

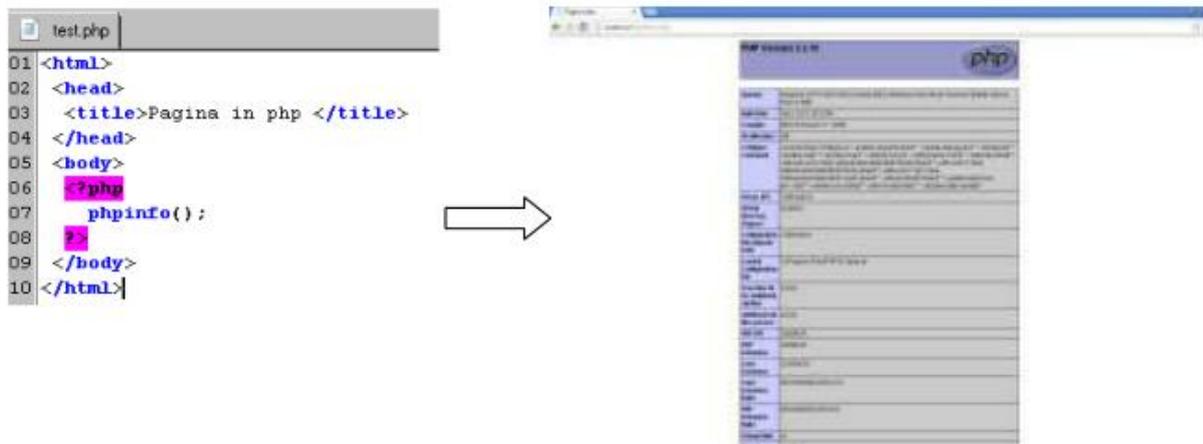
PHP (Hypertext Preprocessor)

Il linguaggio php permette di creare pagine HTML dinamicamente, cioè in risposta alle richieste dell'utente di tipo client e/o server. La sua sintassi è di derivazione C++, Java e Perl. Esso può essere usato da diversi sistemi operativi (LINUX, WINDOWS, UNIX, MAC OS X, RISC OS) ed inoltre supporta la maggior parte dei server web (programma in esecuzione che fornisce le pagine web a tutti i client che ne facciano richiesta) esistenti.

Php è un linguaggio di scripting a lato server (scripting server side) formato da istruzioni che vengono elaborate dal server che generano HTML come risultato dell'elaborazione. Il file HTML viene inviato al client browser come risposta http. Mentre una pagina formata dalle sole istruzioni HTML è interpretata dal browser e le istruzioni di HTML non sono elaborate dal server web, ma solo inviate al browser client come parte della risposta http ed è poi elaborato dal client che ne visualizza il risultato.

Dopo questa introduzione vediamo per prima cosa come il php interagisce con HTML. All'interno della sezione BODY di HTML per scrivere istruzioni in php dette script si apre il seguente tag `<?php` e per la chiusura `?>`.

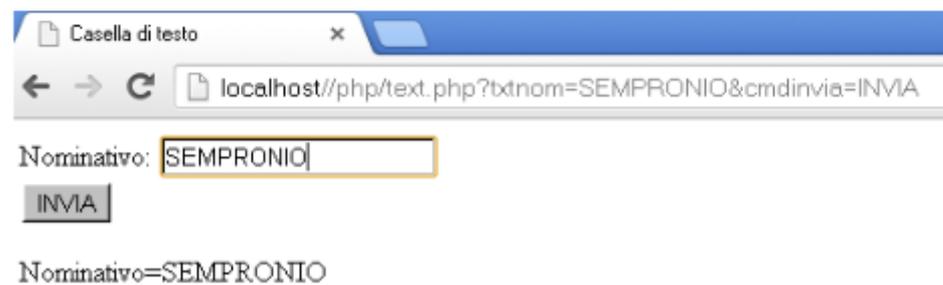
Un esempio:



Vediamo come i valori attribuiti agli oggetti (text, textarea, casella combinata, radio e checkbox) di HTML possono essere riconosciuti ed eseguiti da php.

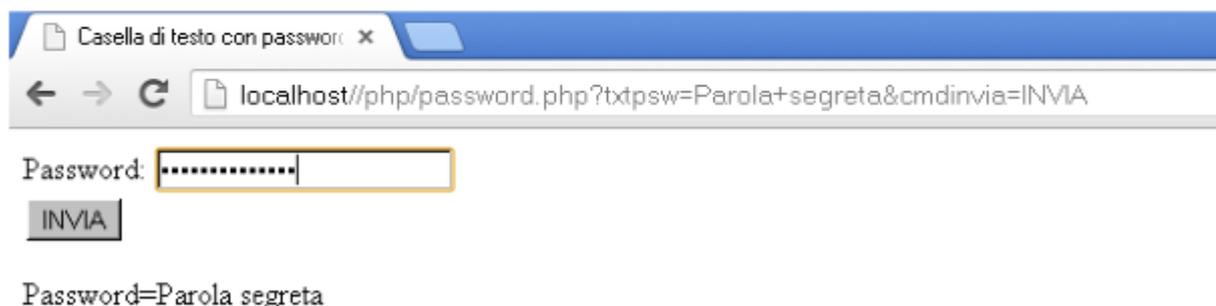
CASELLA DI TESTO

```
text.php
01 <html>
02 <head>
03   <title>Casella di testo</title>
04 </head>
05 <body>
06   <form>
07     Nominativo:
08     <input type=text name=txtnom size=20>
09     <br>
10     <input type=submit name=cmdinvia value=INVIA>
11   </form>
12   <?php
13     echo "Nominativo=" . $_REQUEST["txtnom"];
14   >
15 </body>
16 </html>
```



CASELLA DI TESTO con password

```
password.php
01 <html>
02 <head>
03   <title>Casella di testo con password</title>
04 </head>
05 <body>
06   <form>
07     Password:
08     <input type=password name=txtpsw size=20>
09     <br>
10     <input type=submit name=cmdinvia value=INVIA>
11   </form>
12   <?php
13     echo "Password=" . $_REQUEST["txtpsw"];
14   >
15 </body>
16 </html>
```



L'AREA DI TESTO

```
01 <html>
02 <head>
03   <title>Area di testo</title>
04 </head>
05 <body>
06   <form>
07     Suggestimenti
08     <textarea name=txtsugg rows=2 cols=40>
09   </textarea>
10   <br>
11   <input type=submit name=cmdinvia value=INVIA>
12 </form>
13 <?php
14   echo "Suggestimenti:" . $_REQUEST["txtsugg"];
15 >
16 </body>
17 </html>
```

Area di testo x

localhost/php/textarea.php?txtsugg=In+quest%27area+posso+scrivere+quello+che+voglio%21&cmdinvia=INVIA

Suggestimenti

In quest'area posso scrivere quello che voglio!

INVIA

Suggestimenti:In quest'area posso scrivere quello che voglio!

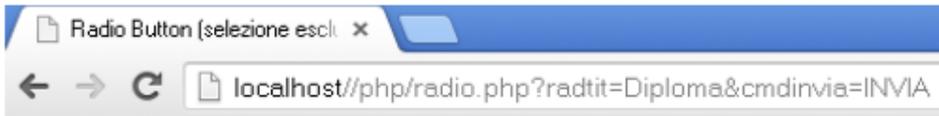
CASELLA COMBINATA

```
01 <html>
02 <head>
03   <title>Casella combinata</title>
04 </head>
05 <body>
06   <form>
07     Impiego
08     <select name=cboimp multiple size=3>
09       <option>Dirigente</option>
10       <option>Impiegato</option>
11       <option>Operaio</option>
12       <option>Libero professionista</option>
13       <option>Imprenditore</option>
14       <option>Studente</option>
15       <option>Disoccupato</option>
16     </select>
17     <br>
18     <input type=submit name=cmdinvia value=INVIA>
19   </form>
20   <?php
21     echo "Impiego:" . $_REQUEST["cboimp"];
22   ?>
23 </body>
24 </html>
```



RADIO BUTTON (scelta esclusiva)

```
radio.php
01 <html>
02 <head>
03   <title>Radio Button (selezione esclusiva)</title>
04 </head>
05 <body>
06   <form>
07     Titolo di studio<br>
08     <input type=radio name=radtit value=Nessuno>Nessuno<br>
09     <input type=radio name=radtit value="Licenza Elementare">Licenza Elementare<br>
10     <input type=radio name=radtit value="Licenza Media">Licenza Media<br>
11     <input type=radio name=radtit value=Diploma checked>Diploma<br>
12     <input type=radio name=radtit value="Laurea I Livello">Laurea I Livello<br>
13     <input type=radio name=radtit value="Laurea II Livello">Laurea II livello<br>
14     <br>
15     <input type=submit name=cmdinvia value=INVIA>
16   </form>
17   <?php
18     echo "Titolo di studio=" . $_REQUEST["radtit"];
19   ?>
20 </body>
21 </html>
```



Titolo di studio

- Nessuno
- Licenza Elementare
- Licenza Media
- Diploma
- Laurea I Livello
- Laurea II livello

INVIA

Titolo di studio=Diploma

CHECK BOX (scelta multipla)

```
checkbox.php
01 <html>
02 <head>
03   <title>CheckBox(selezione multipla)</title>
04 </head>
05 <body>
06   <form>
07     Genere Film<br>
08     <input type=checkbox name=chk1 value=Thriller>Thriller<br>
09     <input type=checkbox name=chk2 value=Horror>Horror<br>
10     <input type=checkbox name=chk3 value=Avventura>Avventura<br>
11     <input type=checkbox name=chk4 value=Comico>Comico<br>
12     <input type=checkbox name=chk5 value=Commedia>Commedia<br>
13     <input type=checkbox name=chk6 value=Altro>Altro<br>
14     <br>
15     <input type=submit name=cmdinvia value=INVIA>
16   </form>
17   <?php
18     echo "Genere scelto:."<br>;
19     echo $_REQUEST["chk1"]."<br>";
20     echo $_REQUEST["chk2"]."<br>";
21     echo $_REQUEST["chk3"]."<br>";
22     echo $_REQUEST["chk4"]."<br>";
23     echo $_REQUEST["chk5"]."<br>";
24     echo $_REQUEST["chk6"]."<br>";
25   ?>
26 </body>
27 </html>
```



Genere Film
 Thriller
 Horror
 Avventura
 Comico
 Commedia
 Altro

INVIA

Genere scelto:
Thriller
Horror
Avventura

LA SINTASSI

Come si è visto all'inizio di questa dispensa, per scrivere istruzioni di php è necessario utilizzare il tag di apertura `<?php` e di chiusura `?>`.

Inoltre negli esempi precedenti, la funzione echo è molto simile alla funzione msgbox di Visual Basic e la concatenazione è rappresentata dal punto in luogo di &. Un'altra osservazione molto importante è riferita alla possibilità di "incapsulare" tag di HTML nella funzione echo come nell'esempio relativo alla checkbox.

La funzione `$_REQUEST` rappresenta il passaggio del contenuto delle variabili da HTML a PHP.

Le variabili in php sono denominate facendo precedere il nome dal simbolo \$.

I TIPI DI DATI

Premesso che le variabili non vengono dichiarate, ma si adattano a ciò che è contenuto in esse, si distinguono in booleano, intero, decimale e stringa.

Una cosa fondamentale è che se una variabile assume il contenuto di un determinato tipo, la natura della variabile non potrà più cambiare.

Booleano

I valori che può assumere una variabile booleana sono false o true:

```
$varbool=false oppure $varbool=true
```

Intero

Il valore di un intero può essere espresso in diverse basi (decimale, ottale ed esadecimale):

```
$varint=23 (numero intero decimale)
```

```
$varint=023 (numero intero ottale)
```

```
$varint=0x23 (numero intero esadecimale)
```

I valori che può assumere un numero intero in base 10 vanno da -2.147483648 a +2.147.483.647, se si dovesse uscire da questo range, il tipo di variabile diventerebbe decimale (float).

Decimale

Il decimale è un numero espresso in virgola mobile e la virgola è rappresentata dal punto.

Una variabile decimale può essere espressa anche in formato esponenziale.

```
$vardec=3.5
```

```
$vardec=3.5e2 (3,5 x 102)
```

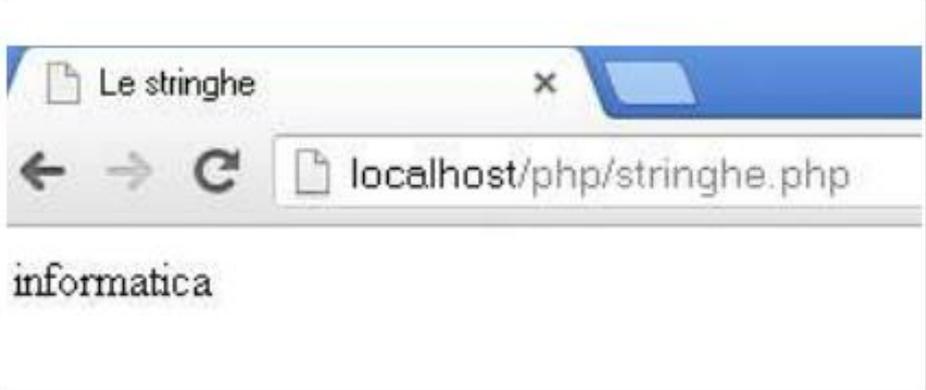
```
$vardec=5E3 (5 x 103)
```

Stringa

Una stringa è un insieme di caratteri che può essere inizializzata con un apice singolo o con apice doppio.

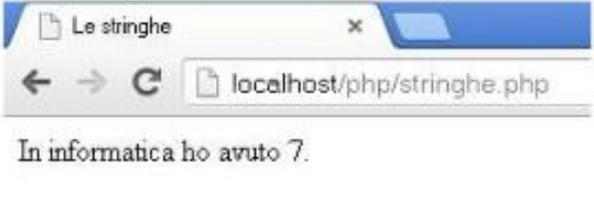
È consigliabile l'apice doppio, in quanto con l'apice singolo non è possibile introdurre caratteri di escape (\n (simbolo LineFeed), \r (simbolo CarriageReturn), \t (simbolo HorizontalTab), \\ (simbolo BackSlash), \\$ (simbolo \$), \" (simbolo doppio apice) e _ (simbolo euro)).

Esempio:

<pre><?php \$a="info"; \$a=\$a."rmatrica"; echo \$a; ?></pre>	
---	--

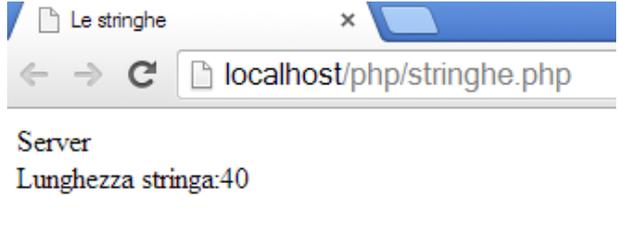
Come visto dall'esempio il punto rappresenta la concatenazione delle stringhe. È possibile concatenare stringhe con variabili numeriche, in quanto il numero viene convertito in stringa.

Esempio:

<pre><?php \$a="In info"; \$b="rmatrica "; \$c=" ho avuto "; \$d=7; echo \$a.\$b.\$c.\$d."."; ?></pre>	
--	--

In php le parentesi graffe servono ad indicizzare una stringa tenendo presente che il primo carattere occupa la posizione 0. Mentre per conoscere la lunghezza della stringa si utilizza la funzione strlen.

Esempio:

<pre><?php \$a="Sistemi, Server, DB e Programmazione Web"; echo \$a{9}.\$a{10}.\$a{11}.\$a{12}.\$a{13}.\$a{14}; echo "
"; \$lunghezza=strlen(\$a); echo "Lunghezza stringa:" .\$lunghezza; ?> </body> </html></pre>	
--	--

LE FUNZIONI DI ESISTENZA

Queste funzioni permettono di verificare l'esistenza della variabile(`iisset(variabile)`), l'eliminazione definitiva(`unset(variabile)`) e la verifica se essa è vuota(`empty(variabile)`).

La funzione `isset()` è importante, in quanto, essendo il php server side, è necessario che il server sappia se il client ha scritto qualcosa o meno nella variabile.

La funzione `unset()` serve per eliminare una variabile in modo che sia liberata lo spazio di memoria.

Infine la funzione `empty()` applicata alla variabile restituisce `true` se la variabile è vuota e `NULL` altrimenti.

LE FUNZIONI DI VERIFICA TIPO

Queste funzioni permettono di verificare se una variabile contiene dati e di che tipo. Esse restituiscono un valore booleano oppure il tipo.

Le funzioni sono: `settype()`, `gettype()` e `var_dump()`.

COSTANTI E VARIABILI

Una costante è definita in php come: `define("nome","valore");`

Una variabile ha un nome, di lunghezza teoricamente indefinita e obbligatoriamente preceduta dal simbolo `$`. L'operatore di assegnazione è rappresentato dal simbolo `=`.

Le variabili d'ambiente sono visibili attraverso l'istruzione `phpinfo()`.

GLI OPERATORI

Gli operatori aritmetici sono: `+`, `-`, `*`, `/` e `%`.

<code>\$a + \$b</code>	Somma	} di \$a e \$b
<code>\$a - \$b</code>	Differenza	
<code>\$a * \$b</code>	Prodotto	
<code>\$a / \$b</code>	Quoziente	
<code>\$a % \$b</code>	Resto della divisione	

Gli operatori di confronto sono: `==`, `!=`, `<>`, `<`, `<=`, `>`, `>=`.

Uguaglianza	<code>\$a==\$b</code>
Disuguaglianza	<code>\$a!=\$b</code> oppure <code>\$a<>\$b</code>
Minore	<code>\$a<\$b</code>
Minore uguale	<code>\$a<=\$b</code>
Maggiore	<code>\$a>\$b</code>
Maggiore uguale	<code>\$a>=\$b</code>

Gli operatori logici sono: `and`, `or`, `xor`, `!`, `&&` e `||`.

<code>and</code>	<code>\$a and \$b</code> oppure <code>\$a && \$b</code>
<code>or</code>	<code>\$a or \$b</code>
<code>not</code>	<code>!\$a</code>
<code>xor</code>	<code>\$a xor \$b</code>

In questa parte del PHP vedremo la sintassi delle strutture di controllo quali la selezione, la sequenza e l'iterazione seguita da alcuni esempi.

LA SELEZIONE

Il costrutto di selezione permette di effettuare una scelta basata su una o più condizioni.

Sintassi:

```
if (condizione)
{
    elenco delle istruzioni da eseguire per condizione vera;
}
else
{
    elenco delle istruzioni da eseguire per condizione falsa;
}
```

In alternativa possono essere omesse le parentesi graffe con la seguente sintassi:

```
if (condizione):
    elenco delle istruzioni da eseguire per condizione vera;
else:
    elenco delle istruzioni da eseguire per condizione falsa;
endif;
```

ESEMPIO:

Supponendo che la bolletta del telefono venga calcolata nel seguente modo:

- minimo euro 15,00 per i primi 100 scatti;
- più euro 0,05 per gli scatti successivi.

scrivete un programma che calcoli l'importo da pagare, avendo come input il numero di scatti.

```
<html>
<head>
  <title>La struttura: selezione</title>
</head>
<body>
  <form>
    Numero scatti:
    <input type=text name=txtscatti size=20>
    <br>
    <input type=submit name=cmdinvia value=INVIA>
  </form>
  <?php
  $scatti=$_REQUEST["txtscatti"];
  echo "Numero scatti=" . $scatti . "<br>";
  if ($scatti<=100)
    $importo=15.00;
  else
    $importo=15.00+($scatti-100)*0.05;
  echo "Importo da versare: " . $importo . "<br>";
  ?>
</body>
</html>
```



Numero scatti:

INVIA

Numero scatti=116
Importo da versare: 15.8

Come si può notare sono state omesse le parentesi graffe, in quanto, il numero delle istruzioni da eseguire è fissato ad uno sia con condizione vera e sia con condizione falsa.

LA SEQUENZA E L'ITERAZIONE

L'iterazione con controllo in testa(while)

Il ciclo while è la struttura iterativa che permette di eseguire le istruzioni nel caso in cui la condizione sia vera.

Sintassi:

```
while (condizione)
{
    elenco delle istruzioni da eseguire;
}
```

In alternativa possono essere omesse le parentesi graffe con la seguente sintassi:

```
while (condizione):
    elenco delle istruzioni da eseguire;
endwhile;
```

ESEMPIO:

Calcolare la potenza di un numero dopo aver fornito in input il valore della base e dell'esponente.

```
<html>
<head>
  <title>Potenza di un numero</title>
</head>
<body>
  <form>
    BASE:
    <input type="text" name="txtbase" size="5"><br>
    ESPONENTE:
    <input type="text" name="txttespo" size="5"><br>
    <input type="submit" name="cmdcalcola" value="CALCOLA">
  </form>
  <?php
    $base=$_REQUEST["txtbase"];
    $espo=$_REQUEST["txttespo"];
    echo "Base=".$base."<br>";
    echo "Esponente=".$espo."<br>";
    $i=0;
    if ($base==0 and $espo==0)
      $potenza="Forma indeternimata";
    if ($base>0 and $espo==0)
      $potenza=1;
    if ($base==0 and $espo>0)
      $potenza=0;
    if ($base>0 and $espo>0)
    {
      $potenza=1;
      while ($i<$espo)
      {
        $potenza=$potenza*$base;
        $i=$i+1;
      }
    }
    echo "Potenza=".$potenza;
  ?>
</body>
</html>
```



BASE:
ESPONENTE:

Base=5
Esponente=3
Potenza=125

L'iterazione con controllo in coda (do-while)

Il ciclo do-while è la struttura iterativa che permette di eseguire le istruzioni e successivamente valutare la condizione, nel caso sia vera le istruzioni verranno rieseguite.

Sintassi:

```
do
{
    elenco delle istruzioni da eseguire;
}
while (condizione);
```

ESEMPIO:

Calcolare la potenza di un numero dopo aver fornito in input il valore della base e dell'esponente.

```
<html>
<head>
  <title>Potenza di un numero</title>
</head>
<body>
  <form>
    BASE:
    <input type="text" name="txtbase" size="5"><br>
    ESPONENTE:
    <input type="text" name="txtespo" size="5"><br>
    <input type="submit" name="cmdcalcola" value="CALCOLA">
  </form>
  <?php
    $base=$_REQUEST["txtbase"];
    $espo=$_REQUEST["txtespo"];
    echo "Base=".$base."<br>";
    echo "Esponente=".$espo."<br>";
    if ($base==0 and $espo==0)
      $potenza="Forma indeterminata";
    if ($base>0 and $espo==0)
      $potenza=1;
    if ($base==0 and $espo>0)
      $potenza=0;
    if ($base>0 and $espo>0)
    {
      $potenza=1;
      $i=0;
      do
      {
        $potenza=$potenza*$base;
        $i=$i+1;
      }
      while ($i<$espo);
    }
    echo "Potenza=".$potenza;
  ?>
</body>
</html>
```



BASE:
ESPONENTE:

Base=3
Esponente=4
Potenza=81

Il ciclo enumerativo for

Il ciclo for esegue determinate istruzioni per un certo numero di volte.

Sintassi:

```
for (istruzione di partenza, condizione di esecuzione delle istruzioni, passo)
{
    elenco delle istruzioni da eseguire;
}
```

L'**istruzione di partenza** viene eseguita all'inizio del ciclo una sola volta. La **condizione di esecuzione** viene interrogata ad ogni ciclo, se risulta essere vera le istruzioni verranno eseguite, altrimenti terminerà l'iterazione. Infine il **passo** rappresenta il conteggio delle iterazioni.

ESEMPIO:

Calcolare la potenza di un numero dopo aver fornito in input il valore della base e dell'esponente.

```
<html>
<head>
  <title>Potenza di un numero</title>
</head>
<body>
  <form>
    BASE:
    <input type="text" name="txtbase" size="5"><br>
    ESPONENTE:
    <input type="text" name="txtespo" size="5"><br>
    <input type="submit" name="cmdcalcola" value="CALCOLA">
  </form>
  <?php
    $base=$_REQUEST["txtbase"];
    $espo=$_REQUEST["txtespo"];
    echo "Base=".$base."<br>";
    echo "Esponente=".$espo."<br>";
    if ($base==0 and $espo==0)
      $potenza="Forma indeterminata";
    if ($base>0 and $espo==0)
      $potenza=1;
    if ($base==0 and $espo>0)
      $potenza=0;
    if ($base>0 and $espo>0)
    {
      $potenza=1;
      for ($i=0;$i<$espo;$i=$i+1)
      {
        $potenza=$potenza*$base;
      }
    }
    echo "Potenza=".$potenza;
  ?>
</body>
</html>
```



BASE:
ESPONENTE:

Base=2
Esponente=4
Potenza=16

La scelta multipla con switch

Lo switch rappresenta l'alternativa alla struttura di selezione nidificata a patto che la condizione sia valutata su un insieme di valori associati ad una variabile che può essere di tipo numerico o alfanumerico.

Sintassi:

switch (\$variabile)

```
{
  case valore 1:
  {
    elenco delle istruzioni da eseguire;
    break;
  }
  ...
  case valore n:
  {
    elenco delle istruzioni da eseguire;
    break;
  }
  default:
  {
    elenco delle istruzioni da eseguire;
    break;
  }
}
```

ESEMPIO:

Un'azienda distributrice di metano applica le tariffe secondo la seguente tabella:

TIPOLOGIA	COSTO
A1	0,15 €/mc
A2	0,20 €/mc
A3	0,22 €/mc

Dato in input la tipologia ed il consumo, si visualizzi l'importo da pagare.

```
<html>
<head>
<title>La scelta multipla switch</title>
</head>
<body>
<form>
<fieldset style=width:200px;><legend>Tipologia Utenza</legend>
<input type=radio name=radtipo value=A1>Residenti<br>
<input type=radio name=radtipo value=A2>Non residenti<br>
<input type=radio name=radtipo value=A3>Aziende<br>
</fieldset>
CONSUMO:
<input type="text" name="txtcons" size="5"><br>
<input type="submit" name="cmdcalcola" value="CALCOLA">
</form>
<?php
$tipo=$_REQUEST["radtipo"]; $cons=$_REQUEST["txtcons"];
echo "Tipologia=".$tipo."<br>";echo "Consumo=".$cons."<br>";
switch($tipo)
{
case "A1":
{$importo=$cons*0.15;
echo "Importo da pagare:".$importo;
break;}
case "A2":
{$importo=$cons*0.20;
echo "Importo da pagare:".$importo;
break;}
case "A3":
{$importo=$cons*0.22;
echo "Importo da pagare:".$importo;
break;}
default:
{echo "Scelta errata-Tipologia inesistente-Valori ammessi A1,A2,A3.";
break;}
}
?>
</body>
</html>
```



Tipologia Utanza

Residenti
 Non residenti
 Aziende

CONSUMO:

Tipologia=A1
Consumo=111
Importo da pagare:16.65

In questa parte del PHP vedremo la sintassi delle funzioni che hanno lo scopo di rendere il programma più snello ed esente da ripetizioni di codice.

Nel php abbiamo le funzioni definite dall'utente e funzioni predefinite.

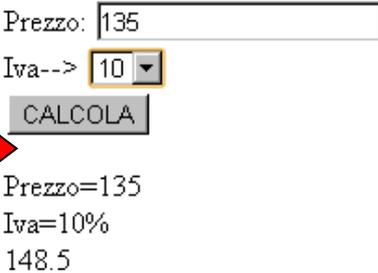
Le **funzioni definite dall'utente** sono create dal programmatore all'interno dello script php e la sintassi è la seguente:

```
function nome (elenco dei parametri separati da virgola)
{
elenco delle istruzioni da eseguire;
return valore di ritorno;
}
```

L'istruzione return non è obbligatoria: una funzione può anche non restituire alcun valore.

Esempio:

```
<html>
<head>
  <title>Le funzioni </title>
</head>
<body>
  <form>
    Prezzo:
    <input type="text" name="txtprz"><br>
    Iva-->
    <select name="cboiva">
      <option>4</option>
      <option>10</option>
      <option>21</option>
    </select>
    <br>
    <input type="submit" name="cmdcalc" value="CALCOLA">
  </form>
  <?php
  function prezzoivato($prezzo,$iva)
  {
    echo $prezzo + ($prezzo*$iva/100);
    return;
  }
  $prz=$_REQUEST["txtprz"];
  $iva=$_REQUEST["cboiva"];
  echo "Prezzo=".$prz."<br>";
  echo "Iva=".$iva."%<br>";
  echo prezzoivato($prz,$iva);
  ?>
</body>
</html>
```



Passaggio dei parametri per valore

Il passaggio dei parametri avviene per valore se all'interno della funzione le variabili definite come parametri non subiscono modifiche. Qualora dovessero subire modifiche le variazioni non avranno effetto.

Nell'esempio precedente i parametri **\$prezzo** e **\$iva** non subiscono modifiche e dunque gli stessi vengono passati per valore allo script php.

Passaggio dei parametri per riferimento

Analogamente quando i parametri subiscono variazioni all'interno della **function**, questi potranno essere passati per riferimento. I nomi dei parametri vengono preceduti dal simbolo &.

Esempio

```

<html>
<head>
<title>Le funzioni </title>
</head>
<body>
<form>
Prezzo:
<input type="text" name="txtprz"><br>
Iva-->
<select name="cboiva">
<option>4</option>
<option>10</option>
<option>21</option>
</select>
<br>
<input type="submit" name="cmdcalc" value="CALCOLA">
</form>
<?php
function prezzoivato($prezzo,$iva)
{
    $prezzo = $prezzo + ($prezzo*$iva/100);
    return;
}
$prz=$_REQUEST["txtprz"];
$iva=$_REQUEST["cboiva"];
echo "Prezzo=".$prz."<br>";
echo "Iva=".$iva."%."<br>";
prezzoivato($prz,$iva);
echo "<br>";
echo "Prezzo nuovo=" . $prz . "<br>";
?>
</body>
</html>

```

Prezzo:

Iva-->

Prezzo=750
Iva=10%

Prezzo nuovo=825

In php esistono le **funzioni predefinite** che rispondono a particolari esigenze del programmatore.

In tabella vengono riportate le funzioni più comuni:

Funzioni Matematiche	
abs(n)	Valore assoluto di n
bindec()	Converte da binario a decimale
ceil(numero)	Arrotonda un numero decimale all'intero superiore
floor(numero)	Arrotonda un numero decimale all'intero inferiore
round(numero,n)	Arrotonda un numero non intero ad n posizioni decimali
sin(),cos() e tan()	Seno, coseno e tangente dell'argomento in radianti
exp()	Funzione esponenziale dell'argomento
is_nan()	Indica se l'argomento non è un numero
log10()	Logaritmo decimale dell'argomento
log()	Logaritmo naturale dell'argomento
pi()	Valore di pi greco
pow(b,e)	Potenza di un numero
rand(min,max)	Genera un valore casuale tra min e max
sqrt()	Radice quadrata
srand()	Inizializza il generatore di numeri casuali

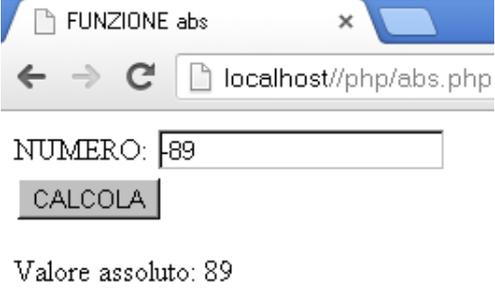
Funzioni Stringa	
chr()	Restituisce il carattere corrispondente al codice ASCII
ord()	Restituisce il codice ASCII corrispondente al carattere
explode()	Divide una stringa in molte sottostringhe
trim()	Elimina eventuali spazi a destra e a sinistra della stringa
ltrim()	Elimina eventuali spazi a sinistra della stringa
rtrim()	Elimina eventuali spazi a destra della stringa
strlen()	Restituisce la lunghezza di una stringa
strpos()	Cerca la posizione della prima occorrenza della stringa
str_replace()	Trova e sostituisce una sottostringa in una stringa
strtolower()	Trasforma una stringa in minuscolo
strtoupper	Trasforma una stringa in maiuscolo
str(st1,st2,st3)	Sostituisce la stringa st2 nella st1 memorizzando nella str3
substr_count(st1,st2)	Restituisce il numero di volte che la st2 appare nella st1
substr()	Restituisce la parte intera di una stringa

ESEMPI SU ALCUNE FUNZIONI MATEMATICHE

```

<html>
<head>
  <title>FUNZIONE abs</title>
</head>
<body>
  <form>
    NUMERO:
    <input type="text" name="txtnum">
    <br>
    <input type="submit" name="cmdcalc" value="CALCOLA">
  </form>
  <?php
    $num=$_REQUEST["txtnum"];
    echo "Valore assoluto: ".abs($num);
  ?>
</body>
</html>

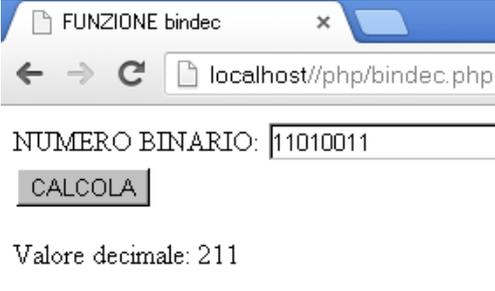
```



```

<html>
<head>
  <title>FUNZIONE bindec</title>
</head>
<body>
  <form>
    NUMERO BINARIO:
    <input type="text" name="txtnum">
    <br>
    <input type="submit" name="cmdcalc" value="CALCOLA">
  </form>
  <?php
    $num=$_REQUEST["txtnum"];
    echo "Valore decimale: ".bindec($num);
  ?>
</body>
</html>

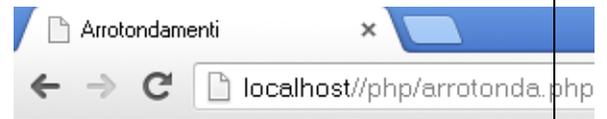
```



```

<html>
<head>
<title>Arrotondamenti</title>
</head>
<body>
<form>
NUMERO:
<input type="text" name="txtnum">
<br>
<input type="submit" name="cmdcalc" value="CALCOLA">
</form>
<?php
$num=$_REQUEST["txtnum"];
echo "Funzione ceil:".ceil($num)."<br>";
echo "Funzione floor:".floor($num)."<br>";
echo "Funzione round:".round($num,2);
?>
</body>
</html>

```



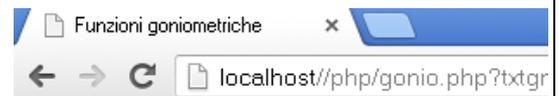
NUMERO:

Funzione ceil:4
 Funzione floor:3
 Funzione round:3.46

```

<html>
<head>
<title>Funzioni goniometriche</title>
</head>
<body>
<form>
Misura angolo in gradi:
<input type="text" name="txtgradi">
<br>
<input type="submit" name="cmdcalc" value="CALCOLA">
</form>
<?php
$gradi=$_REQUEST["txtgradi"];
$radianti=$gradi*pi(5)/180;
echo "Funzione seno:".round(sin($radianti),2)."<br>";
echo "Funzione coseno:".round(cos($radianti),2)."<br>";
echo "Funzione tangente:".round(tan($radianti),2)."<br>";
?>
</body>
</html>

```



Misura angolo in gradi:

Funzione seno:0.5
 Funzione coseno:0.87
 Funzione tangente:0.58

```

<html>
<head>
<title>Funzioni exp,log10,log,pow,sqrt</title>
</head>
<body>
<form>
Numero:
<input type="text" name="txtnum">
<br>
<input type="submit" name="cmdcalc" value="CALCOLA">
</form>
<?php
$num=$_REQUEST["txtnum"];
echo "exp(".$num.")=".exp($num)."<br>";
echo "log10(".$num.")=".log10($num)."<br>";
echo "log(".$num.")=".log($num)."<br>";
echo "pow(".$num.",2)=".pow($num,2)."<br>";
echo "sqrt(".$num.")=".sqrt($num);
?>
</body>
</html>

```

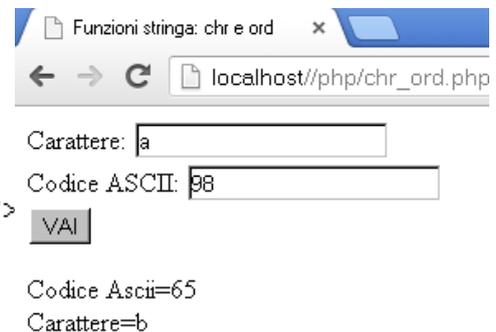


Numero:

exp(4)=54.598150033144
 log10(4)=0.60205999132796
 log(4)=1.3862943611199
 pow(4,2)=16
 sqrt(4)=2

ESEMPI SU ALCUNE FUNZIONI STRINGA

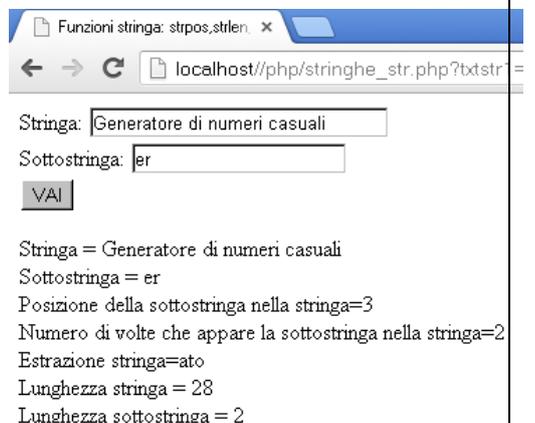
```
<html>
<head>
  <title>Funzioni stringa: chr e ord</title>
</head>
<body>
  <form>
    Carattere:
    <input type="text" name="txtcar">
    <br>
    Codice ASCII:
    <input type="text" name="txtascii">
    <br>
    <input type="submit" name="cmdvis" value="VAI">
  </form>
  <?php
  $car=$_REQUEST["txtcar"];
  $ascii=$_REQUEST["txtascii"];
  echo "Codice Ascii=".ord($car)."<br>";
  echo "Carattere=".chr($ascii);
  ?>
</body>
</html>
```



```
<html>
<head>
  <title>Funzioni stringa: strtolower e strtoupper</title>
</head>
<body>
  <form>
    Stringa
    <input type="text" name="txtstringa">
    <br>
    <input type="submit" name="cmdvis" value="VAI">
  </form>
  <?php
  $stringa=$_REQUEST["txtstringa"];
  echo "Stringa = ". $stringa."<br>";
  echo "STRINGA = ". strtoupper($stringa)."<br>";
  echo "stringa = ". strtolower($stringa)."<br>";
  ?>
</body>
</html>
```



```
<html>
<head>
  <title>Funzioni stringa: strpos, strlen, substr e substr_count</title>
</head>
<body>
  <form>
    Stringa:
    <input type="text" name="txtstr1" size=30>
    <br>
    Sottostringa:
    <input type="text" name="txtstr2" size=20>
    <br>
    <input type="submit" name="cmdvis" value="VAI">
  </form>
  <?php
  $stringa=$_REQUEST["txtstr1"];
  $sottostringa=$_REQUEST["txtstr2"];
  echo "Stringa = ". $stringa."<br>";
  echo "Sottostringa = ". $sottostringa."<br>";
  echo "Posizione della sottostringa nella stringa=".
  strpos($stringa,$sottostringa)."<br>";
  echo "Numero di volte che appare la sottostringa nella stringa=".
  substr_count($stringa,$sottostringa)."<br>";
  echo "Estrazione stringa=".substr($stringa, 5,3)."<br>";
  echo "Lunghezza stringa = ". strlen($stringa)."<br>";
  echo "Lunghezza sottostringa = ". strlen($sottostringa)."<br>";
  ?>
</body>
</html>
```



In questa parte del PHP vedremo l'utilizzo degli array e relativa gestione attraverso alcune funzioni predefinite.

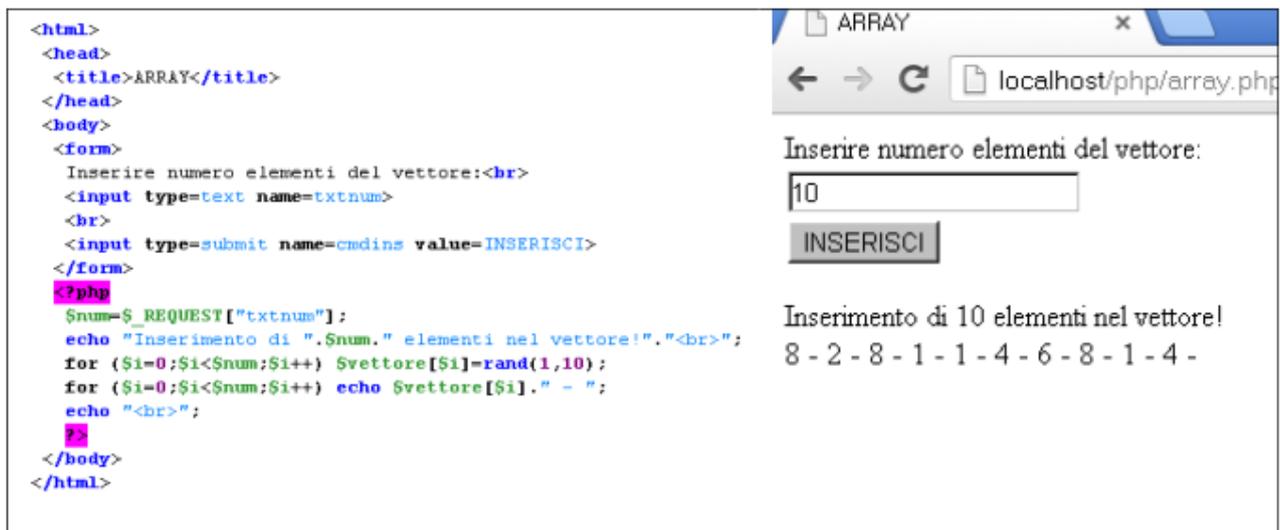
GLI ARRAY

L'array è una struttura di dati omogenei. Per definirlo è necessario specificare il nome, la dimensione e il tipo. La differenza con una variabile elementare sta nel fatto che dopo il nome vi sono delle parentesi quadre.

A differenza dei linguaggi di programmazione quali il VisualBasic, il C++, ecc., non viene specificato il numero di elementi dell'array (es. `$voti[10]` o `$voti[]` rappresentano degli errori).

Per definire un vettore è necessario inizializzare almeno un elemento (es. `$voti[4]=8`).

Supponiamo di voler caricare un vettore di numeri generati dalla funzione random come nell'esempio seguente:



The screenshot shows a web browser window titled 'ARRAY' with the URL 'localhost/php/array.php'. The page contains a form with the text 'Inserire numero elementi del vettore:' followed by a text input field containing '10' and a submit button labeled 'INSERISCI'. Below the form, the output of the PHP script is displayed: 'Inserimento di 10 elementi nel vettore!' followed by the sequence of numbers '8 - 2 - 8 - 1 - 1 - 4 - 6 - 8 - 1 - 4 -'.

```
<html>
<head>
  <title>ARRAY</title>
</head>
<body>
  <form>
    Inserire numero elementi del vettore:<br>
    <input type="text" name="txtnum">
    <br>
    <input type="submit" name="cmdins" value="INSERISCI">
  </form>
  <?php
    $num=$_REQUEST["txtnum"];
    echo "Inserimento di ".$num." elementi nel vettore!";<br>";
    for ($i=0;$i<$num;$i++) $vettore[$i]=rand(1,10);
    for ($i=0;$i<$num;$i++) echo $vettore[$i]." - ";
    echo "<br>";
  <?>
</body>
</html>
```

Le funzioni key(), current(), next() e prev()

Le funzioni key() e current() forniscono informazioni sulla situazione in termini di tracciabilità rispettivamente del valore dell'indice e del contenuto. Mentre le funzioni next() e prev() spostano l'indice rispettivamente in avanti ed indietro.

Un esempio chiarirà meglio le definizioni date:

```

<html>
<head>
<title>Le funzioni key, current, next e prev</title>
</head>
<body>
<?php
echo "Inserimento di 5 elementi nel vettore!."<br>";
for ($i=0;$i<5;$i++) $vettore[$i]=rand(10,50);
for ($i=0;$i<5;$i++) echo $vettore[$i]. " - ";
echo "<br>";
echo "Valore indice vettore=". key($vettore)."<br>";
echo "Valore contenuto vettore=". current($vettore)."<br>";
echo "<br>";
echo "Elemento successivo."<br>";
$successivo=next($vettore);
echo "Valore indice vettore=". key($vettore)."<br>";
echo "Valore contenuto vettore=". current($vettore)."<br>";
echo "<br>";
echo "Elemento precedente."<br>";
$precedente=prev($vettore);
echo "Valore indice vettore=". key($vettore)."<br>";
echo "Valore contenuto vettore=". current($vettore)."<br>";
echo "<br>";
?>
</body>
</html>

```

Le funzioni reset() ed end()

Le funzioni reset() ed end() riposizionano l'indice rispettivamente all'inizio e alla fine del vettore e ne forniscono il contenuto.

Esempio:

```

<html>
<head>
<title>Le funzioni reset e end</title>
</head>
<body>
<?php
echo "Inserimento di 5 elementi nel vettore!."<br>";
for ($i=0;$i<5;$i++) $vettore[$i]=rand(10,50);
for ($i=0;$i<5;$i++) echo $vettore[$i]. " - ";
echo "<br>";
echo "Contenuto iniziale indice vettore=". reset($vettore)."<br>";
echo "Contenuto finale indice vettore=". end($vettore)."<br>";
echo "<br>";
?>
</body>
</html>

```

Le funzioni count(), list() ed each()

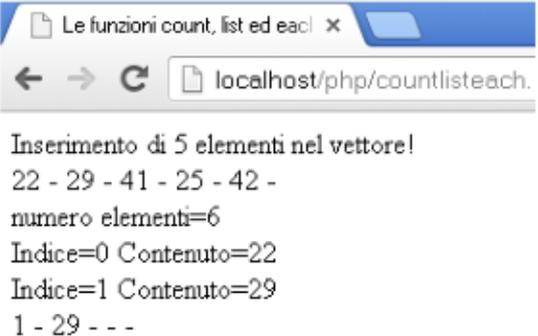
La funzione count() restituisce il numero di elementi dell'array. La funzione each() estrae dall'elemento corrente del vettore l'indice ed il contenuto, mettendoli in un array costituito da due elementi. La funzione list() estrae ciò che la funzione each ha memorizzato.

Esempio:

```

<html>
<head>
<title>Le funzioni count, list ed each</title>
</head>
<body>
<?php
echo "Inserimento di 5 elementi nel vettore!".<br>";
for ($i=0;$i<5;$i++) $vettore[$i]=rand(10,50);
for ($i=0;$i<5;$i++) echo $vettore[$i]. " - ";
$vettore[5]=65;
echo "<br>";
echo "numero elementi=". count($vettore).<br>";
$info=each($vettore);
echo "Indice=".$info[0]. " Contenuto=".$info[1];
echo "<br>";
$info=each($vettore);
echo "Indice=".$info[0]. " Contenuto=".$info[1];
echo "<br>";
$vettorex=(list($info[0],$info[1])=$info);
for ($i=0;$i<count($vettorex);$i++) echo $vettorex[$i].
?>
</body>
</html>

```



Esiste un'altra notazione per la rappresentazione degli array. Supponiamo di voler memorizzare in un array le province della regione Puglia:

```

$puglia=array("bari","bat","brindisi","foggia","lecce","taranto")
cioè equivale a scrivere:
$puglia[0]="bari" . . . $puglia[5]="taranto"

```

```

Possiamo inizializzare anche l'indice ponendolo ad un valore che vogliamo ad esempio:
$mese=array(1 => "gennaio","febbraio",. . . ,"dicembre")
pertanto otterremo:
$mese[1]="gennaio", . . . ,$mese[12]="dicembre"

```

Ordinamento degli array

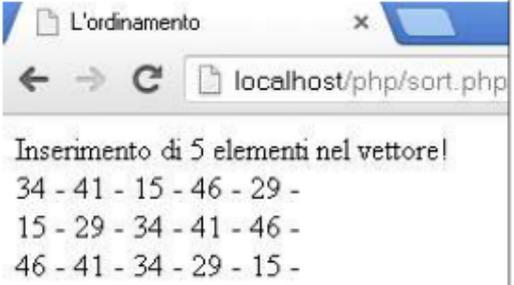
Esistono due funzioni che ordinano il vettore in ordine crescente e decrescente con le funzioni rispettivamente sort() e rsort().

Esempio:

```

<html>
<head>
<title>L'ordinamento</title>
</head>
<body>
<?php
echo "Inserimento di 5 elementi nel vettore!".<br>";
for ($i=0;$i<5;$i++) $vettore[$i]=rand(10,50);
for ($i=0;$i<5;$i++) echo $vettore[$i]. " - ";
echo "<br>";
sort($vettore);
for ($i=0;$i<5;$i++) echo $vettore[$i]. " - ";
echo "<br>";
rsort($vettore);
for ($i=0;$i<5;$i++) echo $vettore[$i]. " - ";
echo "<br>";
?>
</body>
</html>

```



Gli array multidimensionali

Gli array multidimensionali sono array di array di array e così via. Vediamo ad esempio un array articoli che contiene le seguenti informazioni: codice, descrizione, prezzo, iva:

```
$articoli = array (1 => array("codice" => "pcep604",  
    "descrizione" => "Computer Epson modello 604",  
    "prezzo" => 575.55,  
    "iva" => 21),  
2 => array("codice" => "pchp100",  
    "descrizione" => "Computer HP modello 100",  
    "prezzo" => 435.50,  
    "iva" => 21),  
3 => array("codice" => "nbepm250",  
    "descrizione" => "Netbook Epson modello 250",  
    "prezzo" => 275.50,  
    "iva" => 21),  
4 => array("codice" => "tbip2",  
    "descrizione" => "Tablet IPAD modello 2",  
    "prezzo" => 650.00,  
    "iva" => 21));
```

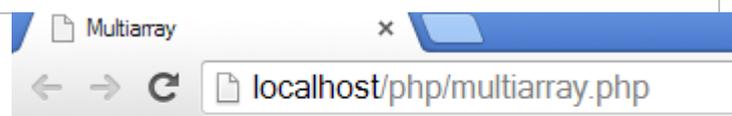
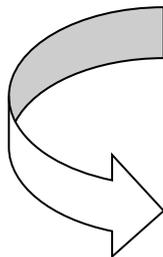
Vediamo un esempio con la visualizzazione dell'array in una tabella:

```

<html>
<head>
  <title>Multiarray</title>
</head>
<body>
  <?php
    $articoli = array (1 => array("codice" => "pcep604",
                                "descrizione" => "Computer Epson modello 604",
                                "prezzo" => 575.55,
                                "iva" => 21),
                      2 => array("codice" => "pchp100",
                                "descrizione" => "Computer HP modello 100",
                                "prezzo" => 435.50,
                                "iva" => 21),
                      3 => array("codice" => "nbepm250",
                                "descrizione" => "Netbook Epson modello 250",
                                "prezzo" => 275.50,
                                "iva" => 21),
                      4 => array("codice" => "tbip2",
                                "descrizione" => "Table IPAD modello 2",
                                "prezzo" => 650.00,
                                "iva" => 21));

  <?>
  <table border=1>
    <tr>
      <td><b>Codice</td><td><b>Descrizione</td><td><b>Prezzo</td><td><b>Iva
    </tr>
    <?php
      while (list($articolo,$dato)=each($articoli))
      {
        echo "<tr>";
        while (list($chiave,$valore)=each($dato))
        {echo "<td>".$valore;
        }
      }
    <?>
  </table>
</body>
</html>

```

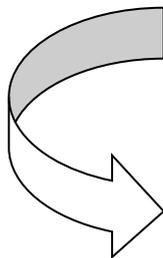


Codice	Descrizione	Prezzo	Iva
pcep604	Computer Epson modello 604	575.55	21
pchp100	Computer HP modello 100	435.5	21
nbepm250	Netbook Epson modello 250	275.5	21
tbip2	Table IPAD modello 2	650	21

Utilizzo della funzione foreach

```
<html>
<head>
  <title>Multiarray</title>
</head>
<body>
  <?php
    $articoli = array (1 => array("codice" => "pcepm604",
                                "descrizione" => "Computer Epson modello 604",
                                "prezzo" => 575.55,
                                "iva" => 21),
                      2 => array("codice" => "pchp100",
                                "descrizione" => "Computer HP modello 100",
                                "prezzo" => 435.50,
                                "iva" => 21),
                      3 => array("codice" => "nbepm250",
                                "descrizione" => "Netbook Epson modello 250",
                                "prezzo" => 275.50,
                                "iva" => 21),
                      4 => array("codice" => "tbip2",
                                "descrizione" => "Table IPAD modello 2",
                                "prezzo" => 650.00,
                                "iva" => 21));

  <?>
  <table border=1>
    <tr>
      <td><b>Codice</td><td><b>Descrizione</td><td><b>Prezzo</td><td><b>Iva
    </tr>
    <?php
      foreach ($articoli as $record)
      {echo "<tr>";
        foreach($record as $key => $valore)
        {echo "<td>".$valore;
          }
        }
      }
    <?>
  </table>
</body>
</html>
```



Multiarrray x

localhost/php/foreach.php

Codice	Descrizione	Prezzo	Iva
pcepm604	Computer Epson modello 604	575.55	21
pchp100	Computer HP modello 100	435.5	21
nbepm250	Netbook Epson modello 250	275.5	21
tbip2	Table IPAD modello 2	650	21

In questa parte del PHP la nostra attenzione verrà rivolta alla possibilità di gestire i file e di utilizzare informazioni memorizzate su dispositivi di memoria secondaria.

Per lavorare con i file è necessario conoscere il significato di puntatore al file. Questo puntatore è rappresentato da un numero assegnato al file al momento della sua apertura e lo identificherà univocamente per tutto il tempo di utilizzo. Al primo file aperto verrà assegnato al puntatore il valore 1. Dopo l'apertura non si dovrà più fare riferimento all'URL, ma al valore assegnato al puntatore.

Per gestire un file è necessario che il suo descrittore sia caricato nella RAM. Questa operazione viene chiamata apertura del file: **fopen()**.

Sintassi: `$numfile=fopen("/path.../nome del file","modalità di apertura");`

La modalità di apertura può contenere i seguenti valori:

Modalità	Significato
r	sola lettura
r+	lettura e scrittura
w	sola scrittura con perdita del contenuto precedente
w+	lettura e scrittura con perdita del contenuto
a	solo per aggiungere
a+	per lettura e aggiunta

Per la modalità di apertura relativa ai soli valori **w**, **w+**, **a** e **a+** se il file non dovesse esistere, il php tenterà di crearlo.

L'esempio seguente mostra come è possibile verificare l'esistenza o meno di un file e, nel caso di inesistenza esso verrà creato.

```
$nomedelfile="archivio.txt";
if (file_exists($nomedelfile)
{
    $numfile=fopen($nomedelfile,"r+");
    if (!$numfile) die ("il file $nomedelfile non e' stato aperto<br>");
    $messaggio="apertura del file $nomedelfile<br>";
}
else
{
    $numfile=fopen($nomedelfile,"w+");
    if (!$numfile) die ("il file $nomedelfile non e' stato aperto<br>");
    $messaggio="creazione del file $nomedelfile<br>";
}
echo $numfile;
echo $messaggio;
```

Nota: die (**stringa**) è una funzione di php che fa terminare lo script di php visualizzando la **stringa**.

Al termine dell'elaborazione il file deve essere chiuso con il comando **fclose()**. Mentre per la lettura e scrittura vengono utilizzati i rispettivi comandi: **fread()** e **fwrite()**.

Sintassi:

`fclose(puntatore del file);`

`fread(puntatore del file, numero dei byte da leggere);`

`fwrite(puntatore del file, stringa da scrivere);`

Scriviamo uno script per contare gli accessi ad una pagina web:

```
<html>
  <head>
    <title>Contatore di accessi</title>
  </head>
  <body>
    <?php
    $nomedelfile = "accessi.txt";
    if (file_exists($nomedelfile))
    {
      $numfile=fopen($nomedelfile,"r+");
      if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
      $contaccessi = (int) fread($numfile,10);
      fclose($numfile);
    }
    else
    {
      $numfile=fopen($nomedelfile,"w+");
      if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
      $contaccessi = 0;
      fclose($numfile);
    }
    $contaccessi++;

    $numfile=fopen($nomedelfile,"w+");
    if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
    fwrite($numfile,$contaccessi);
    fclose($numfile);
    echo($contaccessi);
  ?>
</body>
</html>
```

Come abbiamo visto nell'esempio, il file può essere incrementato da qualunque utente che si collega al nostro sito, ciò genera una situazione di condivisione del file e quindi di concorrenza. Nel nostro caso il file rappresenta l'area critica su cui deve essere applicato il meccanismo di eliminazione della **race condition**. Pertanto il metodo più immediato è quello della mutua esclusione, ovvero può accedere alla sezione critica un utente per volta. Php ci permette di applicare questo meccanismo attraverso l'utilizzo della sincronizzazione basata sui semafori lock e unlock. La problematica riguarda essenzialmente la fase di scrittura. Infatti quando un processo accede in scrittura ad una sezione critica, il file deve essere allocato in modo esclusivo a tale processo (locked), mentre se due o più processi accedono in lettura alla sezione critica, il file potrà essere condiviso tra processi (shared).

La funzione in Php che ci consente di applicare tale meccanismo è:

flock(int fp,int modo)

dove fp è il puntatore al file e modo è un valore intero che può assumere i seguenti valori:

Valore	Costante predefinita	Modalità	Descrizione
1	LOCK_SH	Condivisa – SHared	Per operazioni di lettura
2	LOCK_EX	Bloccata – EXclusive	Per operazioni di scrittura
3	LOCK_UN	Sbloccata – UNlock	Rilascia la risorsa
4	LOCK_NB	Disabilita il semaforo	Realizza un lock non bloccante

La funzione flock() viene utilizzata in combinazione con la funzione fopen():

```
//fase di lettura
$numfile=fopen($nomedelfile,"r");
flock($numfile, LOCK_SH); //blocco il file in modalità condivisa
.
.
.
operazioni di lettura
.
.
.
flock($numfile, LOCK_UN); //sblocco il file
fclose($numfile);
```

```
//fase di scrittura
$numfile=fopen($nomedelfile,"r");
flock($numfile, LOCK_EX); //blocco il file in modalità esclusiva
.
.
.
operazioni di scrittura
.
.
.
flock($numfile, LOCK_UN); //sblocco il file
fclose($numfile);
```

Accesso sequenziale ed accesso diretto ad un file

Prima di parlare dell'accesso ad un file è necessario conoscere alcune funzioni quali **feof**(numero del file) che restituisce il valore true se si è raggiunti la fine del file e false in caso contrario, **file_exists**(numero del file) anch'essa funzione booleana che avrà valore true in caso di esistenza del file e false in caso di inesistenza, **fseek**(numero del file, posizione) funzione che permette di posizionarsi direttamente sull'informazione fornita dal parametro posizione, **rewind**(numero del file) funzione che riposiziona l'indice all'inizio del file, **ftell**(numero del file) che fornisce un valore intero contenente la posizione corrente del

file ed infine **filesize**(*numero del file*) che fornisce un valore intero indicante la dimensione in byte del file, in caso di inesistenza del file restituirà il valore false.

L'applicazione di alcune funzioni determinano il tipo di accesso. Infatti le funzioni **fseek**, **rewind** e **ftell** sono tipiche di un accesso diretto. Pertanto l'accesso non si differenzia nella tipologia, ma in un'applicazione delle funzioni.

Nei tre esempi che seguono, il primo fa riferimento alla lettura/scrittura di un file di testo e vengono utilizzate le funzioni **fread** e **fwrite**. Mentre il secondo esempio memorizza un file contenente due campi (nominativo e numero telefonico) e ad ogni scrittura (**fputs**) del record si effettua un fine riga attraverso il carattere ASCII 10 corrispondente al Carriage Return, in fase di lettura (**fgets**) verranno elencati le righe che sono state scritte.

Infine il terzo esempio vi è un elenco dei comandi tipici di un accesso diretto al file.

Esempio 1

Supponiamo di voler gestire un file di testo contenente una sequenza di caratteri che nel caso di esistenza andremo a leggere, in caso contrario andremo a scrivere:

```
<html>
  <head>
    <title>Gestione file testo</title>
  </head>
  <body>
    <?php
      $nomedelfile = "testo.txt";
      if (file_exists($nomedelfile))
        //fase di lettura dal file testo.txt
        {
          $numfile=fopen($nomedelfile,"r+");
          flock($numfile, LOCK_SH);
          if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
          echo "Contenuto del file $nomedelfile."<br>";
          $stringa="";
          while (!feof($numfile))
            {
              $stringa=$stringa.fread($numfile,1);
            }
          echo "Testo: ".$stringa.<br>";
          flock($numfile, LOCK_UN);
          fclose($numfile);
        }
      else
        //fase di scrittura nel file testo.txt
        {
          $numfile=fopen($nomedelfile,"w+");
          flock($numfile, LOCK_EX);
          if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
          echo "Scrivo sul file $nomedelfile."<br>";
          $stringa="DINAMICI CON LA TECNOLOGIA PHP";
          fwrite($numfile,$stringa);
          flock($numfile, LOCK_UN);
          fclose($numfile);
        }
    ?>
  </body>
</html>
```



Scrittura	Lettura
 <p>Gestione file testo x localhost//php/fileseq.php</p> <p>Scrivo sul file testo.txt</p>	 <p>Gestione file testo x localhost//php/fileseq.php</p> <p>Contenuto del file testo.txt</p> <p>Testo: DINAMICI CON LA TECNOLOGIA PHP</p>

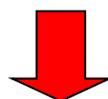
Esempio 2

Fase di lettura

```
<html>
<head>
  <title>File in modalità lettura</title>
</head>
<body>
  <?php
    $nomedelfile = "rubrica.txt";
    //Controllo esistenza del file:
    // - in caso di esistenza lettura delle informazioni
    // - in caso di inesistenza messaggio d'errore
    if (file_exists($nomedelfile))
    {
      $numfile=fopen($nomedelfile,"r+");
      flock($numfile, LOCK_SH);
      if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
      echo "Sto leggendo dal file $nomedelfile."<br>";
      while (!feof($numfile))
        echo fgets($numfile)."<br>";
      flock($numfile, LOCK_UN);
      fclose($numfile);}
    else
      echo "File $nomedelfile inesistente."<br>";
  <?>
</body>
</html>
```

Fase di scrittura

```
<html>
<head>
  <title>File in modalità scrittura</title>
</head>
<body>
  <form>
    Nominativo <input type=text name=txtnome maxlength=20 size=11><br>
    Numero tel.<input type=text name=txtnumero maxlength=20 size=11><br>
    <input type=submit name=cmdins value=INSERISCI>
  </form>
  <?php
    $nome=$_REQUEST["txtnome"]; $numero=$_REQUEST["txtnumero"];
    echo "Dati da inserire: $nome $numero <br>";
    $nomedelfile = "rubrica.txt";
    //Controllo esistenza del file:
    // - in caso di esistenza accodamento delle informazioni
    // - in caso di inesistenza scrittura delle informazioni
    if (file_exists($nomedelfile))
    {
      $numfile=fopen($nomedelfile,"a+");
      flock($numfile, LOCK_EX);
      if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
      echo "Sto scrivendo sul file $nomedelfile."<br>";
      $stringa=$nome.$numero."\n";
      fputs($numfile,$stringa);flock($numfile, LOCK_UN);
      fclose($numfile);}
    else
    {
      $numfile=fopen($nomedelfile,"w+");
      flock($numfile, LOCK_EX);
      if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
      echo "Sto scrivendo sul file $nomedelfile."<br>";
      $stringa=$nome.$numero."\n";
      fputs($numfile,$stringa);flock($numfile, LOCK_UN);
      fclose($numfile);}
  <?>
</body>
</html>
```



<p>File in modalità scrittura x</p> <p>localhost/php/filedirwrite.php</p> <p>Nominativo <input type="text" value="Mario Rossi"/></p> <p>Numero tel. <input type="text" value="0881-1234567"/></p> <p><input type="button" value="INSERISCI"/></p> <p>Dati da inserire: Mario Rossi 0881-1234567 Sto scrivendo sul file rubrica.txt</p>	<p>File in modalità lettura x</p> <p>localhost/php/filedirread.php</p> <p>Sto leggendo dal file rubrica.txt</p> <p>Mario Rossi0881-1234567 Carlo Verdi0881-222222 Antonio Bianchi080-5656567</p>
--	--

Esempio 3

```

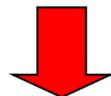
<html>
<head>
  <title>File ad accesso diretto</title>
</head>
<body>
  <?php
    $nomedelfile = "rubrica.txt";
    //Controllo esistenza del file:
    // - in caso di esistenza lettura delle informazioni
    // - in caso di inesistenza messaggio d'errore
    if (file_exists($nomedelfile))
    {
      $numfile=fopen($nomedelfile,"r+");
      flock($numfile, LOCK_SH);
      if (!$numfile) die ("Il file $nomedelfile non e' stato aperto<br>");
      echo "Sto leggendo dal file $nomedelfile"."<br>";
      while (!feof($numfile))
        echo fgets($numfile)."<br>";
      echo "<b>Funzioni applicate al file ad accesso diretto</b> <br>";

      echo "Funzione rewind:posizionamento all'inizio del file ->";
      rewind($numfile);
      echo fgets($numfile)."<br>";

      echo "Funzione seek:posizionamento al 50° carattere del file ->";
      fseek($numfile,50);
      echo fgets($numfile)."<br>";

      echo "Funzione ftell:Fornisce la posizione attuale del file ->";
      echo ftell($numfile)."<br>";
      flock($numfile, LOCK_UN);
      fclose($numfile);}
    else
      echo "File $nomedelfile inesistente"."<br>";
  ?>
</body>
</html>

```

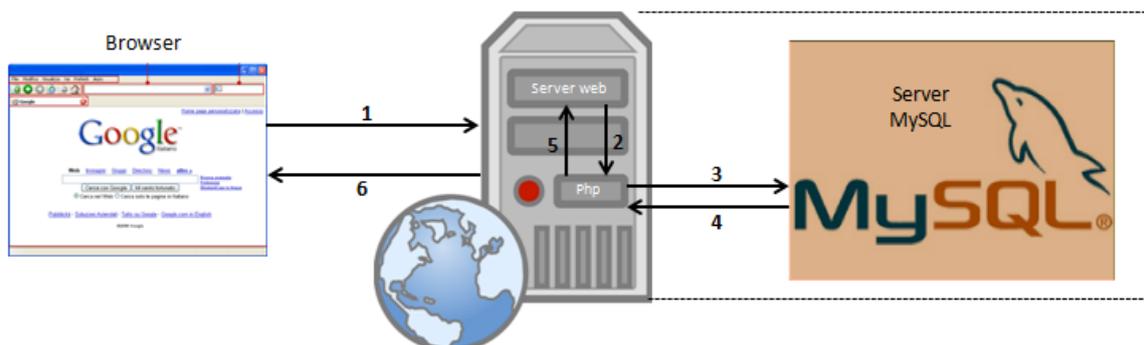




Connessione al database MySQL tramite script Php

In un sito web dinamico il contenuto delle pagine risiede in un database e ogni volta che un utente ne fa richiesta, questo contenuto viene recuperato e mostrato tramite le pagine web realizzate con il linguaggio HTML.

La seguente figura rappresenta la situazione che si verifica quando un utente richiede una pagina web dinamica:



1. Il browser web effettua una richiesta di una pagina web dinamica in formato Php
2. Il Server Web riconosce che la richiesta fa riferimento ad un file con estensione .php e quindi attiva l'interprete Php per eseguire gli script in essa contenuti
3. Lo script Php contiene i comandi per collegarsi al database MySQL e inviare le richieste tramite i comandi in SQL
4. Il server MySQL risponde restituendo i dati richiesti
5. Lo script Php scrive questi dati all'interno di una pagina web e la restituisce al Web Server
6. Il Web Server invia al browser dell'utente la pagina in formato HTML, generata in modo dinamico dallo script Php.

La funzione **mysql_connect** apre una connessione con il server MySQL e restituisce un oggetto:

mysql_connect(hostname, username, password)

- **hostname** rappresenta l'indirizzo IP o il nome del server su cui è in esecuzione il server MySQL; se è lo stesso server su cui è in esecuzione il server web, si indica con *localhost*
- **username** è il nome dell'utente che ha i permessi per accedere al database
- **password** è la password dell'utente.

La funzione restituisce un valore **\$conn** che corrisponde ad un valore booleano che assume il valore *false* qualora la connessione non sia andata buon fine.

Nell'esempio alla password non è stato fornito nessun valore.

Attivazione di una connessione al server MySQL(pagina php)

```

<?
$hostname="localhost";
$username="root";
$password="";
//Connessione server MySQL
$conn=mysql_connect($hostname, $username, $password);
if (!$conn)
{
    echo "Errore durante la connessione a MySQL";
    exit();
}
else
{
    echo "Connessione avvenuta con successo";
}
//Operazioni di gestione con il database
//Inserire le istruzioni appropriate
//Chiusura della connessione
mysql_close($conn);
?>

```

Per creare un nuovo database l'istruzione è:

```
mysql_create_db("magazzino");
```

oppure

```
$stringa_sql="create database magazzino";
mysql_query($stringa_sql);
```

Mentre se il database è già presente l'istruzione è:

```
mysql_select_db("magazzino");
```

Per creare una tabella ad esempio **articoli** nel database **magazzino**:

```
$stringa_sql="create table articoli (codice varchar(10) primary key, descrizione
varchar(40), prezzo dec(5,2), iva int)";
mysql_query($stringa_sql);
```

Con la stessa modalità vengono effettuate le operazioni tipiche di gestione.

Infine vediamo come vengono visualizzate le informazioni su una specifica richiesta dell'utente. Ad esempio si vogliono visualizzare tutti gli articoli che hanno un'iva fornita in input.

Utilizzeremo due pagine, la prima che consentirà all'utente di scegliere da tre radio button valore dell'iva e nella seconda pagina si visualizzeranno gli articoli che hanno l'iva richiesta. Il risultato verrà fornito in forma tabellare.

Pagina web (cerca_articoli.php)

```
<html>
<head>
<title>Cerca articoli</title>
</head>
<body>
<form action="visualizza_articoli.php" method=POST>
  Scegli aliquota iva<br>
  <input type=radio name=radiiva value=4>4%<br>
  <input type=radio name=radtit value=10>10%<br>
  <input type=radio name=radtit value=21 checked>21%<br>
  <br>
  <input type=submit name=cmdinvia value=INVIA>
</form>
</body>
</html>
```

Pagina web (visualizza_articoli.php)

```
<html>
<head>
<title>Visualizza articoli</title>
</head>
<body>
<?php
    $hostname="localhost";
    $username="root";
    $password="";
    $iva=$_POST["radiva"];
    //Connessione server MySQL
    $conn=mysql_connect($hostname,$username,$password);
    if (!$conn)
        echo "Connessione fallita.";
    else
    {echo "Connessione avvenuta con successo."."<br>";
    mysql_select_db("magazzino");
    $query="SELECT * FROM articoli where iva=$iva";
    $risultato=mysql_query($query);
    if (!$risultato)
        echo "Errore nella query.";
    else
        {($riga=mysql_fetch_array($risultato));
        if (!$riga)
            echo "Dati non presenti.";
        else
            {while ($riga)
            {echo $riga["cod"]." ".$riga["des"]." ".$riga["prz"]." ".$riga["iva"]."<br>";
            $riga=mysql_fetch_array($risultato);
            }
            }
        }
    }
    mysql_close($conn);
?>
</body>
</html>
```