Vznik, historie

PHP je programovací jazyk, který pracuje na straně serveru. S PHP můžete ukládat a měnit data webových stránek. Původní význam zkratky PHP byl **Personal Home Page**. Vzniklo v roce 1996 a bylo vytvoreno programatorem Rasmusem Lerdorfem, od té doby prošlo velkými změnami a nyní tato zkratka znamená **PHP: Hypertext Preprocessor**.

Základní princip fungování

PHP je **interpretovaný jazyk**, server čte script realtimově při jeho zavolání, script není předkompilovaný a nečeká pouze na spuštění v binární podobě. Všechny proměnné a datové typy se dosazují dynamicky a ani samotné PHP nepožaduje určovaní datových typů, o vše se stará server.

Webové aplikace často používají linuxový operační systém, webserver **Apache**, databázi **MySQL** a programovací jazyk **PHP**. Tato čtveřice je často označována zkratkou **LAMP**. Všechny zmíněné technologie jsou zadarmo. Pokud následující technologie provozujeme pod Windows, používá se někdy zkratka **WAMP**, pokud pod MAC OS, tak **MAMP**.

Asi nejpoužívanější instalační balíček, který obsahuje **Apache**(HTTP server), **PHP**(interpretovaný programovací jazyk) a **MySQL**(Databázový server), se nazývá XAMPP. Instalace je velmi jednoduchá.

Syntaxe

Základní syntaxe

PHP se zapisuje do souboru mezi prvky HTML a soubory se liší se pouze koncovkou a to **.php**. V těchto souborech se php script zapisuje mezi značky **<?php ?>**. K vypsaní do html slouží základní funkce **echo();** a proměnné se deklarují pomocí **$** (**$nazev_promenne**). **Není** vyžadováno učovat datový typ proměnné. O tuto funkci se

stará samotný server při interpretaci scriptu a v průběhu scriptu se může datový typ proměnné i změnit.

Pole se inicializuje pomocí proměnné, do které se dosadí objekt **array()** (**$nazev_promenne = array();**) nebo hranaté závorky (**$nazev_promenne = [];**). Hodnotu do pole můžeme dosadit buďto na specifický index (**$promenna[1] = 305;**) nebo na konec pole (**$promenna[] = 1;**) nebo pomocí předdefinovaných funkcí **array_push()** a **array_unshift()**.

Pro vypsání informací, které jsou v, pro člověka, čitelné formě je v php funkce **print_r()**. Pokud se tato funkce použije na pole, funkce vypíše celé pole se všemi hodnotami. V PHP existuje tzv. asociativní pole, které umožňuje poli ukládat hodnoty do páru s klíčem, který nemusí být pouze číselný, ale může být i ve formě textového řetězce.

```
$oblibeneVeci = array(
    'homer' => 'kobliha',
    'marge' => 'trouba',
    'bart' => 'prak',
    'liza' => 'kniha',
    'meggie' => 'dudlík',
);
```

Řídící prvky, podmínky, porovnávání proměnných

V php existují podmínkové bloky **if**, **elseif** a **else**, které fungují stejně jako ve všech ostatních programovacích jazycích, za slovem **if** a **elseif** je podmínka, která pokud je splněna tak se provede blok kódu za podmínkou, pokud podmínka není pravdivá(není splněna), pokračuje program blokem **else** nebo pokračuje dál ve čtení kódu. Podmínky u **if** a **elseif** lze řetězit pomocí logických operátorů **&&(AND)** nebo **||(OR)**. V php je speciální možnost porovnávat i datový typ proměnných pomocí **===**, protože php datové typy přiřazuje dynamicky a proměnné při běhu programu mohou měnit svůj datový typ

Switch je jeden z řídících prvků, který příjmá vstup, který může být jak číselný tak textový řetězec a switch poté najde ve svém těle blok kódu, který je pojmenován jako vstup, který

switch dostal. Tyto bloky se nazývají **case** a mohou být označeny číselnou i textovou hodnotou. Case je ukončen příkazem **break**, pokud switch není ukončen, je prováděn kód sekvenčně, dokud nenarazí na konec bloku switche nebo příkaz break.

PHP má k dispozici všechny cykly. Patří mezi ně for, while, do while a foreach. Jediným neobvyklým cyklem může být foreach, protože není ve všech programovacích jazycích defaultně implementován. Jeho zápis je **foreach($kolekce as $prvek)** a prochází celé pole a v jedné iteraci ukládá aktuální prvek pole do proměnné, s kterou je v těle cyklu možno manipulovat.

Zpracování formulářů

HTML má možnost odesílat data z formulářů pomocí dvou metod **GET** a **POST**. PHP data odeslané pomocí těchto dvou metod umí zpracovat a přistoupit k nim.

GET metoda posílá data přes textový řetězec v odkazu stránky, který se nazývá **query string**. Je zobrazován ve formátu názvu proměnné s přiřazenou hodnotou nacházející se za otazníkem. Proměnné jsou odděleny pomocí **&** (soubor.php?cislo1=10&cislo2=20). Touto metodou by se neměly posílat citlivé informace jako hesla. Slouží především k zobrazení daného produktu nebo stránky na daném webu. PHP může přistoupit k datům pomocí pole **$_GET[]**.

POST metoda posílá data přes **HTTP hlavičky(HTTP headers)**. Přenechává tak zabezpečení na HTTP. Pokud je zabezpečen přenos HTTP jsou zabezpečeny i data posílané přes hlavičku. Data nejsou viditelné v odkazu a k přistoupení k datům slouží v php pole **$_POST[]**.

Objektově orientované programování

Objektové programování v PHP je podobné spíše programovacímu jazyku C++. Třída je definována slovem **class** a má své tělo. Třída obsahuje proměnné a metody, které mohou mít přístupové modifikátory **public**, **private** a **protected**.

- Public - viditelné v celém programu
- Private – viditelné pouze ve třídě, kde je definováno

- Protected - viditelné v třídě, ve které je prvek definován a ve třídě, která od třídy dědí

Třída může být také abstraktní, třída se tak stává jakýmsi plánem pro třídy, které z ní budou dědit. Abstraktní třída může být pouze zděděna, nemůže mít vlastní instanci a nemůže obsahovat funkce s definovaným tělem.

Třídy mohou obsahovat statické proměnné, ke kterým se přistupuje pomocí **:: (Scope resolution operator)** (**NazevTridy::$nazevStatickePromenne**). Pokud chceme ve třídě, ve které je statická proměnná definována, přistoupit ke statické proměnné můžeme tak udělat pomocí klíčového slova **self(self::$nazevStatickePromenne)**.

Třída nebo metoda v ní může být definována s klíčovým slovem **final**. U třídy to znamená, že už dále nemůže být děděna. U metody to znamená, že nemůže být přepsána.

```php
<?php
   class Books {
      /* Member variables */
      var $price;
      var $title;

      /* Member functions */
      function setPrice($par){
         $this->price = $par;
      }

      function getPrice(){
         echo $this->price ."<br/>";
      }

      function setTitle($par){
         $this->title = $par;
      }

      function getTitle(){
         echo $this->title ." <br/>";
      }
   }
?>
```

V php jsou implementovány také **rozhraní (interface)**. Datová struktura, která uchovává metody bez těla a třída, která implentuje rozhraní pomocí slova **implements** musí všechny metody, nacházející se v rozhraní, přepsat.

```php
$physics->getTitle();
$chemistry->getTitle();
$maths->getTitle();
$physics->getPrice();
$chemistry->getPrice();
$maths->getPrice();
```

```php
$physics = new Books;
$maths = new Books;
$chemistry = new Books;
```

Databáze

Jedním ze základních principů webových stránek je komunikace s databázovým systémem. PHP umožnují tuto komunikaci **rozšíření/ovladačů MySQLi** nebo **PDO**(PHP Data Objects). Oba ovladače mají svůj vlastní zápis připojení k databázi a následného zadávání vyhledávacích řetězců. **PDO** má pouze objektový zápis a dokáže komunikovat až s 12 typy databází. **MySQLi** dokáže komunikovat pouze s MySQL databází a nabízí jak objektový, tak procedurální zápis. Oba ovladače také podporují **Prepared Statements**, které ochrání náš web před SQL injection útokem.

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = new mysqli($servername, $username, $password);

// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}
echo "Connected successfully";
?>
```

Připojení k databázi přes MySQLi objektově orientovaný zápis

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";

// Create connection
$conn = mysqli_connect($servername, $username, $password);

// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}
echo "Connected successfully";
?>
```

Připojení k databázi přes MySQLi procedurální zápis

```php
<?php
$servername = "localhost";
$username = "username";
$password = "password";

try {
    $conn = new PDO("mysql:host=$servername;dbname=myDB", $username, $password);
    // set the PDO error mode to exception
    $conn->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    echo "Connected successfully";
    }
catch(PDOException $e)
    {
    echo "Connection failed: " . $e->getMessage();
    }
?>
```

Připojení k databázi pomocí PDO

Session

PHP má možnost krátkodobě uchovávat informace, které mohou být použity na každé stránce webové aplikace. Od cookies se liší tím, že session informace nejsou uloženy v počítači uživatele, ale na serveru. Pro dlouhodobější uchovávání dat je vhodnější databáze.

```php
<?php
   session_start();

   if( isset( $_SESSION['counter'] ) ) {
      $_SESSION['counter'] += 1;
   }else {
      $_SESSION['counter'] = 1;
   }

   $msg = "You have visited this page ".  $_SESSION['counter'];
   $msg .= "in this session.";
?>

<html>

   <head>
      <title>Setting up a PHP session</title>
   </head>

   <body>
      <?php  echo ( $msg ); ?>
   </body>

</html>
```

Cookies

Je malý soubor, který server vytvoří v uživatelově počítači. Často se používá k identifikaci uživatele a pokaždé, když si uživatel vyžádá stránku, tak se cookies posílají společně s požadavkem.

PHP umožňuje vytváření i přístup k souborům cookies. K vytváření cookies slouží funkce **setcookie(name, value, expire, path, domain, secure, httponly);** a k přístupu slouží pole **$_COOKIE[]**.