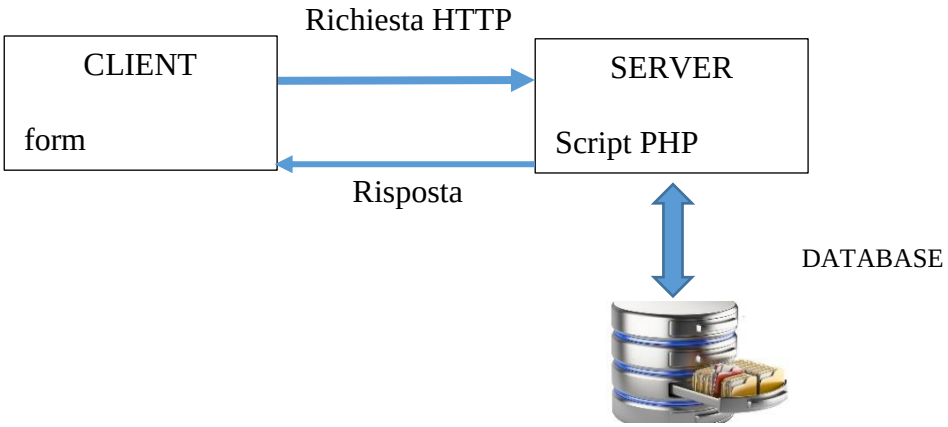
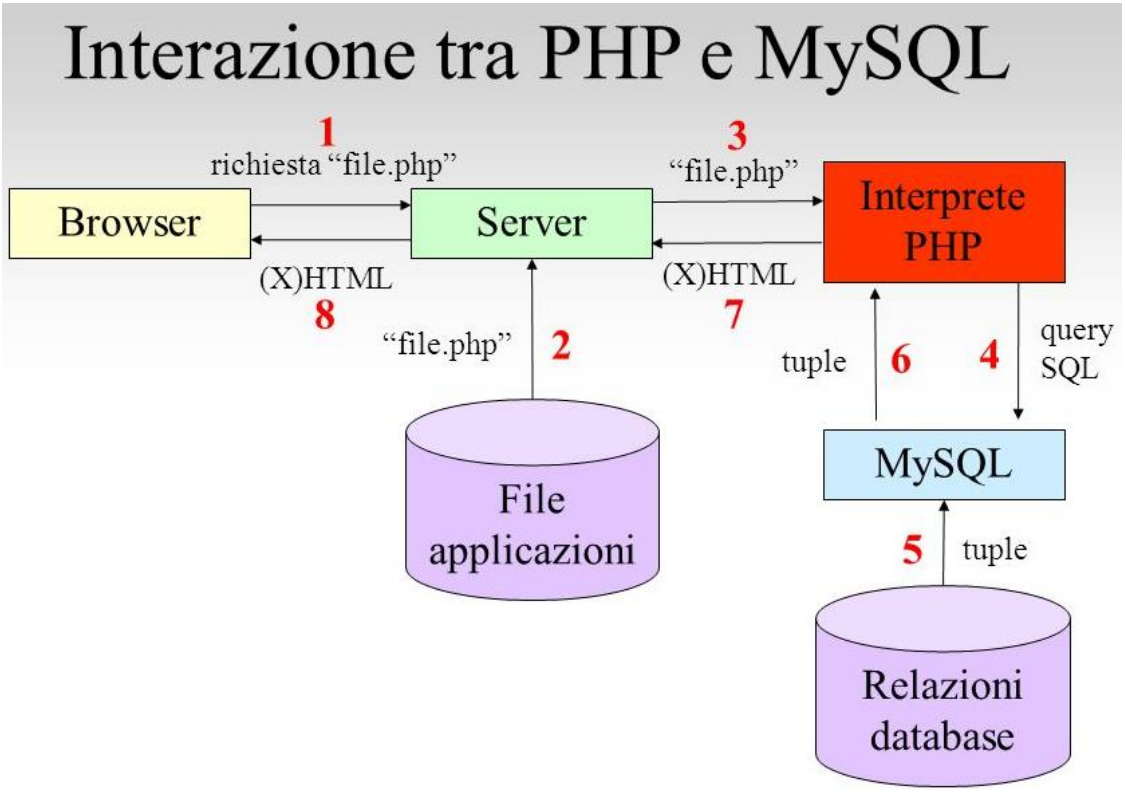
LINGUAGGIO PHP

Il linguaggio PHP è un linguaggio di programmazione lato server il cui script se eseguito produce pagine dinamiche o interattive il cui contenuto può variare a seconda delle scelte dell'utente.
Sul server un apposito modulo PHP interpreta ed esegue lo script. Uno script pertanto deve essere caricato sul server; se si vuole vederlo in esecuzione anche su un host occorre trasformare il computer in un server (es: easyPHP MyAdmin).

ARCHITETTURA CLIENT-SERVER





Esempio:

1. L'utente digita nella barra degli indirizzi il nome della pagina web contenente un forma da compilare (**form.htm**).
2. Se la pagina HTML esiste sul server quest'ultimo la invia al client.
3. L'utente inserisce delle informazioni e preme il tasto INVIA DATI per inviare i dati (mediante un metodo detto GET) al server.

4. Sul server gira uno script (contenuto nel file **elaboraForm.php** indicato come valore dell'attributo ACTION del tag FORM) che intercetta i dati inviati dal client, li elabora e restituisce al client i risultati sottoforma HTML.

5. Il client riceve i dati dal server e il suo browser li interpreta al fine di visualizzarli in una pagina web.

Tag delimitatori del codice PHP

```
<?php
echo "Ciao... questo è PHP!";
?>
```

Il codice in PHP deve essere scritto fra i precedenti tag delimitatori.

La variabile

Una variabile in PHP è etichettata in questo modo:

```
$nomeVariabile
```

È un contenitore che può contenere un solo valore alla volta:

```
$x = 10 ;
```

Commenti

nota su una sola riga

$a = 0; //azzero questa variabile

nota su più righe

/* da questo punto in poi
il codice è stato copiato
da quello di Pierino */

Comunicare dati al server

Una delle principali possibilità offerte dai linguaggi di scripting lato server è quella di generare contenuti (dinamicamente) sulla base delle richieste degli utenti. Questa interattività si realizza anche attraverso le variabili GET e POST che consentono, appunto, agli utenti di passare al server le loro richieste o preferenze attraverso i form (i classici moduli html) o semplici QueryString.

In pratica, attraverso GET e POST è possibile "raccogliere" gli input degli utenti i quali possono essere utilizzati per "indirizzare" il comportamento dei nostri script. Esistono, infatti, due diversi modi per passare dati al server: il metodo POST (generalmente usato nei form) ed il metodo GET (generalmene usato nelle QueryString). Vediamoli separatamente.

Metodo di invio dati GET e Querystring

Alcuni link hanno la seguente sintassi:

pagina.php?par1=valore&par2=valore

la parte dell'URL alla destra del punto di domanda (?) è chiamata **querystring**.

La **query string** server per conservare delle informazioni che l'utente vuole passare allo script chiamato. Tali informazioni, separate dal resto dell'URL attraverso il punto interrogativo, sono presenti sotto forma di coppie chiave=valore separate dal carattere **&**.

In questo modo, con il metodo GET, i dati vengono passati direttamente all'interno dell'indirizzo web (URL).

Esempio

L'utente inserisce il proprio nome e cognome in un form. I dati vengono inviati allo script:

http://localhost/Es02_dati_utente/elaboraForm.php?txtNome=paolo&txtCognome=rossi

lo script recupera i dati mediante il metodo GET e li salva in variabili locali:

```
$nome=$_GET['txtNome'];
$cognome=$_GET['txtCognome'];
```

La funzione explode

La funzione **explode** di **PHP** ha il compito di suddividere una stringa sulla base di un dato separatore.

La sintassi è la seguente:

```
explode(separatore, stringa);
```

- separatore: è il carattere separatore dell'array (spazio vuoto, virgola, etc.).

- stringa: è la stringa di testo da splittare (esplodere) in un array.

La funzione restituisce un array

Esempio:

```
$testo = "Max, Luca, Claudio, Paolo";
$arr = explode(",", $testo);
echo $arr[0] . "<br/>"; // Max
echo $arr[1] . "<br/>"; // Luca
echo $arr[2] . "<br/>"; // Claudio
echo $arr[3] . "<br/>"; // Paolo
```

Metodo POST

Il metodo POST viene utilizzato per inviare i dati ad una applicazione PHP tramite i form (moduli HTML). Prima di parlare nello specifico della variabile $_POST conviene fare un piccolo ripasso e vedere (brevemente) come funzionano i form HTML.

il tag <form> viene sempre accompagnato da due attributi fondamentali - "method" e "action" - vediamo a cosa servono:

- l'attributo "method" determina, appunto, il metodo con cui i dati saranno inviati al server; può avere come valore sia GET (che genera una QueryString) che POST;
- l'attributo "action" ha come valore il percorso dell'applicazione a cui saranno inviati i dati.

Facciamo un esempio:

```
<form method="post" action="applicazione.php">
Tuo Nome: <input type="text" name="nome">
<input type="submit" name="submit" value="invia">
</form>
```

Diversamente dal metodo GET, il metodo POST spedisce i dati in maniera non direttamente visibile per l'utente, attraverso la richiesta HTTP che il browser invia al server.

Tornando all'esempio visto sopra (il form HTML), per recuperare il valore del campo "nome" all'interno della nostra applicazione PHP useremo la variabile $_POST. Ecco il codice del file "applicazione.php" cui punta il nostro form:

```
<?php
//Recupero il valore del parametro "nome"
$nome_utente = $_POST['nome'];
//Ora stampo semplicemente a video il risultato
echo "Ciao " . $nome_utente;
?>
```

Funzione echo

La funzione echo è utilizzata per stampare a video delle informazioni (messaggi o contenuti di variabili.

```
$alunno=''paolo.rossi'';
$voto=7;
echo $alunno . '<br />' . $voto;
```

NB IL simbolo “punto” permette di concatenare elementi diversi (variabili e/o stringhe di testo).

Funzione count

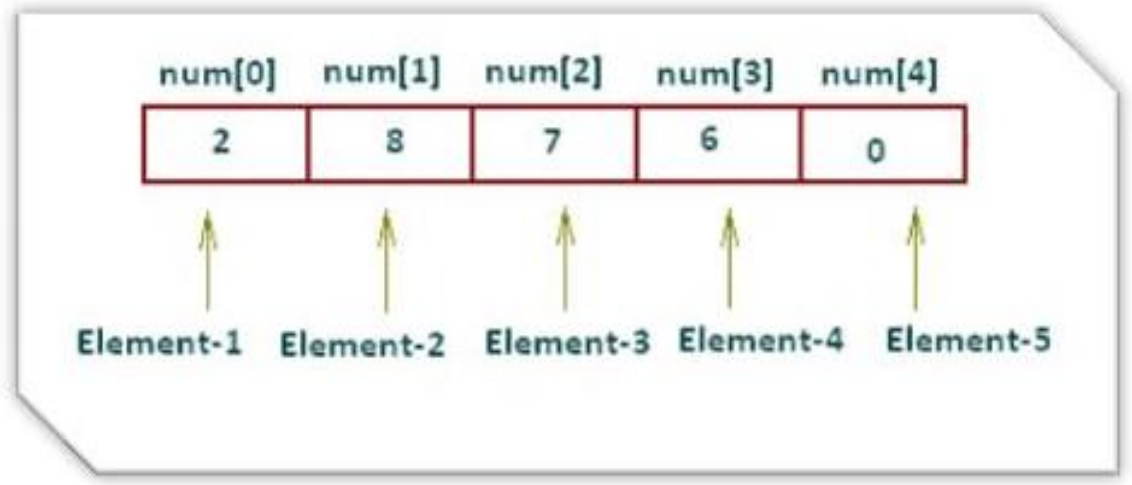
La funzione count restituisce il numero di elementi contenuti in un array.

```
$amici = array("Luca", "Jacopo", "Felice", "Peppo");
$conta=count($amici);
```

Array e ciclo iterativo

Un array è una struttura dati utilizzata per memorizzare una sequenza di elementi (testuali e/o multimediali).

Per accedere ad un elemento dell'array si utilizza un indice. Per definizione il primo elemento ha indice (posizione) 0.



Il ciclo iterativo FOR (ciclo a conteggio)

```
for ($i=0; $i<$n; $i++) {
        istruzione;
        istruzione;
        ... ;
};
```

Il ciclo (tra parentesi graffe) viene ripetuto un numero fisso n di volte.

Ad una variabile è assegnato un valore iniziale (nel nostro caso 0) che è incrementato automaticamente ad ogni ciclo.

Esempio:

```
$amici = array("Luca", "Jacopo", "Felice", "Peppo");
$conta=count($amici);
for($i=0; $i<$conta; $i++)
{
        echo $amici[$i] . " ";
}
```

La funzione array() consente di inserire dei valori in un array. In alternativa posso scrivere:

```
$amici[0] = "Luca";
```

```
$amici[1] = "Jacopo";
$amici[2] = "Felice";
$amici[3] = "Peppo";
$conta = count($amici);
for($i=0; $i<$conta; $i++)
{
        echo $amici[$i] . " ";
}
```

Il ciclo iterativo a condizione finale DO…WHILE e a condizione iniziale WHILE

```
do
{

   // set of actions to be performed

}while(condition);
while(condition)
{

   // set of actions to be performed

}
```

Ciclo WHILE

Il ciclo (tra parentesi graffe) viene ripetuto quando la condizione è VERA.

Se la condizione è falsa dall'inizio, il ciclo non viene ripetuto neanche una volta. Attenzione: se la condizione rimane sempre vera, il ciclo non si interrompe mai.

Ciclo DO…WHILE

Il controllo è alla fine, quindi il ciclo (tra parentesi graffe) viene ripetuto almeno una volta. Dopo il ciclo è ripetuto solo se la condizione è FALSA, altrimenti si interrompe. Se la condizione rimane sempre falsa, il ciclo non si interrompe mai.

The foreach Loop

```
foreach (array as value){

    //actions to be performed

}
```

The syntax simply means that “for each individual element of the array, perform a particular set of actions corresponding to that element.”

For example

```
<?php
$name = array("John", "Drake", "Mary", "Tim", "Jack");

foreach($name as $value){
     echo "$value </br>";
}
?>
```

SELEZIONE

```
if (condizione)
{BLOCCO_istruzione1;}
else
{BLOCCO_istruzione2;}
;
```

Se la condizione ha valore VERO si eseguono le istruzioni tra le prime graffe, altrimenti si eseguono quelle tra le graffe dopo else.