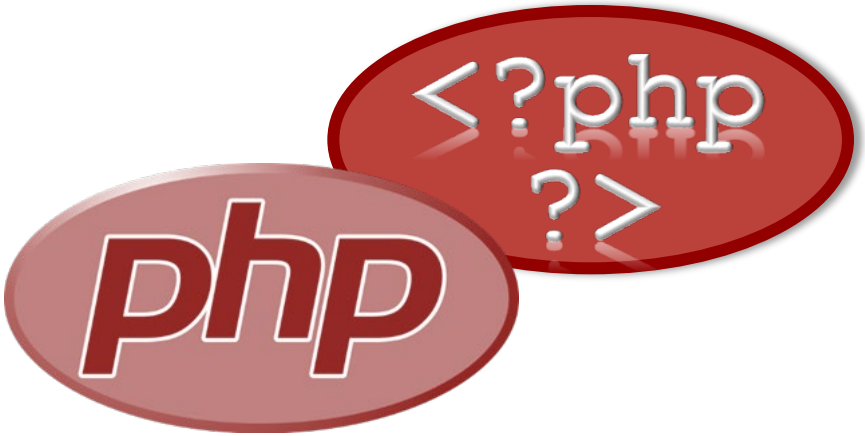
PHP sintassi



Assegnazione Secondo Esercizio

- Assegnato il secondo esercizio (CSS)
- Disponibile su Virtuale
- Deadline: **venerdì 17 novembre (FIRM DEADLINE!!!)**
- Punti bonus: **max 10 punti**
- Consegna del file tramite Virtuale
- I NOMI DEI FILE DEVONO ESSERE: **vostraemailunibo.html** e **vostraemailunibo.css**

Assegnazione Secondo Esercizio

Dato il file **`esercizio_css.html`** (che si trova nelle risorse della sezione *Preparazione al compito*), realizzare il file **`.css`** (esterno) così da ottenere il layout e lo stile riportati in **`css.png`** (browser: Chrome), che trovate tra le *Risorse*, tenendo in considerazione quanto segue:

- Tutti i font devono avere la stessa font-family, che deve essere monospace. La dimensione deve essere del 100%.
- L'elemento di intestazione **`<header>`** ha come sfondo un gradiente che varia dai seguenti colori (in diagonale, dall'angolo in alto a sinistra all'angolo in basso a destra): rosa, giallo, rosa, viola. La sua larghezza è pari al 100%. Margin e padding devono essere simili a quelli riportati nello screenshot. Il bordo inferiore è di colore viola ed ha uno spessore simile a quello mostrato nello screenshot.

Assegnazione Secondo Esercizio

- Il testo in **<header>** è di colore viola ed è allineato a destra.
- L'elemento **<nav>**, l'elemento **<aside>** e l'elemento **<section>** sono affiancati, uno accanto all'altro.
- L'elemento **<section>** deve occupare il 50% della larghezza e deve avere padding e margin simili a quelli mostrati nello screenshot.
- I paragrafi figli di **<section>** sono allineati sinistra e il testo è di colore nero.
- L'elemento **<nav>** deve occupare il 15% della larghezza e deve avere padding e margini simili a quelli mostrati nello screenshot.
- Gli elementi della lista in **<nav>** mostrano un quadratino come simbolo, sono allineati a sinistra ed hanno interlinea pari a 2 volte il valore di default. Lo sfondo di **<nav>** consiste in un gradiente uguale a quello di **<header>**.
- I link in **<nav>** in caso di **hover** vengono mostrati con sfondo viola, colore bianco, in grassetto e con un padding pari ad almeno l'1%.
- In caso di **hover** sull'elelmento **<nav>** allora l'ordine dei colori del gradiente viene invertito.

Assegnazione Secondo Esercizio

- L'elemento **<aside>** deve occupare il 20% della larghezza e i suoi paragrafi devono avere colore viola.
- L'elemento **<footer>** ha come sfondo un gradiente che varia dai seguenti colori (in diagonale, dall'angolo in basso a destra all'angolo in alto a sinistra): rosa, giallo, rosa, viola. La sua larghezza è pari al 100%. Margin e padding devono essere simili a quelli riportati nello screenshot.
- Il testo in **<footer>** è di colore viola, è allineato al centro ed ha una dimensione pari all'80%.
- NB: I NOMI DEI FILE DEVONO ESSERE vostraemailunibo.html e vostraemailunibo.css
- NB2: NON È POSSIBILE APPORTARE MODIFICHE AL FILE HTML (ECCETTO OVVIAMENTE IL NOME DEL FILE CSS)
- NB3: DOPO AVER RINOMINATO I FILE CON LA VOSTRA MAIL, ASSICURATEVI CHE IL FILE CSS SIA COLLEGATO CORRETTAMENTE AL VOSTRO FILE HTML

Assegnazione Secondo Esercizio

Lorem Ipsum

- Lorem ipsum
- Adhuc veniam
- Eu novum graecis ius

Ea suas mollis antiopam duo. Causae dignissim vix ex, civibus dissentias complectitur ea his. Pri luptatum definiebas definitiones an, his choro signiferumque id. Vix dico expetendis eu, magna habeo at his.

Est ut rebum rationibus. Id vidit iuvaret singulis sea, ad suas illud sonet pro. Usu sale accumsan detraxit ex, ponderum conceptam te quo. Detraxit salutatus definitiones ad eos. Falli pertinax te cum.

Lorem ipsum dolor sit amet, sed vide prima id, mea cu elit affert. Ad integre rationibus his, duo rebum urbanitas an. Vel et elit prodesset adversarium, vel ut dolorem assentior vituperatoribus. No graeci cotidieque intellegebat eam.

Adhuc veniam te duo, omnium perpetua imperdiet eos an, est justo choro insolens id. Mel at audire euismod, per eu inermis omnesque. Ea sea movet veritus incorrupte, ut eos modo iracundia, qui magna virtute referrentur ad. Sit cu graece periculis corrumpit.

Eu novum graecis ius. No movet affert homero mei. An mea error everti mollis, eum id ubique adipiscing honestatis. Munere scaevola mei ea, modo movet moderatius qui in. Eos fabulas corpora perpetua an, id qui dolorum honestatis, pericula reformidans mel eu. Ferri ubique labitur te ius.

Ne diam legimus voluptua quo. Est vidisse nusquam in, ornatus sententiae pri cu, vix cibo detraxit id. Per mundi dolores maiestatis ea. Cu nostrum nominavi delicatissimi vis, id saepe iisque eum. Vel stet ludus iudicabit eu. Vim te probo persecuti, et vix erant molestie.

Doctus atomorum an eam. Ut zril ancillae vituperata mei, ne duo unum dolore comprehensam. Eu eros iriure nusquam duo, an vero possim lobortis usu. Fabulas contentiones cum an. Mea indoctum evertitur no, an ornatus incorrupte vim. Ut nam homero disputationi, eam nisl unum adhuc at.

Lorem ipsum dolor sit amet, sed vide prima id, mea cu elit affert. Ad integre rationibus his, duo rebum urbanitas an.

Prossime lezioni

- Venerdì 10 novembre: in lab (PHP)
  - Turno 1 A-L
  - Turno 2 M-Z
- Martedì 14 novembre: in aula (Bootstrap, con il tutor Kelvin Olaiya)
- Venerdì 17 novembre: in lab (Bootstrap + PHP)
  - Turno 1 M-Z
  - Turno 2 A-L

Argomenti

- Sintassi PHP
  - Elementi di base
  - GET e POST
  - Gestione di Cookie e Sessioni
  - Gestione del DB MySQL

Siti di riferimento

- W3Schools: https://www.w3schools.com/php/
- Sito web di PHP: https://secure.php.net/
- NB: conviene comunque impratichirsi con il W3School perché è l'unico disponibile durante l'esame!

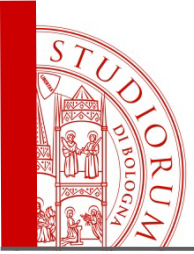
Variabili

- Tutte le variabili iniziano con il carattere **$**

$nome

- È debolmente tipizzato
  - Possiamo associare alla stessa variabile più tipi di dato
    - Automaticamente PHP converte la variabile nel tipo opportuno

```
$qualcosa = 3;
$qualcosa = true;
$qualcosa = “ciao”;
```



Tipi di dato

- Boolean
- Integer
- Float
- String
- Array
- Object
- NULL
- Resource

String

- È possibile definire una stringa in 4 modi:
  1. **Single quoted**: variabili non vengono espanse e gli unici caratteri con escape ammessi sono \ e ’. Esempio: `$stringa = 'stringa';`
  2. **Double quoted**: variabili vengono espanse, ammesse le più comuni sequenze di escape. Esempio: `$stringa = "stringa";`

String

3. **Heredoc**: si comporta come le double quoted ma senza usarle (quindi il carattere " non deve essere preceduto da \).
   Esempio: `$stringa = <<<ID`
   stringa
   ID;
4. **Newdoc**: si comporta come le single quoted ma senza usarle, quindi \ e ‘ sono sempre trattati letteralmente.
   Esempio: `$stringa = <<<'ID'`
   stringa
   ID;

Array

- Gli array vengono definiti con la funzione **`array()`**.
  Esempio: **`$esempio = array(1,2,3);`**
- In PHP ci sono tre tipi di array:
  1. Indexed.
  2. Associative.
  3. Multidimensional.

Array - Indexed

- Sono gli array *classici* con indice numerico.
- Esempio:

```
$gelati = array("cornetto","ghiacciolo","ricoperto");
```

- Esempio alternativo equivalente:

```
$gelati[0] = "cornetto";
$gelati[1] = "ghiacciolo";
$gelati[2] = "ricoperto";
```

Array - Associative

- Array che hanno stringhe come indici.
- Esempio:

```
$age = array("Peter"=>"35", "Ben"=>"37",
"Joe"=>"43");
```

- Esempio alternativo equivalente:

```
$age['Peter'] = "35";
$age['Ben'] = "37";
$age['Joe'] = "43";
```

Array - Multidimensional

- Array che contengono uno o più array.
- Esempio:

```
$age = array(1, array(2,3,5,6), "prova",
array("ciao",3, True));
```

Object oriented

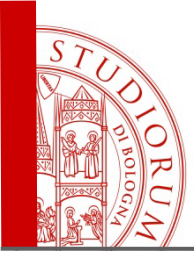
- PHP consente di definire classi e istanziare oggetti.
- Supporta i principali meccanismi dell'OOP:
  - Proprietà e metodi public, private, protected e static.
  - Ereditarietà.
  - Classi astratte.
  - Interfacce.
  - Tratti.

Object - Esempio

```
class Persona {
	private $nome;
	public $cognome;

	public function __construct($nome, $cognome) {
		$this->nome = $nome;
		$this->cognome = $cognome;
	}

	public function presentati() {
		echo "Sono ".$this->nome." ".$this->cognome;
	}
}
```



Object – Esempio (2)

```
$gino = new Persona("Gino", "Pino");

$gino->presentati();
//Mi chiamo Gino Pino.

echo $gino->nome;
//Fatal error

echo $gino->cognome;
//Pino
```

Resource

- Una risorsa non è un vero e proprio tipo
- Si tratta di una variabile speciale che contiene il riferimento ad una risorsa esterna.
- Sono create e usate da funzioni speciali.
- Esempi: file aperti o connessioni ad un database.

Output

- In PHP ci sono due modi per ottenere un output:
  - **`Echo`**: può stampare 1 o più stringhe e non ha valore di ritorno.
  - **`Print`**: può stampare 1 sola stringa e restituisce sempre 1.
- Solitamente viene usata **`echo`** in quanto leggermente più veloce.

var_dump

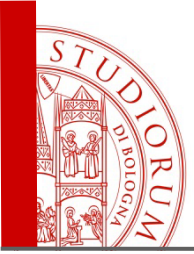
- Funzione per stampare tipo e contenuto di un'espressione, utile in fase di debug

```
<?php
$a = array(1, array("a", "b", "c"));
var_dump($a);
?>
```

```
array(2) {
  [0]=>
  int(1)
  [1]=>
  array(3) {
    [0]=>
    string(1) "a"
    [1]=>
    string(1) "b"
    [2]=>
    string(1) "c"
  }
}
```

Variabili Superglobali

- Le **variabili superglobali** sono variabili accessibili ovunque.
- Esempi:
  - **`$GLOBALS`**: memorizza tutte le variabili globali
  - **`$_SERVER`**: gestisce informazioni sul server
  - **`$_GET`**: usato per collezionare dati inviati con metodo **`GET`**
  - **`$_POST`**: usato per collezionare dati inviati con metodo **`POST`**
  - **`$_COOKIE`**: gestisce i cookie
  - **`$_REQUEST`**: usato per collezionare dati inviati sia con metodo **`GET`** che con metodo **`POST`** e i cookie
  - **`$_SESSION`**: gestisce le sessioni



Esempio Get

- Abbiamo due pagine:
  - **`esempio_get.html`**
  - **`process_get.php`**
- **`Esempio_get.html`** contiene un form con un campo di testo e il bottone di submit
- Una volta compilato il form, vogliamo mandare il testo inserito alla pagina **`process_get.php`** che leggerà i dati e restituirà una pagina HTML contenente l'informazione inserita

Esempio Get

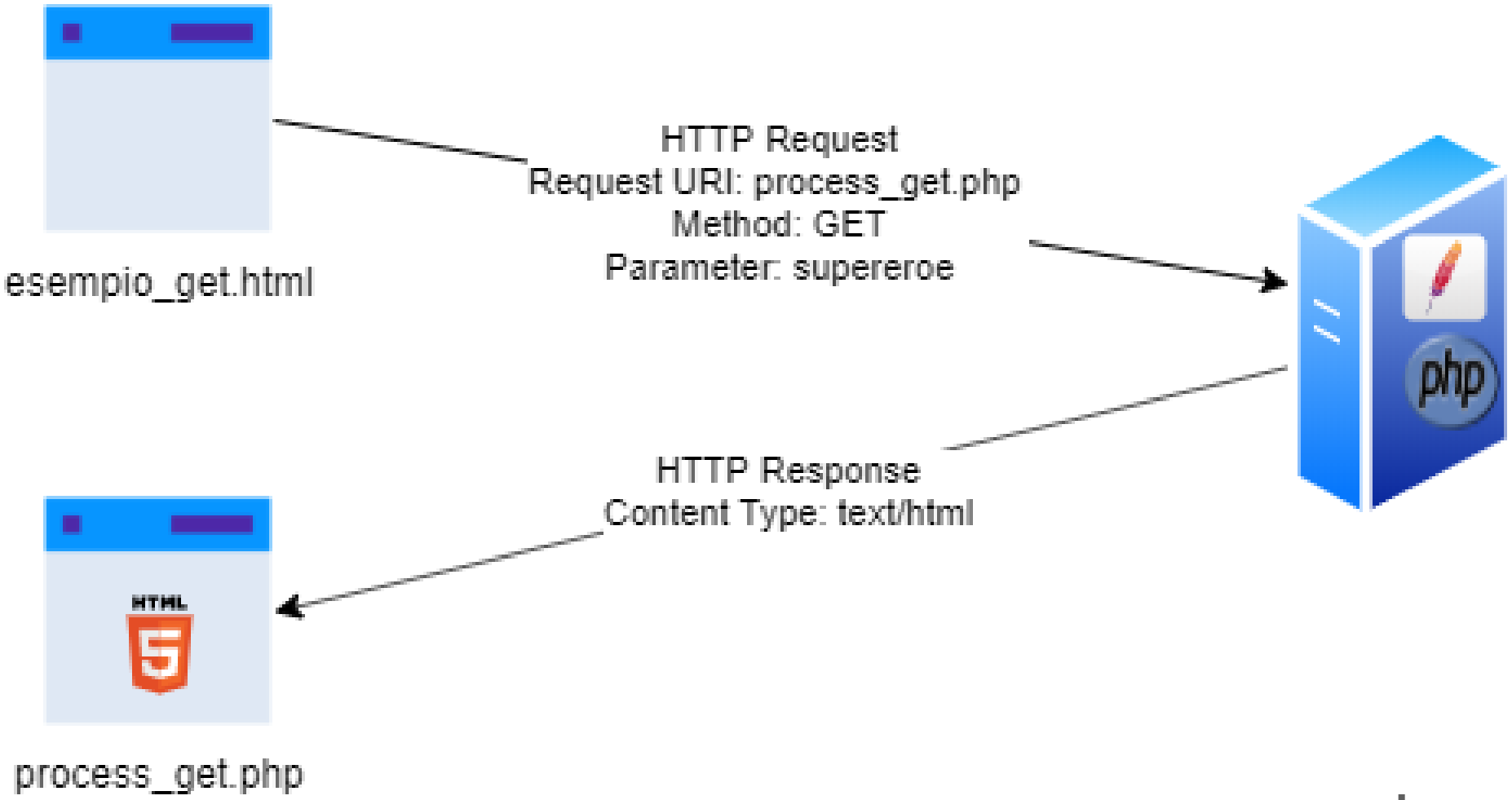
- **esempio_get.html**

```
…
<form action="process_get.php" method="get">
    <label for="idsupereroe">Supereroe</label>
    <input type="text" id="idsupereroe"
name="supereroe">
    <input type="submit">
</form>
…
```

- **process_get.php**

```
…
<p>GET: <?php echo $_GET['supereroe'] ?></p>
<p>POST: <?php echo $_POST['supereroe'] ?></p>
<p>REQ:<?php echo $_REQUEST['supereroe'] ?></p>
…
```

Esempio Get


esempio_get.html
HTTP Request
Request URI: process_get.php
Method: GET
Parameter: supereroe
HTTP Response
Content Type: text/html
process_get.php

Output Esempio Get (batman)

```
GET: batman


POST:
Notice: Undefined index: supereroe
in path\process_get.php on line 9
REQ: batman
```

Esempio Get - Considerazioni

- In pratica, nell'esempio, abbiamo usato il form per generare una richiesta HTTP con metodo GET e con un particolare parametro (supereroe).
- È possibile ottenere lo stesso risultato SENZA usare il form?

Esempio Get - Considerazioni

- In pratica, nell'esempio, abbiamo usato il form per generare una richiesta HTTP con metodo GET e con un particolare parametro (supereroe).
- È possibile ottenere lo stesso risultato SENZA usare il form?
- **Ovviamente sì!**
- **È possibile scrivere l'url direttamente nel browser o usare un qualsiasi software che consenta di effettuare richieste HTTP.**

Esempio Post

- Esempio speculare a quello di prima. L'unico cambiamento è il metodo usato (POST anziché GET)

Esempio Post

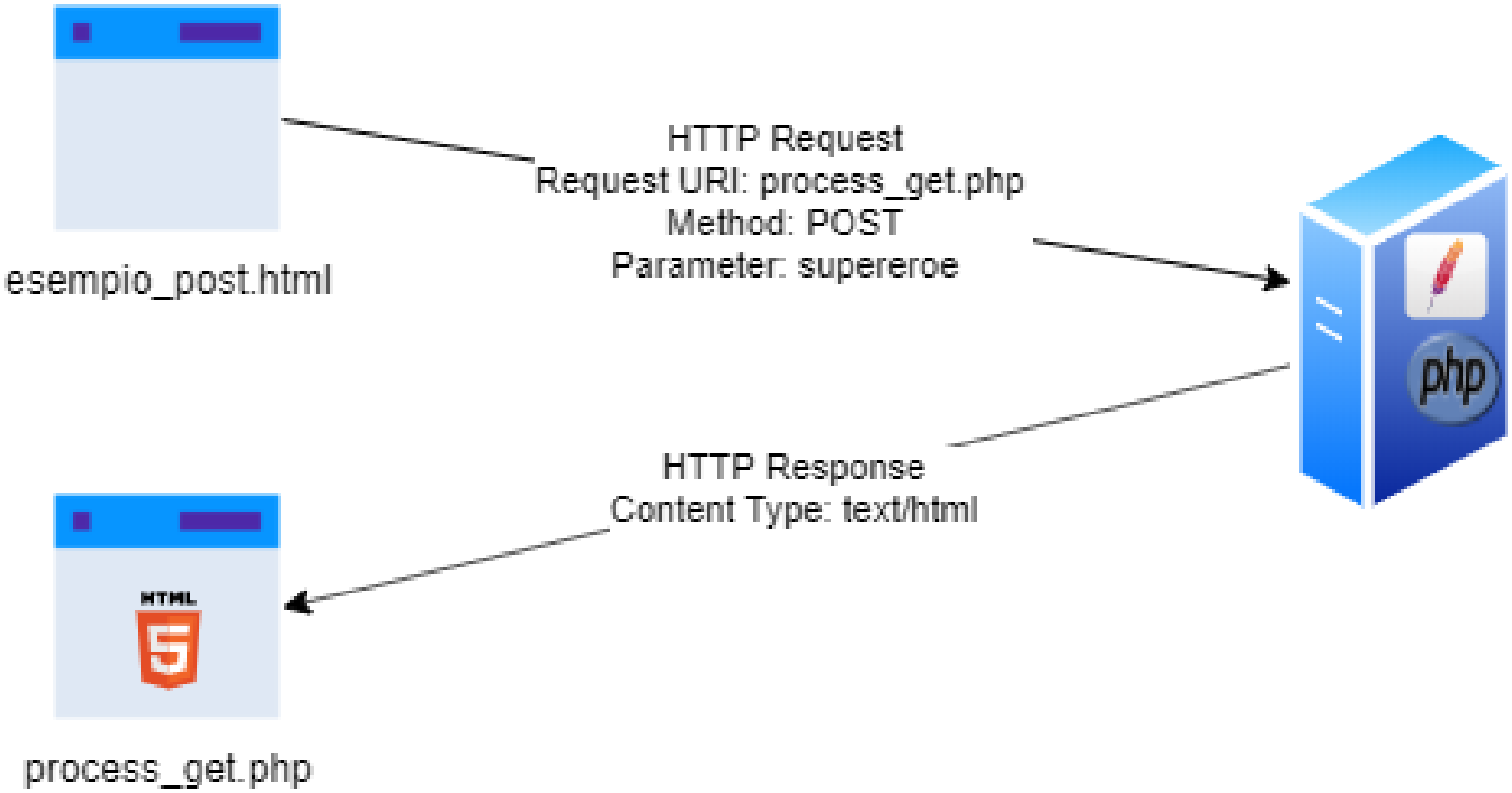
- **esempio_post.html**

```
…
<form action="process_get.php" method="post">
    <label for="idsupereroe">Supereroe</label>
    <input type="text" id="idsupereroe"
name="supereroe">
    <input type="submit">
</form>
…
```

- **process_get.php**

```
…
<p>GET: <?php echo $_GET['supereroe'] ?></p>
<p>POST: <?php echo $_POST['supereroe'] ?></p>
<p>REQ:<?php echo $_REQUEST['supereroe'] ?></p>
…
```

Esempio Post



Output Esempio Post (batman)

```
GET:
Notice: Undefined index: supereroe
in
C:\xampp\htdocs\esempi\esempio_post\
process_get.php on line 8
POST: batman

REQ: batman
```

Esempio Post - Considerazioni

- Valgono le stesse considerazioni dell'esempio precedente.
- I parametri inviati tramite POST sono trasmessi in chiaro?

Esempio Post - Considerazioni

- Valgono le stesse considerazioni dell'esempio precedente.
- I parametri inviati tramite Post sono trasmessi in chiaro?
- **Ovviamente sì!**

Esempio Get v2

- Gli stessi esempi mostrati in precedenza per GET e POST possono essere realizzati anche utilizzando un'unica pagina che si occupa sia di visualizzare il form, se i dati non sono stati inviati, sia di visualizzare eventuali dati.

Gestione dello stato

- Come ricordato HTTP è un protocollo **stateless**, che non tiene traccia delle diverse interazioni che sono parte di una specifica attività.
- Per superare questo limite, si utilizzano alcuni meccanismi che consentono di gestire le singole interazioni che avvengono tra client e server come parte di una stessa attività.
- **PHP** fornisce due strumenti per la **gestione dello stato**:
  1. Cookie
  2. Session

Cookie

- Un cookie consente di salvare un'informazione nel browser dell'utente.
- In PHP è possibile creare un cookie utilizzando la funzione **`setcookie()`**, specificando nome (unico parametro obbligatorio), valore, validità e percorso.

Esempio Cookie

Si vuole salvare in un cookie il numero di volte in cui l'utente ha visitato il sito web.

- Esempio:

```
$nome_cookie = "numero_accessi";
if(!isset($_COOKIE[$nome_cookie])) {
    echo "Cookie '" . $nome_cookie . "' non settato!
       Lo setto adesso!";
    $valore_cookie = 1;
    setcookie($nome_cookie, $valore_cookie, time() +
       (60 * 60 * 24 * 30), "/");
} else {
    echo "Cookie '" . $nome_cookie . "' settato!<br>";
    $num_visite = $_COOKIE[$nome_cookie]+1;
    setcookie($nome_cookie, $num_visite, time() +
       (60 * 60 * 24 * 30), "/");
    echo "Il sito è stato visitato: ".$num_visite." volte!";
}
```

Cookie (2)

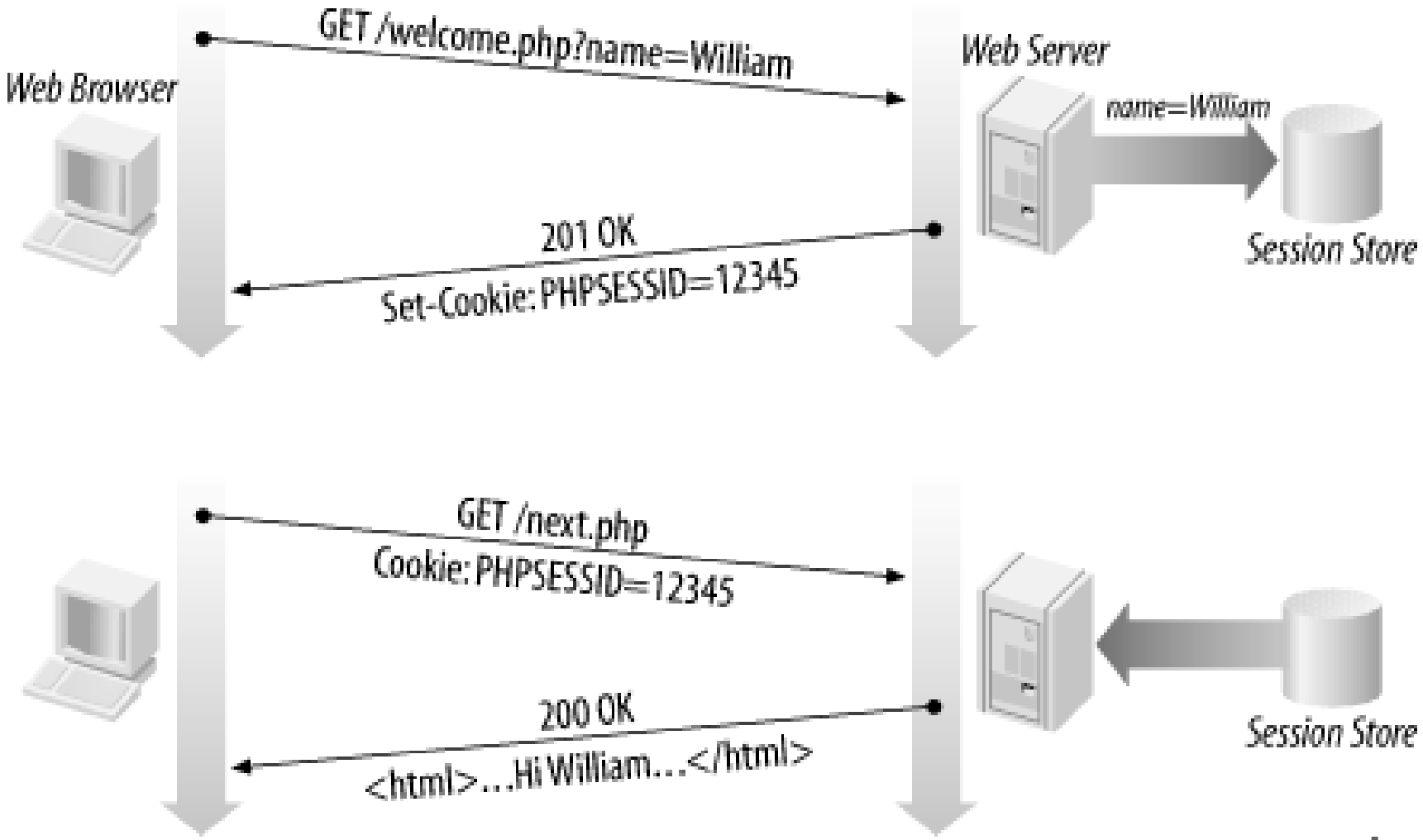
- È possibile cancellare un cookie semplicemente impostando un tempo di validità «passato».
- Esempio:

```
$nome_cookie = "numero_accessi";
setcookie($nome_cookie, "", time() - 100, "/");
echo "Cookie '" . $nome_cookie . "' cancellato!";
```

Session in PHP

- Con una session è possibile salvare un'informazione direttamente sul server.
- Al browser viene assegnato un identificatore di sessione che viene registrato in un cookie (**`PHPSESSID`**) sul browser dell'utente.
- Alla successiva interazione HTTP, PHP controllerà automaticamente se un id è stato inviato con la richiesta. Se l'id è stato inviato, PHP controlla se esiste una sessione con quell'id e, in caso positivo, rende accessibili le informazioni salvate aggiungendole alla variabile super globale **`$_SESSION`**.
- La gestione del cookie lato client avviene in maniera trasparente.
- Una sessione rimane aperta fino a quando il browser non viene chiuso.

Esempio Session



Session in PHP

- È possibile salvare una variabile nella sessione in due modi:
  - Usando direttamente la variabile super globale **`$_SESSION`**:
    ```
    $_SESSION['name'] = "William";
    ```
  - Usando la funzione **`session_register()`**:
    ```
    $name = "William";
    session_register("name");
    ```
- Per rimuovere dati è possibile usare:
  - La funzione **`unset()`** per rimuovere una singola variabile
    ```
    unset($_SESSION['name'])
    ```
  - La funzione **`session_unset()`** per rimuovere TUTTE le variabili
  - La funzione **`session_destroy()`** per rimuovere tutte le informazioni della sessione (non solo le variabili) ma NON il cookie **`PHPSESSID`** (da usare con cautela!).

Session in PHP

- *Una sessione rimane aperta fino a quando il browser non viene chiuso.*
- Ma come fa il server a sapere che il browser è stato chiuso?

Session in PHP

- *Una sessione rimane aperta fino a quando il browser non viene chiuso.*
- Ma come fa il server a sapere che il browser è stato chiuso?
- **Semplicemente non lo sa!**
- Ha un suo parametro di configurazione (che si trova nel file **`php.ini`**) **`session.gc_maxlifetime`** che indica il tempo di vita di una sessione. Scaduto quello, la sessione viene ritenuta scaduta e ci sarà un garbage collector che si occuperà di liberare la memoria.
- Lato browser invece, il cookie **`PHPSESSID`** è impostato per scadere alla chiusura del browser.

Prima domanda

DOMANDA 1 :

Cosa contiene la variabile super globale $_REQUEST[] in PHP?

- ❑ E' un array associativo che contiene il contenuto delle variabili super globali $_GET, $_POST e $_COOKIE
- ❑ E' un array associativo che contiene tutti i log del server
- ❑ E' una variabile speciale che contiene le informazioni riguardanti headers e percorsi assoluti del server
- ❑ E' una variabile speciale che permette di modificare i tipi di richieste accettabili dal server

PHP e MySQL

- Perché MySQL?
  - È il DBMS più usato in ambito web.
  - È cross-platform
  - Usa lo standard SQL
  - È gratuitamente scaricabile ed usabile
  - È sviluppato, distribuito e supportato da Oracle

MySQL

- È un DBMS relazionale.
- Le interrogazioni si esprimono usando SQL. Trovate una guida su SQL sul W3Schools http://www.w3schools.com/sql/default.asp
- Può essere utilizzato per memorizzare **enormi** quantità di dati (con innoDB una tabella può pesare fino a 64TB).

PHP connesso a MySQL

- PHP 5 e successivi possono lavorare con MySQL usando
  - ~~MySQL API~~ (deprecate)
  - MySQLi API (i sta per improved)
  - PDO (PHP Data Object)
- Quale dei due scegliere?
  - Ovviamente dipende dal contesto!
  - Entrambi hanno vantaggi/svantaggi: PDO si può usare con 12 differenti DB mentre MySQLi solo con MySQL ma ha prestazioni leggermente migliori
- In laboratorio useremo MySQLi.

MySQLi

- MySQLi mette a disposizione API sia object-oriented che procedurali. Noi useremo la versione object-oriented.
- Esempio:

```
$servername = "localhost";
$username = "username";
$password = "password";
$sql = "CREATE DATABASE dbname;";

//object-oriented
$conn = new mysqli($servername, $username, $password);
$conn->query($sql) === TRUE

//procedural
$conn = mysqli_connect($servername, $username, $password);
mysqli_query($conn, $sql)
```

Aprire/chiudere una connessione

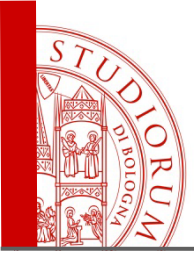
```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "dbname";

// Aprire la connessione
//$conn = new mysqli($servername, $username,
$password, $dbname);

//Chiudere la connessione
$conn->close();
?>
```

Eseguire una query

- È possibile eseguire una query SQL di **qualsiasi tipo** utilizzando il metodo *query()*.
- Ovviamente il risultato sarà diverso in base al tipo di query eseguito:
  - Query per la creazione di database o tabelle restituiranno semplicemente TRUE o FALSE.
  - Query per la selezione di dati restituiranno i dati.



Prepared Statement

- Per motivi di sicurezza, è preferibile usare i prepared statement al posto del metodo query.
- Con i prepared statement inizialmente la query viene creata con dei placeholder, che vengono valorizzati successivamente.
- Esempio:

```
$stmt = $conn->prepare("SELECT *
FROM table WHERE columnname = ?");
$stmt->bind_param('s',$variabile);
$stmt->execute();
```

Prepared Statement

- Quando si esegue il binding è necessario specificare il tipo di parametro.
- Valori ammessi:
  - i: interi
  - d: double
  - s: stringhe
  - b: BLOB

Creazione Tabelle

- In uno degli esempi precedenti avete visto come creare il db.
- La creazione del db, così come quella delle tabella, è buona norma NON gestirle con PHP ma con strumenti più adatti, ad esempio MySQL Workbench https://www.mysql.com/it/products/workbench/

Domande?