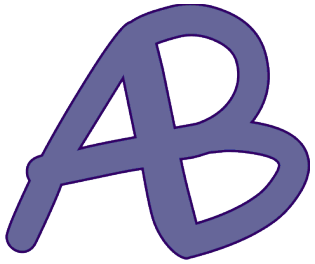


I. PHP

dr Artur Bartoszewski
UTH Radom

Język PHP

Tablice c.d.

Przeglądanie tablicy o znanej liczbie elementów

```php
<?php
for ($i = 0; $i < 10; $i++) {
  echo $tablica[$i];
}
?>
```

Przeglądanie tablicy, gdy nie znamy liczby elementów

```php
<?php
for ($i = 0; $i < sizeof($tablica); $i++) {
  echo $tablica[$i];
}
?>
```

Przeglądanie tablicy asocjacyjnej

```php
<?php
foreach ($tablica as $klucz => $wartosc) {
  echo "$klucz => $wartosc <br />";
}
?>
```

Zamiana tablic na stringi i odwrotnie

Funkcja explode() rozdziela ciąg znaków i tworzy z powstałych elementów tablicę.
Pobiera dwa argumenty:

- ciąg znaków lub znak, który stanowi element rozdzielający,
- zmienną zawierającą ciąg, który należy rozdzielić.

Przykład: Podział tekstu na pojedyncze zdania - separatorem będzie kropka

```
<?php
$tekst = "Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum convallis, quam nec malesuada rhoncus, odio metus vestibulum quam, in laoreet ex odio scelerisque mi. Sed sit amet egestas augue. Proin fermentum sit amet nibh eget facilisis. Suspendisse ultrices et lacus at placerat";

$tablica = explode('.', $tekst);
foreach($tablica as $wpis){
  echo "$wpis <br>";
}
?>
```

Zamiana tablic na stringi i odwrotnie

Funkcja implode() pobiera dwa argumenty:

- ciąg znaków, którymi mają być połączone elementy tablicy,
- nazwa zmiennej tablicy z danymi.

Przykład: Zapis do jednego łańcucha tekstu wszystkich elementów tablicy rozdzielonych średnikami

```php
<?php
$tablica = array("Tekst 01", "Tekst 02", "Tekst 03", "Tekst 04");

$tekst = implode(";", $tablica);

echo $tekst;
?>
```

Zamiana tablic na stringi i odwrotnie

Funkcja implode() pobiera dwa argumenty:

- ciąg znaków, którymi mają być połączone elementy tablicy,
- nazwa zmiennej tablicy z danymi.

Przykład: Zapis do jednego łańcucha tekstu wszystkich elementów tablicy rozdzielonych średnikami

```
<?php
$tablica = array("Tekst 01", "Tekst 02", "Tekst 03", "Tekst 04");

$tekst = implode(";", $tablica);

echo $tekst;
?>
```

Sortowanie tablic

Istnieje wiele funkcji sortujących tablicę oto kilka z nich:

- asort() – sortuje rosnąco tablice asocjacyjne według wartości kluczy, zachowując przypisanie kluczy do wartości,
- arsort() – sortuje malejąco tablice asocjacyjne według wartości kluczy, zachowując przypisanie kluczy do wartości,
- ksort() – sortuje rosnąco tablice asocjacyjne według kluczy, zachowując przypisanie kluczy do wartości,
- krsort() – sortuje malejąco tablice asocjacyjne według kluczy, zachowując przypisanie kluczy do wartości,
- sort() – sortuje rosnąco zwykłe tablice,
- rsort() – sortuje malejąco zwykłe tablice,
- uasort() – funkcja sortująca tablice asocjacyjne za pomocą zdefiniowanej przez użytkownika funkcji porównującej elementy (nazwa funkcji jest podawana za pomocą drugiego parametru),
- usort() – funkcja sortująca zwykłe tablice za pomocą funkcji zdefiniowanej przez użytkownika,
- uksort() – funkcja sortująca tablice asocjacyjne według klucza za pomocą funkcji zdefiniowanej przez użytkownika.

Sortowanie tablic

asort() sortuje tablice asocjacyjne według wartości zachowując przypisanie kluczy do wartości

```php
<?php
$owoce = array("d" => "mango", "a" => "papaja", "b" => "banan", "c" => "aronia");
asort($owoce);
foreach ($owoce as $klucz => $wartosc) {
  echo "$klucz = $wartosc<br>";
}
?>
```

c = aronia
b = banan
d = mango
a = papaja

ksort() sortuje tablice asocjacyjne kluczy

a = papaja
b = banan
c = aronia
d = mango

```php
<?php
$owoce = array("d" => "mango", "a" => "papaja", "b" => "banan", "c" => "aronia");
ksort($owoce);
foreach ($owoce as $klucz => $wartosc) {
  echo "$klucz = $wartosc<br>";
}
?>
```

Tablice - przydatne funkcje

- array_change_key_case (array wejście [, int wielkość]) – zmienia klucze w tablicy wejście, tak aby były one pisane tylko dużymi lub tylko małymi literami. Zmiana zależy od ostatniego opcjonalnego parametru case. Można do niego przekazać jedną z dwóch stałych:
  - CASE_UPPER lub CASE_LOWER. Domyślną wartością jest CASE_LOWER. Indeksy liczbowe będą pozostawione,
- array_flip (trans) – zwraca tablicę w odwróconym porządku, tzn. klucze z tabeli trans stają się wartościami, a wartości trans stają się kluczami. Zauważmy, że wszystkie wartości tablicy trans muszą mieć poprawne klucze, tzn. muszą być albo typu integer, albo string. Jeśli wartość nie ma prawidłowego typu, wyświetlone zostanie ostrzeżenie, a para klucz i wartość nie będzie odwrócona,
- array_fill (indeks_początkowy, num, wartość) – wypełnia tablicę wartością wartość, począwszy od indeksu indeks_początkowy przez num kolejnych elementów tablicy,
- array_pop (tablica) – zdejmuje i zwraca ostatnią wartość tablicy tablica, skracając tę tablicę o jeden element. Jeśli tablica jest pusta (lub nie jest tablicą), zwracana jest wartość NULL,
- array_push (tablica, wartosc [, wartosc ...]) – traktuje zmienną tablica jako stos i wstawia przekazane parametry na koniec podanej tablicy. Długość parametru tablica zwiększa się oliczbę przekazanych wartości. Funkcja zwraca nową liczbę elementów tablicy,

Tablice - przydatne funkcje

- array_shift (tablica) – usuwa pierwszą wartość parametru tablica i zwraca go, skracając tę tablicę o jeden element i przesuwając wszystkie pozostałe elementy w dół. Jeśli tablica jest pusta (lub nie jest tablicą), zwracana jest wartość NULL,
- array_unshift (tablica, wartość, [wartosc...]) – wstawia jeden lub więcej przekazanych jako parametry elementów na początek tablicy tablica. Zauważmy, że lista elementów jest wstawiana jako całość, więc elementy zostają w takim samym porządku. Funkcja zwraca nową liczbę elementów w tablicy tablica,
- array_search (ciąg, tablica[, ścisły]) – przeszukuje tablicę w poszukiwaniu parametru ”ciag” i zwraca odpowiedni klucz, jeśli został on znaleziony, lub FALSE w przeciwnym wypadku. Jeżeli trzeci parametr ścisły jest ustawiony na TRUE, to array_search() porówna także typy parametru „ciąg”z tymi z parametru „tablica”.

Język PHP

Formularze

Do pobierania danych od użytkownika w HTML stosuje się formularze.

Przykład: Poniższy formularz pozwoli wczytać 2 liczby które zostaną przesłane do skryptu *wynik.php*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title></title>
  </head>
  <body>
    <h1>Dodawacz</h1>
    <form action="wynik.php">
      <input name="dana_1" type="number" /><br />
      <label for="dana_2">Druga liczba:</label>
      <input name="dana_2" type="number" /><br />
      <label for="dana_1">Pierwsza liczba:</label>
      <input type="submit" value="Oblicz" />
    </form>
  </body>
</html>
```

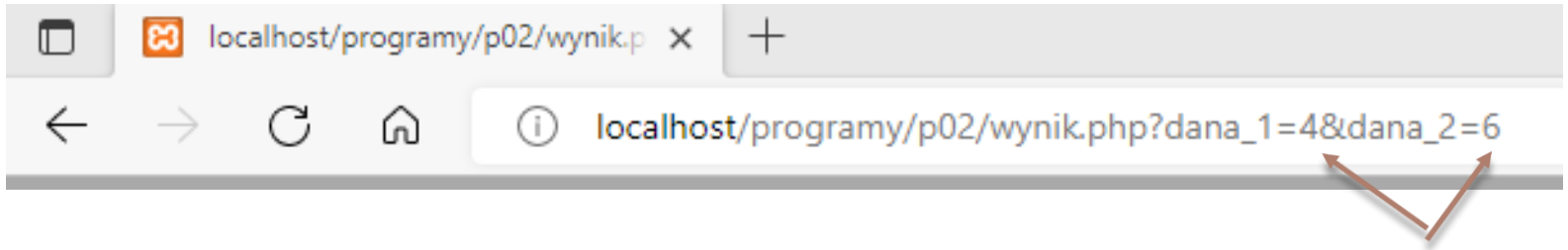
Formularze

Skrypt otrzymuje dane i wykonuje ich dodawanie

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8" />
    <title></title>
  </head>
  <body>
    <?php
    $x1 = $_GET['dana_1'];
    $x2 = $_GET['dana_2'];
    $wynik = $x1 + $x2;
    echo '<h1>'.$x1.' + '.$x2.' = '.$wynik.'</h1>';
    ?>
  </body>
</html>
```

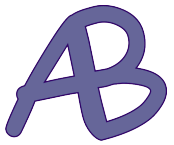
Metody przesyłania wartości

Domyślnie dane przesyłane są przez formularza metodą GET. Nazwy zmiennych ich wartości doklejane są do adresu wywoływanego skryptu.



Przesłano liczby 4 i 6

Metody przesyłania wartości



Wybranie metody w formularzu polega na dodaniu atrybutu *method* z wartością *POST* lub *GET* do znacznika *<form>*

```
<form action="index.php" method="get">

<form action="index.php" method="post">
```

Aby odebrać dane przesłane należy odwołać się do nazwy zmiennej przez użycie tablicy:

- dla metody POST $_POST["zmienna"],
- dla metody GET jest to tablica $_GET["zmienna"].

Metody przesyłania wartości

GET

Dane wysyłane metodą GET dołączane są do adresu URL i stanowią jego część. Z taką metodą spotykasz się m.in. w formularzach stosowanych w wyszukiwarkach, dzięki czemu widać, jakie pola zostały przesłane i jaką mają wartość.

Zalety:

- nie potrzeba formularza aby zmodyfikować zawartość tych zmiennych.
- łatwo = dodać taką stronę do "ulubionych" - zakładka będzie kierować wprost do strony z wynikami wyszukiwania.

Wady:

- Informacje umieszczone w adresie są umieszczane w logach serwera, więc nie nadaje się do przesyłania ważnych danych, jak np. haseł, numerów kont, informacji autoryzacyjnych.
- adres URL nie może być dowolnie długi i bezpiecznie jest przyjąć, że nie powinien być dłuższy niż 1024 znaki, choć niektóre przeglądarki obsługują poprawnie adresy o długości ponad 2 tys. znaków.

Metody przesyłania wartości

POST

Post przesyła dane na standardowe wejście skryptu, więc nie posługuje się w tym celu adresem URL. Przesyłanych danych nie widać w adresie i użytkownik też ich nie zobaczy. Dlatego ten sposób przesyłania danych stosowany jest do haseł, przy autoryzacji, czy przesyłaniu innych danych, które nie powinny zostać ujawnione w prosty sposób lub zapisane do logów systemowych serwera.

- ✓ POST nie ma ograniczenia długości danych, więc można przesyłać nie tylko długie pola formularza, ale też całe pliki binarne.

- ✓ Metoda ta musi być więc używana gdy długość danych w formularzu przekroczy 1000 znaków i wszędzie tam, gdzie dane z formularza nie powinny być widoczne w adresie strony.

Formularze

```
<form action="wynik.php" method=„get">
   <label for="dana_1">Pierwsza liczba:</label>
   <input name="dana_1" type="number" /><br>
   <label for="dana_2">Druga liczba:</label>
   <input name="dana_2" type="number" /><br>
   <input type="submit" value="Oblicz">
 </form>
```

…wynik.php?dana_1=10&dana_2=20

```
<?php
$x1 = $_GET['dana_1'];
$x2 = $_GET['dana_2'];
$wynik = $x1 + $x2;
echo '<h1>'.$x1.' + '.$x2.' = '.$wynik.'</h1>';
?>
```

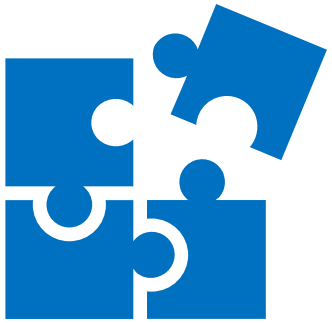
Formularze

```
<form action="wynik.php" method=„post">
   <label for="dana_1">Pierwsza liczba:</label>
   <input name="dana_1" type="number" /><br>
   <label for="dana_2">Druga liczba:</label>
   <input name="dana_2" type="number" /><br>
   <input type="submit" value="Oblicz">
 </form>
```

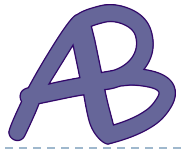
…wynik.php

```
<?php
$x1 = $_POST['dana_1'];
$x2 = $_POST['dana_2'];
$wynik = $x1 + $x2;
echo '<h1>'.$x1.' + '.$x2.' = '.$wynik.'</h1>';
?>
```

Przykład do wykonania



Należy napisać skrypt który otrzyma tablicę i wyświetli ją w tabeli



Przykład do wykonania

```
<!doctype html>
<html>
  <head>
    <meta charset="UTF-8" />
      <link rel="stylesheet" type="text/css" href="styl.css">
    <title></title>
  </head>
  <body>
    <form action="wynik.php" method="post">
        <div>
            <p>Wprowadź dane:</p>
            <input type="text" name="dane[]"><br>
            <input type="text" name="dane[]"><br>
            <input type="text" name="dane[]"><br>
            <input type="submit" name="Submit" value="Wyślij">
        </div>
    </form>
  </body>
</html>
```

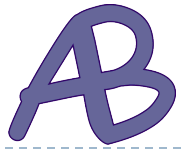
Trzy linie na dane. Używamy notacji tablicowej [] aby zaznaczyć użycie wielu wierszy tablicy

Przykład do wykonania

```
table
{
  border: none;
  margin-top: 20px;
  margin-left: 50px;
}
table td, table th, div
{
  font-size: 1.5rem;
  border: 1px solid rgb(130, 123, 114);
  padding: 10px;
  background-color: antiquewhite;
}
div {
    max-width: fit-content;
}
```

Po dodaniu stylu otrzymamy w efekcie:

Ten sam styl zastosujemy też do skryptu php



Z tablicy globalnej post pobieramy element o identyfikatorze dane

```php
<?php
  $tab = $_POST['dane'];
  if (is_array($tab) && (count($tab) > 0)) {
    echo "<table>";
    echo  " <tr><th>Klucz</th><th>Wartość</th></tr>";
    foreach ($tab as $klucz => $wartosc) {
      echo  "<tr>";
      if (empty($wartosc)) {
        echo "<td>$klucz</td><td><i>pusty</i></td>";
      } else {
        echo "<td>$klucz</td><td><i>$wartosc</i></td>";
      }
      echo "</tr>";
    }
    echo "</table>";
  } else {
    echo "<i>pusty lub nieprawidłowy</i>";
  }
?>
```

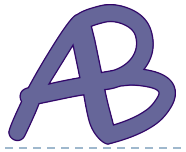
Upewniamy się, czy $tab jest na pewno tablicą

Rozpoczęcie tabeli I wyświetlenie nagłówka

Wyświetlenie wszystkich par klucz/wartość z tabeli

Sprawdzamy czy wartość jest pusta jeżeli tak, informujemy o tym

Klucz	Wartość
0	*dana 01*
1	*pusty*
2	*dana III*



Literatura

W prezentacji użyto przykładów z książki:
- Żygłowicz Jerzy - PHP - Kompendium wiedzy, Helion

- https://www.php.net