VARIABILI E FUNZIONI

PROF. SANTANIELLO

COSTANTI E VARIABILI D'AMBIENTE

In php le costanti (a differenza delle variabili) devono essere sempre dichiarate.

Le costanti sono create con l'istruzione **define** e sono formate da un identificatore (nome) e da un valore

```
<?php
    define ("PIGRECO", 3.14);
    echo PIGRECO;
?>
```

Variabili d'ambiente	Descrizione
$_GET	Contiene i campi HTTP ricevuti in GET
$_POST	Contiene i campi HTTP ricevuti in POST
$_FILES	Consente di effettuare l'upload di variabili
$_SERVER	Contiene informazioni riguardanti il server
$_COOKIE	Rappresenta i cookies HTTP
$_SESSION	Rappresenta le variabili di sessione
$_GLOBALS	Rende le variabili globali

VARIABILI D'AMBIENTE

In php esistono anche particolari strutture dati predefinite, chiamate **variabili d'ambiente** o **superglobals**, che consentono di agevolare la programmazione coinvolgendo il server.

VISIBILITA' DELLE VARIABILI

La variabile $b possiede uno scope limitato alla funzione prova_scope() in cui è stata definita. Quindi al di fuori di questa funzione non esiste

```
<!DOCTYPE html>
<body>
<?php

    function prova_scope()
     {
        $a=true;
        if($a)
          {
             $b="Sono una variabile";
          }
    }
     prova_scope();
    if(isset($b)){ //verifica se una var. è stata definita
        echo $b;
     }
    else{
        echo "variabile non visibile";
    }
?>
</body>
</html>
```

COME RENDERE VISIBILI LE VARIABILI

```php
<!DOCTYPE html>
<body>
<?php
    function prova_scope(){
        $a=true;
        if($a){
            $b="Sono una variabile";
            return $b;
        }
    }
    $b = prova_scope();
    if(isset($b)){
        echo $b;
    }
    else
    {
        echo "variabile non visibile";
    }
?>
</body>
</html>
```

```php
<!DOCTYPE html>
<body>
<?php
    function prova_scope(){
        $a=true;
        if($a){
            $GLOBALS['b'] ="Sono una variabile";
        }
    }
     prova_scope();
    if(isset($GLOBALS['b'])){
        echo $GLOBALS['b'];
    }
    else
    {
        echo "variabile non visibile";
    }
?>
</body>
</html>
```

Prof. Santaniello Assunta

LE FUNZIONI UTENTE

Le funzioni possono essere richiamate prima o dopo il blocco di definizione. Inoltre, le funzioni non possiedono alcun tipo di ritorno, il tipo viene infatti adeguato secondo la tecnica della conversione implicita.

PHP ammette tre tipologie di passaggio dei parametri alle funzioni:

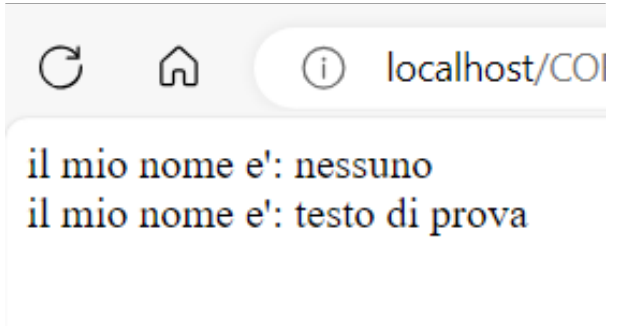
- Per default;
- Per valore;
- Per riferimento;

PASSAGGIO PARAMETRI PER DEFAULT

```php
<!DOCTYPE html>
<body>
<?php
    function chiama($nome = "nessuno")
    {
        echo "il mio nome e': ".$nome;
        echo "<br>";
        return;
    }

    chiama();
    chiama("testo di prova");
?>
</body>
</html>
```

La funzione indicata stampa il parametro passato, se tuttavia non riceve alcun valore stampa un valore predefinito



Nel passaggio parametri **per valore**, la funzione non modifica i parametri passati e all'uscita i parametri non vengono restituiti modificati al segmento di codice chiamante.

Prof. Santaniello Assunta

PASSAGGIO PARAMETRI PER VALORE

Nel passaggio parametri **per valore**, la funzione non modifica i parametri passati e all'uscita i parametri non vengono restituiti modificati al segmento di codice chiamante.

Il passaggio parametri **per riferimento (indirizzo)** consente invece a una funzione di restituire gli argomenti modificati. Per passare un argomento per riferimento a una funzione, si deve anteporre il simbolo di **ampersand** o **&** al nome dell'argomento nella definizione della funzione. In questo caso, la modifica dei parametri all'interno della funzione influisce sui parametri attuali che sono stati passati.

PASSAGGIO PARAMETRI PER VALORE E PER RIFERIMENTO: ESEMPIO PRATICO

```php
<!DOCTYPE html>
<head>
<?php
    function passaValore($numero1, $numero2)
    {
        $numero1=$numero1+100;
        $numero2=$numero2+100;
    }
    function passaIndirizzo (&$numero1,&$numero2)
    {
        $numero1=$numero1+100;
        $numero2=$numero2+100;
    }
?>
</head>
<body>
    <?php
        $a=20;
        $b=30;
        echo"<hr>Prima della chiamata a passaValore<br>";
        echo "$a, $b <br>";
        passaValore($a,$b);
        echo"Dopo la chiamata passaValore <br>";
        echo "$a, $b <br>";
        echo "<hr> Prima della chiamata a passaIndirizzo<br>";
        echo "$a, $b <br>";
        passaIndirizzo($a,$b);
        echo "Dopo la chiamata passaIndirizzo <br>";
        echo "$a,$b<br>";
    ?>
</body>
</html>
```

Prima della chiamata a passaValore
20, 30
Dopo la chiamata passaValore
20, 30

Prima della chiamata a passaIndirizzo
20, 30
Dopo la chiamata passaIndirizzo
120,130

Nel seguente codice possiamo notare che, mentre dopo la chiamata della funzione passaValore() i valori vengono restituiti invariati, dopo la chiamata della funzione passa indirizzo() i valori vengono invece restituiti incrementati di 100.

FACCIAMO UN PO' DI PRATICA

1. Crea una funzione che, passati due numeri e la relativa operazione da effettuare (+-*/).
   Se il parametro vale 0, restituisce «errore», altrimenti il risultato;

2. Crea una funzione che sottolinei una frase passata come parametro

3. Crea una funzione che riceva la base e l'esponente come argomenti e restituisca la potenza.