

Szerver oldali technológiák

Szerver oldali script nyelvek – PHP

Áttekintés

- ▶ Szerver oldali technológiák
- ▶ Szerver oldali script nyelvek – PHP
 - ▶ Általános jellemzők
 - ▶ Formkezelés
 - ▶ Állománykezelés
 - ▶ Cookie (süti)
 - ▶ Szessziókövetés
 - ▶ MySQL adatbázishozzáférés

Szerver oldali technológiák

- ▶ CGI (Common Gateway Interface)
- ▶ ColdFusion – HTML kiterjesztés, új tag-ekkel, melyek speciális műveletek elvégzését teszik lehetővé (pl. adatbázishozzáférés)
- ▶ Server Side Includes (SSI) – szerver oldali scriptek lefuttatását teszi lehetővé a HTML-be ágyazott parancsok segítségével.
- ▶ Server-Side JavaScript (SSJS) – JavaScript kiterjesztése újabb, szerver-oldalon működő objektumokkal

Szerver oldali technológiák

- ▶ PHP
- ▶ Servlet – Sun által 1996-ban bevezetett technológia
- ▶ Java Server Pages (JSP)– Servlet technológiára épül
- ▶ Active Server Pages (ASP) – Microsoft által támogatott szerver-oldali script-nyelv – főként Windows-os platformokra
- ▶ Active Server Pages.NET (ASP.NET) – a Microsoft .NET keretrendszer része
 - új elemeket hoz az ASP-hez képest (C# ↔ Java, CLR ↔ JVM)

Szerver oldali script nyelvek

általános jellemzők

- ▶ a script végrehajtására a szerver gépen kerül sor
- ▶ a HTML-be beágyazott script-ről van szó
- ▶ egy script-értelmező motor dolgozza fel – nincs szükség fordításra

pl.

- ▶ PHP
- ▶ ASP (Microsoft)

Dinamikusan létrehozott HTML oldal

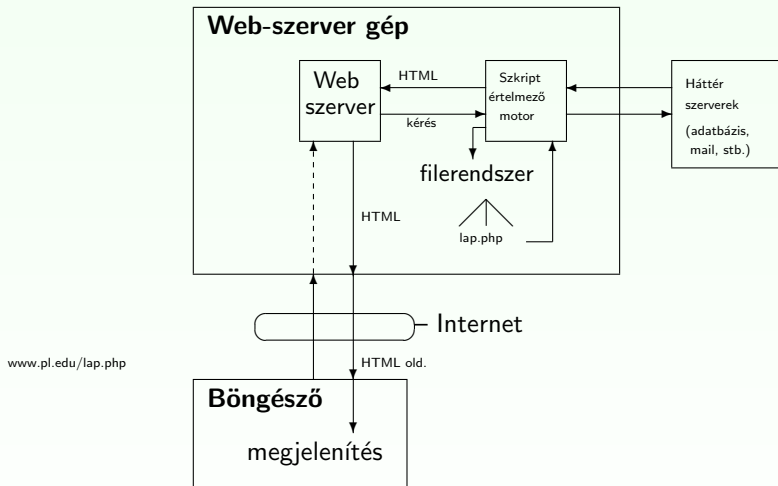


Figure: HTTP kérés web-alkalmazás esetén

általános jellemzők:

- ▶ PHP – **PHP: Hypertext Preprocessor** (eredetileg: Personal Home Page – Rasmus Lerdorf, 1994)
- ▶ jelenleg a legelterjedtebb szerver-oldali script nyelv
- ▶ nyílt forráskódú, ingyenes (<http://www.php.net>)
- ▶ beágyazható a HTML oldalba
- ▶ platformfüggetlen – a legelterjedtebb op. rendszereket támogatja (Unix alapú op. rendszerek, Microsoft Windows, Mac OS X)
- ▶ leggyakrabban az **Apache** Web-szerverrel (más: Microsoft Internet Information Server, Personal Web Server, Xitami, stb.)
- ▶ PHP állományok kiterjesztése: “.php”, “.php[3 | 4| 5]”, “.phtml”

lehetőségek:

- ▶ dinamikus és interaktív web-oldalak készítése
- ▶ különböző adatbázisok támogatása (**MySQL**, Oracle, PostgreSQL, Generic ODBC, stb.)
- ▶ kommunikáció egyéb szolgáltatásokat nyújtó szerverekkel (pl. LDAP, IMAP, POP3, SMTP)
- ▶ használható szövegfeldolgozásra, illetve XML állományok feldolgozására
- ▶ (használható parancssorból, illetve kliens oldali alkalmazásként)

PHP

Kellékek

- ▶ Web-szerver: Apache (<http://www.scs.ubbcluj.ro/>)
- ▶ PHP (Script-értelmező motor: Zend)
- ▶ (adatbázis-szerver: MySQL)

otthoni használatra

- ▶ Apache (<http://httpd.apache.org/>)
- ▶ PHP (<http://www.php.net/downloads.php>)
 - ▶ installálás CGI bináris állományként
 - ▶ installálás Apache modulként
- ▶ MySQL (<http://dev.mysql.com/downloads/>)
- ▶ XAMPP (apache+php+mysql egyben):
<http://www.apachefriends.org/en/xampp-windows.html>

PHP script beszúrása

beágyazása a HTML oldalba (standard):

```
<?php  
...  
?>
```

nem standard:

```
<script language="php"> </script>; <? ?>; <% %>
```

include vagy *require* függvény segítségével

- ▶ a paraméterként megadott állomány tartalmát egyszerű szöveggént szűrik be
- ▶ ha PHP kódot vagy kódrészletet szeretnénk beszúrni, azt “<?php”, “?>” tag-ek közé kell írni
- ▶ lásd: beszur.php, allatnev.inc

Szintaxis

három helyről örökölt:

- ▶ C, shell script, Perl script

általános jellemzők:

- ▶ “engedékeny”
(hibákra való érzékenysége állítható: `error_reporting(E_ALL)`)
- ▶ változónevek esetén számít a kis- vagy nagybetű, függvénynevek vagy kulcsszavak esetén nem

hasonlóságok a C szintaxissal:

- ▶ operátorok (=, +=, &&, ||, !, <, >, ==, +, -, *, %, stb.)
- ▶ kulcsszavak (if, while, for, switch, break, continue)
- ▶ számos függvénynév (pl. printf, string-, illetve filekezelő műveletek, stb.)
- ▶ kommentek (// vagy /* ... */)
- ▶ a fehér karakterek száma nem számít
- ▶ ";"-vel lezárt parancsok

Hasonlóságok a shell illetve Perl script-el:

- ▶ lehet (Perl vagy) shell script-szerű megjegyzést is használni (#)
- ▶ változók neve \$ jellel kezdődik (Perl)
- ▶ echo, (egy másik alternatíva: print("egyetlen argumentum"))
- ▶ idézőjelek: " (a közé írt változók ki lesznek értékelve),
' (\\, \' kivételével betű szerint értelmezi a szöveget)
- ▶ reguláris kifejezések használata

változók:

- ▶ típusuk meghatározása dinamikusan történik, értékadáskor (egész: 21, valós: 3.14, karaktorsor: "szoveg", logikai: TRUE/FALSE, tömb, objektum)
- ▶ nem kell deklarálni
- ▶ nevük: \$ karakterrel kezdődik
 - csak betűt, számjegyet vagy "_" karaktert tartalmazhat (\$ után nem kezdődhet számjeggyel)
- ▶ kis- vagy nagybetű számít
- ▶ használható referencia: \$valt1 = &\$valt2;

változók:

- ▶ alapértelmezett kezdeti értékek: FALSE, 0, üres string, üres tömb (*isset()* segítségével megvizsgálható, hogy volt-e már érték adva egy változónak)
 - ▶ függvényen belül minden változó alapértelmezés szerint lokális (globális változók elérése függvényen belül: `global` kulcsszóval, vagy `$GLOBALS[<változónév>]`)
lásd: [globalis.php](#)
 - ▶ statikus változók: lásd [statikus_valt.php](#)
 - ▶ változó változónevek: lásd [valt_valt.php](#)
 - ▶ automatikus típuskonverzió
-
- ▶ `var_dump()`; `print_r()` – információt ad a megadott változóról

Tömbök

- ▶ PHP-ban a tömb gyakorlatilag egy sorszámozott asszociatív tömb (map)

létrehozása:

- ▶ `array(kulcs => ertekek, ...)`
ahol a kulcs lehet numerikus vagy string, az érték pedig bármilyen típusú lehet
- ▶ egyenkénti értékadással

Tömbök

törlés

- ▶ `unset()`
`$arr = array(5 => 1, "szam" => 21);`
 - ▶ elem törlése: `unset($arr["szam"]);`
 - ▶ tömb törlése: `unset($arr);`

tömb elemeinek bejárása:

- ▶ `for` – numerikus index esetén
- ▶ `foreach`:
`foreach ($tomb as $ertek) ...`
`foreach ($tomb as $kulcs => $ertek) ...`

Pl.

lasd: Tomb → `tomb.php`

Függvények

saját függvény definiálása:

```
<?php
function fgnev($arg_1, $arg_2, ..., $arg_n)
{
    ...
    return $retval;
}
?>
```

- ▶ feltételes függvények (csak a feltétel teljesülését követően érhető el)
- ▶ függvényen belül definiált függvény (csak a külső függvény meghívását követően válik elérhetővé)
- ▶ változófüggvények

Objektumok használata (PHP 5)

osztálydefiníció (lásd. [egyszerupl.php](#))

```
<?php
class MyClass {
    // adattag deklaráció
    public $var = 'alapertelmezett ertek';
    // metódus deklaráció
    public function displayVar() {
        echo $this->var;
    }
} ?>
```

Osztálypéldány létrehozása, hivatkozások

```
<?php
$obj = new MyClass();
echo $obj->var;
$obj->displayVar();
?>
```

Objektumok – öröklődés

öröklődés

- ▶ egyszeres öröklés – **extends** kulcsszó
- ▶ az örökölt metódusok és adattagok felülírhatóak (kivétel: szülő osztályban **final**-al deklarált metódus)
- ▶ felülírt metódusra vagy adattagra hivatkozhatunk **parent::** segítségével
- ▶ amennyiben a gyerek rendelkezik saját konstruktorral (`__construct` függvény) vagy destruktoral (`__destruct` függvény), akkor a szülő konstruktora/destruktora NEM lesz meghívva automatikusan

Pl.

lásd. `oroklodes.php`

Objektumok

metódusok/adattagok láthatósága

- ▶ public, protected, private

egyéb lehetőségek:

- ▶ absztrakt osztály definiálása (**abstract** kulcsszó)
- ▶ interfész (**interface**, **implements** kulcsszavak)

statikus adattagok/metódusok

lásd.: [egyebek.php](#)

PHP beépített globális változók (superglobals)

PHP beépített globális változók (asszociatív tömbök):

- ▶ **\$GLOBALS** – az összes globális változót tartalmazó tömb
- ▶ **\$_SERVER** – a Web-szerver által beállított értékek, illetve az aktuális script-el kapcsolatos értékek ('PHP_SELF', 'argv' – GET, query string, 'SERVER_ADDR', 'REQUEST_METHOD', 'DOCUMENT_ROOT', stb.)
- ▶ **\$_GET** – a (GET módszerrel küldött) kérés paraméterek
- ▶ **\$_POST** – (POST módszerrel küldött) változók
- ▶ **\$_COOKIE** – HTTP-n küldött süтик
- ▶ **\$_FILES** – HTTP post módszerrel feltöltött állományok
- ▶ **\$_ENV** – környezeti változók
- ▶ **\$_REQUEST** – GET, POST, és COOKIE változók
- ▶ **\$_SESSION** – az aktuális szesszióhoz tartozó változók

Formkezelés

- ▶ az elküldött kérés (submit) paraméterei automatikusan elérhetőek a PHP scriptből: `$_POST`, `$_GET` , vagy `$_REQUEST` tömbök segítségével

Pl.

lásd: Formpl → form_pl.php, form_post.htm, form_get.htm,
file_upload_form.htm

Filekezelés

gyakran használt függvények:

- ▶ fopen, fclose, fwrite, fread, fgets, filesize, feof, file_exists, delete, rename, stb.

pl.

lásd: Filekezeles → file_modosit.php, fwrite.php

Süti (cookie)

- ▶ kisméretű szöveges információ, mely a kliens gépén van tárolva
- ▶ leggyakrabban a kliens azonosítására szolgál

cookie létrehozás:

- ▶ `setcookie(nev, ertek, lejar, ut, domenium);`
a `<html>` tag előtt kell megjelenjen

elérhetősége

a `$_COOKIE` változón keresztül

süti törlés lejárási időpont állításával

```
<?php setcookie("user", "", time()-3600); ?>
```

pl.

lásd: `Sutik` → `cookie_set.php`, `szamlalo.php`

Szessziókövetés

Szesszió (munkamenet, session):

- ▶ egy konkrét felhasználóra vonatkozó információkat a *szesszióban* tároljuk ideiglenesen
- ▶ az itt tárolt változók az egész web-alkalmazáson belül hozzáférhetőek
- ▶ egy egyedi *azonosító* lesz minden egyes felhasználóhoz hozzárendelve, ennek alapján történik a felhasználó beazonosítása
- ▶ az azonosító sütiben van tárolva, vagy az URL-en keresztül történik a közvetítése

Szessziókövetés

szesszió indítás:

```
<?php session_start(); ?>
```

a <html> tag ELŐTT kell szerepeljen

változók lementése

- ▶ a \$_SESSION tömb elemeként: `$_SESSION['nr']=1;`

törlés:

- ▶ változó törlése: `unset($_SESSION['nr'])`
- ▶ a teljes szesszió törlése: `session_destroy();`

PI

Szessziokovetes → szessziotovabbitas.php, kov.php; bevasarol.php,
adatok_a_szessziobol.php

MySQL adatbázishozzáférés

gyakori függvények:

- ▶ csatlakozás az adatbázishoz: `mysql_connect()`;
- ▶ adatbázis kiválasztása: `mysql_select_db()`;
- ▶ SQL parancs végrehajtása: `mysql_query()`;
- ▶ egy bejegyzés lekérése a lekérdezés eredményéből:
`mysql_fetch_array()`;
- ▶ kapcsolat bezárása: `mysql_close()`;

Pl.

lásd: MySQL → 1_lekerdez.php, 2_lekerdez.php, form_adat_bevitel.php,
pl_komplex.php