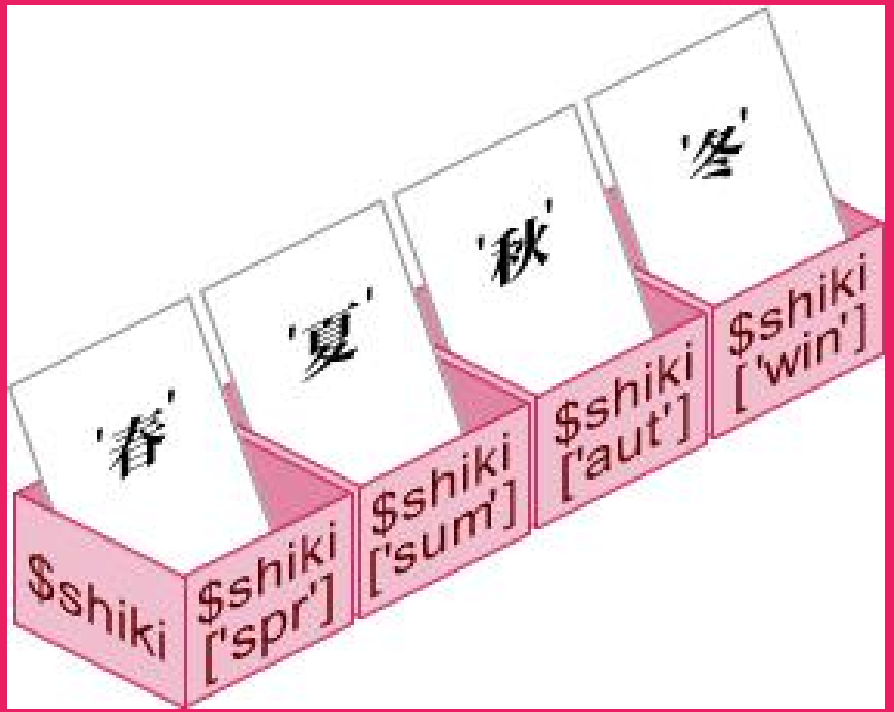
web разработка

PHP работа с форми

Съдържание

- Масиви
- HTTP протокол
- GET
- POST
- GET vs POST
- user input
- PHP сесии

Асоциативни масиви



Масиви

Масивът е променлива, която може да съхранява едновременно повече от една стойност.

```
$cars1 = "Volvo";
$cars2 = "BMW";
$cars3 = "Toyota";
```

този списък с променливи, може да бъде обединен в една - масив $cars

```
$cars = array("Volvo", "BMW", "Toyota")
```

елементи на масива

$cars = ["Volvo", "BMW", "Toyota"] - масив с три елемента.

```
$cars = array("Volvo", "BMW", "Toyota")
```

Достъпваме елементите в масива чрез индекси, започвайки броенето от 0.

елемент с индекс 0 - "Volvo" - $cars[0]
елемент с индекс 1 - "BMW" - $cars[1]
елемент с индекс 2 - "Toyota"- $cars[2]

<?php echo $cars[0];?>//”Volvo”

Не е задължително данните в масива да са от един и същи тип!

Асоциативни масиви

Асоциативните масиви са масиви, чиито индекси са имена/стрингове (ключ).

В декларацията за масива изреждаме ключ => стойност за всеки елемент от масива.

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

или

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

Асоциативни масиви - 2

```
$user_info = ["username"=>"Peter22", "password"=>"33237", "first_name"=>"Peter", “second_name” => “Petrov”, “age”=>35];
```

```
$user_info = ["username"=>"Peter22",
                "password"=>"33237",
                "first_name"=>"Peter",
                “second_name” => “Petrov”,
                “age”=>35];
```

Асоциативни масиви - 3

```
$user_info ["username"] = "Peter22";
$user_info ["password"] = "33237";
$user_info ["first_name"] = "Peter";
$user_info [“second_name”] = “Petrov”;
$user_info [“age”] = 35;
```

Опечатване на целия масив

```
print_r($user_info)
```

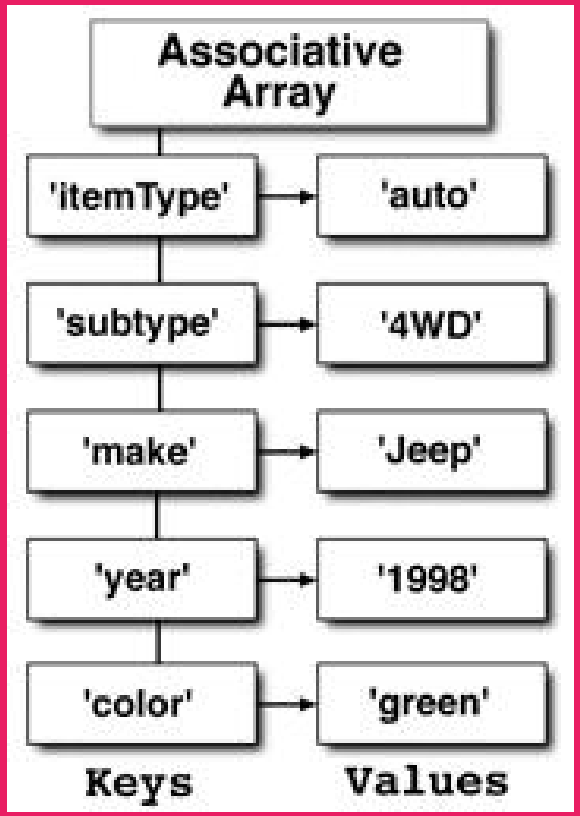
Отпечатване на елемент от масива

```
echo $user_info [“age”]
```

В процеса на работа достъпваме масива

```
var_dump($user_info)
```

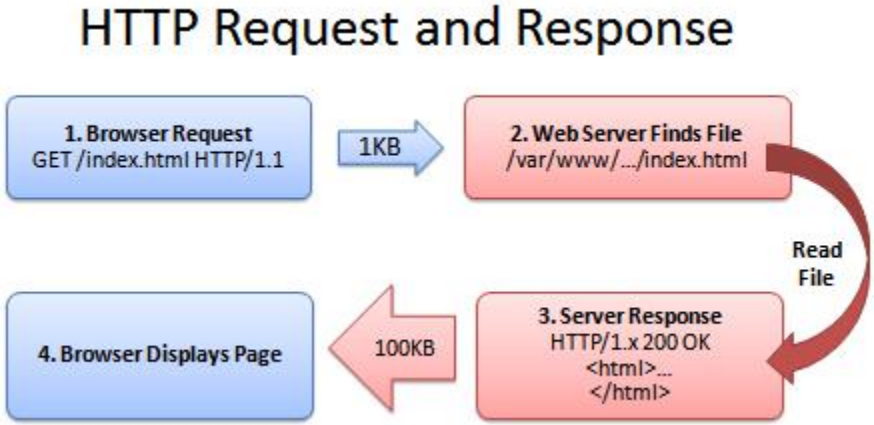
Асоциативни масив
демо
задачи



http протокол

http протокол

Работи с request & response



Stateless протокол, т.е. всяка заявка е независима от предходните

http протокол - 2

Действия: GET, POST

GET: извличане от съществуващ ресурс чрез URL

POST: създава нов ресурс

Статус кодове:

1xx: Informational Messages

2xx:Successful

3xx: Redirection

4xx: Client Error

5xx: Server Error

Глобални масиви

Глобални масиви

$_GET[] <form action=“index.php” method=“get”>

$_POST[] <form action=“index.php” method=“post”>

$_FILE[] <form action="upload.php" method="post" **enctype="multipart/form-data"**>
Select image to upload:
<input type="file" name="fileToUpload" id="fileToUpload">
<input type="submit" value="Upload Image" name="submit">
</form>

$_SESSION[]

работа с форми

работа с форми

html

```
<form action="index.php" method="get">
    <input type="text" name="age" />
    ….
    <input type="submit" name="submit" value="Send" />
</form>
```

php

```
<?php $age = $_GET['age'];

    /*след което използваме $age в решението на поставена задача*/
?>
```

работа с форми - 2

<form action="new_file.php" method="get">

с action="new_file.php"

- отвеждаме към нов php файл

action="same_file.php"

action=""

- оставаме в същия файл за да обработим, постъпилите с формата данни

работа с форми - get vs. post

<form action="file.php" method="get"> vs. <form action="file.php" method="post">

GET requests can be cached	POST requests are never cached
GET requests remain in the browser history	POST requests do not remain in the browser history
GET requests can be bookmarked	POST requests cannot be bookmarked
GET requests should never be used when dealing with sensitive data - passwords, credit-card numbers	Use POST requests instead
GET requests should be used only to retrieve data	
GET requests have length restrictions - up to 250 signs	POST requests have no restrictions on data length

работа с форми
демо + задачи

Данните, постъпващи с формата …
/user input/

user input

Видове полета за въвеждане на данни от потребителя и как “събираме” и работим с тези данни?

input type = “text”
input type="hidden"
input type="radio"
input type = “checkbox” - A checkbox is only submitted if it's actually checked

Създайте валидна форма с тези полета.
Тествайте c method = “get”
c method = “post”
Какво получавате след изпращане на формата?

user input

```
<form action="index.php" method="get">
<input type="text" name="age" />
<input type="submit" name="submit" value="Send" />
</form>
```

```
<?php $age = $_GET['age'];

if ($age < 18) {

echo "You are not allowed here!";

} else {

echo "Welcome, user!";

}
?>
```

User input
демо + задачи