8. Operátory porovnávání hodnot v PHP

Operátorem se nazývá zápis matematické operace. Při použití se ptáme, zda je jedno číslo menší než druhé nebo např. zda se tyto čísla rovnají apod.

Základní operátory porovnávání hodnot:

Příklad	Kdy je podmínka pravdivá
$a == $b	Hodnota proměnné a je rovna hodnotě proměnné b
$a != $b $a <> $b	Hodnota proměnné a není rovna hodnotě proměnné b
$a < $b	Hodnota proměnné a je menší než hodnota proměnné b
$a <= $b	Hodnota proměnné a je menší nebo rovna hodnotě proměnné b
$a > $b	Hodnota proměnné a je větší než hodnota proměnné b
$a >= $b	Hodnota proměnné a je větší nebo rovna hodnotě proměnné b

Příklad:

```
<?php
$a = 100;
// Zkuste zadat i jiné číslo.
if ($a == 100)
//pokud je proměnná rovna číslu 100, pak se vypíše následný text
echo 'Proměnná $a je rovna číslu 100.';
?>
```

Datový typ boolean

Když PHP pracuje s podmínkami, zajímá ho vlastně jen jedno: zda je podmínka pravdivá nebo nepravdivá. Používá k tomu hodnoty true – pravda a false – nepravda. Tomuto typu dat říkáme logický datový typ – v PHP boolean. Pro ilustraci jeden příklad, kdy se hodnota vypíše:

```
<?php
$a = (3 != 4);
// Do proměnné a se uloží výsledek vyhodnocení podmínky, zda se číslo 3
nerovná číslu 4
var_dump($a);
// Vypíše bool(true)
?>
```

Jak již bylo napsáno, funkce var_dump vypisuje informaci o typu dat i její hodnotu (diagnostická funkce). Tak že v tomto případě je to typ dat bool (takto se označuje logický datový typ) a hodnota, tedy true (neboť je pravda, že 3 není rovno 4).

Operátory porovnávání hodnot s kontrolou typu
Kromě základních operátorů, které byly vyjmenovány v předchozí kapitole, PHP používá ještě dva speciální nejen pro porovnání hodnoty, ale zároveň také pro porovnání typu těchto dat:

$a === $b	Hodnota proměnné a je rovna hodnotě proměnné b a navíc jsou stejného typu
$a !== $b	Hodnota proměnné a není rovna hodnotě proměnné b nebo nejsou stejného typu

Rozdíl mezi běžným porovnáváním a porovnáváním s kontrolou typu dat je vidět na dalším příkladu:

```
<?php
var_dump(1.0 == 1);
// Podmínka je pravdivá, protože reálné číslo 1.0 se rovná celému číslu 1.
// Vypíše se bool(true)
echo '<br>';
// Zajistí, aby se další výpis vypsal na další řádku.
var_dump(1.0 === 1);
// Podmínka je nepravdivá, protože 1.0 a 1 nejsou shodné typy.
// Číslo 1.0 patří do typu reálných čísel a číslo 1 patří do typu celých čísel.
?>
```

Porovnávání řetězců

Porovnávání čísel není nic složitého. **Ovšem pokud potřebujeme porovnat řetězce (tedy text), je situace o něco složitější.** PHP k tomuto účelu využívá tzv. tabulku ASCII (American Standard Code for Information Interchange). Každému znaku je přiřazen nějaký číselný kód (např. A má přiřazeno 65, B má 66, atd…). Kompletní ASCII tabulku najdete např. na adrese:

http://cs.wikipedia.org/wiki/ASCII

Postup porovnávání řetězců:

1. Začnou se porovnávat znaky obou řetězců od začátku
2. Nejdříve se porovnají první znaky obou řetězců, pak druhé, třetí atd.
3. Jakmile se narazí na první rozdíl nebo jeden z řetězců skončí, je rozhodnuto.
4. Pokud se při porovnávání řetězců narazí na konec jednoho z nich, je ten kratší menší než ten delší.

Podle tabulky ASCII platí, že libovolné velké písmeno je menší než libovolné malé písmenko.

Ukázky porovnávání řetězců:

První řetězec	**Druhý řetězec**	**Výsledek porovnání**
‚abc‘	‚abx‘	**První řetězec je menší než druhý**
‚XX‘	‚xx‘	**První řetězec je menší než druhý**
‚sova‘	‚sova‘	**Oba řetězce jsou shodné**
‚xy‘	‚xyz‘	**První řetězec je menší než druhý**

Složitější porovnávání:

Následující příklad ukazuje porovnávání hodnot s kontrolou typu:

```
<?php
var_dump ('xy' === 'xyz');
// Vypíše bool(false)
?>
```

Výsledkem je výpis bool(false)

Porovnávání kombinovaných řetězců:

Řetězce se porovnávají poněkud složitěji. PHP se totiž nejdříve podívá, jestli jdou oba řetězce převést na číslo. Pokud ano, porovná je jako čísla. Pokud minimálně jeden nejde převést na číslo, porovná je jako texty. Příklad:

```
<?php
var_dump('30' == '+30');
// Podmínka je pravdivá, protože PHP porovná řetězce jako čísla, vypíše se tedy bool(true)
echo '<br>';
// Zajistí, aby se další výpis vypsal na další řádek.
var_dump('30x' == '+30x');
// Podmínka není pravdivá, protože PHP porovná řetězce jako texty, vypíše se tedy bool(false)
?>
```

Ukázky porovnávání složitějších řetězců:

Pozn.: ***řetězce jsou „kousky“ textu, řetězec poznáme podle uvozovek nebo apostrofů.***

Podmínka	**Výsledek podmínky**
‚+10.0′ == ’10‘	**pravdivá (porovnávají se jako čísla)**
‚2‘ == ‚2a‘	**nepravdivá (porovnávají se jako řetězce)**
‚abc‘ != ‚xyz‘	**pravdivá (porovnávají se jako řetězce)**
‚8‘ == ‚+8‘	**pravdivá (porovnávají se jako čísla)**

Porovnávání řetězce s číslem

Pokud PHP porovnává řetězec s číslem, chová se jednoznačně tak, že převede řetězec na číslo a pak provádí porovnání. Např.:

```
<?php
var_dump('100' > 40);
// Výsledkem je pravda, protože PHP zde porovnává jakoby se jednalo o dvě čísla.
// Tedy zjišťuje, zda je číslo 100 větší, než číslo 40. Vypíše se tedy bool(true)
?>
```

Převádění řetězce na číslo provádí PHP tak, že se „podívá“ na první znak.

Pokud zde nenajde číslo, výsledkem je číslo nula (0). Výjimku tvoří znaky + a -. Ty fungují jako v matematice.

Ilustrační příklady převodu řetězce na číslo:

Řetězec před převodem	**Výsledek po převodu PHP na číslo**
„10slonů“	**10**
‚postel‘	**0**
‚1.2 m‘	**1.2**
‚333 stříbrných stříkaček‘	**333**
„9e2“	**900 (zápis 9e2 je zápis reálného čísla ve vědecké notaci)**

Nejvíce se převod řetězce na čísla projeví při matematických výpočtech:

Zápis	**Výsledek**
1 + „1 pes“	**2**
‚1.2 kg‘ – ‚0.8 kg‘	**0.4**
8 * „lev“	**0**
0 + „3e4xxx“	**30000 (zápis 3e4 je zápis reálného čísla ve vědecké notaci)**

Další ukázky porovnávání řetězců s čísly:

Podmínka	**Výsledek podmínky**
‚+1‘ > 0	**pravdivá** (řetězec ‚+1‘ se převede na jedničku)
’23 psů‘ == 23	**pravdivá** (řetězec ’23psů‘ se převede na číslo 23)
‚-2abc‘ > 10	**nepravdivá** (řetězec ‚-2abc‘ se převede na číslo -2)
0 != ‚cihla‘	**nepravdivá** (řetězec ‚cihla‘ se převede na nulu)