

3. PHP (2)

27.2.2006

Obsah

- "Pokročilejšie" kombinovanie PHP a HTML
- Jednoduché polia
- Reťazce
- Dátum a čas
- Funkcie

"Pokročilejšie" kombinovanie PHP a HTML

```
<?php
```

```
  for ($i = 1; $i <= 10; $i++) {
```

```
?>
```

číslo

```
<?php
```

```
  echo $i . '<br/>';
```

```
}
```

```
?>
```

Jednoduché polia

- Index/klúč poľa môže byť celé číslo alebo reťazec.
- Tieto rôzne klúče sa môžu kombinovať.
- Číselné klúče zvyčajne začínajú nulou.
- V prípade, že neuvedieme klúč, hodnote je priradený klúč **maximum+1**.
- Pole môžeme inicializovať rôznymi spôsobmi (priamo alebo pomocou funkcie `array()`).

Inicializácia poľa

```
$predmety[3] = "Delphi";
```

```
$predmety[0] = "PHP";
```

```
$predmety[1] = "C++";
```

```
$predmety[] = "Databázy";
```

```
$predmety["d"] = "Delphi";
```

```
$predmety["p"] = "PHP";
```

```
$predmety["c"] = "C++";
```

```
$predmety["db"] = "Databázy";
```

Inicializácia poľa pomocou funkcie array()

```
$predmety = array(3=>"Delphi", 0=>"PHP",  
1=>"C++", "Databázy");
```

iný zápis

```
$predmety = array("PHP", "C++", 3=>"Delphi",  
"Databázy");
```

```
$predmety = array("d"=>"Delphi", "p"=>"PHP",  
"c"=>"C++", "db"=>"Databázy");
```

Prechádzanie poľa (1)

- Ak sú kľúče poľa spojené, môžeme použiť cyklus FOR

```
$predmety = array("PHP", "C++", "Delphi", "Databázy");  
for ($i = 0; $i < count($predmety); $i++) {  
    echo "$predmety[$i] <br/>\n";  
}
```

Prechádzanie poľa (2)

- Čo ak nie sú kľúče spojené?
- Využijeme funkcie `each()`, `list()`

```
while (list($index, $hodnota) = each($predmety)) {  
    echo "$index = $hodnota <br/>\n";  
}
```

Vyskúšajte si tieto upravené cykly **while**:

```
while (list($index) = each($predmety)) { ... }
```

```
while (list(, $hodnota) = each($predmety)) { ... }
```

Polia – dôležité funkcie

- `count($pole)` – vráti počet prvkov poľa
- `reset($pole)` – nastaví vnútorný ukazovateľ na 1. prvok poľa
- `each($pole)` – postupne prejde všetky prvky poľa
- `list($prem1, $prem2,...)` – vloží do premenných prvky poľa

Polia – ďalšie funkcie

názov funkcie	popis
<code>in_array(\$co, \$pole, strict)</code>	Funkcia vráti true, ak sa hodnota <code>\$co</code> nachádza v poli <code>\$pole</code> . Ak je <code>strict = true</code> , tak funkcia kontroluje aj typ parametrov.
<code>current(\$pole)</code>	vypíše hodnotu aktuálneho prvku
<code>key(\$pole)</code>	vypíše index aktuálneho prvku
<code>prev(\$pole)</code>	posunie vnútorný ukazovateľ na predchádzajúci prvok
<code>next(\$pole)</code>	posunie vnútorný ukazovateľ na nasledujúci prvok
<code>sort(\$pole), asort(\$pole), ksort(\$pole), krsort(\$pole), rsort(\$pole), arsort(\$pole)</code>	triedenie poľa
<code>shuffle(\$pole)</code>	zamiešanie prvkov poľa

Práca s reťazcami (1)

<code>strlen(\$ret)</code>	Dĺžka reťazca
<code>substr(\$ret, zac [, pocet])</code>	Vráti podreťazec
<code>strpos(\$ret, \$co [, zac])</code>	Pozícia \$co v reťazci \$ret
<code>strtolower(\$ret)</code>	Vráti reťazec prevedený na malé písmená
<code>strtoupper(\$ret)</code>	Vráti reťazec prevedený na veľké písmená
<code>strrev(\$ret)</code>	Vráti prevrátený reťazec
<code>nl2br()</code>	Prevedie konce riadkov (<code>\n</code>) na <code>
</code>

Ret'azce – príklady

predpokladáme \$vstup = "PHP v príkladoch";

príkaz	vypíše sa
substr(\$vstup, 4)	v príkladoch
substr(\$vstup, 2, 5)	P v p
substr(\$vstup, -3)	och
substr(\$vstup, 4, -3)	v príklad
substr(\$vstup, -7, 4)	klad

Práca s reťazcami (2)

- `trim($ret)`, `ltrim($ret)`, `rtrim($ret)` – odstráni tzv. whitespace z príslušného konca
- `chr(ascii-kód)`
- `ord(znak)`
- `implode($oddelovac, $pole)` – `$pole` prevedie na reťazec
- `explode($oddelovac, $ret)` – reťazec rozdelí do poľa

```
$predm = explode(";", "TWD;UWA;Delphi")
```

Dátum a čas

- `checkdate(mesiac, den, rok)`
- `date("format" [, timestamp])`

príkaz	vypíše sa
<code>date("j.n.Y")</code>	27.2.2006
<code>date("j. F Y")</code>	27. February 2006
<code>date("M j, Y h:I A")</code>	Feb 27, 2006 8:30 AM
<code>date("G:i:s")</code>	8:30:15

Vlastné funkcie

```
function menoFunkcie(parametre) {  
    // telo funkcie  
}
```

```
function mocnina3($cislo) {  
    return $cislo * $cislo * $cislo;  
}
```

Globálne premenné

Veľký rozdiel oproti Delphi

```
$sucet = 0;  
function pridaj($co) {  
    $sucet += $co;  
}
```

Premenná **\$sucet** sa nikdy nezmení

Globálne premenné – správne

```
$sucet = 0;  
function pridaj($co) {  
    global $sucet;  
    $sucet += $co;  
}
```

Parametre volané adresou – chybné

```
$sucet = 0;
```

```
function zvys($sum, $co) {  
    $sum += $co;  
}
```

```
zvys($sucet, 10);
```

Parametre volané adresou – správne

```
$sucet = 0;
```

```
function zvys(&$sum, $co) {  
    $sum += $co;  
}
```

```
zvys($sucet, 10);
```

Parametre – inicializačná hodnota

```
$sucet = 0;
```

```
function zvys(&$sum, $co = 1) {  
    $sum += $co;  
}
```

```
zvys($sucet, 1);
```

```
zvys($sucet);
```

Premenné

- `isset($premenna)` – zistí, či existuje
- `unset($premenna)` – zruší premennú

**Ďakujem za
pozornosť 😊**

Priestor na vaše otázky