

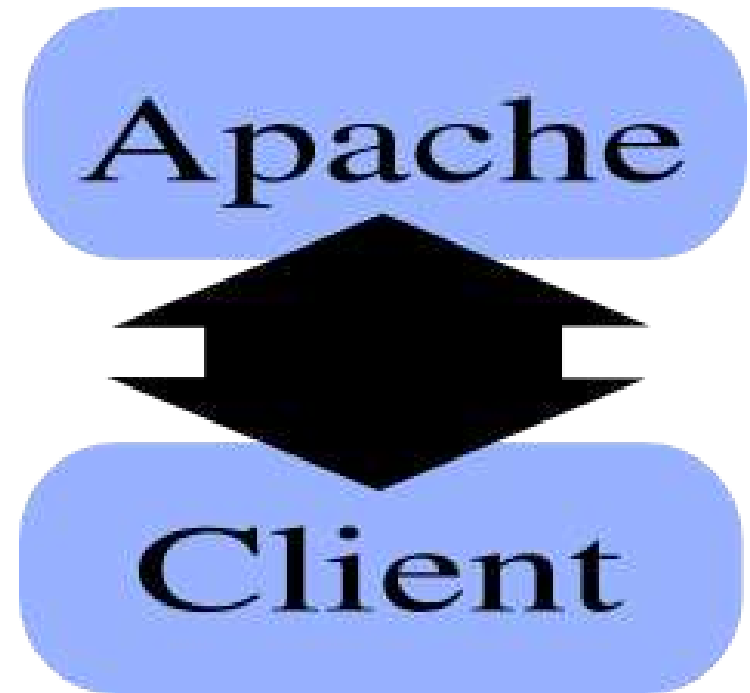
# Applicazioni Client/Server

- Una applicazione Client/Server è un sistema costituito da più applicativi che comunicano tra di loro ma sono indipendenti l'uno dall'altro
- Es. Browser/WebServer



# Applicazioni Web

- Una applicazione Web è una applicazione Client/Server che utilizza il protocollo HTTP permettendo agli applicativi di comunicare tra loro.



# Protocollo HTTP

- Nasce per la visualizzazione e scambio di documenti remoti.
- Si compone di due fasi.
  - 1. Il client richiede un file.
  - 2. Il server trasmette il contenuto del file richiesto.
- Request - Response

# Protocollo HTTP

- Request:

```
GET /file HTTP/1.0
User-Agent: Explorer...
Host: www.host.com
.....
```

- Response:

```
HTTP/1.1 200 OK
Date: Wed, 23 Nov 2005 10:10:44 GMT
Server: Apache/2.0.55
Last-Modified: Wed, 23 Nov 2005 10:09:01 GMT
ETag: "1000009-7f86-b591a940"
Accept-Ranges: bytes
Content-Length: 32646
Content-Type: text/plain
<HTML>..... <- Contenuto del File Richiesto
```

# Applicazioni Web

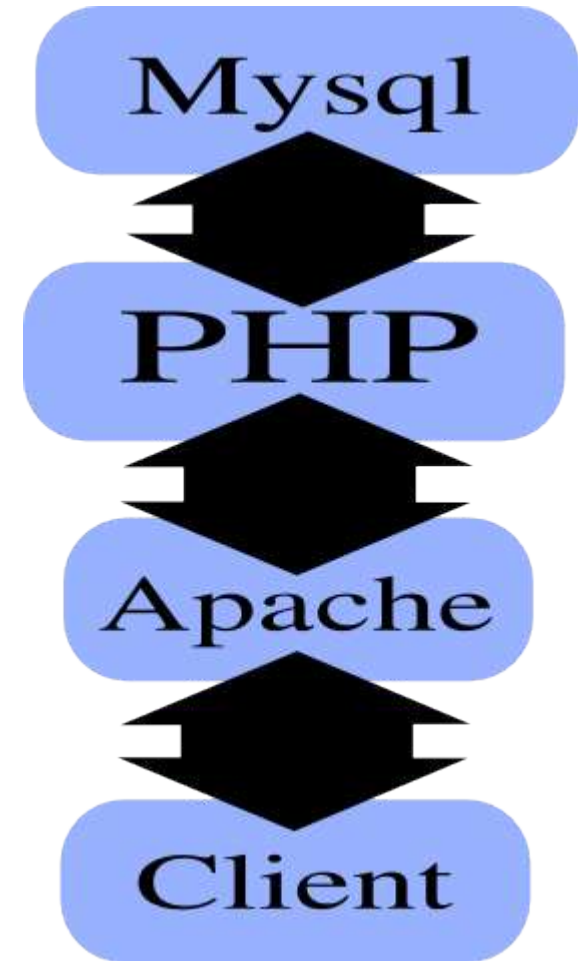
- Una applicazione Web è un programma che viene fatto eseguire sul server.
- Al posto di restituire il contenuto del file richiesto, tale file viene interpretato/eseguito a nome del server Web.
- Una volta eseguito lo script, il server Web restituisce l'output dello script.

# Applicazioni Web

- Il passaggio dei parametri dal client al server avviene durante la fase di Request del client in due modi:
  - 1. GET /file?**par=val1&par2=val2**... HTTP/1.0 ....
  - 2. POST /file HTTP/1.0...  
Content-Length: <lunghezza del contenuto>  
**par=val1&par2=val2**

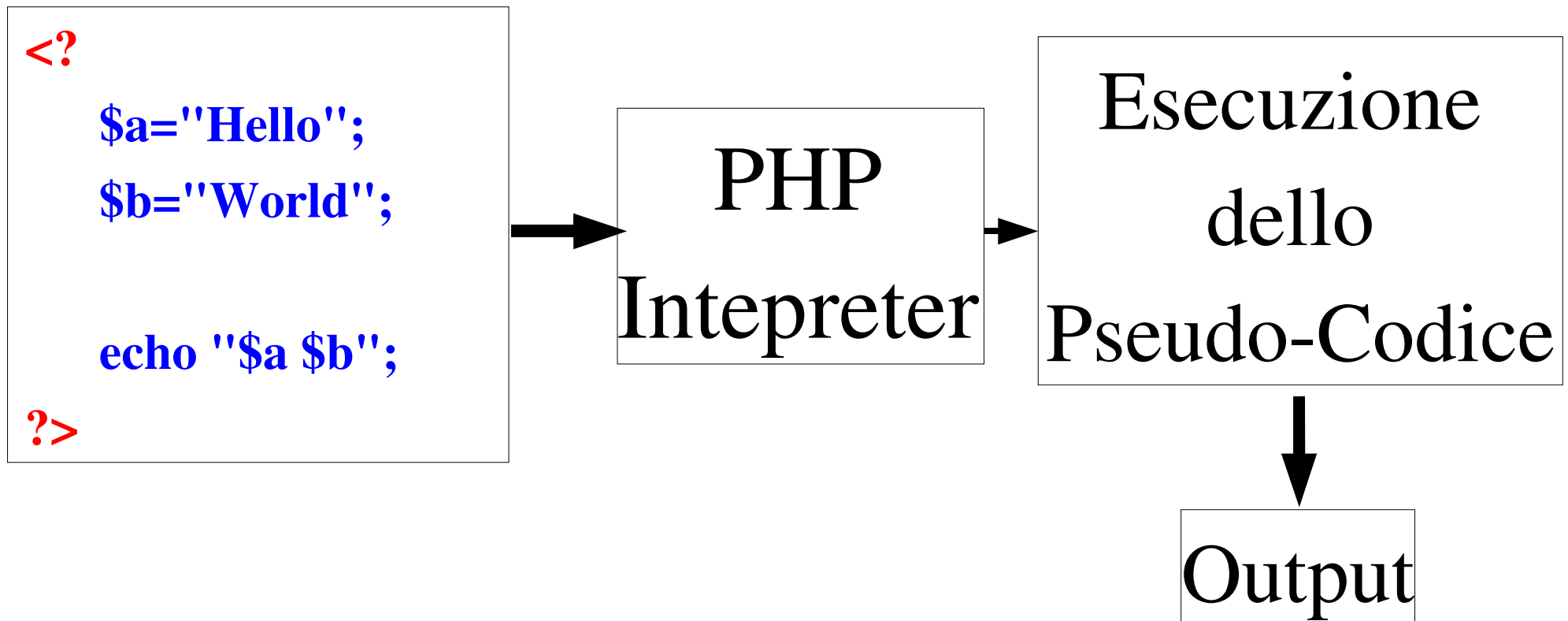
# Interfacciamento a DB via Web

- Una applicazione Web che si interfaccia con un DB è una applicazione che sfrutta il protocollo Http per evadere le richieste dei client.



# PHP

- Il Php è un linguaggio pseudo compilato.
- Esempio:





# Funzioni PHP

<?

```
function printMe( ){  
    $a="Hello";  
    $b="World";  
    echo "$a $b";  
}
```

```
printMe( );
```

?>

# Funzioni PHP e Parametri

```
<?  
function printMe($r, $s="World" )  
{  
    echo "$r $s";  
}  
$a="Hello";  
$b="World";  
printMe( $a,$b );  
?>
```

**Passaggio per Valore**

```
<?  
function printMe(&$r, $s="World" )  
{  
    $r="$r $s";  
}  
$a="Hello";  
$b="World";  
printMe( $a,$b );  
echo $a;  
?>
```

**Passaggio per Riferimento**

# Funzioni PHP – Valori di Ritorno

```
<?  
function printMe($r, $s="World" )  
{  
    return "$r $s";  
}  
$a="Hello";  
$b="World";  
echo printMe( $a,$b );  
?>
```

# Funzioni PHP - Scope

```
function printMe($r, $s="World" ){  
    $a="non e' la stessa variabile esterna alla funzione";  
    return "$r $s\n";  
}  
$a="Hello";  
$b="World";  
echo printMe( $a,$b );  
echo $a;
```

# Tipi di Dato in PHP

- PHP è un linguaggio non tipizzato.
- Cioè le variabili assumono il tipo di dato che rappresentano.

```
$a="stringa"; // Stringa
$a=true;     // Booleano
$a=3;       //intero
$a=3.4;     //float
$a=array("1"); // array $a[0]="1"
$a[0]="1";   // array $a[0]="1"
$a['key']="1"; // Array associativo
$a=new Class(); // Oggetto
$a=fopen();  // Risorsa
```

# PHP e Request

- Il passaggio degli input da Client a PHP avviene tramite l'accesso a due array associativi:
  - `$_GET` e `$_POST`
  - Contengono le variabili (chiave dell'array) e il loro valore (valore associato).

*`$a=$_GET['variabile'];`*

# Html e PHP

```
<html>
```

```
<body>
```

```
<?
```

```
function printMe($r, $s="World" ){  
    return "$r $s\n";
```

```
}
```

```
$a="Hello";
```

```
$b="World";
```

```
echo printMe( $a,$b );
```

```
echo $a;
```

```
?>
```

```
</body>
```

```
</html>
```

# Html e PHP

**<html>...**

**<?**

```
function printMe($r, $s="World" ){
```

```
    return "$r $s\n";
```

```
}
```

```
$a="Hello";
```

```
$b="World";
```

**?>**

**<b>**

**Questo e' il valore restituito da printME(\$a,\$b);**

**<?**

```
echo printMe( $a,$b );
```

**?>**

**</html>**



# Oggetti in PHP - Classi

```
class Vegetable {  
    var $edible;  
    var $color;  
    function Vegetable($edible, $color="green")  
    {  
        $this->edible = $edible;  
        $this->color = $color;  
    }  
    function is_edible()  
    { return $this->edible; }  
    function what_color()  
    {  
        return $this->color;  
    }  
}
```

```
class Spinach extends Vegetable {  
    var $cooked = false;  
    function Spinach()  
    {  
        $this->Vegetable(true, "green");  
    }  
    function cook_it()  
    {  
        $this->cooked = true;  
    }  
    function is_cooked()  
    {  
        return $this->cooked;  
    }  
}
```

# Oggetti in PHP - Creazione

```
include("classes.inc.php");  
  
$vegetale = new Vegetable(true, "blue");  
  
$spinacio = new Spinach();  
  
$spinacio->cook_it();  
  
echo $spinacio->is_cooked;
```

# Stateless, Cookies e Sessioni

- Http è un protocollo *state-less*  
ovvero *senza memoria*
- Ogni connessione è una Request indipendente dalle precedenti.
- Per avere memoria di una sequenza di Request che sono dipendenti tra loro si usano due tipi di tracciamento: *Sessioni (Server)* e *Cookie (Client)*

# Cookies - Server

- Sono stringhe del tipo:
  - **Set-Cookie:**  
**PREF=ID=1894766deff31a06:TM=1132901004:LM=1132901004:S=xb  
Nhoez\_l2vA8io4; expires=Sun, 17-Jan-2038 19:14:07 GMT; path=/  
domain=.google.it**
- Vengono elargiti nella fase di Response del server. Sono nell'header.

# Cookies - Client

- Il client li memorizza e quando fa una richiesta al server li invia inserendoli nell'header.
  - *Cookie:*  
*PREF=ID=1894766deff31a06:TM=1132901004:LM=1132901004:S=xb  
Nhoez\_l2vA8io4*
- I cookie hanno una vita definita dal server. Allo scadere del periodo vitale vengono cancellati dal client.

# Cookies in PHP

- `int setcookie(string nome, string valore, int scadenza, string percorso, string dominio, int sicuro);`

```
$nome="NomeUtente";  
$valore="Luca76";  
$scadenza=time()+600; //600 Secondi  
$percorso="/";  
$dominio="127.0.1";  
setcookie($nome,$valore,$scadenza);
```

# Accesso ai Cookies in PHP

- `$_COOKIE` è un array associativo contenente tutti i cookies che il client ci invia.

```
$nome=$_COOKIE['NomeUtente'];
```

# Sessioni

- Una sessione è un file o una struttura che contiene informazioni memorizzate dal server.
- E' identificata attraverso un ID casuale che viene inviato al client come Cookie:
  - Set-Cookie: PHPSESSID: 123456789;



# Sessioni e PHP

- Per avere coscienza della presenza di una sessione php ha bisogno di una funzione da inserire in ogni script che ne fa uso:
  - `Session_Start()`;
- Per inserire e accedere ai valori di sessione si usa l'array associativo `$_SESSION`.
- Quando non serve più deve essere cancellata con:
  - `Session_Destroy()`;

# Sessioni PHP - Esempio

**// Inserimento**

```
session_start();
```

```
$_SESSION["foo"] = 'bar';
```

```
session_write_close()
```

**// Recupero**

```
session_start();
```

```
$tmp = $_SESSION["foo"];
```

```
session_write_close();
```

# Interfacciarsi con MySQL

- PHP ha un gruppo di funzioni scritte per tale scopo.
- Le funzioni sono presenti solo se la libreria (o modulo o estensione) che le contiene è stata caricata dall'interprete PHP all'avvio del server.
- Il file di configurazione del PHP dice quali estensioni caricare. Tale file si chiama PHP.INI.

# Le estensioni Mysql.

- I moduli di interfacciamento a MySQL.
  - php\_mysql (MySQL<=4.0 php<5.0 di default)
  - php\_mysqli (tutte le versioni di MySQL e php>=5.0)
- Nel file php.ini deve essere presente una riga del tipo:
  - extension: mysql.so (o per windows mysql.dll)
- Dove mysql.so è il nome della libreria.