Maturitní témata 2020/21

Otázka č. 20 – Počítačové sítě a programování

PHP:

- Hypertext Preprocessor, nejrozšířenější skriptovací programovací jazyk, je určený především pro dynamické internetové stránky a webové aplikace např. ve formátu HTML nebo XHTML, dynamický jazyk → datové typy nemusíme u proměnných zadávat (jako třeba v C) ale PHP si typ podle obsahu proměnné nastaví samo, mezi typy také PHP samo převádí

- syntaxe je inspirována několika jazyky (Perl, C, Pascal a Java), je nezávislý na platformě (pracovní prostředí, zajišťující bezproblémovou činnost programu)

- podporuje mnoho knihoven pro různé účely - např. zpracování textu, grafiky, práci se soubory, přístup k většině databázových systémů (MySQL, ODBC, Oracle, SQLite)

- výhody: specializace na webové stránky, nativní podpora v mnoha databázových systémů, rozsáhlý soubor funkcí v základní knihovně PHP, jednoduchý,

- nevýhody: nekonzistentní pojmenování funkcí (např. strpos(), str_replace()), nejednotné pořadí parametrů, oslabuje výkon

Proměnné:

- místo v paměti kam si můžeme ukládat data a potom s nimi pracovat, bez proměnných se neobejde žádný skript, proměnné deklarujeme pomocí znaku dolaru $ a pojmenováváme je libovolným názvem bez mezer a diakritiky

```
$dolar = "1$";
$den="středa";
$datum =            14;
```

- proměnné se objevují v několika typech, každá proměnná je určitého typu:

Typ	Význam
String	Text, řetězec (sada znaků), příklad: $retezec = "obyčejný text";
Integer	Celé číslo, se kterými je možné pracovat, počítat, příklad: $cislo = 2;
Float, real nebo double	Desetinné číslo
Boolean	Logická proměnná, hodnota PRAVDA, NEPRAVDA (1, 0), zapisuje se TRUE nebo FALSE

- proměnnou vypisujeme pomocí příkazu echo()

```php
$dolar="1$";
echo($dolar);
echo("$dolar");
echo("mám jen" . $dolar . "<br />");
echo("nemám ani $dolar");
```

- pomocí operátorů můžeme proměnné měnit, násobit, porovnávat, atd.

aritmetické operátory	
$a+$b	přičte k A B
$a-$b	odečte B od A
$a/$b	vydělí A proměnnou B
$a*$b	vynasobí A proměnnou B
přirovnávací operátory (zkracující zápisy)	
$a++	$a = $a + 1
$a--	$a = $a - 1
$a += $b	$a = $a + $b
$a -= $b	$a = $a - $b
$a *= $b	$a = $a * $b
$a /= $b	$a = $a / $b

porovnávací operátory	
$a == $b	$a má stejnou hodnotu jako $b
$a != $b	$a má jinou hodnotu než $b
$a > $b	$a je větší než $b
$a < $b	$a je menší $b
logické operátory	
$a || $b	vrátí 1 pokud alespoň jedna proměnná je pravdivá
$a && $b	vrátí 1 pokud jsou obě proměnné vyhodnoceny jako pravda
!$a	negace (opak) $a

Funkce:

- způsob zkrácení většího počtu procesů pokud je nechceme celé vypisovat, kromě předem zavedených a vyhrazených funkcí PHP si můžete deklarovat své vlastní, funkce zapisujeme klíčovým slovem function

```php
function napis(){
echo("ahoj");
}
```

- function deklaruje funkci, napis je název funkce, v závorkách jsou argumenty funkce (může být i bez argumentu)

- k vyvolání funkce zapíšeme pouze název funkce a závorky

- funkce může mít argumenty a může vracet hodnotu, hodnota se vrací pomocí klíčového slova return

```php
function sude($cislo) {
  // zkontroluje, zda je číslo sudé
  // Číslo je sudé, pokud číslo modulo 2 je 0
  return (($cislo % 2) == 0) ? true : false;
}
```

Třídy a objekty:

- **Konstruktor** - funkce, která má stejný název jako třída a která se při incializaci třídy okamžitě provede, používá se hlavně k počátečnímu nastavení proměnných tříd

```php
class clsZarovka {
 .
 .
//kod z předešlého příkladu

 .
 function clsZarovka ($stav) {
//zde je náš konstruktor

  $this->Sviti=$stav;
//nastavení členské proměnné

 }
}
```

- **Dědičnost** - pomocí dědičnosti můžeme vytvořit třídu která zdědí proměnné funkce nějaké jiné třídy, pro dědění používáme klíčové slovíčko extends

```php
class clsLepsiZarovka extends clsZarovka
 var $Barva, $Jas;
 function NastavBarvu($b) {
  $this->Barva=$b;
//nastavení členské proměnné

 }
 function NastavJas($j) {
  $this->Jas=$j;
//nastavení členské proměnné

 }
 function clsLepsiZarovka ($s,$j,$b) {
  $this->Sviti=$s;
//nastavení členské proměnné

  $this->Barva=$b;
//nastavení členské proměnné

  $this->Jas=$j;
//nastavení členské proměnné

 }
}
```

- nyní si vytvoříme instanci této třídy (datový objekt, instance bývá vytvořena pomocí konstruktoru a klíčového slova new):

```
$SuperZarovka = new clsLepsiZarovka(true, „yellow“,“2“);
$SuperZarovka->Rozsvit();
```

- **Přímé volání funkce ve třídě** - v PHP můžeme zavolat funkci v libovolné třídě aniž bychom tuto třídu dříve inicializovali

```
class trida {
  function funkce() {
    echo „Funkce spuštěna“;
  }
}
```

- jestliže chceme spustit funkci obyčejným zápisem funkce();

Zpracování formulářů:

- pomocí PHP lze pracovat s formuláři, díky formulářům je možné zapojit návštěvníka do průběhu skriptu, zde se podíváme na základní způsob práce s formulářem

- budu mít dva soubory - 1. formular1.html s jedním vstupním polem (obyčejné HTML),

2. zpracovani_formulare1.php s php kódem

Soubor formular1.html (Vypisuji bez hlaviček.):

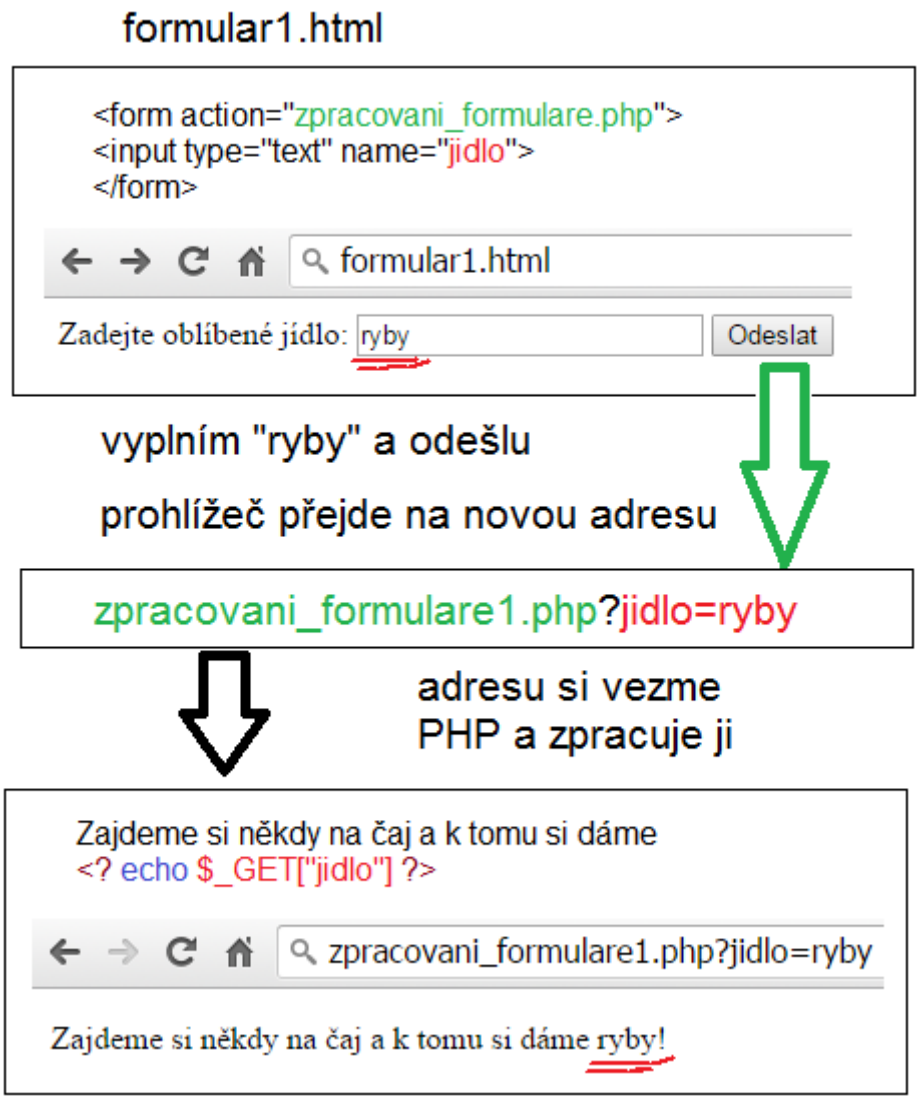
```
<body>
<form method="GET" action="zpracovani_formulare1.php" >
<p>Zadejte oblíbené jídlo:
<input type="text" name="jidlo" size="20">
<input type="submit" value="Odeslat"></p>
</form>
</body>
```

A soubor zpracovani_formulare1.php:

```
<body>
<H1>Zpracování formuláře</h1>
<p>Výborně! Zajdeme si někdy na čaj a k tomu si dáme
<?php echo $_GET["jidlo"] ?>!
</body>
```

- formulář má nastavenou action na jméno php souboru který to bude zpracovávat,

- PHP příkaz echo pak už jenom vypíše hodnotu proměnné a celé to odešle prohlížeči



- když se formulář odesílá metodou $_POST místo $_GET, GET je výchozí metoda, přes POST se posílají až větší data typicky dlouhé texty, maily, atd.

- při metodě GET komunikují stránky pomocí URL (jsou vidět v URL adrese, maximum znaků 1024), výhodou je převážně jednoduchost a možnost uvést odkaz na výsledek zpracování

- data odesílané metodou POST nejsou vidět v URL adrese což řeší problém maximální délky odesílaných dat, metodou POST bychom měli vždy odesílat formulářová pole, protože tím zabezpečíme že nebudou vidět například hesla a na stránku se zpracováním výsledku konkrétního vstupu nepůjde uvést odkaz

Session:

- Session nám umožňují sdílet údaje mezi jednotlivými http requesty a identifikovat uživatele

- HTTP protokol je bezestavový, server vždy klientovi odpovídá výhradně na základě jeho http requestu (tj. žádost o konkrétní stránku), aby bylo možné vytvářet složitější aplikace, je nezbytná přítomnost mechanismu sdílejícího určité informace mezi jednotlivými htttp požadavky

- po zobecnění všech potřeb zjistíme že nám stačí správně identifikovat opětovné přístupy od téhož uživatele, samotná data přiřazená ke konkrétnímu uživateli již není třeba mezi stránkami sdílet, ale lze je ukládat na server a odkazovat se na ně

- při vytváření session je uživateli vygenerován unikátní identifikátor sloužící k jeho rozpoznání a umožňující přistupovat ke konkrétnímu datovému souboru, pro sdílení mezi jednotlivými stránkami tento identifikátor uložíme do cookies (mechanismus pro ukládání dat), kterou prohlížeč na server odesílá při každém načítání stránky

Příklad http requestu

```
GET /wiki/Wikipedie HTTP/1.1
Host: cs.wikipedia.org
Accept-Charset: UTF-8,*
```

```
GET /programujeme-v-php/sessions?rev=1 HTTP/1.1
Host: pehapko.cz
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en,cs;q=0.8
Cookie: PHPSESSID=9g0jg1mgs26c1i0e4hok7rgbd1; nette-browser=spx2kdfq2y
```