Informační systémy

Dynamické webové stránky a jazyk PHP

Martin Trnečka

Katedra informatiky
Univerzita Palackého v Olomouci

Předchozí přednášky

- Jazyk HTML - nástroj pro vytvoření **statické webové stránky**.
- CSS - nástroj pro upravení vzhledu www stránky.
- I přes určitou dynamiku, která přichází s CSS jsou stránky stále statické.
- Potřeba **dynamické webové stránky** (měnící se dle požadavků uživatele, akceptující vstup, zobrazující výstup) - nutnost plnohodnotných webových IS (ale i rozsáhlejší webu).
- JavaScript první opravdu „dynamické“ možnosti. Omezení: veškerá funkcionalita probíhá na straně klienta. Nesplňuje požadavky na plnohodnotný dynamický web.
- Dynamické stránky na straně serveru - skripty (skriptování) na straně serveru.

Dynamické webové stránky

Co je všechno potřeba:

- Webový server (Apache, IIS).
- Databázový server (MySQL, PostgreSQL a další).
- Podpora použitého skriptovacího jazyka (PHP, Ruby on Rails, Java a další).
- V praxi je zapotřebí server (hosting), doména (nebo IP adresa).
- Vývoj lze emulovat například pomocí virtuální serverů WAMP, XAMP, MAMP či jiných nástrojů (lokální instalace).

Většinu znáte z počítačových sítí.

Připomenutí, jak to funguje (neformálně):

1. Server přijme požadavek od klienta a spustí příslušný skript.
2. Provede se výpočet, dotazy na databázi a další.
3. Skript vygeneruje HTML stránku a tu odešle klientovy.

Jazyk PHP

- Rekurzivní zkratka: PHP: Hypertext Preprocessor.
- Skriptovací jazyk primárně pro tvorbu www stránek (lze vytvářet i konzolové a desktopové aplikace, není primární účel).
- Interpretovaný jazyk (existuje i kompilovaná verze).
- Nejrozšířenější skriptovací jazyk pro web (82%).

`http://w3techs.com/technologies/overview/programming_language/all`

- Proč PHP: Technologická volba, zdarma dostupný, velká komunita, ...
- Proč ne PHP: Pro „mastodontní“ projekty ne příliš šťastná volba. Problémy s přechody na vyšší verze, syntaxe.
- Na druhou stranu: Celá řada velkých projektů v PHP.
- Dynamicky typovaný.

Historie vývoje jazyka PHP

- 1994, Common Gateway Interface (CGI) v C, (Rasmus Lerdorf) + Personal Home Page, původně pro nahrazení PERL skriptů.
- 1995, Oficiální název „Personal Home Page Tools (PHP Tools)“. Poprvé byl použit název PHP.
- 1996, PHP 2
- 1998, PHP 3, zcela přepsány původní zdrojové kódy.
- 2000 - 2005, PHP 4, značné vylepšené, poslední PHP 4.4.
- 2004, PHP 5, přichází s objektovým programováním, zlepšení výkonu, PHP 5.2 v roce 2006, poslední nepodporovaná verze, 5.3 (jen bezpečnostní mezery).
- 2013, červen, PHP 5.5, ohlášena verze 5.6.
- ????, PHP 6, zatím v nedohlednu (kromě knihkupectví) celá řada specifikací je již známá (neoficiálně). Velké pročištění jazyka, nativní podpora Unicode.

Jazyk PHP

Soubory s koncovkou `.php`. Vnořený skriptovací jazyk (HTML a PHP lze používat v jednom souboru). PHP kód bude interpretován pouze pokud bude soubor označen jako PHP soubor (koncovkou). Výsledek (HTML) je odeslán klientovy.

Obecně: Cílem webových skriptů na straně serveru je vytvořit HTML kód.

Příklad (Uvození PHP kódu)

```
1 <?php
2
3 echo "Hello world";
4
5 ?>
```

PHP může generovat celou HTML stránku, ale i jen její část.

Poslání dat klientovy

Příklad

```php
<?php
  echo "Hello world"; // nevraci zadnou hodnotu
  echo 'Hello world'; //  print 'Hello world';
  print "Hello world"; // vraci jedna, bere pouze jeden argument
  echo("Hello world"); // echo i print je funkce
  echo "This", " string", " was", " made", " with multiple parameters.";

  // escape
  echo "She said, \"How are you?\"";
  echo 'I\'m.';

  echo 'This sentence is not
  printed over two lines.';
  # dalsi komentar
  /* porad se neco komentuje */
  echo 'This sentence is not \n
  printed over two lines.';
?>
```

PHP je case sensitive (kromě jmen funkcí). White space jsou v duchu HTML.

Proměnné

- Vždy začínají $ (dolar).
- Datové typy: Boolean (true, false), integer, floating point, strings (řetezce), pole, objekty a NULL. Typ lze zjistit funkcí `gettype()`.

Příklad (Proměnná)

```
1 $moje_promenna = 10;
```

Příklad (Odeslání dat)

```
1 $name = "Jon Doe";
2
3 // tohle bude fungovat
4 echo "Hello, $name";
5 // tohle nebude fungovat
6 echo 'Hello, $name';
```

Předchozí příklad detailněji

Příklad (Odeslání dat)

```php
$name = "Jon Doe";

// tohle bude fungovat
echo "Hello, $name";

// tohle nebude fungovat (nevypise Jon Doe)
echo 'Hello, $name';
```

V PHP je možné použít pro uvození řetězce uvozovky nebo apostrof. **Je zde ale zásadní rozdíl**. Escape znaků je v tomto případě druhotná záležitost.

Řetězce

Příklad (Zřetezení)

```
1 $author = $first_name . ' ' . $last_name;
```

Zajímavé funkce:

- `implode`
- `explode`
- `strcmp`
- `strpos`
- `htmlspecialchars`
- `substr_replace`
- `trim`

Práce s řetězci: `http://cz1.php.net/manual/en/ref.strings.php` (mirror)

Syntaxe jazyka PHP

- Konstrukty: `if`, `else`, `else if`, `switch`, `for`, `while`, `do/while`, `foreach`, `break`, `continue`.
- Inkrementace, dekrementace, operátory typu +=.

Příklad (Větvení)

```
1  $t=date("H");
2
3  if ($t<"10") {
4    echo "Have a good morning!";
5  }
6
7  elseif ($t<"20") {
8    echo "Have a good day!";
9  }
10
11 else {
12   echo "Have a good night!";
13 }
```

Funkce

Příklad (Funkce)

```
1 function vrat( $cislo ) {
2   return $cislo*2;
3 }
4
5 echo(vrat(20));
```

Příklad (Funkce s výchozími parametry)

```
1 function vrat( $cislo=100) {
2   return $cislo*2;
3 }
4
5 echo(vrat(20));
6 echo(vrat());
```

Pro globální proměnné prefix: `global`. Lze i elegantněji.

Příklad (Anonymní funkce)

```php
<?php
  $greet = function($name) {
    printf("Hello %s\r\n", $name);
  };

  $greet('PHP');
?>
```

Příklad (Předání funkce hodnotou)

```php
<?php
  function add_some_extra(&$string) {
    $string .= 'and something extra.';
  }

  $str = 'This is a string, ';
  add_some_extra($str);
  echo $str;    // outputs 'This is a string, and something extra.'
?>
```

Příklad (Přetížení funkce, autor Lubalov)

```php
function overload() {
  $overloadFn = array();

  $overloadFn[0] = function() {
    return print nl2br("Zero argument. \n");
  };
  $overloadFn[1] = function($a) {
    return print nl2br("One argument. Result: $a \n");
  };
  $overloadFn[2] = function($a,$b) {
    return print nl2br("Two argument. Result: $a, $b \n");

  $countArgs = func_num_args();
  $valueArgs = func_get_args();

  if( isset($overloadFn[ $countArgs ]) ) {
    return call_user_func_array( $overloadFn[ $countArgs ], $valueArgs );
  }
}

overload (); // Zero argument.
overload (1); // One argument. Result: 1
overload (1,2); // Two argument. Result: 1, 2
```

Pole

Příklad (Pole)

```php
$pole["br"] = "brambor";
$pole["kv"] = "kvetak";
echo("nemam rad ".$pole["br"]." ani ".$pole["kv"]);

$age=array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
echo($age["Ben"]);

// uzitecna konstrukce
$pole = array();
$pole[] = 1;
$pole[10] = 2;
$pole[] = 3;

// dvou rozmerne pole
$cars = array (
  array("Volvo", 100, 96),
  array("BMW", 60, 59),
  array("Toyota", 110, 100)
 );
```

Procházení polem

Příklad (Procházení polem pomocí `foreach`)

```
1  $arr = array(1, 2, 3, 4);
2  foreach ($arr as $value) {
3      echo $value;
4  }
5
6  $arr = array("one", "two", "three");
7  reset($arr);
8  while (list($key, $value) = each($arr)) {
9      echo "Key: $key; Value: $value<br>";
10 }
11
12 foreach ($arr as $key => $value) {
13     echo "Key: $key; Value: $value<br>";
14 }
```

Lze použít i přímí přístup k prvkům pole `[]` a jiný typ cyklu. Funkce `count()`.

Práce s polem: `http://www.w3schools.com/php/php_ref_array.asp`

Setřídění pole

Metody:

- `sort()` - setřídění vzestupně.
- `rsort()` - setřídění sestupně.
- `asort()` - vzestupně vzhledem k hodnotě.
- `ksort()` - vzestupně vzhledem ke klíči.
- `arsort()` - sestupně, vzhledem k hodnotě.
- `krsort()` - sestupně vzhledem ke klíči.

Příklad (Třídění)

```
1 $numbers = array(4, 6, 2, 22, 11);
2 sort($numbers);
3
4 $age = array("Peter"=>"35", "Ben"=>"37", "Joe"=>"43");
5 asort($age)
```

Objekty

Příklad

```
1  class MyClass {
2   public $prop1 = "I'm a class property!";
3
4   public function setProperty($newval) {
5     $this->prop1 = $newval;
6   }
7
8   public function getProperty() {
9     return $this->prop1;
10  }
11 }
12
13 $obj = new MyClass;
```

- Metody: `__construct()`, `__toString()`, `__destruct()`.
- `extends` rozšíření třídy (možnost přepsání původních metod).
- `public, protected, private, static`.

Objekty

Příklad (Statické vlastnosti)

```
1 class MyClass {
2   public static $index = 0;
3 }
4
5 while(MyClass::$index < 10) {
6   echo MyClass::$index;
7   MyClass::$index++;
8 }
```

Příklad

```
1 $obj = (object) array ('a' => '1', 'b' => '2');
2
3 echo $obj->a;
4 echo $obj->b;
```

DocBlock

- Styl pro komentování tříd.
- Není součástí standardu PHP.
- Široce rozšířený, použití zejména různých IDE či ORM.
- Možnost použít různé tagy uvozené „@“.

Příklad (DocBlock)

```
1  /**
2   * Sets $foo to a new value upon class instantiation
3   *
4   * @param string $val a value required for the class
5   * @return void
6   */
7
8  public function __construct($val) {
9    $this->foo = $val;
10 }
```

Konstanty

- Název konstanty musí začínat číslicí nebo znakem _
- Obsah nelze změnit.

Příklad (Konstanty)

```
1 define("GREETING", "Welcome to W3Schools.com!");
2 echo GREETING;
3
4 // case sensitive konstanta
5 define("GREETING", "Welcome to W3Schools.com!", true);
6 echo greeting;
```

Matematika

- Celá řada vestavěných matematických funkcí a konstant.
- Více na: `http://www.w3schools.com/php/php_ref_math.asp`

Příklad (Zaokrouhlení)

```
1 echo ceil(2.3); //  nejblizsi  nahoru
2 echo floor(2.6); //  nejblizsi  dolu
3 echo round(2.5); // klasicke  zaokrouhleni
```

Regulární výrazy

- preg_match()
- preg_match_all()
- preg_replace()

Příklad

```
1 if (preg_match("/php/i", "PHP is the web scripting language of choice.")) {
2     echo "A match was found.";
3 } else {
4     echo "A match was not found.";
5 }
6
7 preg_match_all("/\d{3}-\d{4}/", "Call 555-1212, 555-2444 or 555-2345", $phones);
```

Více na: http://cz1.php.net/manual/en/ref.pcre.php

Odeslání dat na server

Základní možnosti:

1. Metodou GET (přes url adresu)
2. Metodou POST

Příklad (Zřetezení)

```
1 <form action="script.php" method="post">
2 ...
3 </form>
```

Otázka k zamyšlení: Jakou metodu používá prvek <a>?

Předefinované proměnné

Globální (superglobální) pole.

- $GLOBALS - References all variables available in global scope
- $_SERVER - Server and execution environment information
- $_GET - HTTP GET variables
- $_POST - HTTP POST variables
- $_FILES - HTTP File Upload variables
- $_REQUEST - HTTP Request variables
- $_SESSION - Session variables
- $_COOKIE - HTTP Cookies

Jsou vždy dostupné. V nových verzích PHP není možné registrovat nová glob. pole.

Příklad (Zřetezení)

```
1 echo 'Hello ' . htmlspecialchars($_POST["name"]) . '!';
```

Vkládání souborů

Příklad

```
1 require('somefile.php');
2 require 'somefile.php';
3 include 'somefile.php'; // varovani pokud dojde k chybe
4
5 // navic overi zda jiz soubor nebyl vlozen
6 require_once 'somefile.php';
7 include_once 'somefile.php';
```

Příklad (Vložení HTML stránky)

```
1 <?php include_once("menu.html") ?>
```

Důležité: první nástroj na udržení pořádku v kódu.

Session

Příklad (Základní práce)

```php
session_start();

// ulozeni dat
$_SESSION['views'] = 1;
$_SESSION['other'] = 2014;

if(isset($_SESSION['views'])) unset($_SESSION['views']);

// zruseni vsech ulozenych informaci
session_destroy();
```

Práce s cookies v PHP

Příklad (Základní práce)

```php
setcookie("user", "Alex Porter", time()+3600);

echo $_COOKIE["user"];

setcookie("user", "", time()-3600);
```

Pokud jsou cookies zakázány nebudou uložena žádná data.

Vyjímky

Příklad

```
1  function checkNum($number) {
2    if($number>1) {
3      throw new Exception("Value must be 1 or below");
4    }
5
6    return true;
7  }
8
9  try {
10   checkNum(2);
11   echo 'If you see this, the number is 1 or below';
12 }
13
14 catch(Exception $e) {
15   echo 'Message: ' . $e->getMessage();
16 }
```

Více na: `http://www.w3schools.com/php/php_exception.asp`

Práce se soubory

Příklad (Čtení souboru po řádcích)

```
1 $file = fopen("welcome.txt", "r") or exit("Unable to open file!");
2
3 while(!feof( $file )) {
4   echo fgets( $file ) . "<br>";
5 }
6 fclose( $file );
```

Příklad (Čtení souboru po znacích)

```
1 $file = fopen("welcome.txt","r") or exit("Unable to open file!");
2
3 while (!feof( $file )) {
4   echo fgetc( $file );
5 }
6 fclose( $file );
```

Práce se soubory

Příklad (Čtení celého souboru a zápis)

```
1  $file = 'people.txt';
2
3  // Open the file to get existing content
4  $current = file_get_contents($file);
5
6  // Append a new person to the file
7  $current .= "John Smith\n";
8
9  // Write the contents back to the file
10 file_put_contents ( $file , $current );
```

- Celá řada funkcí pro manipulaci se soubory, změnu oprávnění a čtení a zapisování souborů.
- Více na: `http://www.w3schools.com/php/php_ref_filesystem.asp`

Čtení obsahu adresáře I.

Funkce `dir()` vrací objekt reprezentující adresář. `getcwd()` vrací ukazatel na aktuální pracovní adresář.

Příklad

```
1 $d = dir(getcwd());
2
3 echo "Handle: " . $d->handle . "<br>";
4 echo "Path: " . $d->path . "<br>";
5
6 while (( $file  = $d->read()) !== false){
7   echo "filename: " . $file  . "<br>";
8 }
9 $d->close();
```

Metoda `rewind()`. Soubory jsou vraceny v pořadí v jakém jsou uloženy v systému (včetně složek . a ..).

Čtení obsahu adresáře II.

Příklad (Varianta bez objektů)

```php
$dir = "/images/";

// Open a directory, and read its contents
if (is_dir($dir)){
  if ($dh = opendir($dir)){
    while (( $file = readdir($dh)) !== false){
      echo "filename:" . $file . "<br>";
    }
    closedir($dh);
  }
}
```

Příklad (Skenování adresáře)

```php
$dir = "/images/";
$a = scandir( $dir );
```

Práce s datem a časem

Příklad (Základní práce)

```php
echo date("Y/m/d");
echo date("Y.m.d");
echo date("Y-m-d");

$tomorrow = mktime(0, 0, 0, date("m"), date("d")+1, date("Y"));

$date = date_create("2013-03-15");
date_add($date, date_interval_create_from_date_string("40 days"));

$date1 = date_create("2013-03-15");
$date2 = date_create("2013-12-12");
$diff  = date_diff($date1, $date2); // m = 8, d = 27

$t = time(); // timestamp Leden 1 1970 00:00:00 GMT
```

Více na `http://www.w3schools.com/php/php_ref_date.asp`.

Užitečné funkce

- `print_r` - přívětivé zobrazení obsahu proměnných (vhodné zejména pro složené datové typy).
- `var_dump` - informace o datovém typu.
- `header` - slouží k přesměrování přes HTTP.

Příklad

```
1 header('Location: http://www.example.com/');
```

Filtrace

- Validace dat z neznámých zdrojů.
- Málo používané, většinou vlastní validace.

Příklad

```
1 $int = 'q';
2
3 if(! filter_var ($int, FILTER_VALIDATE_INT)) {
4   echo("Integer is not valid");
5 }
6 else {
7   echo("Integer is valid");
8 }
```

Filtrace

Příklad

```
1  $var=300;
2
3   $int_options  = array(
4  "options"=>array
5    (
6    "min_range"=>0,
7    "max_range"=>256
8    )
9  );
10
11 if(! filter_var ($var, FILTER_VALIDATE_INT, $int_options)) {
12   echo("Integer is not valid");
13 }
14 else {
15   echo("Integer is valid");
16 }
```

Filtrace

Příklad (Filtrace vstupního super globálního pole)

```
1  $filters = array
2   (
3   "name" => array("filter"=>FILTER_SANITIZE_STRING),
4   "age" => array
5     (
6     "filter"=>FILTER_VALIDATE_INT,
7     "options"=>array
8       (
9       "min_range"=>1,
10      "max_range"=>120
11      )
12    ),
13  "email"=> FILTER_VALIDATE_EMAIL
14  );
15
16 $result = filter_input_array (INPUT_GET, $filters);
17 var_dump($result);
```

Více na: `http://www.w3schools.com/php/php_filter.asp`

Magické konstanty

Obsah konstant je určen kontextem jejich použití.

- `__LINE__`
- `__FILE__`
- `__DIR__`
- `__FUNCTION__`
- `__CLASS__`

Více na:
`http://us3.php.net/manual/en/language.constants.predefined.php`

Odbočka: Model, view, controller (MVC) v PHP

Model:

- Zodpovídá za správu dat.
- Obvykle napojen na databázi nebo datové uložiště.

View (Presentation):

- Zodpovídá za zobrazení dat (webu).
- Někdy též označován jako šablona.

Controller:

- Reaguje na požadavky od klienta.
- Je napojen na Model.

Ačkoliv MVC architektura není nijak dogmatická, je osvědčená a doporučená pro celou řadu situací včetně návrhu webové aplikace. Především vede na pořádek ve zdrojových kódech a určuje jasnou logiku aplikace.

Jak je to tedy v praxi

Model:

- Celá řada podob. PHP společně s databázovým jazykem např. MySQL.
- Striktně vzato pouze zapozdření databáze.

View (Presentation):

- V nejednoduší podobě HTML a jazyk PHP.
- Lze využít i pokročilejší možnosti(šablonovací systémy, webové frameworky).
- Obvykle každá stránka má svůj vlastní View, který je součástí hlavního view.

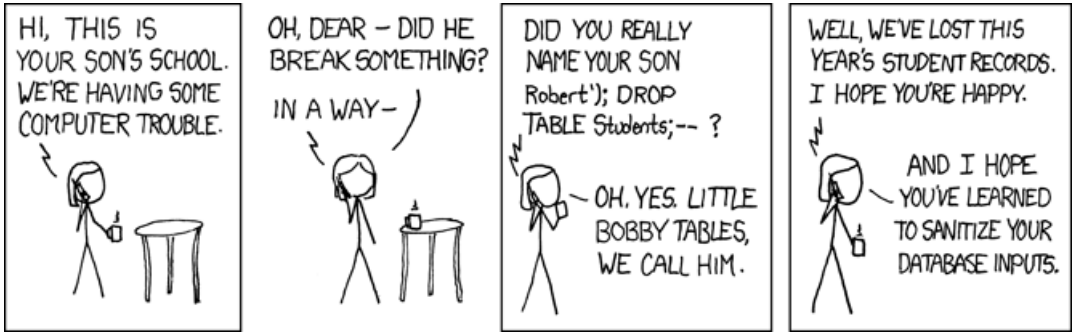
Controller:

- Skriptovací jazyk (PHP).
- Podle události se vykonávají jednotlivé části kódu.
- Obvykle každá stránka má svůj vlastní Controller.

Aplikace jsou obvykle mnohem komplikovanější. Například inicializace, ověření uživatele, . . . často označované jako jádro systému.

Bezpečnost

- Cross-site scripting (XSS) - neošetřené vstupy.
- SQL injection - neošetřené vstupy.
- Spouštění kritických skriptů bez oprávnění a kontextu.



XKCD, Exploits of a Mom, `http://xkcd.com/327/`

Ostatní

O čem jsme nemluvili:

- Práce s e-mailem.
- Funkce pro práci se ZIP soubory.
- Práce s FTP.
- Time Zone.
- Další pokročilé možnosti.

Více na `http://www.w3schools.com/php/` a jiné zdroje.

Závěr

Důležité pojmy:

- Dynamická webová stránka, jazyk PHP.
- Syntaxe a použití jazyka PHP.

Čtení na doma:

- Ullman L., PHP and MySQL for Dynamic Web Sites, 4th Edition, 2012, ISBN 978-0-321-78407-0. (kapitoly 1-3).
- `http://cz1.php.net/` (mirror)
- `http://www.w3schools.com/php/`

Kniha: PHP and MySQL for Dynamic Web Sites

- Český ekvivalent neexistuje. Alternativa: „PHP 6 Programujeme profesionálně“, „PHP 6, MySQL, Apache“, „PHP 5 a MySQL 5“ od cpressu.