Sikos László:

Szerver oldali webprogramozás

Sikos László:

Szerver oldali webprogramozás

BBS-INFO Kiadó – 2004.

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

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

AJÁNLÁS

Ajánlom ezt a könyvet szüleimnek, kedvesemnek és barátaimnak, akik buzdítottak a könyv írására és rádöbbentettek, 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) 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 írásával, 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 addícionális 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:

Kliens	**Szerver**
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.	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.

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 felhasználótól 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 klienstől 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 történeti áttekintést is megnézünk a nyelvről.

Sokan vannak, akik egy tévhittel küszködnek: nem tudják, mi a Java és a JavaScript közötti különbség. Valakik felváltva hol Javaról, 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