2017

PHP

Guida PHP di base

Progetto di Alternanza Scuola-Lavoro – Anno scolastico 2016-2017 Istituto Tecnico Enrico Fermi, Siracusa

Eleonora
SikeWEB
03/05/2017

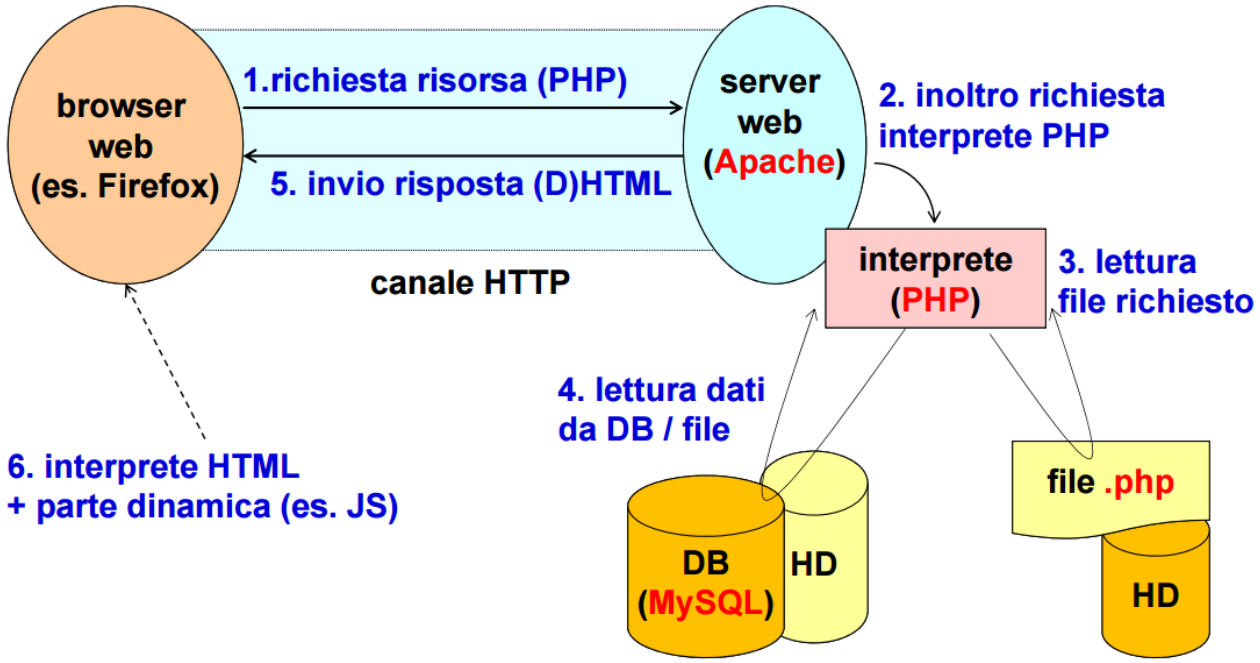
Ambiente di sviluppo: XAMPP

Un'applicazione web richiede l'utilizzo di diversi componenti:

- HTTP server per rendere disponibili le pagine web – es. Apache
- database per gestire i dati – es. MySQL
- application engine per l'esecuzione di programmi – es. interprete PHP, Tomcat per JSP

XAMPP è un insieme integrato di pacchetti software che:

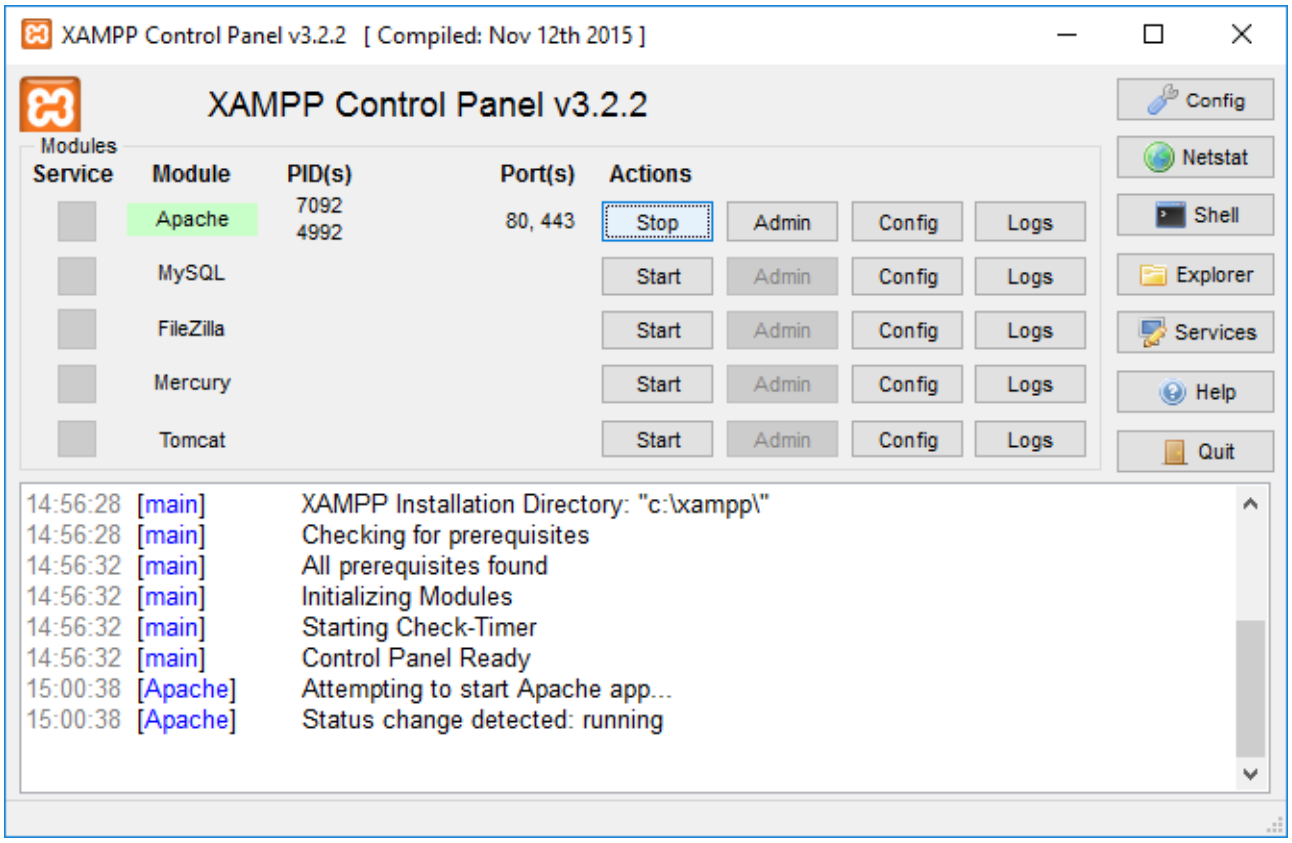
- include Apache, PHP, MySQL, phpMyAdmin
- è orientato allo sviluppo/test di applicazioni web
- facile da installare, configurazione predefinita
- presenta un'interfaccia per operazioni semplici, es. avvio/stop servizi, configurazione porte web server (per operazioni avanzate è necessario invece agire sulle configurazioni specifiche dei servizi)



Apache

Dal pannello di gestione XAMPP è possibile avviare il Server Apache cliccando su START. All'avvio verranno mostrati:

- il PID (identificativo del processo)
- le Porte occupate (Porta 80 e porta 443 di default)



Apache – file di configurazione e cartelle

- cartella “DocumentRoot”
  - dove posizionare le pagine/applicazioni web
  - “C:\xampp\htdocs” (in Windows)
- cartella “ServerRoot”
  - dove Apache viene installato
  - “C:\xampp\apache” (in Windows)
- file configurazione principale
  - porte in ascolto, moduli da caricare
  - “C:\xampp\apache\conf\httpd.conf” (in Windows)
- file configurazione XAMPP
  - usato per integrare i componenti (es. PHP, database)
  - “C:\xampp\apache\conf\extra\httpd-xampp.conf” (in Windows)

Apache - Verifica installazione

- caricare il file "ciao.html" nella DocumentRoot di Apache C:\xampp\htdocs
- aprire il browser alla URL: http://localhost/ciao.html

```
<!-- File ciao.html -- >

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Saluti HTML</title>
</head>
<body>
<p>Ciao!</p>
</body>
</html>
```

PHP

Php – file di configurazione e cartelle

- integrato e preconfigurato in XAMPP
- installato in C:\xampp\php\ (in Windows)
- file configurazione Apache C:\xampp\apache\conf\extra\httpd-xampp.conf (in Windows)
- file configurazione modulo PHP C:\xampp\php\php.ini (in Windows)

PHP - verifica installazione

- caricare il file "ciao.php" nella DocumentRoot di Apache C:\xampp\htdocs" (in Windows)
- aprire il browser alla URL: http://localhost/ciao.php

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<title>Saluti PHP</title>
</head>
<body>
<p><?php echo "Ciao da PHP!" ?></p>
</body>
</html>
```

Fondamenti di PHP

Concetti generali

Il php è stato sviluppato nel 1994, è un linguaggio lato server, simile al C ed è il maggior linguaggio di sviluppo del web. E' importante ricordare la sua interazione con database come MYSQL.

Per seguire il corso di Php è necessario possedere:

- Un qualsiasi editor html

- Uno spazio web con connessione ftp e supporto php oppure in alternativa un server locale come Xampp

Il mio primo programma: echo e html

PHP nasce come linguaggio di scripting per il Web, caratteristica che ci permette di modificare il codice HTML di una pagina e di modificarne il comportamento e quindi l'output, in base alle nostre esigenze.

Quando il Web server riceve una richiesta per una pagina PHP, essa viene analizzata dall'interprete del linguaggio, il quale restituisce un file contenente solo il codice che deve essere inviato al browser. In particolare l'interprete elabora unicamente il sorgente contenuto all'interno dei tag <?php ?>.

Questo ci dà la possibilità di inserire del codice PHP all'interno di una pagina contenente codice HTML (*embedding*) e quindi di rendere le nostre pagine Web dinamiche.

Sviluppiamo una pagina .php che visualizzi la scritta "Ciao Mondo!".

```
<?php
echo "Ciao Mondo!";
?>
```

Dal codice sovrastante notiamo come si utilizza il comando echo, che serve per stampare(Cioè mostrare/visualizzare) un testo o un codice html.

Come è possibile notare, nell'esempio proposto il codice contenuto all'interno dei tag delimitatori di PHP viene interpretato e, di conseguenza, viene stampato soltanto l'output che noi abbiamo stabilito in sede di stesura del sorgente (Ciao Mondo!)

Altro esempio:

```
<?php
// QUESTO E' UN COMMENTO
echo "<img src='immagine.jpg/>"
//Qui notiamo perfettamente come il codice html si integri nel php
?>
```

Possiamo concludere, dunque che per integrare un codice html nel php basta utilizzare il comando echo (echo "Codice html";) e inserire il codice html sostituendo però tutti gli " con ' .Quindi, se io ad esempio scrivo:

echo “”

è scorretto, poiché la forma corretta è questa:

echo “”

Per integrare invece codice php all’interno di una pagina html basta inserirlo all’interno dei tag

```
<?php
 // Codice Php
?>
```

I commenti in php vengono realizzati tramite l’uso:

- del doppio carattere di slash : //
- del doppio carattere cancelletto ##
- delimitatore di inizio e fine commento /* */ per commenti più elaborati e distribuiti su più righe

```
<?php
// questo è un commento
## questo è un commento
/* questo
È
Un
Commento
*/
echo “ questo non è un commento”
?>
```

Le variabili in PHP

Con il termine variabile e costante si identificano in informatica aree di memoria del sistema di elaborazione, identificate da un nome e destinate a contenere valori di uno specifico tipo di dato.

Un’area di memoria identificata da una variabile è destinata a contenere valori di dati che possono essere modificati dall’elaborazione di un determinato algoritmo.

Una costante invece identifica un’area di memoria il cui contenuto si dichiara immutabile per tutta la durata dell’elaborazione.

Le variabili nella programmazione e nel php sono degli elementi che vanno a sostituire un valore. Ogni variabile, infatti ha nome e rispettivo valore. Le variabili si definiscono anteponendo il simbolo $ al nome, nel seguente modo:

```
<?php
$variabile="valore";
?>
```

Il nome di una variabile è sensibile alla distinzione tra lettere maiuscole e minuscole.

Assegnazione

Per assegnare un valore ad una variabile si utilizza l'operatore =

Esempio pratico di variabile:

```
<?php
$testo = "Ciao ragazzi"; //OSSERVATE E MEMORIZZATE, ECCO COME SI DEFINISCE UNA VARIABILE !!
echo $testo; //Qui apparirà la scritta Ciao ragazzi!
$ a = 55;
?>
```

Ma ci sono alcune eccezioni in cui non è obbligatorio mettere "" e sono tutte qui sotto elencate:

- Nel caso di numeri, esempio:
  - $variabile = 2; //intero
  - $variabile = 5.3; // variabile di tipo double
- Nel caso di true o false, esempio: $variabile = true;

Proviamo ad incrociare tutto ciò che abbiamo appreso fino ad ora

```
<?php
$nome="Eleonora";
$anni=34; // i numeri, es 34 non hanno bisogno di "" ma si possono scrivere direttamente
$infos="Guida php";
echo "<b>Mi chiamo $nome, ho $anni anni</b>"; //Avremo come risultato la scritta Mi chiamo Eleonora, ho 34 anni, grassetata con sotto scritto: Guida php
echo "".$infos."";
?>
```

Operazioni Matematiche

E' possibile sommare variabili con contenuti numerici. Ecco un'esempio:

```
<?php
$primonumero = 2;
$secondonumero = 3;
$somma = $primonumero + $secondonumero;
echo "Il risultato è $somma"; //Qui apparirà il testo "Il risultato è 5"
?>
```

Così come, nell'esempio sopra abbiamo eseguito un'addizione è possibile eseguire una sottrazione (mettendo al posto di + -), eseguire una moltiplicazione (mettendo al posto di + *) e una divisione (mettendo al posto di + /).

Per incrementare e decrementare una variabile si ricorre agli operatore di :

- Incremento: ++
- Decremento: --

```
<?php
$k = 2;
$k++;
$k--
?>
```

Stringhe

Le stringhe rappresentano il modo più immediato per presentare all'utente testi generici e messaggi.

È possibile assegnare una stringa ad una variabile delimitandola tramite una coppia di apici singoli o doppi apici.

E' possibile concatenare due o più stringhe tramite l'operatore punto .

```
<?php
$a =  "mi chiamo";
$b = 'Eleonora' ;
$c = $a." ".$b;
echo $c; // verrà stampato -> mi chiamo Eleonora
?>
```

SikeWEB
Click different

Operatori booleani

Gli operatori booleani o operatori logici disponibili in PHP sono:

Nome	Operatore	Esempio	Operatore	Esempio	Risultato	Operazione
And	and	$a and $b	&&	$a && $b	True se entrambi sono true	Prodotto logico
Or	or	$a or $b	||	$a || $b	True se almeno uno dei due è true	Somma logica
Not	!	! $a	!		True se $a non è true	negazione
Xor	xor	$a xor $b	^		True se uno dei due è true ma non entrambi	

Operatori di confronto

Operatore	Confronto	Sintassi	Esempio
==	Uguale	espr1 = espr2	$a = $b
!=	Diverso	espr1 != espr2	$a != $b
===	Identico (uguale per valore e per tipo)	espr1 === espr2	$a === $b
>	maggiore	espr1 > espr2	$a > $b
>=	Maggiore o uguale	espr1 >= espr2	$a >= $b
<	minore	espr1 < espr2	$a < $b
<=	Minore o uguale	espr1< = espr2	$a <= $b

Le istruzioni condizionali, di controllo e i cicli

Istruzione IF

L'istruzione if offre la possibilità di valutare una condizione e di eseguire una specifica espressione nel caso in cui questa sia verificata.

If (condizione) espressione

Oppure

If(condizione){

Espressione 1;

espressione 2;

….

}

```
if($a>18) echo “sei maggiorenne”;
if($a<18) echo “sei minorenne”;
if($a=18){
echo “hai 18 anni, sei maggiorenne”;
echo “quindi puoi avere la patente”;
}
```

Se sorge la necessità di enunciare più espressioni è necessario ricorrere alle parentesi {}

Istruzione IF-ELSE

La parola chiave else permette di inserire un'opzione alternativa che viene eseguita nel caso in cui la condizione è falsa.

If(condizione){
Espressione 1;
espressione 2;
….
}
Else{
Espressione a;
Espressione b
……
}

```
if($a>=18) {
echo “sei maggiorenne”;
}
Else{
echo “sei minorenne”;
}
```

Istruzione Switch

L'istruzione Switch case permette il confronto di una variabile con una molteplicità di valori.

switch (variabile){
case valore1:
espressione1;
break;
case valore2:
espressione2;
break;
case valoren:
espressionen;
break;
default:
espressione;
}

```
switch ($a){
case “Giorgio”:
        echo “ciao Giorgio”;
        break;
case “Vittorio”:
        echo “ciao Vittorio”;
        break;
default:
        echo “mi dispiace non ti conosco”;
}
```

Ciclo For

L’istruzione For serve per l’implementazione di un ciclo.

For(valore iniziale; condizione; incremento){

Espressione;

}

```
for($k=1;$k<10;$k++){
echo 'il contatore vale:'.$k.'<br>';
}
echo “ciclo terminato”;
```

Ciclo While

L’istruzione For serve per l’implementazione di un ciclo.

while(condizione){

Espressione;

}

```
$k=1;
while ($k<10){
echo 'il contatore vale:'.$k.'<br>';
$k++;
}
echo “ciclo terminato”;
```

Post e Get

Post e controlli variabili

Nel php sono presenti alcune variabili predefinite: $_POST e $_GET.

La variabile $_POST è obbligatoriamente "collegata" a un form fisico html, serve dunque per ricavare un dato inserito in un campo di un form html che abbia come metodo "post". Andiamo alla pratica. Creeremo una pagina in html che conterrà la parte fisica dello script(Il form e i campi in html) e una pagina che conterrà la parte php/azione dello script.

pagina.html

```
<form action="pagina.php" method="post">
Eta' : <input type="text" name="eta"/>
<br>
Nome: <input type="text" name="nome"/>
<br>
<input type="submit" name="send" value="Invia Form"/>
</form>
```

Pagina.php

```
echo $_POST['eta'];
//Qui verrà mostrato ciò che ha inserito l'utente nel campo eta.
echo $_POST['nome'];
//Qui verrà mostrato ciò che ha inserito l'utente nel campo nome
```

E' possibile controllare tutte le variabili tramite alcune funzioni:

- empty : serve a controllare se la variabile è vuota. La sintassi è

```
if (empty($nomevariabile)){ ..... }
```

- isset : serve a controllare se la variabile esiste. La sintassi:

```
if (isset($nomevariabile)){  ..... }
```

Nel php è possibile usare "!" per invertire il significato della funzione che lo sussegue.

Ad esempio:

```
if (empty($nomevariabile)){} // Serve a dire se la variabile $nomevariabile è vuota
```

Invece

```
if (!empty($nomevariabile)){ //Con davanti ! Serve a dire se la variabile $nomevariabile non è vuota
```

Esempio:

Pagina.php

```
if (isset($_POST['send'])){
if (!empty($_POST['eta'])){
echo $_POST['eta'];
} else {
echo "Hai lasciato vuoto il campo età";
}
//Qui verrà mostrato ciò che ha inserito l'utente nel campo di nome eta.
if (empty($_POST['nome'])){
echo "Il campo nome è vuoto!";
} else {
echo $_POST['nome'];
}
//Qui verrà mostrato ciò che ha inserito l'utente nel campo nome
}
```

C'è anche da dire che per controllare se una variabile esiste è possibile anche non usare alcuna funzione, ma metterla normalmente in una parentesi.

Ad esempio:

```
if ($nomevariabile){
echo "Ok, ci siamo!";
}
```

significa: Se la variabile $nomevariabile esiste stampa il testo "Ok, ci siamo"

Get

Premetto che tutti i controlli e le funzioni valide per $_POST sono valide per $_GET.

Ma allora qual è la differenza tra $_GET e $_POST?

- $_POST è più sicuro e viene utilizzato per ottenere dei dati da un form.
- $_GET viene utilizzato per gestire dei dati tramite "url".

Ad esempio

esempio.php

```
<?php
echo "Ciao";
echo $_GET['nome'];
?>
```

Se dal browser raggiungiamo la pagina esempio.php in questo modo:

- esempio.php?nome=Matteo ----> il messaggio che verrà mostrato sarà "Ciao Matteo";
- esempio.php?nome=Giovanni ---->il messaggio che verrà mostrato sarà "Ciao Giovanni";
- e così via.

Esempio

Pagina.php

```
<?php
if ($_GET['stato'] == "usa"){
echo "Washigton";
} elseif ($_GET['stato'] == "italia"){
echo "Roma";
} elseif ($_GET['stato'] == "francia"){
echo "Parigi";
} else {
echo "Stato non definito!";
}
?>
```

Se raggiungiamo la pagina esempio.php dal nostro browser in questo modo:

- pagina.php?stato=italia ---->ci apparirà la scritta "Roma"
- pagina.php?stato=francia ------> ci apparirà la scritta "Parigi"
- e cosi' via
- Invece, raggiungendo la pagina in questo modo: pagina.php?stato= oppure cosi': pagina.php (Cioè con il $_GET "stato" vuoto) ci apparirà la scritta: "Stato non definito!";

Gli array

Iniziamo con un'esempio:

```
<?php
$nomearray = array ("giorgio","nicola","filippo"); //definiamo questa variabile/array
?>
```

per mostrare il contenuto, poi faremo cosi':

```
<?php
echo $nomearray[0] ; //Verrà mostrata la scritta giorgio
echo $nomearray[1]; //Verrà mostrata la scritta nicola
echo $nomearray[2]; //Verrà mostrata la scritta filippo e cosi' via..
?>
```

Ricerca di un elemento in un array

E' possibile, in ambito di array utilizzare la funzione in array(in_array()). La funzione è strutturata in questo modo:

in_array("paroladacercare", "arrayincuicercare");

Basandoci sul codice di sopra potremmo usare la funzione in_array in questo modo:

```
<?php
if ( in _array ("giorgio",$nomearray)) {
echo "si c'e giorgio";
}else{
echo "no, non c'e ";
}
?>
```

Funzioni

Possiamo dividere le funzioni in tre grandi gruppi:

- Quelle per ricavare informazioni e gestire variabili, stringhe e array

- Quelle per la gestione delle date

- Quelle per la gestione, le operazione e la gestione dei database mysql

- Quelle per la gestione dei files

Include e Require

Entrambe queste funzioni servono per implementare una pagina in un'altra pagina, quindi per "unire" due pagine. Solo che se si usa include e la pagina da implementare non esiste, non c'è nessun problema. Se invece, si usa require e la pagina da implementare non esiste, appare un'errore.

- Include serve a dire: se la pagina esiste implementala, altrimenti non fa niente
- Require serve a dire: se la pagina esiste implementala, altrimenti arrabbiati!
  Esempio di utilizzo:

Creiamo due pagine.

include.php

```
<?php
echo "ciao";
?>
```

pagina.php

```
<?php
include("include.php");
?>
```

Funzioni per ricavare informazioni e gestire variabili, stringhe e array

Ecco a voi le principali:

- isset (Controlla se una variabile esiste)
- empty (Controlla se una variabile è vuota)
- stripslashes (Elimina gli slash in una variabile, molto utilizzata per fixare i $_POST e i $_GET)

```
$nomevariabile=stripslashes($_POST['nomecampo']);
```

- strlen (Conta il numero di caratteri presenti una variabile)

```
$email=”info@miosito.it”;
strlen($email); //restituisce il valore 15
```

- str_replace (Serve per sostituire una lettera o una parola in una variabile , si usa cosi':

```
<?php
$variabile="Mi chiamo Matteo";
$variabile_nuova = str_replace("Matteo", "Giovanni", $variabile);
//Significa sostituisci Matteo con Giovanni nella varaibile di nome $variabile
echo $variabile_nuova; //IL testo che apparirà sarà Mi chiamo Giovanni, perchè abbiamo
sostituito Matteo con Giovanni
?>
```

- preg_match (E' possibile controllare se un testo o una variabile contiene un determinato valore. Ad esempio:

```
$email = 'info@miosito.it';
if (preg_match("/@/", $email)) {
```

```
 echo "Email valida";
} else {
 echo "Email non valida";}
```

Funzioni Personalizzate

Come creare una funzione in php? Semplice:

function nomefunzione($var){ echo $var;}

dopodichè la richiamiamo come si richiamano tutte le funzioni del php:

nomefunzione("matteo"); //Apparirà la scritta Matteo

Esempio:

```
function somma($var1,$var2){
        $somma=$var1 + $var2;
        echo $somma;
}
somma(5,4);
```

Funzione per gestire le Date

Con php è possibile mostrare la data corrente(Ora e/o giorno e/o mese e/o anno), sotto qualsiasi forma.

La funzione chiave è date();

Ecco a voi una tabella di utilizzo:

- d giorno del mese numerico 01-31
- D giorno della settimana in abbreviazione di 3 caratteri
- m mese numerico 01-12
- M mese in abbreviazione di 3 caratteri
- F mese in parola
- Y anno a quattro cifre
- y anno a due cifre
- H ore 00-24
- h ore 00-12
- i minuti
- s secondi

I valori che vedete sopra sono i valori da inserire nella funzione date, in questo modo date("valore");
Quindi, ad esempio se vogliamo stampare giorno, mese e anno corrente:

```
echo date("d.m.Y");
```

Funzioni per gestire File

Con php è possibile creare file, eliminarli, modificarli, creare cartelle, eliminare files da cartelle e tante altre cose. Ecco a voi la lista delle principali funzioni(ed esempi di utilizzi) per la gestione dei files.

- fopen($filedaaprire, $mode); (Apre un file)
- fread($filedaleggere,$filesize); (Legge un file)
- unlink($daeliminare); (Elimina un file)
- file_exists($dacontrollare); (Controlla se un file esiste)
- fclose($filedachiudere) (Chiude un file)
- fwrite($filedascrivere, "testodascrivere"); (Con questa funzione è possibile scrivere un file, eliminando però il vecchio contenuto.

Esempio lettura di un file

```
<?php
$filename = "nomefile.txt";
$handle = fopen($filename, "r");
$contents = fread($handle, filesize($filename));
echo $contents;
fclose($handle);
?>
```

Si nota che la variabile $mode sta ad indicare la modalità di apertura file. Scrivendo w+ potrete sia leggere che scrivere il file, scrivendo w potrete solo scriverlo, scrivendo r potrete solo leggerlo.

La variabile $filesize indica invece la dimensione del file, e si calcola con la funzione filesize($filedaleggere)

Esempio scrittura su file

```
<?php
$fp = fopen("nomefile.txt", "w+"); //apriamo il file in modalità "w",per poterlo leggere e scrivere
fwrite($fp, "Ciao");
fclose($fp); //Chiudiamo il file, serve più che altro per motivi di sicurezza.
?>
```

Funzione di redirect

Possiamo effettuare un redirezionamento dell'utente (cioè trasferirlo automaticamente da una pagina all'altra) attraverso la funzione header

```
header(“Location: http://www.sito.it/nuova-pagina.html”);
```

Si noti che all’interno della funzione possiamo passare un percorso assoluto o relativo.

```
header("location:home.php");
```

SikeWEB
Click different

Funzione di invio mail

per spedire un messaggio di posta elettronica dalle pagine del nostro sito web, infatti, è sufficiente richiamare la funzione mail().

La funzione mail(), una volta richiamata all'interno della nostra applicazione PHP, "contatterà" il sistema postale del nostro server intimandogli di spedire una mail con le caratteristiche definite dallo sviluppatore.

La sintassi è la seguente

```
mail($destinatario, $oggetto, $messaggio, $headers)
```

Esempio form html

```
<form action="send.php" method="post">
Mail : <input type="email" name="email">
<br>
Telefono:  <input type="tel" name="tel">
<br>
Nome: <input type="text" name="nome"/>
<br>
<input type="checkbox" name="consenso1" value="consenso1"> Acconsento al trattamento dei dati
<br>
<input type="checkbox" name="consenso2" value="consenso2"> Acconsento a ricevere pubblicità
<br>
<textarea rows="4" cols="50" name="message"></textarea>
<br>
<input type="submit" name="send" value="Invia Form"/>
</form>
```

Esempio send.php

```
<?php
$nome = $_POST['nome'];
$email = $_POST['email'];
$tel = $_POST['tel'];
$cons1 = $_POST['consenso1'];
$cons2 = $_POST['consenso2'];
$message = $_POST['message'];

$destinatario = 'email@gmail.com';
$messaggio = "Mittente: $nome \r\n Email: $email \r\n Tel: $tel \r\n Consenso dati: $cons1 \r\n Consenso pubblicita: $cons2 \r\n Messaggio: $message";
$headers = "From: $email";
$oggetto ="Richiesta info";
        if ($nome !="" and $message!=""){
                mail($destinatario, $oggetto, $messaggio, $headers); //This method sends the mail.
```

```
                echo "Email inviata con successo";
                }
        else{
                echo "Email non inviata";
                }
?>
```

Se vogliamo che venga mostrato un messaggio di Invio avvenuto con successo o Invio messaggio non riuscito nella pagina contenente il form html allora faremo nel seguente modo

Pagina home.php

```
<?php
        if(@$_GET['send']=='ok')
          {echo "Messaggio inviato";}
        elseif(@$_GET['send']=='ko')
        {echo "Messaggio non inviato";}
        else echo "";
?>
<form action="send.php" method="post">
Mail : <input type="email" name="email">
<br>
Telefono:  <input type="tel" name="tel">
<br>
Nome: <input type="text" name="nome"/>
<br>
<input type="checkbox" name="consenso1" value="consenso1"> Acconsento al trattamento dei dati
<br>
<input type="checkbox" name="consenso2" value="consenso2"> Acconsento a ricevere pubblicità
<br>
<textarea rows="4" cols="50" name="message"></textarea>
<br>
<input type="submit" name="send" value="Invia Form"/>
</form>
```

Pagina send.php

```
<?php
$nome = $_POST['nome'];
$email = $_POST['email'];
$tel = $_POST['tel'];
$cons1 = $_POST['consenso1'];
$cons2 = $_POST['consenso2'];
$message = $_POST['message'];
$destinatario = 'eletuccitto@gmail.com';
```

```
$messaggio = "Mittente: $nome \r\n Email: $email \r\n Tel: $tel \r\n Consenso dati: $cons1 \r\n Consenso pubblicita: $cons2 \r\n Messaggio: $message";
$headers = "From: $email";
$oggetto ="Richiesta info";
        if ($nome !="" and $message!=""){
                mail($destinatario, $oggetto, $messaggio, $headers); //This method sends the mail.
                header("location:home.php?send=ok");
                }
        else{header("location:home.php?send=ko");}
?>
```