Sikos László:

JavaScript kliens oldalon

Sikos László:

JavaScript
kliens oldalon

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 3 7
E-book ISBN 9786156364142

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 hosszú esti órákon, melyek a könyvírás és a példaprogramok készítése jegyében teltek. 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. Ugyancsak itt felveheti a kapcsolatot a szerzővel e-mail-en, ha bármilyen kérdése merülne fel.

ELŐSZÓ

Köztudott, hogy napjainkban a számítástechnika térhódítása jellemző. Ha azonban megvizsgáljuk, hogy mi az, ami ezen belül az egyik legintenzívebben fejlődő terület, azt tapasztaljuk, hogy a számítógépes hálózatok a mozgatórugója a számítógépek mindennapos és univerzális felhasználásának.

Egy távoli gép elérése olyan alternatívákat rejt, mely néhány éve még elképzelhetetlen volt. A „számítógépes világháló"-nak titulált internet (a „hálózatok hálózata") az az informatikai eszköz, mely számos médiával vetekedhet: friss információkra vágyva felkereshetjük az egyik internetes újságot; elektronikus levelet küldhetünk (különféle mellékletekkel tarkítva) dél-amerikai barátunknak, akinek válasza akár pár perc múlva meg is érkezhet hozzánk; elutazás előtt megnézhetjük, milyen időjárás van Rómában; érdeklődve tanulmányozhatjuk a japán kalligráfiát; sakkozhatunk egy Ausztráliában tartózkodó személlyel; konferenciát bonyolíthatunk le a világ különböző részein levő üzletfelekkel; letölthetjük egy program legújabb verzióját; beiratkozhatunk egy online nyelvtanfolyamra és még sorolhatnánk. Természetesen nem minden szolgáltatás vehető igénybe pusztán attól a ténytől, hogy számunkra elérhető az internet. Vannak olyan webhelyek, ahol regisztrálni kell magunkat, vannak fizetős szolgáltatások, a programoknak sokszor csak a próbaverziója tölthető le stb. Technikai problémák is adódnak: ezek közül a legjellemzőbb ma Magyarország sok részén a rendszer sebessége. A szépséghibáktól eltekintve az internet egy nagyon hasznos dolog, mai információs társadalmunkban jó segítőtársunk lehet, ha tudjuk, hogyan használjuk.

Jelen könyv elsősorban azoknak a weblapfejlesztőknek készült, akik már rendelkeznek a HTML forrásszintű szerkesztésének ismereteivel (nem pusztán szerkesztőprogramokra alapoznak), és szeretnék honlapjaikat „életre kelteni", dinamikus tartalmat csempészni bele, a böngésző ablakának minden részét uralni akarják, speciális effektusokkal kívánnak látogatókat vonzani oldalaikra.

Haszonnal forgathatják a könyvet azok is, akiknek néhány konkrét igényük volt a honlapjukkal kapcsolatban, de nem tudták megvalósítani elképzeléseiket, esetleg még a szükséges eszközben (script) sem voltak biztosak. Ilyen lehet például az az igény, hogy ketyegjen egy óra a weblapomon, jelezze ki a dátumot, netán írja ki a szilveszterhez közeledvén

az évből hátralevő napokat vagy a mai névnapokat, a betűk „írógép-effektussal" potyogjanak, időzítve nyíljon meg egy új böngészőablak, stb. Ez a JavaScript lehetőségeinek pusztán felvillantása volt, de a szorgalmas Olvasó a könyv végére érve sok-sok hasonló „apróságot" tud majd becserkészni saját webes dokumentumaiba. Két dolog azonban egyértelműen látszik még ebből a – korántsem teljes – felsorolásból is. Először is a JavaScript kódok generálásához némi számítógépes hálózati és forrásszintű HTML-szerkesztési ismeret feltétlenül szükséges. A témáról számos könyv jelent meg magyarul is, ezek áttekintése erősen ajánlott a leendő JavaScript-fejlesztőknek. (A profi weblapfejlesztéshez szükséges egyéb témák áttekintésével az 1. fejezet foglalkozik részletesen.) Másrészt a könyv nem foglalkozik a kiszolgáló oldali (server side) scriptekkel. Ez a téma olyan sokrétű, hogy önmagában is megér egy könyvet. Tömérdek igényünk közül jónéhány vígan kielégíthető a kliens oldali JavaScripttel is. Egyébként pedig először mindenkinek célszerűbb a kliens oldallal foglalkozni, s ha ebben a témában már elég tájékozott, akkor érdemes csak áttérnie a szerver oldal tanulmányozására.

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, de a böngészők elég jól kezelik ezeket a „lazaságokat". A másik ok pedig az új technológiák, a HTML helyett sokkal inkább az XHTML felé történő közeledésem, melyet mindenkinek erősen ajánlok.

Ha úgy érzi a kedves Olvasó, hogy birtokában van a szükséges ismereteknek és be kíván lépni a JavaScriptesek táborába, olvassa végig a könyvet, elemezze a példákat, 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. AMIT A JAVASCRIPTTEL VALÓ MEGISMERKEDÉSHEZ FELTÉTLENÜL TUDNI KELL

Mint az Előszóban már említettük, ahhoz, hogy nekiláthassunk a JavaScript tanulásához, szükséges némi előismeret. Nézzük sorra ezeket!

Az operációs rendszer ismerete. Számtalan egyszerűbb operációs rendszer-szintű fogalomra lehet szükség scriptek írásakor. Egyes lehetőségek kiaknázása pedig az operációs rendszer-szintű műveleteket is érinti (egy külső fájl megnyitása, tallózás a mappák között stb.), így az operációs rendszert is ismerni kell legalább alapszinten.

A böngésző program ismerete. A webes dokumentumok megjelenítéséhez használt böngészőprogramok tudása típustól, verziószámtól, a telepített beépülő moduloktól (plug-in) stb. függően sokféle lehet. Emellett számtalan beállítási lehetőség is kínálkozik, melyek egy része meghatározhatja egyes scriptek működését.

Weblapszerkesztési ismeretek. A weblapok általános felépítése alapkövetelmény a JavaScript fejlesztő számára. Nem elegendő azonban csak a szerkesztőprogramok ismerete, de persze az sem hátrány!
A kódok egy része értelmes angol (szak)szavakból vagy azok rövidítéséből áll. Ezért erősen ajánlott az angol nyelv és szakzsargon legalább alapszintű ismerete.

Forrásszintű HTML kód-generálás. A legfontosabb dolog, a JavaScript nyelv elsajátításának alapfeltétele. Mivel a scriptek java részét beleimplementáljuk a HTML forrásba, nem árt, ha tudjuk, melyik rész hova tartozik és mi a feladata (ha egy HTML forrást megnyitva annak minden karaktere (!) érthető számunkra, ez nem okoz problémát).

Számítógépes grafikai ismeretek nélkül ne is álljunk neki a weblapfejlesztésnek. Egy digitális fénykép feljavítása, montázs készítése, egyszerű bit- és vektorgrafikus gépek elkészítése nem szabad, hogy gondot okozzon. Ha egyéb webes ismeretek birtokában van az Olvasó (pl. Flash), az további előnyt jelenthet, hiszen így a hatások fokozhatók, ha az eszközöket kombináljuk, illetve felváltva alkalmazzuk.

Alapvető hálózati ismeretek. Még a kliens oldalon is szükséges tisztában lennünk a számítógépes hálózatok működésének alapjaival és ennek kapcsán néhány fogalommal. Ha csak arra gondolunk, hogy a JavaScript implementációi hol futnak, egyértelmű, miért kell értetünk ehhez is. A böngészők alatt ugyan futtathatunk scripteket ún. kapcsolat nélküli módban is (ha nincs internet-elérésünk), de ekkor bizonyos dolgokról le kell mondanunk.

Programozási ismeretek. Már az egyszerűbb scripteknél is szükség lehet változók deklarálására, egy-egy ciklusra, elágaztatásra, tömbök kezelésére stb. Szükséges tehát egy kis programozási előismeret is. Mivel a JavaScript a C-hez hasonló szintaktikát használ, nem árt, ha legalább kezdő szinten ismerjük a C nyelvet.
Mivel a JavaScript objektumalapú nyelv, feltétlenül szükségeltetik az objektum mint fogalom pontos ismerete.

1.1. Mit értünk webes programozáson?

Ha szó esik a webes programozásról, akkor ahány ember, annyiféle dolog jut az eszébe. Van, aki egy Netscape Navigator-hoz hasonló böngésző megírására, van, aki Excite-féle keresőprogramok készítésére asszociál, de olyan is akad, aki web-oldalak HTML nyelven történő tervezésére gondol. Annyi arca van a WWW-nek, hogy igen sokféle szempontból vizsgálhatjuk mind felhasználóként, mind fejlesztőként.
Ha már rendelkezünk némi HTML-ismerettel, már el tudjuk magyarázni a böngészőnek, hogyan jelenítse meg webes dokumentumainkat. A HTML azonban egy egyszerű szöveges leírónyelv, mely képtelen válaszolni a weblapot megtekintő érdeklődőnek, nem tud döntéseket hozni és automatikusan végrehajtani előre meghatározott műveleteket. Az ilyen jellegű feladatok megoldására bonyolultabb nyelvre (ún. parancsnyelvre) van szükség (ilyen a JavaScript is).
Az összetett programozási nyelveknél a parancsnyelvek lényegesen egyszerűbbek mind szintaktikájukat, mind pedig parancsaikat tekintve. Kézenfekvő lehetőség tehát weboldalaink „életre keltése" scriptekkel. Az érdeklődésünk középpontjában levő JavaScript parancsnyelv egyesíti a HTML leíró nyelv és a bonyolult programozási nyelvek előnyeit.
A webes programozás tehát nem más, mint a csillogó-villogó vagy éppen szerényen egyszerű webes felület mögött rejtőző szöveges sorok, pontosabban a parancsnyelvi kódok generálása.
Ha a teljes weblapot a semmiből hozzuk létre (vagy akár mások ötleteiből is merítünk) és mi tervezzük meg, valamint minden, a weblap megjelenéséhez szükséges egyéb fájlt, objektumot stb. (pl. animált GIF-et, digitális fényképeket, digitális adatbázisokat) is létrehozunk, akkor már általánosabb fogalmakat szokás használni, úgy mint weblap-tervezés (webdesign), weblap-fejlesztés vagy weblap-készítés. E fogalmak között azonban nincs éles határ, sok dolog akár a fent felsorolt fogalmak mindegyikével lefedhető.

1.2. Mikor alkalmazható a webes felület?

Egy ilyen dinamikus és igen sokrétű tartalommal rendelkező médium, mint a WEB esetében azt hiszem, egyértelmű a felhasználás egyik fő előnye: a hálózati alkalmazások készítésével nem kell pusztán egy-egy különálló számítógép adataira hagyatkoznunk. Nemcsak a gépek közötti fájlátvitel oldható meg a webes programozással, de többek között olyan feladatok is, mint dinamikus tartalmú dokumentumok szolgáltatása és letöltése, számítógépek távoli elérése, interaktív kapcsolat számítógépek között stb.

Napjainkban a programozók rendelkezésére állnak olyan programozási nyelvek, mint a Perl, amely adatszűrőként, biztonsági átjáróként, adatbázis-előtétként, script fájlok programozásához is használható, vagy például a Java, amellyel egyetlen végrehajtható állományt készítve ugyanazt használhatjuk Windows, UNIX vagy Macintosh környezetben is. A hálózati alkalmazások használatával jelentősen kibővül a számítógép felhasználhatóságának köre.
A webes programok két helyen futhatnak: a kliens- és a szerveroldalon. Jelen könyv csak a kliens oldali résszel foglalkozik. A szerveren futtatható programok készítése és működése jelentős eltérést mutat. Egy kliensoldali és egy szerveroldali JavaScript szintaxisa ugyan megegyezik, de a programozói interfész teljesen más. A kliens oldali scriptek gyakorlatilag az összes, a JavaScriptet támogató böngésző (napjainkban ez már alapkövetelmény a böngészőkkel szemben) alatt felhasználhatók, tehát akár webes felületű dokumentumok tízezreit is vidáman kezelhetjük velük saját gépünk merevlemezén.

1.3. Milyen lehetőségeink vannak a JavaScript felhasználásával?

Azt megfogalmazni, hogy pontosan milyen lehetőségeink vannak a JavaScript felhasználásával, reménytelen vállalkozás. Lehet azonban néhány klasszikus példát kiemelni, mint ahogy azt most meg is tesszük.
Aki programozott már valamilyen programozási nyelven, bizonyára hiányolta a HTML-ből a különböző események lekezelésének, a böngészőablak egyes részei objektumként való elérésének és néhány egyszerű eszköznek (pl. modálablak) igazi támogatottságát. A JavaScript jóvoltából a webes felületeken az ehhez hasonló kívánságainkról sem kell lemondanunk. Ha például egy fotógalériát teszünk közzé az interneten, az oldal betöltésekor feldobhatunk a felhasználónak egy modálablakot, melyben tájékoztatjuk a lehetőségről, hogy egy képre kattintva megtekinthető annak nagyobb felbontású változata is, csupán a gyorsabb letöltés érdekében láthatók kis méretű képek az oldalon.
Készíthetünk olyan „aktív grafikákat" (mozgóképeket), melyek a felhasználó egérmozgatására reagálnak. Mikor az egérkurzor a kép fölé kerül, azt egy másik képpé változik. Gondolunk bele, milyen jó kis trükköt lehet így csinálni, ha a grafika egy szöveget ábrázol! Egy menüben jól láthatóvá válik az egérkurzor alatt levő menüpont, hiszen megváltozik annak színe, mérete, esetleg dőlt betűssé válik. Számítógépes diabemutatóknál, webes felületű prezentációknál érdekes effektusokat érhetünk el ezzel a lehetőséggel.

Pusztán HTML-t használva nem volt kihasználva a böngészőnk állapotsora? JavaScripttel ezen is változtathatunk. Például a weblap bármely linkje fölött állva az állapotsorban a hivatkozásnak megfelelő szöveg jelenhet meg, de akár egy fényreklámot (gördülő szöveget) is elhelyezhetünk itt.
Bár HTML-ben készíthetünk űrlapokat, azok kezelése hagy némi kívánnivalót maga után. A „hiányokat" a JavaScript pótolja, mert képes az űrlap tartalmának ellenőrzésére, az azon levő adatokkal műveletek végzésére.
Hirdetéseink „életre kelhetnek" scriptek alkalmazásával, mert aktívak lesznek azáltal, hogy a felhasználó tevékenységére reagálnak.
Kódból ellenőrizhetjük az oldal megtekintéséhez alkalmazott böngésző típusát (Explorer, Netscape, Opera, Neoplanet stb.) és a böngésző típusának megfelelő tartalmú kód hajtódik végre (bonyolultabb esetekben célszerű külső js fájlokban elhelyezni a különböző böngészőkhöz készített scripteket).
Lekérdezhetjük a telepített bővítményeket, beépülő modulokat (plug-in), melyek esetleges hiánya esetén jelezhetjük a felhasználónak, hogy mire lenne szükség az oldal megtekintéséhez.

Megjegyzés: a fenti lehetőségek közül jónéhányat (nem mindet!) egészen más eszközzel is meg lehet oldani (pusztán a HTML-lel nem), mint ahogy azzal az interneten találkozhatunk is nap mint nap.
A JavaScript nyújtotta lehetőségeink széles tárháza ezzel természetesen nem merült ki. A könyv példáit végigelemezve számos egyéb lehetőségünk is kínálkozik majd, melyeknek igazán már csak egyetlen dolog szab határt: a fantáziánk…

1.4. Hogyan fogjunk hozzá a JavaScriptes fejlesztéshez?

Ahhoz, hogy JavaScriptben tudjunk programozni, szükségünk van egy a kódot fogadó felületre. Ez a leggyakrabban a webes felület. Ez azt jelenti, hogy egy-egy script írását a legtöbb esetben egy HTML dokumentum készítése előzi meg. Ehhez a legtöbb esetben akár egy Jegyzettömbhöz hasonló egyszerű szövegszerkesztő programot használunk (forrásszintű szerkesztés esetén). Ugyanez a program alkalmas a scriptek írására is, hiszen a HTML és a script sokszor nemhogy nem különül el élesen, de egymást alátámasztva, egymást kiegészítve érnek el olyan hatásokat, melyeket pusztán HTML-lel lehetetlen megvalósítani (pl. az eseménykezelőknél majd látni fogjuk, hogy a két forrás teljesen egybeolvad).
A kód futtatása, tesztelése igen egyszerű, azt ugyanis a webböngésző futtatja interpreter jelleggel.

Ahhoz, hogy egy-egy kód a megfelelő, illetve a kívánatos hatást keltse, fontos, hogy a megfelelő helyre legyen implementálva a HTML forrásba (lásd az erről szóló fejezetet). Nem mellékes, hogy a HTML dokumentum fej vagy törzs részébe írjuk a kódot. Nem mindegy, hogy mikor és hogyan hívunk meg egy-egy függvényt.
Összefoglalva tehát elmondható, hogy az egyes nyelvi elemek gyakorlati próbálgatásaihoz, illetve a könyv konkrét példáinak kipróbálásához létre kell hoznunk egy egyszerű szövegfájlt (pl. txt kiterjesztéssel), majd ebbe kell beírnunk a legegyszerűbb HTML dokumentumot (<html> és </html>). Ezt elmentjük, a kiterjesztést átírjuk htm-re (vagy html-re), majd megnyitjuk a fájlt egy böngészőben (ez sokszor csak egy dupla kattintás a fájl ikonján). Ettől kezdve már van egy egyszerű HTML dokumentumunk, melynek forrása innen, a böngészőből is szerkeszthető (Explorerben Nézet menü Forrás menüpont vagy jobb egérgombbal előhívjuk a helyi menüt és onnan kiválasztjuk a Forrás megtekintése menüpontot).
Célszerű a forrás megnyitása mellett meghagyni a HTML dokumentum böngészőbeli ablakát is, így minden előkészületet megtettünk a forrásszintű szerkesztéshez. Ettől kezdve már csak annyi a teendőnk, hogy a forrást igényeink szerint módosítjuk, majd mentjük a fájlt. Átlépünk a böngésző ablakába, majd frissítünk és máris megtekinthetjük az eredményet (Explorert használva a frissítés nemcsak a Frissítés gombba érhető el, célszerű megjegyeznünk, hogy az F5 gomb is ezt a célt szolgálja).

(Megjegyzés: miután a fent leírtak a HTML forrásszintű szerkesztésére is elmondhatók, tulajdonképpen az Olvasónak ezen ismeretekkel már rendelkeznie kell. A dologban kevésbé jártas Olvasók kedvéért mégis szántunk rá pár sort, hogy tisztán láthassák a szerkesztést azok is, akiknek ez korábban még nem volt világos. Aki azonban a HTML forrászintű szerkesztésével bajban van, annak célszerű azt jobban megismerni[*], s csak azután nekiállni a JavaScript tanulásának.)

[*] Ehhez ajánlható a Weblapkészítés házilag c. könyv.

2. A JAVASCRIPT NYELV ÁTTEKINTÉSE

A JavaScript kliens oldali alkalmazásának egyik vitathatatlan erőssége a honlap dinamikus tartalommal és érdekesnél érdekesebb hatásokkal való kiegészítése. A fejezet bemutatja a JavaScriptet is magába foglaló scriptnyelv-család általános jellemzőit, a JavaScript nyelvi változatait és azok böngésző-támogatottságát, a nyelv alternatív elnevezéseit. Ezen bevezető után bemutatásra kerül a nyelv összes fontos kliens oldali része, az egyes nyelvi elemek és azok felhasználása, sok-sok példával tarkítva. A példasorok látszólag erőteljesen megelőzik a JavaScript nyelv-tanulásunk során logikailag következő anyagrészeket. Ez azonban csak látszólagos ellentmondás a könyv céljával, hiszen az anyagok lépcsőzetesen egymás fölött, de párhuzamosan egymás mellett is épülnek egymásra. Így lehetetlen olyan formában megismerkedni a JavaScripttel, hogy pontosan meghatározott elemeket tanulunk egymás után. Az egyéni haladások is jelentősen eltérnek egymástól. Az első kódsorok mintegy olvasmányosan belevezetnek minket a nyelvbe, szinte észrevétlenül ismerkedhetünk meg a JavaScripttel. A válogatott példák arra hivatottak, hogy megkönnyítsék előrehaladásunkat és mentessé tegyék azt az esetleges túl szárazzá válástól. Ebben különbözik ez a fejezet a JavaScript-referenciaanyagoktól, melyek a nyelv legapróbb elemeit is leírják, példák is lehetnek bennük, de szinte olvashatatlanok. Ennek oka, hogy más a céljuk, nem a nyelv megtanítása, hanem sokkal inkább arra szolgálnak, hogy ha fejlesztés közben elakad a programozó, akkor legyen hova nyúlnia, megnézhessen egy-egy szintaktikát, paraméterlistát stb. A fent leírtakból következik, hogy a könyv legnehezebben emészthető fejezetével van dolgunk, de ha ezen átrágjuk magunkat, akkor már nem lesz gond a nyelv gyakorlati, fejlesztésen alapuló elsajátításával konkrét és természetesen egyre nehezedő példák segítségével.