PHP
introduzione al linguaggio

Laboratorio di basi di dati
Stefano Montanelli
Dipartimento di Informatica
Università degli Studi di Milano
http://islab.di.unimi.it/bdlab1

Configurazione di PHP

- Le direttive di configurazione dell'interprete sono contenute nel file **php.ini**
- E' possibile visualizzare la configurazione in uso mediante la funzione phpinfo

  <?php phpinfo() ?>
- Alcune direttive possono essere modificate localmente ad uno script con la funzione ini_set()

  ini_set("max_execution_time", "60");
- La modificabilità di una direttiva dipende dal livello di permesso a cui è associata:
  - *PHP_INI_SYSTEM* - php.ini o httpd.conf
  - *PHP_INI_PERDIR* - php.ini, .htaccess o httpd.conf
  - *PHP_INI_USER* - script utente o registro di sistema
  - *PHP_INI_ALL*

Configurazione di PHP

- Alcune direttive di configurazione
  - **max_execution_time**. Tempo massimo di esecuzione dello script
    - Default = 30 (secondi)
    - Livello di modifica = PHP_INI_ALL
  - **upload_max_filesize**. Definisce la dimensione massima di dati in upload
    - Default = 2M
    - Livello di modifica = PHP_INI_PERDIR
  - **post_max_size**. Definisce la dimensione massima di dati inviabili in POST (deve essere maggiore di upload_max_filesize)
    - Default = 8M
    - Livello di modifica = PHP_INI_PERDIR

Configurazione di PHP

- Altre direttive di configurazione
  - **upload_tmp_dir**. Directory in cui vengono temporaneamente salvati i file in upload
    - Default = NULL
    - Livello di modifica = PHP_INI_SYSTEM
  - **display_errors**. Definisce se gli errori devono essere riportati come parte dell'output
    - Default = Off
    - Livello di modifica = PHP_INI_ALL

Caratteri speciali

- Separatori. Identificano il codice PHP all'interno di un file

  <?php <codice> ?>

- Commenti:

  // e # per commenti su singola riga

  /* commenti

  multi-riga */

- Il terminatore di riga è il punto e virgola (;)

Variabili – 1

- Le variabili sono indicate con il prefisso **$**
- I nomi delle variabili sono case sensitive
- Un nome di variabile deve iniziare con una lettera o con il simbolo tratto basso o underscore (_) seguiti da una serie qualsiasi di lettere, numeri o altri underscore
- PHP è un linguaggio **loose typed**, quindi le variabili non vengono definite dal programmatore, ma il loro tipo è deciso a runtime dall'interprete in base al contesto

Variabili – 2

- La funzione **gettype($<nomevar>)** restituisce il tipo a runtime della variabile $<nomevar>
  - $var = “una stringa”;
    Gettype($var) //restituisce string
- Per ogni tipo di dato <typename>, esiste una funzione booleana **is_<typename>($<nomevar>)** che restituisce true/false se $<nomevar> è di tipo <typename> a runtime
  - $var = 9;
    is_int($var) // restituisce true
  - $var = 9;
    is_string($var) // restituisce false
  - is_null($var) // restituisce true se $var è nulla

Tipi di dato supportati

- Tipi di dato scalari
  - Boolean
  - Integer
  - Float
  - String
- Tipi di dato complessi
  - Array
  - Object
- Tipi di dato speciali
  - Resource
  - Null

Tipo di dato boolean

- Può assumere solo 2 valori
  - $var = true; //true e false sono case insensitive
  - $var = false;
- I seguenti tipi vengono valutati false se convertiti in boolean
  - integer 0 e float 0.0
  - string “” e string “0”
  - array vuoto (zero elementi)
  - Il tipo speciale NULL
- Tutti gli altri oggetti vengono valutati true se convertiti in boolean

Tipo di dato integer

- Può assumere valori interi positivi e negativi
- Possono essere specificati in base decimale, ottale o esadecimale
  - $int = 35; // decimale positivo
  - $int = -35; // decimale negativo
  - $int = 043; // ottale
  - $int = 0x23; // esadecimale
- Il valore boolean True viene valutato 1 se convertito in Integer

Tipo di dato float

- Può assumere valori decimali positivi e negativi
- Anche noto come tipo di dato double o real
  - $float = 1.3245;
  - $float = 1.3e2;
  - $float = 7E-10;

Tipo di dato string

- Una stringa è una sequenza arbitraria di caratteri ASCII.
- Può essere specificata
  - con apici singoli per indicare una stringa statica
  - $string = ‘esempio di stringa statica’;
  - Con apici doppi per indicare una stringa che può contenere caratteri di controllo e la sostituzione delle variabili
    - $nome = ‘Stefano’;
      $string = “il nome è $nome \n”;
      print ($string); /* stampa: *il nome è Stefano* e termina la riga con il carattere a capo */

Tipo di dato Array – 1

- Un array in PHP è una mappa ordinata di chiavi e valori
- Può avere come valori uno o più tipi di dato PHP, quindi anche un altri array
- Si accede ai suoi elementi tramite funzioni o con l'operatore [ ]

Tipo di dato Array – 2

- Usare la funzione *array* per creare un array:
  - $myarray = array(1=>‘Paolo’, 2=>‘Rossi’);
- Le chiavi e i valori di un array possono avere tipi di dato diversi
  - $myarray = array(‘numero’=>23, 2=>‘sole’);
- Se la chiave di un elemento non è specificata, sarà automaticamente utilizzato il valore più alto delle chiavi numeriche aumentato di 1
  - $myarray =
    array(‘tre’, ‘autore’=>‘io’);
  - Produce come risultato il seguente array:
    - array(13=>‘uno’, 16=>‘due’)
  - $myarray =
    array(13=>‘uno’, 16=>‘due’, 17=>‘tre’, ‘autore’=>‘io’);

Tipo di dato Array – 3

- L'operatore [ ] permette di accedere o assegnare elementi di un array
  - $ar[<chiave>]=<val>;
  - Se l'array $ar non esiste viene creato
  - Se l'elemento con indice <*chiave*> non esiste viene creato, altrimenti il valore <val> è assegnato all'elemento $ar[<chiave>]
- L'operatore [] consente di aggiungere elementi a un array
  - $ar[] = <val> /* un nuovo elemento con valore <val> viene aggiunto all'array */

Operatori per array

- Per rimuovere elementi di un array si utilizza la funzione **unset($<nomevar>)**
  - unset($ar[2]) /* rimuove dall'array $ar l'elemento con chiave 2 */
  - unset($ar) // cancella l'intero array $ar
- Per contare il numero di elementi contenuti in un array si utilizza la funzione **count($<nomevar>)**
  - $ar = array(1 => 'primo', 2 => 'secondo');
  - print count($ar) // visualizza la stringa 2

Operatori di comparazione

- == uguaglianza
- === identità (controlla anche i tipi)
- != <> disuguaglianza
- !== non identità
- < minore di
- > maggiore di
- <= minore uguale
- >= maggiore uguale

Strutture di controllo

- Sono quelle tradizionali:
  - if – else if – else
  - while – do while
  - for
  - foreach
  - switch – case

Focus sul costrutto *foreach*

- Struttura iterativa utile per scorrere un array
  - foreach($ar as $val){ codice; }
    - Esegue un'iterazione per ogni elemento dell'array $ar. $val contiene il valore dell'elemento considerato nell'iterazione corrente
  - foreach($ar as $k => $val) { codice; }
    - Analoga alla versione precedente, ma mette a disposizione anche la variabile $k che contiene la chiave dell'elemento considerato nell'iterazione corrente

Costrutto *break*

- Fa terminare l'esecuzione della struttura di controllo in cui viene invocata
- Può essere utilizzata in una delle seguenti strutture di controllo
  - for
  - foreach
  - while
  - do...while
  - switch

Costrutto *continue*

- Usato all'interno di una struttura iterativa, consente di saltare le rimanenti istruzioni dell'iterazione e passare alle istruzioni dell'iterazione successiva

```
for ($i = 1; $i <= 10 ; $i++) {
    if ($i == 5) {
            continue;
    }
    print($i);
}
// Stampa i numeri da 1 a 10 saltando il 5
```

Variabili superglobals – 1

- Il PHP mette a disposizione alcune variabili predefinite usate per mettere a disposizione dello script:
  - Variabili d'ambiente
  - Input inserito dall'utente
  - Dati di sessione
- Tali variabili sono chiamate **superglobals**: sono quindi disponibili automaticamente senza necessità di dichiarazione

Variabili superglobals - 2

- **$GLOBALS**
  - Contiene i riferimenti alle variabili globali dello script
- **$_SERVER**
  - Variabili relative al web server e all'ambiente di esecuzione dello
- **$_REQUEST**
  - Variabili trasmesse allo script mediante i meccanismi GET, POST e COOKIE
- **$_GET**
  - Variabili trasmesse allo script tramite il metodo GET
- **$_POST**
  - Variabili trasmesse allo script tramite il metodo POST

Variabili superglobals – 3

- **$_FILES**
  - Variabili trasmesse allo script mediante upload di file attraverso il metodo HTTP POST
- **$_ENV**
  - Variabili di ambiente trasmesse allo script
- **$_COOKIE**
  - Variabili trasmesse allo script dal browser attraverso i cookie
- **$_SESSION**
  - Variabili mantenute in sessione

Passaggio di parametri

- I principali metodi per consentire il passaggio di parametri/variabili da una pagina web a un'altra sono
  - GET
    - I parametri sono esplicitati nella URL
  - POST
    - I parametri provengono da un form con clausola method impostata su POST

Metodo GET

- In PHP, i parametri passati attraverso il metodo GET sono accessibili tramite l'array $_GET
  - http://indirizzo_server/paginaGET.php?id=20&length=3
  - Nel codice PHP della paginaGET.php l'array $_GET contiene i seguenti elementi:
    - $_GET('id' => 20, 'length' => 3)

Metodo POST

- In PHP, i parametri passati attraverso il metodo POST sono accessibili tramite l'array $_POST

```
<form action="paginaPOST.php" method="POST">
        Inserisci il primo elemento:
        <input type="text" name="item1" />
        Inserisci il secondo elemento:
        <input type="text" name="item2" />
        <input type="submit" value="Invia dati">
</form>
```

  - Nel codice PHP della paginaPOST.php l'array $_POST contiene i seguenti elementi:
    - $_POST('item1' => val1, 'item2' => val2)
    - val1 e val2 sono i valori indicati dall'utente nel form

Definizione di funzioni

- Definizione:

```
function nomefunzione($arg1, ...., $argn){
        codice;
        return($ret);
}
```

- Nella definizione, dopo la parola chiave *function* viene specificato il nome della funzione seguito dai parametri (senza specifiche di tipo). L'istruzione *return* fa terminare l'esecuzione della funzione

Caratteristiche delle funzioni

- Il corpo di una funzione può essere un qualunque pezzo di codice PHP valido (anche la definizione di un'altra funzione o di una classe)
- Una funzione può ritornare un qualsiasi tipo di dato
- Non è previsto l'overloading né la ridefinizione di funzioni

Scope delle variabili

- Le variabili definite all'interno di una funzione sono chiamate *variabili locali* e sono visibili soltanto all'interno di quella funzione
- Le variabili definite al di fuori di una funzione sono chiamate variabili globali e sono accessibili attraverso la variabile predefinita $GLOBALS

Passaggio di parametri a una funzione

- I parametri di una funzione sono passati **per valore** (copia)
- Se si desidera passare un parametro **per referenza** (in modo che eventuali modifiche vengano rese disponibili al di fuori dello scope della funzione) occorre utilizzare il simbolo *ampersand* (&) nella segnatura della funzione:

function nomefunzione(&$par1, &$par2)

- Le modifiche eseguite da *nomefunzione* ai paramteri *$par1* e *$par2* sono visibili anche all'esterno della funzione

Riferimenti

- Consultare il manuale e le guide online per
  - Supporto alla sintassi
  - Esempi e casi d'uso
  - Risoluzione di problemi
- Il manuale ufficiale:
  - http://www.php.net/manual/en/