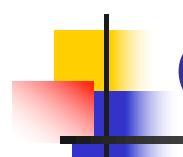


PHP

Introduzione



Caratteristiche (1)

- PHP Hypertext Processor (abbreviato in PHP) è un linguaggio di programmazione che consente di codificare algoritmi all'interno di pagine HTML
- È stato sviluppato a partire dal 1995. L'ultima versione, PHP7, è del 2016 (attualmente 7.2.X)
- Il codice PHP viene sempre eseguito dal lato Server
- È free
- È utilizzabile in molti S.O ...
 - Unix/Linux, Windows, MacOS, ...

Caratteristiche (2)

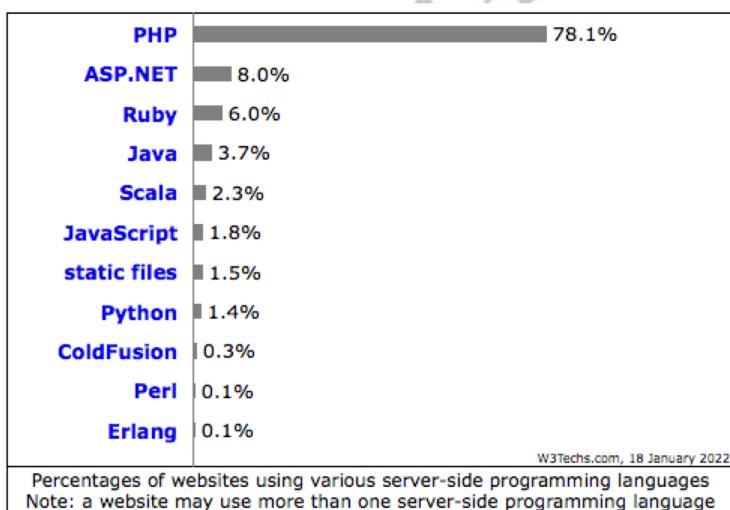
- ... e da diversi tipi di server
 - Apache, IIS, nginx, ...
- È multiparadigma: può essere usato come linguaggio imperativo o "ad oggetti"
- È un linguaggio di programmazione completo, che consente di accedere a tutte le risorse dal lato server

PHP

3

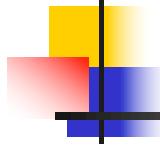
Uso di PHP

W3Techs - World Wide Web Technology Surveys



PHP

4



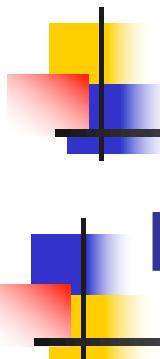
Il Codice PHP

- Per **includere** in codice **PHP** in un documento **HTML**, si può:
 - Utilizzare la forma `<?php <istruzioni_php> ?>` per inserire istruzioni del linguaggio
 - Utilizzare la forma `<?= <testo> ?>` per produrre in output la stringa `<testo>` (forma abbreviata dell'istruzione `echo <testo>`)
- Il codice **PHP** può essere collocato in qualsiasi sezione del documento **HTML** (`<head>`, `<body>`)



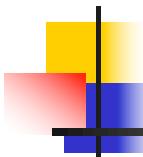
PHP

5



PHP

Variabili e Tipi di Dato



Le Variabili

- Identificatore di variabile

```
<varid>::=$<letter>{<letter>|<digit>|_}
```

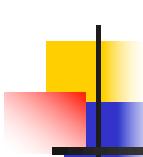
- L'identificatore è '**case sensitive**'
- **Le variabili non vengono dichiarate**, ma **vengono create contestualmente** dell'assegnazione del loro valore

Es: `$indice=10`

- Il **binding variabile-tipo** è **dinamico**

PHP

7



Tipi di Dato

Tipo	Operatori
Integer	+,-,*,/,%,++,--,=,+=,-=,*=,/=%,===%,==!=,!=,!=!=,>,<,>=,<=,?:,&, ,^,~,=>,<>,(int)
Real	Come integer, tranne gli operatori su bit (&, , ^, ~, >>, <<) che mancano e l'operatore di casting (int) che viene sostituito da (double)
Booleano	and, or, xor, &&, , !
Stringhe	., .=, (string)
Array	Dipendenti dal tipo degli elementi, oltre a (array)
Object	new, (object)

PHP

8

Conversione di Tipo

- Conversione stringa --> numero
 - Implicita

```
$Y="3"; $Y*="7"; // $Y=21
```

- Esplicita
 - doubleval(), intval()
- Conversione numero --> stringa

```
$Y=3; $Y.="Tavoli"; // $Y="3Tavoli"
```

PHP

9

Gli Array (1)

- Vengono **definiti** tramite il costrutto

```
array({<elemval>} | {<elemnam>=><elemval>})
```

```
$num=array(1, 17, "uno", 2=>16, "tre");
$num[0] --> 1
$num[1] --> 17
$num[2] --> 16
$num[3] --> "tre"
```

- Possono essere **multidimensionali**

```
$per=array(array("Mario", "Rossi", "Studente", 21),
           array("Giovanni", "Bianchi", "Impiegato", 45));
```

PHP

10

Gli Array (2)

- Possono essere **associativi**, cioè con identificatore dei singoli elementi svincolato dalla loro posizione nell'array

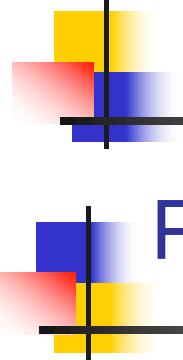
```
$temp=array("Dom"=>15.5, "Lun"=>13.7, "Mar"=>14.9);  
$temp["Dom"] --> 15.5  
$temp["Lun"] --> 13.7  
$temp["Mar"] --> 14.9
```

- Hanno **numerose funzioni** ad essi associate **predefinte nel linguaggio** (`sort()`, `asort()`, `ksort()`, `foreach()`, `list()`, ...)



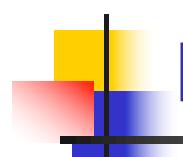
Scope di una Variabile

- In PHP le **variabili** possono essere:
 - **Locali**, definite all'interno di un blocco di programma utilizzabili solo in esso
 - **Globali**, definite all'esterno di un blocco di programma e utilizzabili nel blocco tramite l'uso dell'attributo **global**
- Lo **scope**, in PHP, è quindi **STATICO**



PHP

Funzioni



Le Funzioni

- Dichiarazione

```
function <nome_funzione> ([<parametri>]) {<istruzioni>}
```

```
function somma ($a,$b) {  
    $risultato = $a + $b;  
    return $risultato;  
}
```

- Attivazione: `$c = somma(3,7)`
- Possono essere:
 - Predefinite nel linguaggio
 - Definite in un file esterno (incluse con `require` e `include`)
 - Definite nel documento

Passaggio di Parametri

- In PHP sono possibili due tecniche di passaggio dei parametri:
 - Per valore
`function a ($x, $y) {...}`
 - Per riferimento
`function a (&$x, &$y) {...}`
- L'operatore **&**, che definisce un alias a livello di symbol-table di una variabile, può essere usato in qualsiasi istruzione PHP



Funzioni Predefinite (1)

Funzioni predefinite per gli array

array()	arsort()	krsort()
array_keys()	asort()	list()
array_merge()	count()	rsort()
array_reverse()	current()	sizeof()
array_shift()	each()	sort()
array_slice()	end()	...
array_values()	ksort()	

Funzioni Predefinite (2)

Funzioni matematiche predefinite

abs()	cos()	pow()
acos()	decbin()	rand()
asin()	dechex()	sin()
atan()	exp()	sqrt()
base_convert()	log()	tan()
bindec()	max()	...
ceil()	min()	

PHP

17

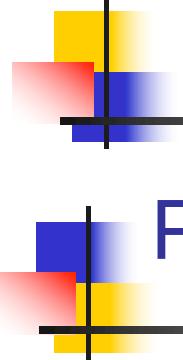
Funzioni Predefinite (3)

Funzioni predefinite per le stringhe

chr()	ord()	strcmp()
count_chars()	printf()	strpos()
crypt()	rtrim()	stristr()
echo()	sscanf()	substr()
explode()	strchr()	wordwrap()
implode()	strcmp()	...
ltrim()	strlen()	

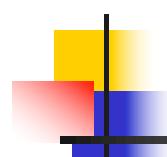
PHP

18



PHP

Strutture di Controllo Condizionale ed Iterativo

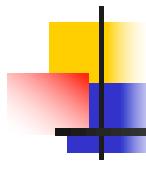


IF - IF ... ELSE

- Sintassi

```
if (<espressione_condizionale>) {  
    <istruzioni>;  
}
```

```
if (<espressione_condizionale>) {  
    <istruzioni>;  
}  
else {  
    <istruzioni>;  
}
```



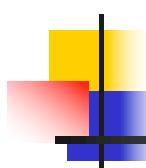
SWITCH

- Sintassi

```
switch(<espressione>) {  
    case <etichetta>: <istruzioni>; break;  
    case <etichetta>: <istruzioni>; break;  
    ...  
    default: <istruzioni>;  
}
```

PHP

21



WHILE – DO ... WHILE

- Sintassi

```
while (<espressione_condizionale>) {  
    <istruzioni>;  
}
```

```
do {  
    <istruzioni>;  
} while (<espressione_condizionale>);
```

- Le **<istruzioni>** vengono ripetute fintanto che l'**<espressione_condizionale>** è **vera**

PHP

22

FOR - FOREACH

- Sintassi

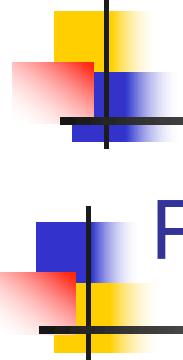
```
for (<inizializzazione>; <condizione>;<step>) {  
    <istruzioni>;  
}  
  
foreach <espressione_array> { <istruzioni>; }
```

- **foreach** consente di scandire gli elementi di un array e di usarne i valori all'interno del ciclo



BREAK, CONTINUE

- **break** consente di interrompere un ciclo iterativo prima che la condizione di fine ciclo sia verificata
- **continue** permette di passare direttamente alla successiva iterazione di un ciclo



PHP

Gestione contenuti delle FORM

PHP e FORM (1)

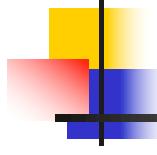
- La gestione dell'input dei dati provenienti da una FORM è estremamente semplice in PHP
 - Ad ogni elemento di <form> con name=<nome> corrisponde, in PHP, una variabile superglobale predefinita:

`$_GET["<nome>"]` Per il metodo GET

`$_POST["<nome>"]` Per il metodo POST

`$_REQUEST["<nome>"]` Per qualsiasi metodo

- Se più elementi della FORM condividono lo stesso nome (<select name=<nome> multiple>) è sufficiente assegnare ad essi in un attributo name="<nome>" perché i valori immessi siano memorizzati nell'array PHP `$_<metodo>["<nome>"]`



Passaggio di Parametri

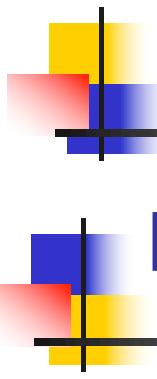
- Con meccanismo **analogo** a quello usato per l'acquisizione di dati da una **form**, è possibile **passare parametri** ad un programma **PHP**

HTTP://localhost/PHP6.php?a=1&b=due



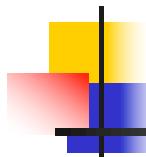
PHP

27



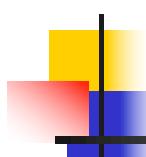
PHP

Interfacciamento a MySQL



PHP e DB

- È possibile configurare il linguaggio **PHP** per inglobare una serie di funzioni che consentono l'accesso alle primitive di gestione di un DataBase
- Nella fase di **installazione** del **PHP**, può essere specificato il **DB** al quale il linguaggio si interfaccia, scelto tra:
 - MySQL
 - PostgreSQL
 - SQLite
 - Oracle
 - ...



MySQL (1)

- È un **RDBMS free**, nato per piattaforme **Linux**, che supporta il linguaggio **SQL**
 - Robusto
 - Affidabile
 - In grado di gestire DB di notevoli dimensioni
 - Con una buona gestione della sicurezza dell'accesso
- Fork del progetto: **MariaDB** (licenza **GNU GPL**)

MySQL (2)

Comandi MySQL

create database
create index
create table
delete
drop database
drop index
drop table

insert
join
load data infile
replace
select
show databases
show index

show tables
show variables
update
use
...

PHP

31

Esempi d'uso (1)

```
> create database biblio;  
> use biblio;  
> create table libri (titolo char(25),autore char(20),  
cod int, prezzo double(5,2), data_vendita date);  
> describe libri;  
> insert into libri values('Il Nome della Rosa','Umberto  
Eco',1,24.32,'2017-04-10');  
> insert into libri (titolo,prezzo) values ('HTML',32.00);  
> select * from libri;  
> select titolo, cod from libri;
```

PHP

32

Esempi d'uso (2)

```
> select * from libri where titolo='HTML' and  
    prezzo > 30.00;  
  
> select * from libri where titolo like 'HTML%';  
  
> select * from libri order by titolo desc;  
  
> update libri set prezzo=34.00 where  
    titolo='HTML';  
  
> delete from libri where titolo='HTML';  
  
> alter table libri change cod codice int not null;  
  
> alter table libri add unique(codice);
```

PHP

33

Interfaccia PHP-MySQL: mysqli

- **mysqli** è un'estensione PHP che fornisce API (con interfaccia Object Oriented o procedurale) per l'accesso al DBMS MySQL
- Alcune **funzioni** dell'interfaccia **procedurale**.
 - `mysqli_affected_rows()`
 - `mysqli_close()`
 - `mysqli_connect()`
 - `mysqli_fetch_assoc()`
 - `mysqli_fetch_field()`
 - `mysqli_free_result()`
 - `mysqli_num_rows()`
 - `mysqli_query()`
 - `mysqli_select_db()`
 - `mysqli_field_count()`

PHP

34

Esempio

- Visualizzazione dei record in una tabella di un DB

The screenshot shows a web browser window titled "Visualizza Dati DB". The URL bar shows "localhost/Esempi_T...". The main content is a table titled "Visualizzazione record nella tabella Libri". The table has columns: Codice, Titolo, Autore, Prezzo, and Data Ultima Vendita. There are two rows of data:

Codice	Titolo	Autore	Prezzo	Data Ultima Vendita
2	La Divina Commedia	Dante Alighieri	16.00	2022-04-15
21	Il nome della rosa	Umberto Eco	12.50	2022-03-30

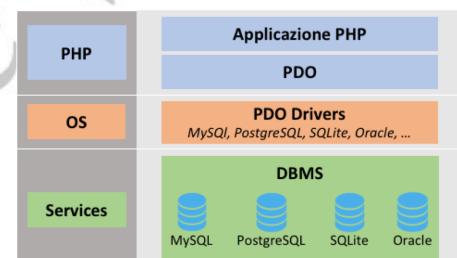


PHP

35

Interfaccia PHP-MySQL: PDO

- PHP Data Object: meccanismo di interfaccia PHP-DBMS che consente l'astrazione dei dati del DB in ambiente PHP in modo indipendente dal DBMS
 - Le istruzioni PHP per l'accesso e la manipolazione dei dati sono generiche e non cambiano in relazione al DB
- Un'istanza della classe PDO è un oggetto PHP che rappresenta la connessione al DB



```
$dsn = "mysql:host=$host;dbname=$db";  
$pdo = new PDO($dsn, $user, $password, $options)  
...  
$pdo = null;
```

PHP

36

PDO

- Le operazioni per l'accesso ai dati sono metodi dell'oggetto PDO

- exec()*: esegue l'istruzione SQL passata come parametro e ritorna il numero di righe del DB interessate. Usata tipicamente per INSERT, DELETE, UPDATE

```
$count = $pdo->exec('DELETE FROM libri WHERE Titolo = "La Divina Commedia"')
```

- query()*: esegue l'istruzione SQL passata come parametro e ritorna i dati prodotti. Usata tipicamente per SELECT

```
$count = $pdo->query('SELECT * FROM libri')
```

- prepare()*: prepara per l'esecuzione l'istruzione SQL passata come parametro creando una sorta di template che ammette l'uso di parametri.

```
$cmd = $pdo->prepare('SELECT * FROM libri WHERE codice = ?')
```

PHP

37

PDO

- I metodi *query()* e *prepare()* producono come risultato un oggetto PDOStatement, con i seguenti metodi:
 - bindParam(\$param, \$value)*: associa il valore \$value al parametro \$param definito nella *prepare()*
 - execute(\$params)*: esegue un'istruzione prepared con valore dei parametri \$parms e ritorna un booleano
 - fetch(\$mode)*: recupera il successivo elemento dal set di quelli estratti dal DB
 - fetchAll(\$mode)*: recupera tutti gli elementi estratti dal DB
- I parametro \$mode dei metodi *fetch* definisce il formato dei dati recuperati

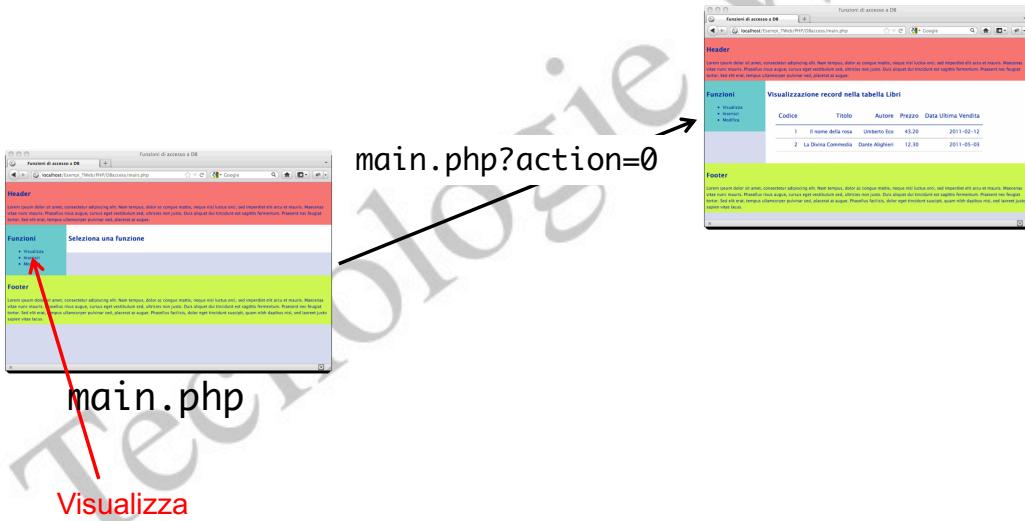
PDO::FETCH_ASSOC, PDO::FETCH_NUM, PDO::FETCH_OBJ, ...



PHP

38

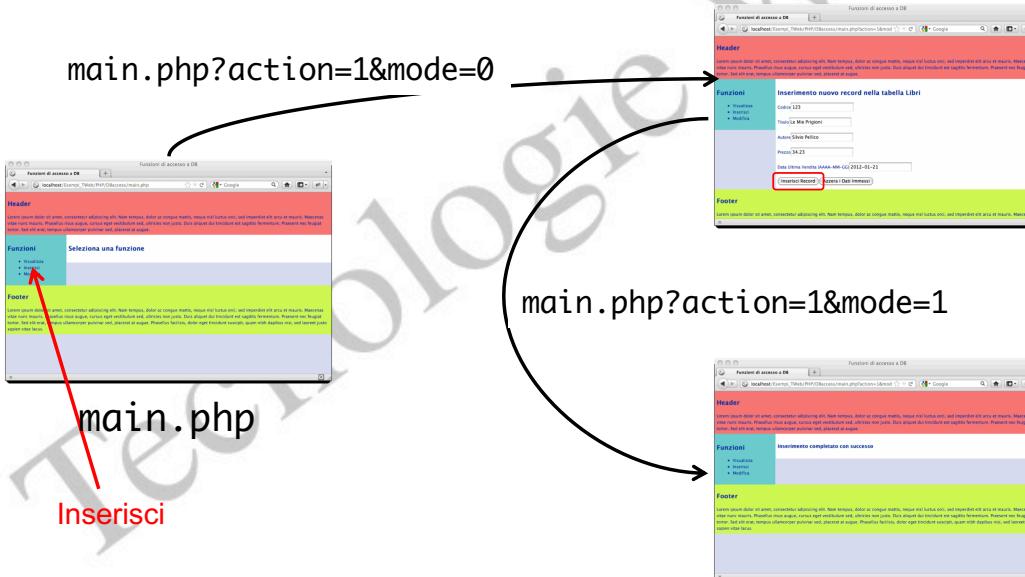
Gestione Integrata: Visualizza



PHP

39

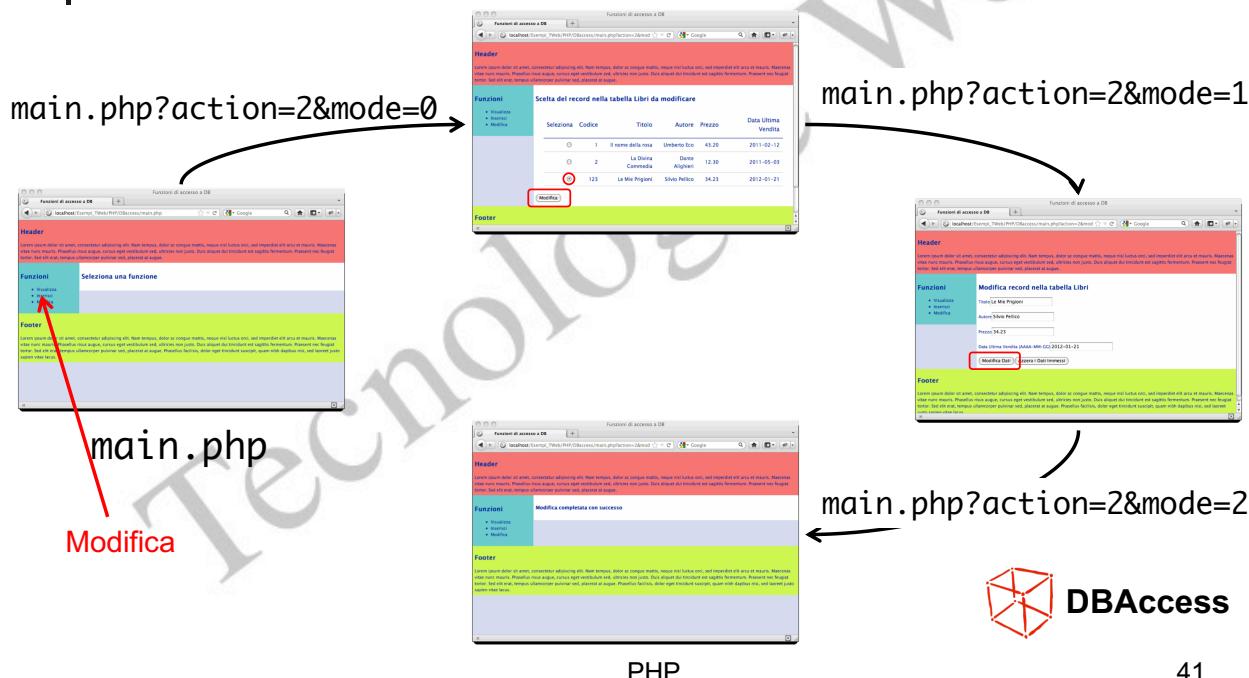
Gestione Integrata: Inserisci



PHP

40

Gestione Integrata: Modifica



41

Accesso Autorizzato

- **Problema:** implementare un meccanismo per l'accesso alle pagine ai soli **utenti autorizzati**
- **Soluzione:**
 - Definire il profilo degli utenti del sito
 - Identificare in modo permanente gli utenti che accedono al sito (**Autenticazione**)
 - Generare le pagine ad accesso controllato solo dopo aver verificato che l'utente collegato sia autorizzato (**Autorizzazione**)



PHP

42