

Sikos László:

## **Szerver oldali webprogramozás**



Sikos László:

# **Szerver oldali webprogramozás**

**BBS-INFO Kiadó – 2004.**

Minden jog fenntartva! A könyv vagy annak oldalainak másolása, sokszorosítása csak a kiadó írásbeli hozzájárulásával történhet.

A könyv nagyobb mennyiségben megrendelhető a kiadónál:  
BBS-INFO Kft. 1630 Bp. Pf. 21. Tel.: 407-17-07

A könyv megírásakor a szerző és a kiadó a lehető legnagyobb gondossággal járt el. Ennek ellenére a könyvben előfordulhatnak hibák. Az ezen hibákból eredő esetleges károkért sem a szerző sem a kiadó semmiféle felelősséggel nem tartozik, de a kiadó szívesen fogadja, ha e hibákra felhívják figyelmét.

ISBN 963 86392 5 3  
E-book ISBN 9786156364173

Kiadja a BBS-INFO Kft.  
1630 Budapest, Pf. 21.  
Felelős kiadó: a BBS-INFO Kft. ügyvezetője  
Készült a debreceni Kinizsi nyomdában  
Felelős vezető: Bördős János ügyvezető igazgató

# TARTALOMJEGYZÉK

<b>AJÁNLÁS</b> .....	<b>11</b>
<b>ELŐSZÓ</b> .....	<b>13</b>
<b>1. HOGYAN FOGJUNK HOZZÁ?</b> .....	<b>15</b>
1.1. Amit feltétlenül tudni kell.....	15
1.2. A szerver-oldali JavaScript lehetőségei.....	16
1.3. A szerver jelentősége.....	18
<b>2. A JAVASCRIPT NYELV ÁTTEKINTÉSE</b> .....	<b>19</b>
2.1. A JavaScript nyelv magja, a kliens és a szerver oldali JavaScript .....	19
2.2. A JavaScript és a Java.....	21
2.3. JavaScript verziók.....	22
2.4. Hibakeresés .....	23
<b>3. SZERVER-OLDALI ALKALMAZÁSOK FEJLESZTÉSE</b> .....	<b>24</b>
3.1. Bevezetés a fejlesztésbe.....	24
3.1.1. A JavaScript alkalmazások felépítése.....	25
3.1.2. Rendszerkövetelmények .....	27
3.1.3. Konfigurálás .....	28
3.1.3.1. A szerver-oldali JavaScript engedélyezése.....	28
3.1.3.2. Védelmi kérdések .....	29
3.1.3.3. Előkészület a LiveConnect-hez .....	30
3.1.3.4. A fordító elhelyezése .....	30
3.2. JavaScript alkalmazások fejlesztési mechanizmusai.....	31
3.2.1. A fejlesztés alaplépései.....	31
3.2.2. Az alkalmazás-menedzselés áttekintése.....	33
3.2.3. Az alkalmazás forrásfájljainak létrehozása .....	34
3.2.4. Új alkalmazás telepítése .....	34
3.2.4.1. Az alkalmazások URL-jei .....	36
3.2.5. Az alkalmazáshoz való hozzáférés vezérlése .....	37
3.2.6. Egy alkalmazás módosítása.....	38
3.2.7. Egy alkalmazás eltávolítása .....	38
3.2.8. Alkalmazás indítása, leállítása, újraindítása.....	38

3.2.9. Alkalmazás futtatása .....	39
3.2.10. Hibakeresés az alkalmazásban.....	40
3.2.10.1. Az alkalmazás-menedzser használata hibakeresésre .....	40
3.2.10.2. Hibakereső URL-ek használata .....	41
3.2.10.3. A debug függvény .....	41
3.2.11. Egy alkalmazás feldolgozása .....	42
3.2.12. Az alkalmazás-menedzser áttekintése.....	42
3.2.12.1. Alapértelmezett beállítások konfigurálása.....	42
3.2.12.2. A színtalpak mögött.....	43
<b>4. A SZERVER-OLDALI JAVASCRIPT .....</b>	<b>45</b>
4.1. Egy példa: Helló, világ! .....	46
4.1.1. A program működése .....	47
4.1.2. A forrás kielemezése .....	47
4.1.3. A program módosítása .....	50
4.2. A futásidejű feldolgozás .....	51
4.3. A nyelvi elemek áttekintése .....	52
4.3.1. Prototípusok .....	53
4.3.2. Használat .....	54
4.3.3. A környezet .....	54
4.3.4. Osztályok és objektumok .....	55
4.4. A HTML-be ágyazás .....	57
4.4.1. A SERVER tag.....	58
4.4.2. Futásidejű feldolgozás a szerveren .....	59
4.5. A HTML oldal létrehozása .....	61
4.5.1. A HTML generálás.....	62
4.5.2. A kimeneti puffer .....	62
4.5.3. Váltás egy új kliens-kérelemre.....	63
4.6. CGI változók elérése .....	63
4.7. Kommunikáció a szerver és a kliens között.....	67
4.7.1. Kliens-szerver adatforgalom .....	68
4.7.1.1. Űrlap-értékek elérése .....	68
4.7.1.2. Legördülő listák használata.....	69
4.7.1.3. Információ-kódolás egy URL-ben .....	69
4.7.2. Szerver-kliens adatforgalom .....	70
4.7.3. Sütik használata.....	70
4.7.4. Hulladék-gyűjtemény .....	70
4.7.5. Hibakezelés a szerver-oldali JavaScriptben .....	71
4.8. Beépített objektumok.....	72
4.8.1. A client objektum.....	73
4.8.1.1. A client objektum tulajdonságai .....	74
4.8.1.2. Egyedi hivatkozás a client objektumra .....	75
4.8.1.3. Saját client objektum létrehozása.....	76
4.8.2. A project objektum .....	77
4.8.2.1. A project objektum tulajdonságai .....	77

4.8.2.2. A project objektum megosztása .....	78
4.8.3. A request objektum .....	78
4.8.3.1. A request objektum tulajdonságai .....	78
4.8.3.2. Térképek használata .....	80
4.8.4. A server objektum .....	80
4.8.4.1. A server objektum tulajdonságai .....	80
4.8.4.2. A server objektum megosztása .....	81
4.8.4.3. Az objektumok és az URL-ek kapcsolata .....	81
4.9. A client objektum fenntartási technikái .....	82
4.9.1. A client-fenntartási technikák összehasonlítása .....	83
4.9.2. Kliens-oldali technikák .....	88
4.9.2.1. A kliens sütis technika .....	88
4.9.2.2. A kliens URL-kódolósos technika .....	89
4.9.3. Szerver-oldali technikák .....	90
4.9.3.1. Az IP-címes technika .....	91
4.9.3.2. A szerver sütis technika .....	91
4.9.3.3. Szerver URL-kódolás .....	92
4.9.4. A kliens objektum élettartama .....	93
4.9.5. A client objektum felszámolása .....	93
4.10. Objektumok biztonságos megosztása lokkolással .....	93
4.11. Levelezési szolgáltatás .....	94
4.12. Fájrendszer-szolgáltatás .....	96
4.12.1. Biztonsági megfontolások .....	96
4.12.2. Egy File objektum létrehozása .....	97
4.12.3. Egy állomány megnyitása és bezárása .....	97
4.12.4. Állományok lokkolása .....	98
4.12.5. Munka fájlokkal .....	99
4.12.5.1. Pozicionálás fájlokban .....	100
4.12.5.2. Olvasás fájlból .....	101
4.12.5.3. Írás fájlba .....	101
4.12.5.4. Adatkonverzió .....	102
4.12.5.5. Fájlinformációk lekérése .....	103
4.12.6. Külső könyvtárak használata .....	103
4.12.6.1. A könyvtár állományainak azonosítása .....	105
4.12.6.2. Külső függvények regisztrálása .....	105
4.12.6.3. Külső függvények használata .....	106
4.12.7. Kérés- és válasz-manipuláció .....	106
4.12.7.1. A kérés fejléce .....	107
4.12.7.2. A kérés törzse .....	107
4.12.7.3. A válasz fejléce .....	109
<b>5. A LIVEWIRE ADATBÁZIS-SZOLGÁLTATÁS .....</b>	<b>110</b>
5.1. Csatlakozás egy adatbázishoz .....	110
5.1.1. A kapcsolat kiépítése .....	111
5.1.2. Adatbázisok kapcsolat-készletei .....	113

5.1.3. Egyszálú és többszálú adatbázisok.....	115
5.1.4. Kapcsolat-készletek menedzselése.....	116
5.1.5. Egyedi adatbázis-kapcsolatok .....	118
5.1.5.1. Kapcsolat fenntartása kéréseken keresztül.....	119
5.1.5.2. Várakozás egy kapcsolatra .....	119
5.1.5.3. Problémás kapcsolat helyrehozása .....	120
5.2. Adatbázisok használata .....	120
5.2.1. Műveletek egy relációs adatbázisban .....	121
5.2.2. Automatikusan megjelenő lekérdezési eredmények .....	121
5.2.3. Tetszőleges SQL utasítások futtatása .....	122
5.2.4. Lekérdezés-eredmények manipulálása mutatókkal .....	122
5.2.4.1. Mutató létrehozása .....	123
5.2.5. Rekordok megjelenítése .....	124
5.2.6. Kifejezések és aggregáló függvények megjelenítése.....	124
5.2.7. Navigálás mutatókkal .....	124
5.2.8. Munka az oszlopokkal.....	124
5.2.9. Adatbázis-információk cseréje .....	125
5.2.10. Tranzakciók menedzselése.....	125
5.2.10.1. A tranzakció-vezérlési metódusok használata.....	126
5.2.11. Bináris adatok feldolgozása.....	127
5.2.12. Tárolt eljárások hívása.....	129
5.2.12.1. Információk cseréje .....	129
5.2.12.2. Tárolt eljárások használati lépései.....	130
5.2.12.3. A tárolt eljárás regisztrálása .....	130
5.2.12.4. Prototípus definiálása tárolt eljáráshoz.....	131
5.2.12.5. A tárolt eljárás futtatása .....	131
5.2.12.6. Eredményhalmazok .....	132
5.2.12.7. Visszatérési értékek.....	134
5.2.12.8. Kimeneti paraméterek .....	134
5.2.12.9. Informix és Sybase kivételek.....	135
5.3. Adattípus-konverziók.....	135
5.3.1. Az adatbázisok adattípus-konverziói.....	135
5.3.1.1. A DB2 adattípus-konverziók .....	136
5.3.1.2. Az Informix adattípus-konverziók .....	136
5.3.1.3. Az ODBC adattípus-konverziók .....	137
5.3.1.4. Az Oracle adattípus-konverziók .....	137
5.3.1.5. A Sybase adattípus-konverziók.....	138
5.3.2. Munka a dátumokkal és adatbázisokkal.....	138
5.4. A LiveWire hibakezelése .....	139
5.4.1. Hibafeltételek .....	139
5.4.2. Visszatérési értékek.....	139
5.4.2.1. A szám visszatérési érték .....	139
5.4.2.2. Az objektum visszatérési érték.....	141
5.4.2.3. A logikai visszatérési érték .....	141
5.4.2.4. A string visszatérési érték.....	142



5.4.2.5. A void .....	142
5.4.3. Hibametódusok.....	142
5.4.4. Állapotkódok.....	143
<b>6. LIVECONNECT .....</b>	<b>145</b>
6.1. A LiveConnect áttekintése .....	145
6.2. JavaScript-Java kommunikáció.....	146
6.2.1. A csomag objektum (Packages).....	147
6.2.2. Java tömbök felhasználása.....	147
6.2.3. Csomag- és osztály-hivatkozások.....	148
6.2.4. A karakter típus argumentumai .....	148
6.3. Java-JavaScript kommunikáció.....	148
6.3.1. A LiveConnect osztályai .....	149
6.3.2. A szerver-oldali JavaScript elérése .....	149
6.4. Adattípus-konverziók.....	149
6.4.1. JavaScript-Java konverziók .....	150
6.4.1.1. Szám értékek konverziója.....	150
6.4.1.2. Logikai értékek konverziója .....	151
6.4.1.3. String értékek konverziója .....	152
6.4.1.4. Null értékek konverziója .....	152
6.4.1.5. JavaArray és JavaObject objektumok konverziója .....	153
6.4.1.6. JavaClass objektumok konverziója.....	154
6.4.1.7. Egyéb JavaScript objektumok konverziója .....	154
6.4.2. Java-JavaScript konverziók .....	155
<b>7. PÉLDATÁR .....</b>	<b>157</b>
7.1. Levélküldés.....	157
7.1.1. Egyszerű e-mail küldése.....	158
7.1.2. Közvetlen üzenetküldés .....	160
7.1.3. Fájlcsatolás.....	162
7.2. JavaScript a kliens és a szerver között.....	165
7.3. CGI változó elérése .....	167
7.4. Példa hibaelhárításra .....	169
<b>FÜGGELÉK.....</b>	<b>171</b>
A. Az SSJS objektum-modell .....	171
B. Az SSJS objektumok listája.....	172
C. Az SSJS függvények.....	175
D. A JavaScript magjának objektumai.....	176
E. A JavaScript magjának függvényei.....	179
F. Objektum-hierarchia .....	180
<b>TÁRGYMUTATÓ.....</b>	<b>181</b>
<b>WEBES KISSZÓTÁR.....</b>	<b>183</b>



## AJÁNLÁS

Ajánlom ezt a könyvet szüleimnek, kedvesemnek és barátaimnak, akik buzdítottak a könyv írására és rádöbentettek, hogy képes vagyok kitartani a könyvírás és a példaprogramok készítése jegyében töltött nehéz órákban.

Köszönet a szeretetért és a támogatásért.

A könyvben található valamennyi példaprogram (és számos más script is) fellelhető az interneten és ingyenesen letölthető. Az oldalra a kiadó honlapján ([www.bbs.hu](http://www.bbs.hu)) a Szerzők menüben a Sikos László menüpont alatt található linkkel juthat el.

A szerver-oldali JavaScript tanulmányozásához elengedhetetlen a kliens-oldali JavaScript ismerete. Ehhez jó kiindulópont lehet a Szerző JavaScript kliens oldalon című könyve, melynek példaprogramjai szintén letölthetők az oldalról.

Ugyanitt található a webes ajánlásokat kiadó és terjesztő W3C konzorcium dokumentumaiból egy válogatás, melyek a Szerző fordítási munkájának köszönhetően olvashatók magyarul.

Ha bármilyen kérdése vagy észrevétele van a könyv témájával kapcsolatban, az oldalon e-mail-ben felveheti a kapcsolatot a Szerzővel.

## ELŐSZÓ

Akik már behatóbban ismerik a JavaScript nyelv kliens oldalát (és az ehhez szükséges egyéb webes technológiákat), azok között sokan vannak, akik szeretnék a nyelv szerver-oldali képességeit is kitanulni. Ennek több oka is lehet, de az egyik legfontosabb indok a hálózatok nyújtotta lehetőségek szélesebb körben való kihasználhatósága iránti vágy.

Jelen könyv elsősorban azoknak a weblapfejlesztőknek és rendszergazdáknak készült, akik már rendelkeznek a kliens-oldali JavaScript szerkesztésének ismereteivel, egy HTML forráskód megírása nem okoz nekik gondot és szeretnék honlapjaikat alkalmazásokkal „életre kelteni”, rajtuk távoli adatbázisok tartalmát megjeleníteni, esetleg e-mail-t küldeni róluk. Megtudhatjuk azt is, hogyan lehet Java nyelven írt komponenseket vagy osztályokat elérni JavaScriptből, hogyan valósulnak meg a különböző adattípusok közötti konverziók, miként lehet hibákat lekezelni szerver oldalon, a szerver típusa milyen módon befolyásolja a szerver-oldali programozást.

Haszonnal forgathatják a könyvet azok is, akik meg szeretnék ismerni a Netscape Enterprise Server (újabb nevén iPlanet Web Server, Enterprise Edition) néhány lehetőségét, netán a kliens és a szerver oldal különbségeire kíváncsiak, esetleg a két oldal közötti, úgynevezett osztott-oldali JavaScript iránt is érdeklődnek.

Két dolog azonban egyértelműen látszik még ebből a korántsem teljes felsorolásból is. Először is a szerver-oldali JavaScript kódok generálásához némi számítógépes hálózati ismeret és forrásszintű HTML-szerkesztési készség feltétlenül szükséges. Emellett kitűnően ismerni kell a kliens-oldali JavaScript programozását\*. Másrészt a szerver-oldali JavaScript környezetfüggő, tehát nem mellékes, mi-

---

\* Ehhez ajánlható a JavaScript kliens oldalon című könyv

lyen webszerveren dolgozunk. Ebből viszont az következik, hogy a könyv sem tárgyalhat minden szerver-típust külön-külön, így a forrásokat egyes szervereken kissé módosítani kell. Ehhez feltétlenül szükség van a webszerver rendszergazdai kézikönyvére.

A forrásokban fellelhető paraméterezések idézőjelek közti írással, a HTML tag-ek kisbetűs megadásával és a kevert leírási formákkal azt kívánom érzékeltetni, hogy a webes dokumentumok sajnálatos módon nagyon gyengén szabványosított forrásúak. Optimális lenne, ha minél többen tartanánk be a webes ajánlásokat kiadó és terjesztő W3C, ECMA és egyéb konzorciumok, egyesületek ajánlásait és szabványait, mert számos előnyhöz juthatunk ezáltal. Ez a kliens oldalon éppúgy igaz, mint a szerver oldalon.

Ne felejtsük el egyetlen pillanatra sem, hogy szerver oldalon azokra a felhasználókra is gondolni kell, akikkel valamilyen módon megosszuk alkalmazásainkat. Egyes adatokat célszerű nem publikussá tenni, másokhoz viszont feltétlenül biztosítani kell a hozzáférés lehetőségét.

Ha úgy érzi a kedves Olvasó, hogy birtokában van a szükséges ismereteknek és kliens oldal mellett már a szerver oldalt is ki akarja használni kedvenc internetes nyelvén, olvassa végig a könyvet, elemezze a példákat, mélyedjen el a függelékek rálátást biztosító táblázataiban, gyakoroljon bátran, mert a fejlesztést csak így lehet tanulni! Ehhez a korántsem egyszerű, de idővel nagyon hasznossá, sőt akár szórakoztatóvá is váló tevékenységhez kívánok sok kitartást:

A szerző

# 1. HOGYAN FOGJUNK HOZZÁ?

Ha elkezdünk programozni valamilyen nyelven, mindig azt kell tisztáznunk, mi mindenre van szükségünk (előismeret, hardver, szoftver stb.) ahhoz, hogy egyáltalán elkezdhessük a munkát.

Mivel a szerver oldalon sok minden másképpen működik, mint kliens oldalon, nem árt, ha már a kezdet kezdetén legalább azt megértjük, hogy még alkalmazásaink futása is jelentősen eltér a megszokott böngészős script-interpretálástól.

Nem nézünk meg mindent, de a lényegesebb tudnivalókkal feltétlenül meg kell ismerkednünk, mielőtt áttérnénk a nyelvi elemek taglalására.

## 1.1. Amit feltétlenül tudni kell

A JavaScript egy Netscape-re készült, objektum-alapú scriptnyelv. A nyelv magja operátorokat, vezérlési szerkezeteket, utasításokat és alapvető objektumokat (Array, Date, Math) tartalmaz. A nyelv magja számos célra használható, ha kibővítjük azt addicionális objektumokkal:

- A kliens oldali JavaScript a nyelv magját a böngésző és annak Dokumentum Objektum Modellje (DOM) vezérléséhez szükséges objektumokkal egészíti ki.
- A szerver oldali JavaScript a nyelv magját olyan objektumokkal egészíti ki, mely lehetővé teszi a script szerveren történő futtatását. Így például lehetővé válik, hogy egy relációs adatbázissal kommunikáljunk a hálózaton keresztül vagy egy szerveren fájl-manipuláló műveleteket végezzünk.

A JavaScripttel tehát az interneten futó alkalmazásokat fejleszthetünk. Amíg azonban a kliens alkalmazások egy böngészőben (pl. Internet Explorer) futnak, addig a szerver alkalmazások egy szerveren (pl. Netscape Enterprise Server). A JavaScript felhasználásával olyan dinamikus oldalakat készíthetünk, melyek feldolgozzák a felhasználó által bevitt adatokat.

A JavaScript LiveConnect alkalmazásával a Java és JavaScript kódok képesek egymással kommunikálni. JavaScriptből példányosíthatunk Java objektumokat, elérhetünk egyes metódusokat, mezőket. A másik oldalról, a Java-ból JavaScript objektumokat, tulajdonságokat, metódusokat érhetünk el.

A szerver-oldali JavaScript a JavaScript nyelv magját ugyanúgy tartalmazza, mint a kliens-oldali JavaScript, melyet a szerver oldal tanulmányozásához feltétlenül ismernünk kell. Egy JavaScript szerveren történő futtatásával járó feladatok meglehetősen különböznek a JavaScript kliens-oldalon történő futtatásától. Más környezet, melyben fejlesztünk és melyben az alkalmazások futnak, mások a feladatok és a hívott függvények. Ezt nézzük meg részletesebben a későbbi fejezetekben.

## 1.2. A szerver-oldali JavaScript lehetőségei

A kliens környezet (böngésző) a weblapok mellett többek között JavaScript alkalmazások megjelenítésére, használatára is szolgálhat. A megtekintett weblapok listájában visszaugorhatunk és előre lapozhatunk a korábban megtekintett oldalak között. Ebben a környezetben a használt objektumok a lapok, az ablakok és a history-k.

Ezzel szemben a szerver környezetben a szerver erőforrásaival dolgozunk. Kapcsolódhatunk relációs adatbázisokhoz, információkat oszthatunk meg egy alkalmazás felhasználói között vagy dolgozhatunk a szerver fájlrendszerén. Ebben a környezetben tehát olyan objektumok vannak, melyek lehetővé teszik a fenti műveletek elvégzését.

Egyébiránt a HTML oldalakat nem a szerveren jelenítjük meg. A szerver továbbítja a fájlt a kliensnek, hogy azt az meg tudja jeleníteni. Az elküldött dokumentum tartalmazhat kliens-oldali JavaScriptet is. Ha a kért oldal egy JavaScript alkalmazás része, a szerver közvetlenül legenerálja az oldalt.



A JavaScript alkalmazások fejlesztésekor tartsuk szem előtt a kliens és a szerver platform közötti különbségeket:

<b>Kliens</b>	<b>Szerver</b>
<p>A kliensek gyakran (bár nem mindig) egyszerű asztali rendszerek. A kliensek általában egyfelhasználós gépek. Az adatok kliensen történő előfeldolgozása a sávszélességi igényeket csökkentheti, ha a kliens alkalmazás képes az adatok összesítésére.</p>	<p>A szerverek általában (de nem mindig) nagy teljesítményű munkaállomások, gyors processzorokkal és nagy tárolási kapacitásokkal. A szerverek túlterheltté válhatnak, ha pár ezer (vagy több) kliens próbálja elérni őket.</p>

Általában többféle lehetőségünk is van az alkalmazás kliens és szerver közötti felosztására. Egyes feladatok csak a kliensen vagy csak a szerveren hajthatók végre, de vannak olyanok is, melyek mindkettőn. Nem mindig egyértelmű, mely feladatot mely oldalon végezzünk el, de létezik néhány általános irányelv:

- A kliens-oldali feldolgozást (a `SCRIPT` tag-et) az alábbi esetekben használjuk:
  - Felhasználói bemenet kiértékelése: az űrlapon megadott adatok érvényesek-e
  - A felhasználotól megerősítést kérni, hibákat vagy információkat megjeleníteni
  - Összesítő számításokat (összeadások, átlagszámítások stb.) végezni vagy más, a szervertől kapott adatot feldolgozni.
  - Feltételekkel kiegészíteni a HTML forrást
  - Függvényeket végrehajtani, melyek nem igényelnek adatot a szervertől
- A szerver-oldali feldolgozást (a `SERVER` tag-et) a következő feladatok elvégzésére használjuk:
  - Kliens-hozzáférések felügyelete
  - Több kliens vagy alkalmazás között megosztott adatok fenntartása
  - A szerveren levő adatbázis vagy fájlok elérése
  - Külső könyvtárak meghívása a szerveren
  - Java alkalmazások dinamikus testreszabása (pl. adatmegjelenítés Java alkalmazással)

A JavaScript Session Management Service szolgáltatása olyan objektumokat kínál, melyek időtálló információkat tárolnak, a kliens-oldali JavaScript sokkal kevésbé "tartós". A kliens-oldali objektumok csak addig léteznek, amíg a felhasználó eléri az oldalt. A szerverek számos kienstől és alkalmazástól gyűjtenek össze adatokat és tárolnak hatalmas adatbázisokban.

### **1.3. A szerver jelentősége**

A szerver-oldali JavaScript tanulmányozásához egy webszerver közelébe kell férkőznünk. Számos webszerver-típus létezik, a különféle gyártók termékei általában nemcsak a hardverben, hanem a szoftverben is különböznek. Így egyáltalán nem mindegy, milyen szerveren dolgozunk.

Az egyes szerver-szoftverek különféle komponenseket és szolgáltatásokat tartalmaznak, melyek merőben különböznek az egyes platformokon. Ezért optimális esetben mindig kéznél van nálunk a szerverünk rendszergazdai kézikönyve.

## 2. A JAVASCRIPT NYELV ÁTTEKINTÉSE

Mivel a JavaScript nem egyszerűen csak egy nyelv, hanem többféle felhasználása is lehetséges, tisztáznunk kell, mi is az a JavaScript mag, melyek azok a nyelvi elemek, melyek csak a kliens oldalon és melyek azok, melyek csak a szerver oldalon alkalmazhatók.

A JavaScript nyelvnek több verziója is létezik, ezek mindegyikével találkozhatunk az interneten nap mint nap. Ezért egy tönténeti áttekintést is megnézünk a nyelvről.

Sokan vannak, akik egy tévhitel küszködnek: nem tudják, mi a Java és a JavaScript közötti különbség. Valakik felváltva hol Javáról, hol JavaScriptről beszélnek. Mások szerint a két nyelvben csak az első négy betű hasonló. Ez persze nem igaz, hiszen ha csak azt nézzük, hogy mindkét nyelv C alapú, máris van némi hasonlóság. Ha viszont azt is hozzátesszük, hogy egyes Java osztályokat vagy komponenseket felhasználhatunk JavaScriptben is, máris egyértelművé válik, hogy a két nyelv között kissé szorosabb a kapcsolat. Vagyis a JavaScript a Java kistestvére...

### 2.1. A JavaScript nyelv magja, a kliens és a szerver oldali JavaScript

A kliens és a szerver oldali JavaScript alábbi elemei közösek:

- a kulcsszavak
- az utasítások szintaxisa, a JavaScript nyelvtan
- a kifejezésekre, változókra és literálokra vonatkozó szabályok
- az objektum-modell alapjai (bár a kliens és a szerver oldali JavaScript előre definiált objektum-halmazai különböznek)
- olyan előre definiált objektumok és függvények, mint az Array, a Date vagy a Math