PHP ESSENTIALS #5

By WI400 Team

: stringhe

Agenda

- definizione delle stringhe
- comparazione
- corrispondenza
- parsing
- formattazione

Stringhe

- Le stringhe sono i tipi di variabili più spesso utilizzate
- Tutti i dati che arrivano da una form vengono passati come stringhe (anche i checkbox)
- Le stringhe possono avere una lunghezza arbitraria; non esiste nessun limite in merito
- PHP mette a disposizione una grande quantità di funzioni per la manipolazione delle stringhe

Definizione delle stringhe

- Esistono tre modi diversi di definire una stringa
  - ‘ ’ - Racchiudendo la stringa tra i singoli apici
  - “ “ - Racchiudendo la stringa tra i doppi apici
  - <<< - Heredoc
- Il metodo da utilizzare dipende da cosa deve contenere la stringa e dall’utilizzo che ne vogliamo fare

```php
<?php

 $domanda = 'vi piace il PHP?';
 $stringa1 = 'Ciao a tutti, $domanda';
 echo $stringa1;
 echo '<br>';

 $stringa2 = "Ciao a tutti, $domanda";
 echo $stringa2;
```

qual'è l'output ?

```
Ciao a tutti, $domanda
Ciao a tutti, vi piace il PHP?
```

Stringhe: Apici singoli

- Le stringhe possono essere definite tramite apici singoli
- I caratteri contenuti saranno memorizzati così come sono senza interpretare caratteri o sequenze di escape

```php
<?php

$variabile = 'stringa';
echo 'Questa è una $variabile.';
```

```
Questa è una $variabile
```

Stringhe: Apici doppi

- Le stringhe possono essere definite tramite i doppi apici
- Le variabili e i caratteri di escape saranno interpretati e sostituiti

```php
<?php

$variabile = 'stringa';
echo "Questa è una $variabile.";
```

```
Questa è una stringa
```

Stringhe: Heredoc

- Le stringhe possono essere definite tramite Heredoc

  Molto simile ai doppi apici ma progettato per dividere la stringa su più righe. Permette inoltre l'utilizzo di apici singoli e apici doppi senza eseguire l'escape

dov'è l'errore ?

```
<?php
$nome = "Mario Rossi";
$variabile = <<<LOGINLOG
    L'utente "$nome" ha eseguito l'accesso al portale
    come amministratore del sistema.


        LOGINLOG;
echo $variabile;
```

- Il tag di chiusura dell'heredoc deve essere all'inizio della riga e l'unico carattere permesso nella stessa riga è il punto e virgola ';'

Stringhe: abbreviata

- è possibile utilizzare la funzione echo con una sintassi abbreviata che funziona solo se nel file di configurazione **php.ini** la direttiva short_open_tag è impostata su on:

  nota:con lo ZendServer questo è già abilitato

```
<?php
$stringa = "ciao";
?>
<!-- così viene stampata la variabile $stringa -->
<?=$stringa?>
```

Stringhe: Conversione

- La funzione *strtoupper()* converte tutte le lettere di una stringa nelle rispettive lettere maiuscole.

```php
<?php
$stringa = "Mario Rossi";
$stringaUpper = strtoupper ( $stringa );

echo $stringaUpper;
```

```
MARIO ROSSI
```

Stringhe: Conversione

- La funzione *strtolower()* converte tutte le lettere di una stringa nelle rispettive lettere minuscole

```php
<?php
$stringa = 'MARIO ROSSI';
$stringaLower = strtolower ( $stringa );

echo $stringaLower;
```

```
mario rossi
```

Stringhe: Comparazione

- E' possibile comparare le stringhe con gli operatori di comparazione più noti:

  (==, !=, >, <, >=, <=, ===, !==)

qual'è l'output ?

```php
<?php
$stringa1 = 'fox';
$stringa2 = 'fox';
if ($stringa1 == $stringa2) {
    echo '<p>$stringa1 = $stringa2</p>';
}
```

→ `$stringa1 = $stringa2`

Stringhe: Comparazione

- E’ possibile utilizzare anche le funzioni di comparazione messe a disposizione dal PHP
- La funzione *strcmp()* accetta due parametri e restituisce un intero
  - ✓ < 0 se la prima stringa è minore della seconda
  - ✓ > 0 se la prima stringa è maggiore della seconda
  - ✓ = 0 se le due stringhe sono uguali

qual'è l'output ?

```php
<?php
$stringa1 = 'fox';
$stringa2 = 'fox';
if (strcmp($stringa1,$stringa2)) {
    echo "<p>$stringa1 = $stringa2</p>";
} else {
    echo 'ci sono delle differenze';
}
```

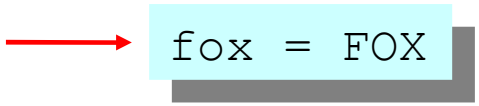
→ ci sono delle differenze

ricorda !
strcmp() restituisce **0** quando le stringhe sono uguali

Stringhe: Comparazione

- La funzione *strcasecmp()* è identica alla funzione *strcmp()* ma non tiene conto delle maiuscole e delle minuscole (*case-insensitive*)

```php
<?php
$stringa1 = 'fox';
$stringa2 = 'FOX';
if (!strcasecmp($stringa1,$stringa2)) {
    echo "<p>$stringa1 = $stringa2</p>";
} else {
    echo 'ci sono delle differenze';
}
```



Stringhe: Comparazione

- La funzione *strncmp()* permette di comparare i primi *n* caratteri di due stringhe:
- strncasecmp() è simile ma è "*case-insensitive*"
- Accetta tre parametri in ingresso e restituisce un intero come risultato

qual'è l'output ?

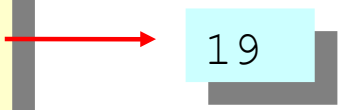
```
<?php
$stringa1 = 'fox';
$stringa2 = 'Foxtrot';
if (!strncasecmp($stringa1,$stringa2,3)) {
    echo 'i primi 3 caratt. di $stringa1 e $stringa2 coincidono';
    echo "<br>";
    print "rilevate ".strncmp($stringa1,$stringa2,3)." diff. maius/minus";
} else {
    echo 'ci sono delle differenze';
}
```

```
i primi 3 caratt. di $stringa1 e
$stringa2 coincidono
rilevate 1 differenze maius/minus
```

Stringhe: Lunghezza

- la funzione strlen() restituisce il numero di caratteri di cui si compone una stringa:

```php
<?php
echo strlen("the quick brown fox");
```



Stringhe: Ricerca

- La funzione strstr() trova la prima occorrenza di una stringa all'interno di un'altra stringa

  stristr() è identica ma "case-insensitive"

- Accetta due parametri:
  - La stringa da controllare
  - La stringa da ricercare
- Restituisce come risultato:
  - Dalla stringa trovata fino al termine della stringa
  - false nel caso in cui la stringa non sia stata trovata

```php
<?php
$stringa = 'Il PHP è il linguaggio che preferisco';
echo strstr ( $stringa, 'PHP' );
```

→ `PHP è il linguaggio che preferisco`

Stringhe: Ricerca

- La funzione strpos() trova la posizione di una stringa all'interno di un'altra stringa

  stripos() è identica ma "*case-insensitive*"
- Accetta due parametri:
  - ✓ La stringa da controllare,la stringa da ricercare
- Restituisce come risultato:
  - ✓ La posizione numerica della stringa trovata (a partire da 0)
  - ✓ false nel caso in cui la stringa non sia stata trovata

```php
<?php
$string = 'The quick brown fox';
if(!strpos($string, 'The')) {
    echo "stringa non trovata";
} else {
    echo "stringa trovata";
}
```

qual'è l'output ?

→ `stringa non trovata`

ricorda !
`strpos()` è zero based, e **0** è uguale a **false**

Stringhe: approccio corretto

qual'è l'output ?

```php
<?php
$string = 'The quick brown fox';
$pos=strpos($string, 'The');
if($pos === false) {
    echo "stringa non trovata";
} else {
    echo "stringa trovata";
}
```



```
stringa trovata
```

Stringhe: Conteggio

- La funzione substr_count() conta il numero delle occorrenze di una sottostringa
- Accetta due parametri:
  - La stringa da controllare, la sottostringa da ricercare
- Restituisce come risultato un intero che corrisponde al numero di occorrenze trovate

qual'è l'output ?

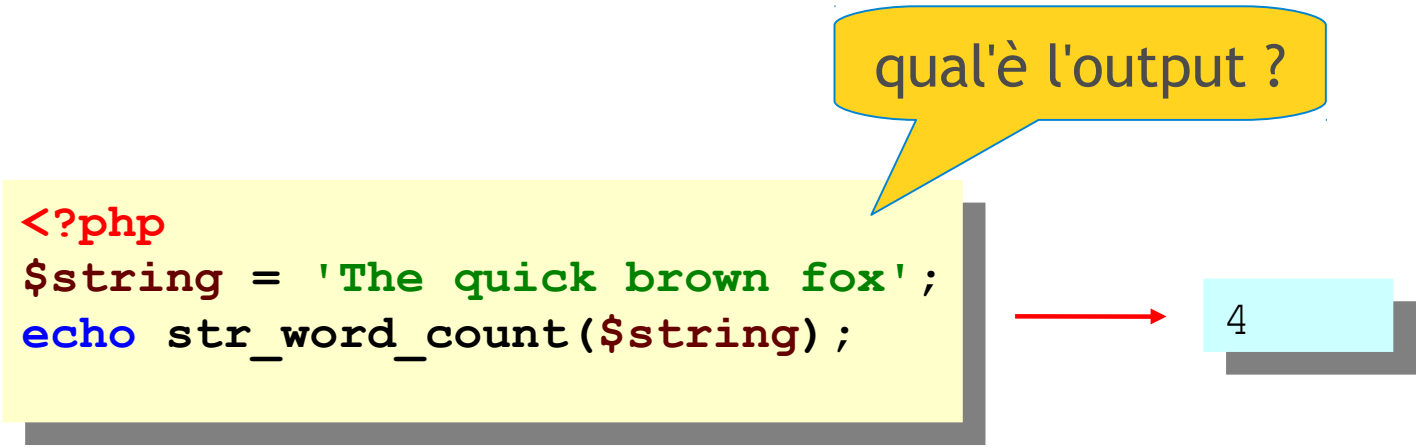
```php
<?php

$string = 'The quick and brown and crafty fox';
echo substr_count($string,'and');
```

2

Stringhe: Conteggio

- La funzione str_word_count() conta il numero delle parole presenti all'interno di una stringa
- Accetta come parametro la stringa da controllare
- Restituisce come risultato un intero che corrisponde al numero delle parole presenti nella stringa



Stringhe

- Il PHP mette a disposizione un'ampia gamma di funzioni per le stringhe
- Quelle viste fino ad ora sono solo una piccola parte
- Per maggiori informazioni:
  http://www.php.net/manual/en/ref.strings.php

Esercizio 5:

- creare la pagina “search.php” che possa ricevere in post una stringa e restituisca:

- il numero delle occorrenze della parola “the” nella stringa

- il totale del numero delle parole della stringa
- la percentuale delle parole “the” nella stringa

Esercizio 5: solution

- creazione della pagina search.html

```
<html>
<form action="search.php" method="POST">
<p><input type="text" name="string" /></p>
<p><input type="submit" /></p>
</form>
</html>
```

- creazione della pagina php

```
<?php
$the = substr_count ( $_POST ['string'], 'the' );
$total = str_word_count ( $_POST ['string'] );
$percentage = ($the / $total) * 100;
echo "$the / $total = $percentage %";
```

Stringhe: Parsing

- E’ possibile trattare una stringa utilizzandola come un array
- E’ possibile fare riferimento ai singoli caratteri di una stringa accedendo alla posizione del carattere
- Anche se la sintassi è molto semplice tutto risulterà più chiaro quando parleremo degli array

Stringhe: Parsing

- Suppondendo di voler accedere al quinto carattere di una stringa è sufficiente utilizzare la seguente sintassi:

```php
<?php
$stringa = 'Il PHP è il linguaggio che preferisco';
echo $stringa [4];
```

→ H

- Il quinto carattere è il carattere 'H' e si accede con indice 4 perchè il conteggio delle lettere inizia da 0 ("*zero-based*")

- E' possibile combinare quanto detto con uno dei cicli di array per scorrere tutti i caratteri di una stringa uno alla volta:

```php
<?php

$stringa = 'Il PHP è il linguaggio che preferisco';
for($i = 0; $i < strlen ( $stringa ); $i ++) {
    $stringa [$i] = strtoupper ( $stringa [$i] );
}
echo $stringa;
```

→ IL PHP è IL LINGUAGGIO CHE PREFERISCO

Stringhe: Parsing

- La funzione substr() restituisce una sottostringa a partire da una stringa
- Accetta due parametri obbligatori:
  - La stringa da analizzare
  - La posizione iniziale della sottostringa da estrarre (può essere negativa)
- Accetta opzionalmente anche un terzo parametro:
  - La posizione finale della sottostringa da estrarre (può essere negativa)

ricorda! : il calcolo è "zero-based", ossia il primo carattere ha posizione 0 e non 1.

```php
<?php
$string = 'The quick brown fox';
echo substr($string, 4);
echo "<br>";
echo substr($string, -3);
```

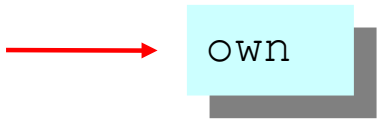
qual'è l'output ?

```
quick brown fox
fox
```

Stringhe: Parsing

- qual'è l'output del seguente codice ?

```php
<?php
$string = 'The quick brown fox';
echo substr($string, -7, -4);
?>
```



Stringhe: Parsing

- funzione interessante è str_replace():
- substring da sostituire
- substring che sostituisce
- stringa su cui lavorare

Problema:
data la stringa **'The quick brown fox'**
facciamola diventare **'The quick red fox'**

```php
<?php
$string = 'The quick brown fox';

echo str_replace('brown', 'red', $string);

?>
```

Stringhe: Validazione

- Le seguenti funzioni vengono spesso utilizzate per validare i dati in input:

- ctype_alnum(): controlla se tutti i caratteri della stringa sono caratteri alfanumerici
- ctype_alpha(): controlla se tutti i caratteri della stringa sono caratteri alafbetici
- ctype_digit(): controlla se tutti i caratteri della stringa sono caratteri numerici
- strip_tags(): consente di eliminare tutti I caratteri html o javascript

Esercizio 6:

- Creare una form che permetta l'inserimento di una frase

- Creare una frase che abbia le lettere alternate
  (una maiuscola e una minuscola)
- Si deve controllare se è una lettera oppure no

- Stampare
  - La frase creata
  - Il numero di sostituzioni eseguite
  - Il numero di "non-lettere" che sono stati incontrati

Esercizio 3: Soluzione

```php
<?php
$frase = " A Londra piove tutti i giorni ";
$lunghezza = strlen ( $frase );
$do_next = true;
$nonLettere = 0;
$sostituzioni = 0;
for($i = 0; $i < $lunghezza; $i ++) {
    if (ctype_alpha ( $frase [$i] )) {
        if ($do_next === true) {
            $frase [$i] = strtoupper ( $frase [$i] );
            $do_next = false;
            $sostituzioni ++;
        } else {
            $do_next = true;
        }
    } else {
        $nonLettere ++;
    }
}
echo "Frase: $frase<br>";
echo "Non-lettere: $nonLettere<br>";
echo "Sostituzioni: $sostituzioni";
```

```
Frase: A LOnDrA pIoVe TuTtI i GiOrNi
Non-lettere: 7
Sostituzioni: 12
```

QUESTION TIME ?

Nome_____ Cognome_____ Data_____

ARRIVEDERCI

TITOLO