$i,[$q])	restituisce una sottostringa di **$s** a partire dal carattere **$i** e lunga **$q**
str_replace ($s1,$s2,$s3)	restituisce **$s3** con **$s2** al posto di **$s1**
strpos($s1,$s2)	restituisce la posizione in cui **$s2** comincia in **$s1** (**false** se **$s2** non è in **$s1**)

FUNZIONI SULLE STRINGHE (2)

strtolower($s)	restituisce **$s** tutta al minuscolo
strtoupper($s)	restituisce **$s** tutta al maiuscolo
ucfirst($s)	restituisce **$s** con la prima lettera maiuscola
ucwords($s)	…tutte le parole

FUNZIONI SULLE STRINGHE (3)

explode($s1,$s2)	restituisce un array che contiene tante stringhe create a partire da **$s2** che viene separata da **$s1**
implode($s,$a)	restituisce una stringa costruita con tutti gli elementi del vettore delimitati dalla stringa indicata
strcmp($s1, $s2)	restituisce 0 se le due stringhe sono uguali, un numero negativo se **$s1** è lessicograficamente inferiore a **$s2**, un numero positivo altrimenti
strcasecmp($s1,$s2)	come **strcmp** ma case insensitive

FUNZIONI SULLE STRINGHE (4)

addsleshes($s)	restituisce la stringa passata alla quale è stato aggiunto un backslash prima di un apice singolo(‘), un apice doppio (“) o un altro backslash (\)
stripcslashes($s)	il contrario di addslashes
strrev($s)	restituisce la stringa rovesciata
htmlspecialchars($s)	restituisce la stringa passata dopo avere convertito i caratteri speciali per html, dopo il primo parametro si possono indicare i seguenti: **ENT_HTML5, “UTF-8”, true**.

ESEMPI

```php
$s = "Questa è una stringa di prova.";
// vogliamo isolare la quarta parola di questa stringa.
// la quarta parola comincia al 14esimo carattere
// ed è lunga 7 caratteri
$t = substr($s, 14, 7);

// $t è una stringa che vale: “stringa”
```


ESEMPI

```php
$vettore = array (1,2,4,8,16,32,64,128);
$separatore = " -> ";
$potenze2 = implode($vettore, $separatore);

// $potenze2 è una stringa che vale:
// 1 -> 2 -> 4 -> 8 -> 16 -> 32 -> 64 -> 128
```

```php
$cognomi = "Rossi, Russo, Ferrari, Esposito, Bianchi";
$vettore = explode(",", $cognomi);

// $vettore è un array così formato:
// cella numero 0: Rossi
// cella numero 1: Russo
// cella numero 2: Ferrari
// cella numero 3: Esposito
// cella numero 4: Bianchi
```

ESEMPI

```php
// Quando PHP incontra il lingiaggio MYSQL possono essere utili
// una serie di funzioni come assslashes

$stringa = "Un testo con l'apice al suo interno";
$stringa = addslashes($stringa);
$query = "insert into nometabella(nomeCampo) values ('$stringa')";
```


ESERCIZI

PHP

I file

QUALI FILE

Impariamo ad interagire con i file di testo in PHP.

APERTURA DI FILE

FUNZIONE	COSA RESTITUISCE
fopen($n,$d)	un puntatore al file $n aperto in modalità $d

Modalità di apertura del file	
r	in lettura
r+	in lettura e scrittura
w	in creazione (o sovrascriuttura)
w+	in creazione (o sovrascrittura) e lettura
a	in append
a+	in append e lettura

OPERAZIONI SU FILE (1)

FUNZIONE	COSA FA
fclose($f)	chiude il file
fwrite($f,$s)	scrive la stringa $s
fread($f,$n)	legge $n caratteri
fgets($f)	legge una riga
fgetc($f)	legge un carattere

Per scrivere un carattere di fine riga su un file usare la stringa:"/r/n"

OPERAZIONI SU FILE (2)

FUNZIONE	COSA FA
file_exists($f)	restituisce true se il file esiste
feof($f)	restituisce true se il file è finito
filesize($f)	restituisce la grandezza del file
ftell($f)	restituisce la posizione del cursore
fseek($f, $n)	si posiziona al carattere numero $n

fseek di default ha come riferimento l'inizio del file; si può specificare un terzo parametro:
- **SEEK_END**: riferimento fine file
- **SEEK_CUR**: riferimento la posizione corrente

ALESSANDRO URSOMANDO

OPERAZIONI SPECIALI CON I FILE

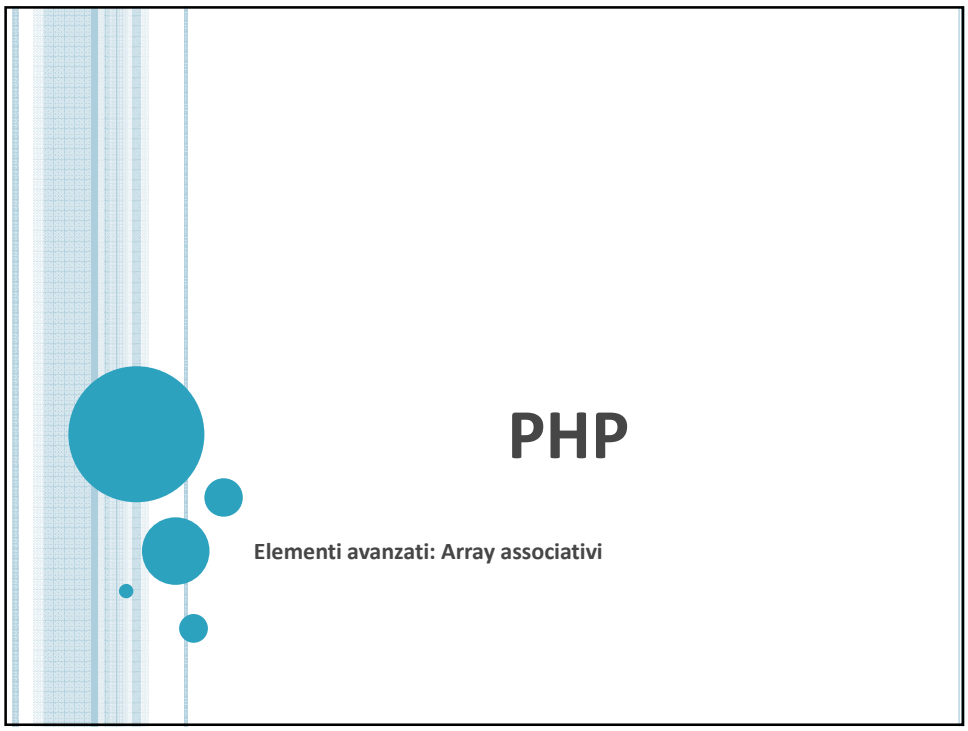
FUNZIONE	COSA FA
file($f)	restituisce un vettore con il contenuto del file (ogni cella contiene una riga del file)
file_get_contents($f)	restituisce una stringa con il contenuto del file
include($f)	sostituisce nel file corrente questa riga di codice con tutto il contenuto del file indicato
require($f)	come include

ALESSANDRO URSOMANDO

OPERAZIONI SU CARTELLE

FUNZIONE	COSA FA
opendir($path)	restituisce un puntatore alla cartella
closedir($d)	chiude la gestione della cartella
readdir($d)	restituisce il nome di un file della cartella i file vengono restituiti uno alla volta, finiti i file viene restituito false (si usi l'operatore identico e non uguale)
rewinddir($d)	punta al primo file della cartella
rmdir($path)	cancella la cartella
mkdir($path)	crea la cartella

ESERCIZI



CREAZIONE

```php
<?php
    // creazione implicita
    $persona["nome"] = "Mario";
    $persona["cognome"] = "Rossi";
    $persona["data_nascita"] = "1980/10/15";
    $persona["residenza"] = "Torino";
?>
```

```php
<?php
    // creazione esplicita
    $persona= array(
        "nome"=>"Mario",
        "cognome"=>"Rossi",
        "data_nascita"=>"1980/10/15",
        "residenza"=>"Torino"
    );
?>
```


ALESSANDRO URSOMANDO

SCORRIMENTO DI VALORI

```php
<?php
    // dichiarato un vettore associativo in un modo qualsiasi
    $persona["nome"] = "Mario";
    $persona["cognome"] = "Rossi";
    $persona["data_nascita"] = "1980/10/15";
    $persona["residenza"] = "Torino";

    // scorro tutti i valori
    foreach ($persona as $valore)
        echo "<p>$valore</p>";
?>
```


SCORRIMENTO DI CHIAVI E VALORI

```php
<?php
    // dichiarato un vettore associativo in un modo qualsiasi
    $persona["nome"] = "Mario";
    $persona["cognome"] = "Rossi";
    $persona["data_nascita"] = "1980/10/15";
    $persona["residenza"] = "Torino";

    // scorro tutti i valori
    foreach ($persona as $chiave => $valore)
        echo "<p>$chiave: $valore</p>";
?>
```

ARRAY ASSOCIATIVI DI ARRAY ASSOCIATIVI

```php
<?php
    $persone=array(
        array("nome"=>"Mario",      "cognome"=>"Rossi",      "residenza"=>"Torino"),
        array("nome"=>"Filippo",    "cognome"=>"Bianchi",    "residenza"=>"Milano"),
        array("nome"=>"Giorgio",    "cognome"=>"Nero",       "residenza"=>"Napoli"),
        "totale" => 3
    );


    echo $persone[0]["cognome"];        // stampa Rossi
    echo $persone[1]["nome"];           // stampa Filippo
    echo $persone["totale"];            // stampa 3
?>
```


ALESSANDRO URSOMANDO

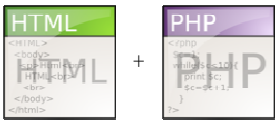
ESERCIZI

ALESSANDRO URSOMANDO

PHP

Interazione con l'utente

INTERAZIONE CON L'UTENTE



Il linguaggio HTML permette all'utente
di inserire dei valori nella pagina WEB,
ma non è in grado di processarli: impariamo a farlo con PHP.

ALESSANDRO URSOMANDO

INTERAZIONE CON L'UTENTE

I **moduli** HTML (**form**) danno all'utente la possibilità di introdurre dei dati.

Ogni oggetto del form ha un **nome** un **valore**

Nome:
Cognome:
Registra:
invia

Il modulo invia le **coppie** (nome, valore) ad un **programma** per mezzo di una delle **due** strategie possibili.

Vediamo come usare queste coppie (nome, valore) in un **programma**.

Vediamo le caratteristiche dei **due metodi di spedizione**.

ESEMPIO

```
<!-- omissis -->
<form action="elabora.php" method="post">
    <p>
        <label for="nome">Nome:</label>
        <input id="nome"
               type="text"
               name="nome"/>
    </p>
    <p>
        <label for="cognome">Cognome:</label>
        <input id="cognome"
               type="text"
               name="cognome"/>
    </p>
    <p>
        <label for="nuovo">Registra: </label>
        <input id="nuovo"
               type="checkbox"
               name="nuovo"
               value="si"/>
    </p>
    <input type="submit"
           name="submit"
           value="invia" />
</form>
<!-- omissis -->
```

Nome:
Cognome:
Registra:
invia

L'alternativa al metodo **post** è il **get**.

`method="get"`

Vedremo tra poco le differenze.

ESERCIZIO 14-00-00-01

ESEMPIO

```
<!-- omissis -->
<form action="elabora.php" method="post">
    <p>
        <label for="nome">Nome:</label>
        <input id="nome"
               type="text"
               name="nome"/>
    </p>
    <p>
        <label for="cognome">Cognome:</label>
        <input id="cognome"
               type="text"
               name="cognome"/>
    </p>
    <p>
        <label for="nuovo">Registra: </label>
        <input id="nuovo"
               type="checkbox"
               name="nuovo"
               value="si"/>
    </p>
    <input type="submit"
           name="submit"
           value="invia" />
</form>
<!-- omissis -->
```

Nome: Alessandro

Cognome: Ursomando

Registra: ☑

invia

Ti chiami Alessandro Ursomando

.. e sei qui per la prima volta!

ESERCIZIO 14-00-08-01

ESEMPIO

In **PHP** la pagina destinataria troverà i dati in un vettore associativo **$_GET** o **$_POST**.

Le etichette saranno il valore della proprietà **name** degli elementi .

Nome: Alessandro

Cognome: Ursomando

Registra: ☑

invia

Ti chiami Alessandro Ursomando

.. e sei qui per la prima volta!

```
<!-- omissis -->
<?php
    echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
    if (isset($_POST['nuovo']))
        echo "<p>.. e sei qui per la prima volta!</p>";
?>
<!-- omissis -->
```

ESERCIZIO 14-00-08-01

GET vs POST

- invia i dati direttamente nella query string
  - dati visibili
  - lunghezza limitata (256 caratteri)

GET

```
http://localhost/esercizi/elabora.php?nome=se&cognome=se&submit=invia
```


GET vs POST

- invia i dati direttamente nella query string
  - dati visibili
  - lunghezza limitata (256 caratteri)

GET

- invia i dati nella richiesta HTTP
  - dati non visibili
  - lunghezza illimitata

POST

Pertanto, se non diversamente indicato, noi useremo sempre il metodo **post**.

ESERCIZI

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST))
        // mostra la pagina di benvenuto
    else
        // mostra il form
?>
```

Lo schema visto prevede la realizzazione di due pagine:

- la pagina **index** che mostra il **form** e spedisce le coppie (nome, valore) a un'altra pagina
- l'altra pagina che legge i dati dal vettore **$_POST** o **($_GET)**

Vediamo un altro schema che prevede la realizzazione di **un'unica** pagina, la quale – sulla base dell'esistenza del vettore **$_POST** (o **$_GET**) – esegue una parte oppure l'altra.

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome']))
        // mostra la pagina di benvenuto
    else
        // mostra il form
?>
```

Purtroppo il vettore **$_POST** esiste sempre e quindi dobbiamo verificare se esiste una componente specifica.

```php
isset($_POST['nome'])
```

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome'])) {
        echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
        if (isset($_POST['nuovo']))
            echo "<p>.. e sei qui per la prima volta!</p>";
    } else {
        // mostra il form









    }
?>
```

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome'])) {
        echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
        if (isset($_POST['nuovo']))
            echo "<p>.. e sei qui per la prima volta!</p>";
    } else {
        echo '<form action="#" method="post">';
        echo '  <p>';
        echo '      <label for="nome">Nome:</label>';
        echo '      <input id="nome" type="text" name="nome"/>';
        echo '  </p>';
        echo '  <p>';
        echo '      <label for="cognome">Cognome:</label>';
        echo '      <input id="cognome" type="text" name="cognome"/>';
        echo '  </p>';
        echo '  <p>';
        echo '      <label for="nuovo">Registra: </label>';
        echo '      <input type="checkbox" name="nuovo"  value="si" id="nuovo">';
        echo '  </p>';
        echo '  <input type="submit" name="submit" value="invia" />';
        echo '</form>';
    }
?>
```

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome'])) {
        echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
        if (isset($_POST['nuovo']))
            echo "<p>.. e sei qui per la prima volta!</p>";
    } else {
        echo '<form action="#" method="post">';
        echo '  <p>';
        echo '      <label for="nome">Nome:</label>';
        echo '      <input id="nome" type="text" name="nome"/>';
        echo '  </p>';
        echo '  <p>';
        echo '      <label for="cognome">Cognome:</label>';
        echo '      <input id="cognome" type="text" name="cognome"/>';
        echo '  </p>';
        echo '  <p>';
        echo '      <label for="nuovo">Registra: </label>';
        echo '      <input type="checkbox" name="nuovo"  value="si" id="nuovo">';
        echo '  </p>';
        echo '  <input type="submit" name="submit" value="invia" />';
        echo '</form>';
    }
?>
```

Attenzione al valore del parametro **action**!

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome'])) {
        echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
        if (isset($_POST['nuovo']))
            echo "<p>.. e sei qui per la prima volta!</p>";
    } else {
        echo '<form action="#" method="post">';
        echo '  <p>';
        echo '      <label for="nome">Nome:</label>';
        echo '      <input id="nome" type="text" name="nome"/>';
        echo '  </p>';
        echo '  <p>';
        echo '      <label for="cognome">Cognome:</label>';
        echo '      <input id="cognome" type="text" name="cognome"/>';
        echo '  </p>';
        echo '  <p>';
        echo '      <label for="nuovo">Registra: </label>';
        echo '      <input type="checkbox" name="nuovo"  value="si" id="nuovo">';
        echo '  </p>';
        echo '  <input type="submit" name="submit" value="invia" />';
        echo '</form>';
    }
?>
```

Ed infine, eliminiamo tutte queste istruzioni **echo**!

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome'])) {
        echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
        if (isset($_POST['nuovo']))
            echo "<p>.. e sei qui per la prima volta!</p>";
    } else {
?>
        <form action="#" method="post">
            <p>
                <label for="nome">Nome:</label>
                <input id="nome" type="text" name="nome"/>
            </p>
            <p>
                <label for="cognome">Cognome:</label>
                <input id="cognome" type="text" name="cognome"/>
            </p>
            <p>
                <label for="nuovo">Registra: </label>
                <input type="checkbox" name="nuovo"  value="si" id="nuovo">
            </p>
            <input type="submit" name="submit" value="invia" />
        </form>
<?php
    }
?>
```

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

```php
<?php
    if (isset($_POST['nome']))
        // mostra la pagina di benvenuto
    else
        // mostra il form
?>
```

Qualora l'inserimento di tutto il codice arrivi ad essere di difficile gestione, si può decidere di isolare le due porzioni codice in due file separati e di linkare gli stessi mediante la funzione **include**.

```php
<?php
    if (isset($_POST['nome']))
        include("mostraPagBenvenuto.php");
    else
        include("mostraPagForm.php");
?>
```

ESERCIZIO 14-00-09-01

TUTTO IN UNA PAGINA

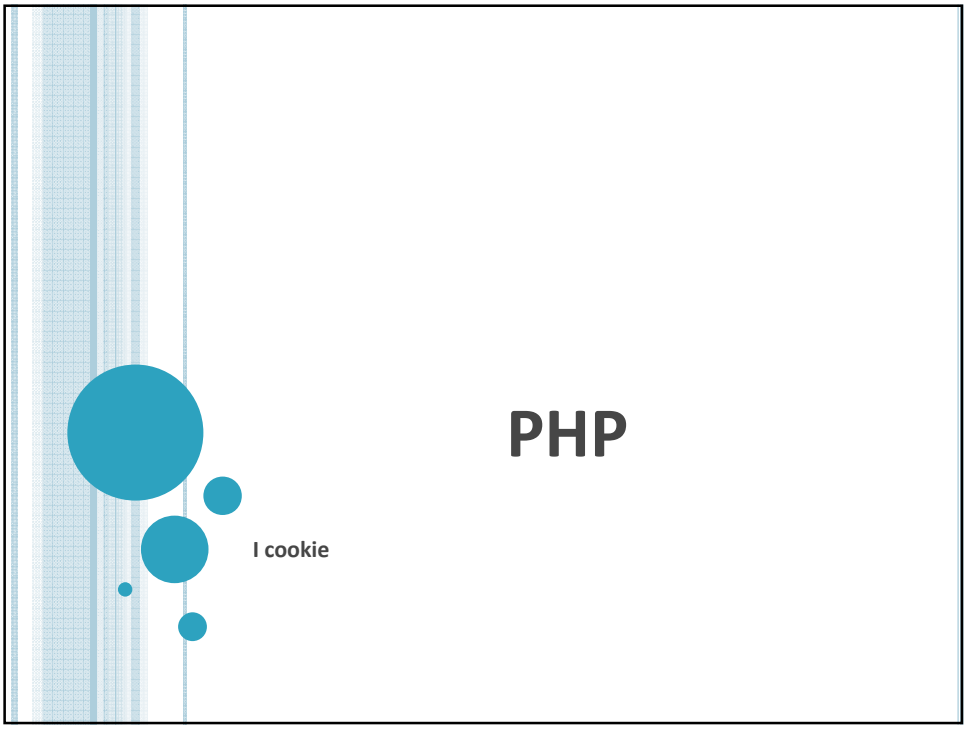
```php
<?php
    echo "<p>Ti chiami ".$_POST['nome']." ".$_POST['cognome']."</p>";
    if (isset($_POST['nuovo']))
        echo "<p>.. e sei qui per la prima volta!</p>";
?>
```

```php
<form action="#" method="post">
    <p>
        <label for="nome">Nome:</label>
        <input id="nome" type="text" name="nome"/>
    </p>
    <p>
        <label for="cognome">Cognome:</label>
        <input id="cognome" type="text" name="cognome"/>
    </p>
    <p>
        <label for="nuovo">Registra: </label>
        <input type="checkbox" name="nuovo"  value="si" id="nuovo">
    </p>
    <input type="submit" name="submit" value="invia" />
</form>
```

```php
<?php
    if (isset($_POST['nome']))
        include("mostraPagBenvenuto.php");
    else
        include("mostraPagForm.php");
?>
```

ESERCIZIO 14-00-09-01

ESERCIZI



DEFINIZIONI

- ❑ È un biscotto
- ❑ È una coppia **(chiave, valore)** salvata sul PC dell'utente
- ❑ Qualsiasi pagina php può **scrivere** cookie
- ❑ Qualsiasi pagina php può **leggere** cookie
- ❑ I cookie possono avere una **scadenza**

COOKIE

setcookie(str1, str2, ts)	scrive (sulla macchina client) un cookie denominato **str1** al valore **str2** che scade al **timestamp** indicato con **ts**
$_COOKIE	è il vettore associativo che si usa per leggere i cookie

Poiché noi non abbiamo ancora il concetto di **timestamp**, per il momento tralasciamo l'uso dell'ultimo parametro.

ALESSANDRO URSOMANDO

CREAZIONE DI UN COOKIE

- ❑ È un biscotto
- ❑ È una coppia **(chiave, valore)** salvata sul PC dell'utente
- ❑ Qualsiasi pagina php può **scrivere** cookie
- ❑ Qualsiasi pagina php può **leggere** cookie
- ❑ I cookie possono avere una **scadenza**

COOKIE

setcookie(str1, str2, ts)	scrive (sulla macchina client) un cookie denominato **str1** al valore **str2** che scade al **timestamp** indicato con **ts**
$_COOKIE	è il vettore associativo che si usa per leggere i cookie

```php
<?php
    setcookie('totaleCarrello', '100');
    // ho scritto sulla macchina client un cookie che..
?>
```


ALESSANDRO URSOMANDO

LETTURA DI UN COOKIE

- È un biscotto
- È una coppia **(chiave, valore)** salvata sul PC dell'utente
- Qualsiasi pagina php può **scrivere** cookie
- Qualsiasi pagina php può **leggere** cookie
- I cookie possono avere una **scadenza**

COOKIE

setcookie(str1, str2, ts)	scrive (sulla macchina client) un cookie denominato **str1** al valore **str2** che scade al **timestamp** indicato con **ts**
$_COOKIE	è il vettore associativo che si usa per leggere i cookie

```php
<?php
    echo "<h1>".$_COOKIE['totaleCarrello']."</h1>";
    // leggo il cookie e lo mostro
?>
```


CANCELLAZIONE DI UN COOKIE

- È un biscotto
- È una coppia **(chiave, valore)** salvata sul PC dell'utente
- Qualsiasi pagina php può **scrivere** cookie
- Qualsiasi pagina php può **leggere** cookie
- I cookie possono avere una **scadenza**

COOKIE

setcookie(str1, str2, ts)	scrive (sulla macchina client) un cookie denominato **str1** al valore **str2** che scade al **timestamp** indicato con **ts**
$_COOKIE	è il vettore associativo che si usa per leggere i cookie

```php
<?php
    setcookie('totaleCarrello', '', time()-1);
    // imposto il cookie a un secondo fa: lo cancello
?>
```

CANCELLAZIONE DI UN COOKIE

- È un biscotto
- È una coppia (chiave, valore) salvata sul PC dell'utente
- Qualsiasi pagina php può scrivere cookie
- Qualsiasi pagina php può leggere cookie
- I cookie possono avere una scadenza

La cancellazione dei cookie
può avvenire anche mediante
gli strumenti a disposizione dal browser.

	al valore str2 che scade al datetime indicato con dt
$_COOKIE	è il vettore associativo che si usa per leggere i cookie

```php
<?php
    setcookie('totaleCarrello', '', time()-1);
    // imposto il cookie a un secondo fa: lo cancello
?>
```


QUANDO È DISPONIBILE UN COOKIE?

L'ultima osservazione è relativa al **momento** in cui
un cookie appena scritto (con la funzione **setcookie**)
diventa disponibile (nel vettore **$_COOKIE**).

Ogni volta che viene **avviato** uno script PHP,
il browser vede se ci sono dei cookie sulla macchina
(scritti **precedentemente** da **altri** script)
e li rende disponibili (mediante il vettore **$_COOKIE**)

Se questo script **crea** dei cookie
questi **non** compariranno nel vettore **$_COOKIE** in uso da **quello stesso** script!

Solo gli script avviati **successivamente** ad esso
riceveranno un vettore **$_COOKIE**
che conterrà tutti i cookie **presenti sulla macchina in quel momento**
e quindi **anche** quelli inseriti dallo script di cui sopra.

ESEMPIO

In questo esempio la pagina **index** si presenta nei due modi che seguono sulla base della presenza (e del valore) del **cookie** trovato sulla macchina **client**.

Nome: invia
Qui poi ci va la pagina vera e propria.

Bentornato Alessandro!!!
Qui poi ci va la pagina vera e propria.

Clicca qui per cancellare il cookie

ESEMPIO

Nel caso non ci sia il **cookie**, una volta inserito un valore nel **textedit**, si richiama una pagina che scrive il **cookie** sulla macchina client.

Inserimento avvenuto con successo.

Clicca qui per cancellare il cookie
Clicca qui per tornare alla home

ESEMPIO

Quando il **cookie** è presente sulla macchina, si dà il bentornato e si propone un link che permette di cancellare il **cookie** stesso.

Inserimento avvenuto con successo.

Clicca qui per cancellare il cookie
Clicca qui per tornare alla home

Bentornato Alessandro!!!
Qui poi ci va la pagina vera e propria.

Clicca qui per cancellare il cookie

ESEMPIO

Lo stesso link (per la cancellazione del **cookie**) è presente anche sulla pagina di avvenuto inserimento.

Inserimento avvenuto con successo.

Clicca qui per cancellare il cookie
Clicca qui per tornare alla home

Bentornato Alessandro!!!
Qui poi ci va la pagina vera e propria.

Clicca qui per cancellare il cookie

Cancellazione avvenuta con successo.

Clicca qui per tornare alla home

ESEMPIO

Quando necessario si propone un link alla pagina iniziale,
la quale si presenterà in accordo con la presenza ed il valore del **cookie**.

Inserimento avvenuto con successo.

Clicca qui per cancellare il cookie
Clicca qui per tornare alla home

Bentornato Alessandro!!!
Qui poi ci va la pagina vera e propria.

Clicca qui per cancellare il cookie

Cancellazione avvenuta con successo.

Clicca qui per tornare alla home

ESEMPIO: LA PAGINA INDEX

```php
<?php
    if (isset($_COOKIE['nome']))
        // mostra messaggio di bentornato
    else {
        // mostra il form
    }

    // la pagina vera e propria


    if (isset($_COOKIE['nome'])) {
    // il footer coi link
    }
?>
```

Nome: invia
Qui poi ci va la pagina vera e propria.

Bentornato Alessandro!!!
Qui poi ci va la pagina vera e propria.

Clicca qui per cancellare il cookie

ESERCIZIO 14-00-10-01

ESEMPIO: LA PAGINA INDEX

```php
<?php
    if (isset($_COOKIE['nome']))
        echo "<h1>Bentornato ".$_COOKIE['nome']."!!!</h1>";
    else {
?>
    <form action="elabora.php" method="post">
        <label for="nome">Nome:</label>
        <input type="text" name="nome" id="nome" />
        <input type="submit" name="submit" value="invia" />
    </form>
<?php
    }
?>
    <small>
        Qui poi ci va la pagina vera e propria.
    </small>
<?php
    if (isset($_COOKIE['nome'])) {
?>
        <footer>
            Clicca <a href="cancella.php">qui</a>
            per cancellare il cookie
        <footer>
<?php
    }
?>
```

ESERCIZIO 14-00-10-01

ESEMPIO: LA PAGINA ELABORA

```php
<?php
    // inserisce il cookie
    // il footer coi link
    }
?>
```

```php
<?php
    setcookie('nome', $_POST['nome']);
?>
<h1>Inserimento avvenuto con successo.</h1>
<footer>
    <p>
        Clicca <a href="cancella.php">qui</a>
        per cancellare il cookie
     </p>
     <p>
        Clicca <a href="index.php">qui</a>
        per tornare alla home
    </p>
<footer>
```

Inserimento avvenuto con successo.

Clicca qui per cancellare il cookie
Clicca qui per tornare alla home

ESERCIZIO 14-00-10-01

ESEMPIO: LA PAGINA CANCELLA

```php
<?php
    // cancella il cookie
    // il footer coi link
    }
?>
```

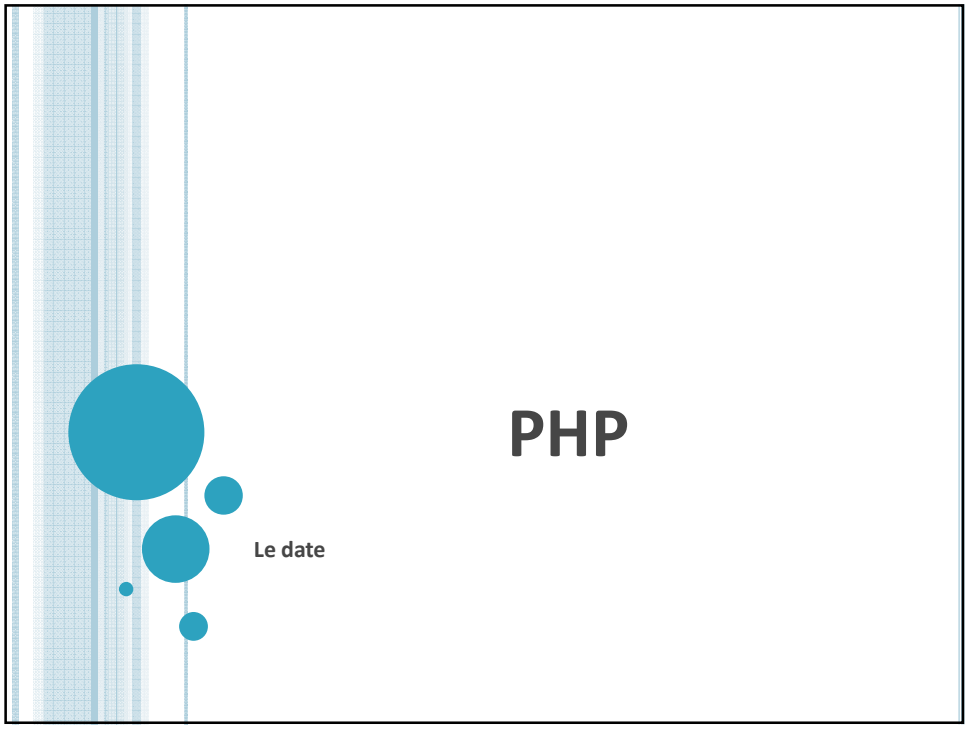
```php
<?php
    setcookie('nome', '', time()-1);
?>
<h1>Cancellazione avvenuta con successo.</h1>

<footer>
    Clicca <a href="index.php">qui</a>
    per tornare alla home
<footer>
```

Cancellazione avvenuta con successo.

Clicca qui per tornare alla home

ESERCIZIO 14-00-10-01

ESERCIZI



FUNZIONI SULLE DATE

time()	restituisce il timestamp di adesso
date($str, $ts)	restituisce una stringa formattata che rappresenta il timestamp **$ts** passato secondo il formato indicato in **$str**
mktime ($ore, $minuti, $secondi, $mese, $giorno, $anno)	restituisce un timestamp prodotto a partire dai parametri indicati
checkdate($m, $g, $a)	restituisce **true** se la data passata è valida

Il **timestamp** è il numero di secondi trascorsi dal 1° Gennaio 1970.

CODICI DI FORMATTAZIONE

Codice	Descrizione
Y	anno su 4 cifre
y	anno su 2 cifre
n	mese numerico (1-12)
m	mese numerico su 2 cifre (01-12)
F	mese testuale ('January' - 'December')
M	mese testuale su 3 lettere ('Jan' - 'Dec')
d	giorno del mese su due cifre (01-31)
j	giorno del mese (1-31)
w	giorno della settimana, numerico (0=dom, 6=sab)
l	giorno della settimana, testuale ('Sunday' - 'Saturday')
D	giorno della settimana su 3 lettere ('Sun' - 'Sat')
H	ora su due cifre (00-23)
G	ora (0-23)
i	minuti su due cifre (00-59)
s	secondi su due cifre (00-59)

ALESSANDRO URSOMANDO

PHP

I cookie
(gestione avanzata)

ESERCIZI

PHP

Interazione con i database

MYSQL

- PHP nasce per gestire basi di dati
- PHP può interfacciarsi a tanti DBMS
- PHP si interfaccia a MYSQL
  - mediante i layer di astrazione
  - con la libreria mysql (funzioni)
  - con la libreria mysqli (oggetti)

- apertura e chiusura della connessione
- invio interrogazioni
- lettura dei risultati

OPERAZIONI SUI DB

FUNZIONI MYSQL (1)

mysql_connect()	restituisce una variabile di tipo **resource** che rappresenta la connessione (o **false** su errore)
mysql_close($c)	restituisce **true** se riesce a chiudere la connessione **$c**
mysql_select_db($nome, $c)	restituisce **true** se è riuscito a selezionare il db indicato in **$nome** sul dbms cui si è connessi con **$c**

FUNZIONI MYSQL (2)

mysql_list_dbs($c)	restituisce un **resource** che contiene un elenco di stringhe che rappresentano i nomi dei db della connessione **$c**
mysql_fetch_array($r)	restituisce un array del **resource $r** e sposta in avanti il puntatore; quando finisce restituisce **false** (l'array può essere usato sia come associativo che semplice)
mysql_list_tables($nome, $c)	restituisce un **resource** che contiene un elenco di stringhe che rappresentano i nomi delle tabelle del db indicato sulla connessione **$c**

FUNZIONI MYSQL (3)

mysql_query($q)	Esegue la query **$q** e restituisce un **resource** che contiene un elenco di array. Ogni array è una riga del risultato della query; ogni array contiene il nome dell'attributo (come chiave) e valore (come valore).
mysql_num_rows($r)	Restituisce la quantità di vettori che contiene il **resource $r**.
mysql_result($r,$x,$y)	Accede alla variabile di tipo **resource $r** e restituisce la cella numero **$y** del vettore numero **$x**.

ESEMPIO

Supponiamo di avere questa tabella nel database **TestSlideDML**

Presidenti

id	nominativo	percentuale	dal	al	idPartiti
122	Enrico De Nicola	72,8	01/07/1946	12/05/1948	325
125	Luigi Einaudi	59,4	12/05/1948	11/05/1955	325
131	Giovanni Gronchi	74,5	11/05/1955	11/05/1962	326
132	Antonio Segni	52,6	11/05/1962	06/12/1964	326
135	Giuseppe Saragat	68,9	29/12/1964	29/12/1971	329
136	Giovanni Leone	52	29/12/1971	15/06/1978	326
145	Alessandro Pertini	83,6	09/07/1978	29/06/1985	380
155	Francesco Cossiga	75,4	03/07/1985	28/04/1992	326
156	Oscar Luigi Scalfaro	66,3	28/05/1992	15/05/1999	326
187	Carlo Azeglio Ciampi	71,4	18/05/1999	15/05/2006	489
221	Giorgio Napolitano	54,8	15/05/2006		501

E supponiamo di volere stampare a video in questo modo i nominativi dei presidenti che hanno avuto una percentuale di preferenza superiore al 60%.

Enrico De Nicola
Giovanni Gronchi
Giuseppe Saragt
Alessandro Pertini
Francesco Cossiga
Oscar Luigi Scalfaro
Carlo Azeglio Ciampi

ESERCIZIO 14-00-13-01

ESEMPIO

```
<!doctype html>
<html>
<head>
    <title>Esercizio</title>
    <meta charset="UTF-8" />
    <link type="text/css" rel="stylesheet" href="reset.css" />
    <link type="text/css" rel="stylesheet" href="stili.css" />
</head>
<body>
<?php
    // Preparo la connessione

    // Seleziono i dati e li mostro a video

    // chiudo la connessione
?>
</body>
</html>
```

ESERCIZIO 14-00-13-01

ESEMPIO

```php
// Preparo la connessione
$dbHost     = "localhost";
$dbUser     = "root";
$dbPasswd   = "";
$dbNome     = "testSlideDML";
$connessione = mysql_connect($dbHost, $dbUser, $dbPasswd);
mysql_select_db($dbNome, $connessione);
```

```php
// Seleziono i dati e li mostro a video
$query = "select nominativo from presidenti where percentuale > 60";
$risultato = mysql_query($query);
$rigaPari = false;
while ($unaRiga = mysql_fetch_array ($risultato)) {
    if ($rigaPari) {
        echo "\t<p class=\"rigaPari\">\n";
        $rigaPari = false;
    } else {
        echo "\t<p class=\"rigaDispari\">\n";
        $rigaPari = true;
    }
    echo "\t\t".$unaRiga['nominativo']."\n";
    echo "\t</p>\n";
}
```

```php
// chiudo la connessione
mysql_close($connessione);
```

ESERCIZIO 14-DB-01

ESERCIZI

PHP

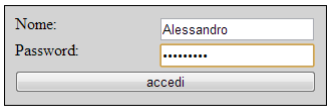
Interazione con i database e gestione degli errori

GESTIONE DEGLI ERRORI

Quando uno script PHP produce un **errore**, il browser lo segnala.

Prendiamo, per esempio, una porzione dell'esercizio
in cui si gestiva l'accesso ad un database e facciamo un **errore di sintassi**.

```
$dbHost   = $_POT['host'];
$dbUser   = $_POST['nome'];
$dbPasswd = $_POST['password'];
```



In questo caso seguendo le indicazioni del browser
(errore occorso riga di codice dove intervenire) possiamo correggere il nostro script.

Notice: Undefined variable: _POT in G:\xampp\htdocs\esercizi\14-13-01\elabora.php on line 13

ALESSANDRO URSOMANDO

GESTIONE DEGLI ERRORI

Il browser però segnala anche gli errori **generati dall'utente**!

Prendiamo ancora come esempio l'esercizio precedente e verifichiamo cosa accade se l'utente prova ad accedere con un account **non** autorizzato.

```
$connessione = mysql_connect($dbHost, $dbUser, $dbPasswd);
if ($connessione == false) {
    echo "<p>Errore in connessione</p>";
    exit();
} else {
    echo "<p>Connessione avvenuta con successo.</p>";
}
```

In questo caso, il browser **sporca** la pagina con un messaggio.

Warning: mysql_connect(): Access denied for user 'Alessandro'@'localhost' (using password: YES) in G:\xampp\htdocs\esercizi\14-13-01\elabora.php on line 17
Errore in connessione

GESTIONE DEGLI ERRORI

Per impedire al browser di mostrare gli errori di **una certa istruzione** anteponiamo una **chiocciola** (@) a questa istruzione.

```
$connessione = @mysql_connect($dbHost, $dbUser, $dbPasswd);
```

GESTIONE DEGLI ERRORI

Per impedire al browser di mostrare gli errori di **una certa istruzione** anteponiamo una **chiocciola** (@) a questa istruzione.

Nel caso l'istruzione vada in errore si può indicare al browser la stringa da mostrare con il costrutto **or die**

```
$connessione = @mysql_connect($dbHost, $dbUser, $dbPasswd)
    or die ("Errore durante la connessione.");
```

Il browser mostrerà la seguente pagina WEB.

Errore durante la connessione.

GESTIONE DEGLI ERRORI

Per impedire al browser di mostrare gli errori di **una certa istruzione** anteponiamo una **chiocciola** (@) a questa istruzione.

Nel caso l'istruzione vada in errore si può indicare al browser la stringa da mostrare con il costrutto **or die**

In questo caso, abbiamo a disposizione le funzioni **mysql_errno()** e **mysql_error()** per conoscere il numero e la descrizione dell'errore verificatosi.

```
$connessione = @mysql_connect($dbHost, $dbUser, $dbPasswd)
    or die ("Errore durante la connessione: ".mysql_errno());
```

Il browser mostrerà la seguente pagina WEB.

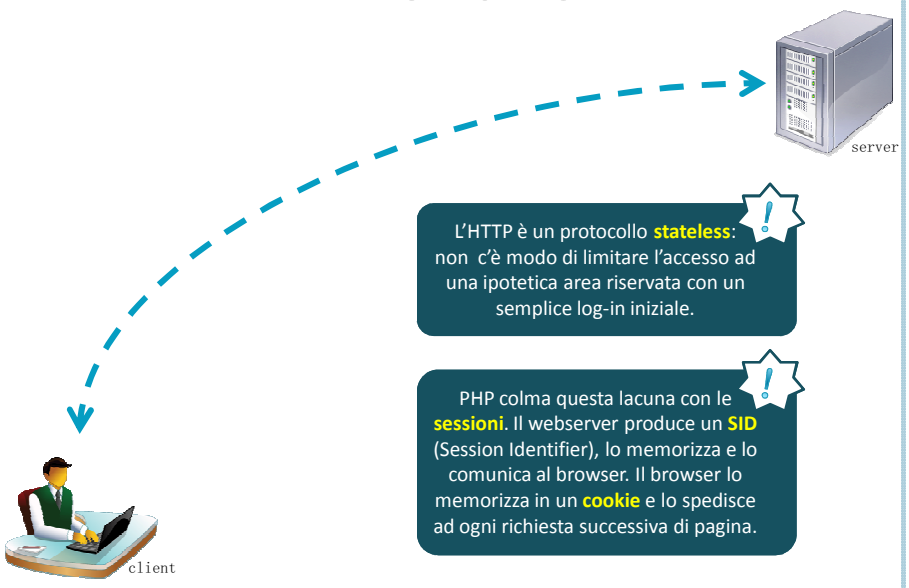
Errore durante la connessione: 1045

ESERCIZI

PHP

Le sessioni

INTRODUZIONE



L'HTTP è un protocollo **stateless**: non c'è modo di limitare l'accesso ad una ipotetica area riservata con un semplice log-in iniziale.

PHP colma questa lacuna con le **sessioni**. Il webserver produce un **SID** (Session Identifier), lo memorizza e lo comunica al browser. Il browser lo memorizza in un **cookie** e lo spedisce ad ogni richiesta successiva di pagina.

ALESSANDRO URSOMANDO