Otázka: Vestavěné a uživatelské (vlastní) funkce a jejich využití

Předmět: Informační technologie

Přidal(a): Kahanek2

– Vestavěné funkce řetězové, matematické, časové.
– Vlastní funkce, jejich struktura a použití.
– Použití proměnných ve vlastních funkcích.

Toto téma se dá nejlépe vysvětlit na našem, dobře známém, programovacím jazyce PHP. PHP je dodáváno s mnoho užitečnými funkcemi a konstrukcemi. Některé mohoufungovat nezávisle na kompilaci jazyka, ovšem některé potřebují speciální nástavbu. Jako příklad si můžeme uvést funkci mysqli_connect(), pro kterou je potřeba podpora MySQL. Některé funkce také pracují nezávisle a vrací nám vlastní výstup, narozdíl od jiných, které pracují přímo s proměnnou a mění ji.Vestavěných funkcí je v PHP nezpočet, jako jsou například řetězcové, nebo variabilní funkce.

1) Vestavěné funkce:

Řetězcové

- pracují s řetězci (v php se žetězce zapisují do uvozovek, v podstatě je to text)

echo – vypíše řetězec
md5($str) – zakóduje řetězec $str do md5
strtolower($str) – převede řetězec $str na malá písmena
strtoupper($str) – převede řetězec $str na velká písmena

příklad změny řetězce na velká písmena:

echo strtoupper(„seto“);
=> SETO

Matematické

- většinou pracují s proměnnými a čísly
- dokáží řešit složité matematické úkony

abs($var) – vrátí absolutní hodnotu hodnoty $var
cos($var) – vypočítá cosinus hodnoty $var
pi() – vrátí hodnotu pi (dá se nahradit funkcí M_PI)
pow($var, $exp) – vrátí hodnotu $var umocněnou hodnotou $exp
rand($a, $b) – vrátí náhodné číslo od $a do $b

příklad výpočtu obsahu kruhu:

$r = 20;
$res = pi()*pow($r,2);
echo $res;

=>1256.6370614359

Časové

- dokáží pracovat s časem – vypisovat přesný datum i čas
- umožňují i měřit určité časové úseky v průběhu programu

time() – vrátí čas v sekundách od 1/1/1970 (tzv. timestamp)
date(string, int) – formátuje ‚timestamp' do uživatelem zvoleného formátu
– pro formát lez použít mnoho předdefinovaných znaků (viz. příklad)

příklad výpisu datumu:

```
$t = time();
echo (date(„Y-m-d“, $t));
```

=> 2013-12-17

```
$t = time();
echo (date(„d M, Y“, $t));
```

=>17 Dec, 2013

2) Vlastní funkce:

Vlastní uživatelské funkce mají podobné vlastnosti jako ty vestavěné. Jediný rozdíl je ten, že si je uživatel programuje sám.

K uživatelským funkcím se pojí pojem OOP (Object Oriented Programming). Je to odlišný druh syntaxe PHP, který využívá hlavně uživatelsky vytvořených funkcí a zabraňuje tak přepisování desítek, ne-li stovek řádků programu.

Jednoduché uživatelské funkce pracují jednoduše bez jakékoliv přidané hodnoty (proměnné).

Dají se využít lehce například pro usnadnění odřádkování uvnitř PHP skriptu. V následujícím příkladě by se musela dvě echa odsadit dalším echem, které by vypisovalo tag
:

```
echo „prvni radek“;
echo „<br>“;
echo „druhy radek“;
```

Toto složíté ‚echování‘ lze nahradit vlastní předdefinovanou funkcí:

```
function br()
{
echo „<br>“;
}

echo „prvni radek“;
br();
echo „druhy radek“;
br();
echo „treti radek“;
```

3) Použití proměnných ve vlastních funkcích

- Použití proměnných nám dovoluje vlastní funkci řídit. To si můžeme ukázat v podobném příkladě:

```
function vypis($count)
{
for ($i=0; $i < $count; $i++) {
echo „.“;
}
}
vypis(3);
```

Funkce je definována tak, že vypíše takový počet teček, kolik bude proměnná $count (v tomto případě 3). Ta se nastaví přímo ve vyvolávání funkce do závorky. Takových proměnných může být ve funkci mnoho, jako na tomto příkladu, který má stejný učel jako ten předchozí, ovšem lze u něj určit i znak, který se bude vypisovat.

```
function vypis($count, $znak)
{
for ($i=0; $i < $count; $i++) {
echo $znak;
}
}
vypis(5, „/“);
```

Výsledek bude tedy pět lomítek.

Všechno poté můžeme zkombinovat do nějaké hezké funkce:

```
function obsahkruhu($r)
{
$res = pi()*pow($r, 2);
echo $res;
}

obsahkruhu(20);
```