POLSKO-JAPOŃSKA WYŻSZA SZKOŁA TECHNIK KOMPUTEROWYCH

Instrukcje PHP

Warunki i pętle

- Instrukcja warunkowa if…else
- Pętla while
- Pętla do…while
- Pętla for
- *Pętla foreach*
- Instrukcje break i contonue
- Instrukcja switch

Instrukcja warunkowa if...else

Czasem chcemy, żeby fragment kodu wykonał się tylko pod jakimś warunkiem. Stosuje się wtedy instrukcję if:

```
if($a>$b)
echo("$a jest większe od $b");
```

Jeżeli wartość $a jest większa od $b wyświetli się napis "$a jest większe od $b". Warunek jest spełniony, jeżeli wyrażenie w nawiasie ma wartość różną od zera. Jeżeli będzie to np. pusta zmienna warunek nie jest spełniony. Polecenie w następnej linii zostanie wykonane, jeżeli warunek jest spełniony.

Instrukcja warunkowa if…else

Aby warunek objął kilka poleceń, stosuje się nawiasy klamrowe:

```
if($a>$b)
{
echo("$a jest większe od $b");
$a++;
}
```

Instrukcja warunkowa if…else

Jeżeli chcemy wykonać inny fragment kodu gdy warunek nie jest spełniony, stosujemy instrukcję else:

```
if($a>$b)
echo("$a jest większe od $b");
else
echo("$a jest mniejsze lubrówne $b");
```

Instrukcja warunkowa if…else

Aby uzyskać bardziej złożony warunek można zastosować operatory logiczne

- i (&&) lub **and**
- lub (||) lub **or**

```
if($a>$b||$a<2)
echo("$a jest większe od $b, lub mniejsze od 2");
```

Pętla while

Aby fragment kodu wykonać wiele razy stosuje się pętle. PHP obsługuje 3 rodzaje pętli: while, do..while i for. Najprostszą z nich jest pętla while:

```
<?
while (warunek) {
//instrukcje do wykonania
}
?>
```

Pętla while

W pętli while najpierw sprawdzany jest warunek (w tym wypadku $a<5). Jeżeli jest spełniony, pętla wykonuje się i wraca do sprawdzenia warunku. Jeżeli warunek nie jest spełniony, wykonanie pętli kończy się. Poniższy skrypt wyświetli liczby od 0 do 4.

```
$a=0;
while($a<5)
{
echo("$a ");
$a++;
}
```

Pętla do…while

To specyficzna odmiana pętli WHILE, bo jeśli we WHILE warunek jest na starcie fałszywy, to pętla ani razu nie wykona bloku instrukcji. W wypadku użycia DO…WHILE pętla wykona blok instrukcji przynajmniej raz (nawet wtedy, gdy warunek jest od początku fałszywy).

```
<?
do {
//instrukcje do wykonania
} while (warunek);
?>
```

Pętla do…while

Pętla do..while różni się od pętli while tym, że najpierw wykonuje się pętla, a dopiero potem sprawdzany jest warunek. Oznacza to, że pętla zawsze wykona się co najmniej 1 raz.

Pętla do...while

```
$a=6;
do
{
echo("$a ");
$a++;
}while($a<5); /* ta pętla wykona się 1 raz */

$a=6;
while($a<5)
{
echo("$a ");
$a++;
} /* instrukcje w tej pętli nie zostaną wykonane */
```

Pętla for

Pętla FOR jest używana tylko wtedy, gdy zachodzi potrzeba wykonania jakiegoś kodu określoną liczbę razy (założoną z góry przez autora lub pochodzącą ze zmiennej).

```
<?
for (inicjalizacja zmiennych; sprawdzenie warunku;
modyfikacja zmiennych) {
//instrukcje do wykonania
}
?>
```

Pętla for

Wykonanie pętli for:

```
for($a=0;$a<5;$a++)
{
echo("$a ");
}
```

Odpowiada wykonaniu pętli:

```
$a=0;
while($a<5)
{
echo($a);
$a++;
}
```

Można to też zapisać jako

```
for($a=0;$a<5;echo("$a "), $a++);
```

Pętla foreach

Ułatwia ona obsługę tablic i tablic asocjacyjnych. Poniższy przykład pokazuje, jak łatwo zamienić pętlę FOR na FOREACH w wypadku korzystania z tablic.

Pętla foreach

Blok instrukcji:

```
<?
for($i=0;$i<sizeof($kolory);$i++) {
echo "Kolorek: ".$kolory[$i]
}
?>
```

jest równoważny zapisowi:

```
<?
foreach ($kolory as $kolor) {
echo "Kolorek: ".$kolor
}
?>
```

Instrukcje break i continue

Wykonanie pętli można w każdym momencie zakończyć. Służy do tego instrukcja break:

```
$a=0;
while($a<10)
{
$a++;
if($a==3)
break;
}
```

Ta pętla nie wykona się 10 razy - gdy $a osiągnie wartość 3, wykonanie pętli zostanie przerwane.

Instrukcje break i continue

Funkcja CONTINUE powoduje przerwanie aktualnej iteracji (przebiegu) pętli i wykonanie jej od nowa.

```
$a=0;
while($a<10)
{
$a++;
if($a==3)
continue;
echo("aaa"); /* ta instrukcja wykona
się tylko gdy $a nie jest równe 3 */
}
```

Instrukcja switch

Switch jest rodzajem skondensowanej instrukcji warunkowej, którą zazwyczaj zastępujemy rozbudowane i wielokrotne użycia ELSE IF.

Instrukcja switch

```
<?
switch ($miasto) {
case 'warszawa': echo "Pochodzisz ze stolicy?"; break;
case 'hel': echo "Mieszkasz nad morzem?"; break;
case 'sanok': echo "A może w Bieszczadach?"; break;
default: echo "Miasto nierozpoznane.";
}
?>
```

Instrukcja switch – porównanie z if

```
if($a==1)
{
echo("a jest równe 1");
}

if($a==3)
{
echo("a jest równe 3");
}

if($a==11)
{
echo("a jest równe 11");
}
```

Instrukcja switch – porównanie z if

```
switch($a)
{
case 1:
echo("a jest równe 1");
break;

case 3:
echo("a jest równe 3");
break;

case 11:
echo("a jest równe 11");
break;
}
```