Kopacz Botond

Internet Explorer 9 a zsebben

Készült a Devportal.hu közösség támogatásával

Kopacz Botond

Internet Explorer 9 a zsebben

Jedlik Oktatási Stúdió
Budapest, 2011

A szerzők a könyv írása során törekedtek arra, hogy a leírt tartalom a lehető legpontosabb és naprakész legyen. Ennek ellenére előfordulhatnak hibák, vagy bizonyos információk elavulttá válhattak.

A könyvben leírt programkódokat mindenki saját felelősségére alkalmazhatja. Javasoljuk, hogy ezeket ne éles környezetben próbálják ki. A felhasználásból eredő esetleges károkért sem a szerzők, sem a kiadó nem vonható felelősségre.

Az oldalakon előforduló márka- valamint kereskedelmi védjegyek bejegyzőjük tulajdonában állnak.

A könyv vagy annak bármely része, valamint a benne szereplő példák a szerzőkkel kötött megállapodás nélkül nem használhatók fel üzleti célú oktatási tevékenység során! A könyv tudásanyaga államilag finanszírozott közép- és felsőoktatásban, illetve szakmai közösségek oktatásában bármely célra felhasználható.

© Kopacz Botond

Borító: Varga Tamás

Anyanyelvi lektor: Venczel Katalin

Kiadó: Jedlik Oktatási Stúdió Kft.
1212 Budapest, Táncsics M. u. 92.
Internet: http://www.jos.hu
E-mail: jos@jos.hu
Felelős kiadó: a Jedlik Oktatási Stúdió Kft. ügyvezetője

Nyomta: LAGrade Kft.
Felelős vezető: Szutter Lénárd

ISBN: **978-615…**
Raktári szám: JO-0335

Tartalom

Előszó 3

1.1. Teljesítmény 5

Chakra, az új JavaScript motor 5

Hardveresen gyorsított grafikus feldolgozás 5

Még több optimalizálás 7

Bővítményteljesítmény-tanácsadó 9

1.2. Megjelenés és használat 10

Webhely központú főképernyő 10

Új lap megnyitása 18

Értesítési sáv 21

Letöltéskezelő 22

Bővítménykezelő 24

1.3. Windows 7 együttműködés 26

Rögzített webhelyek 27

Ugrólisták 28

Windows 7 Snap funkció 28

1.4. Biztonsági funkciók 29

SmartScreen-szűrő 30

ActiveX-szűrő 33

Követésvédelem 33

Szűrő helyközi, parancsfájlt alkalmazó támadások ellen 35

InPrivate-böngészés 36

Lapok elszigetelése és helyreállítása 37

2. fejezet: Az Internet Explorer 9 rendszerüzemeltetői szemmel 39

2.1. Kilenc ok, amiért az IE9 a legjobb böngésző üzleti felhasználók számára 39

Teljesítményközpontú kialakítás 39

Hardveres gyorsítás 40

Webhelyközpontú kialakítás ... 40
A Windows szerves része ... 40
Biztonságosabb böngészés ... 40
Könnyen bevezethető ... 41
Átfogó felügyeleti eszköz ... 41
Kompatibilitás és az áttérés támogatása ... 41
Modern webes szabványok támogatása ... 41
2.2. Internet Explorer Administration Kit (IEAK) ... 42
2.3. Alkalmazásunk ellátása digitális aláírással ... 43
2.4. Egyedi keresési szolgáltató létrehozása ... 45
2.5. Egyedi követésvédelmi lista készítése ... 47
3. fejezet: Az Internet Explorer 9 fejlesztői szemmel ... 50
3.1. Támogatott technológiák ... 50
HTML5 támogatás ... 51
CSS3 támogatás ... 51
ECMAScript 5 támogatás ... 51
SVG támogatás ... 51
Geolocation támogatás ... 51
WebM támogatás ... 52
3.2. Beépített fejlesztő eszközök ... 52
3.3. HTML5 a gyakorlatban ... 53
Áttekintés ... 53
Előkészítés ... 54
HTML5 szemantikus elemek ... 54
HTML5 multimédiás vezérlő elemek ... 56
A HTML5 vászna ... 58
SVG támogatás ... 69
CSS3 támogatás ... 70
ECMAScript 5 újdonságok ... 81
Rögzíthető webhelyek ... 83
WebStorage modul ... 85

Előszó

Közel negyven éve, a számítástechnika hajnalán indult el az az ARPANET-nek nevezett program, ami különböző számítógépek közötti adatátvitelt tett lehetővé. A nyolcvanas évek elején létrejött az a hálózat, ami a mai ismert internet alapját képezte. Kezdetben védelmi, majd kutatói és oktatási célokra használták ezt a hálózatot, azonban mára egy olyan komplex információs hálózattá nőtte ki magát, amely mindenki számára elérhető és számítógépek milliárdjait kapcsolja. Az internet szerves, és elengedhetetlen része lett mindennapjainknak, legyen szó akár szórakozásról, szabadidőtöltésről, információ-szerzésről vagy munkáról.

Mindennapi teendőink során számtalan olyan tevékenységet végzünk, ahol kihasználjuk az internet nyújtotta lehetőséget. Például amikor bankkártyával fizetünk, vagy digitális televíziót nézünk, akkor is az internetet használjunk. Ámde a legtöbbünk az internet fogalmát a világhálón való szörfözéssel, barangolással azonosítja. Ennek a szörfözésnek pedig nélkülözhetetlen kelléke egy megbízható böngésző továbbá tengernyi megjelenítendő weblap.

A jegyzet azzal a célkitűzéssel készült, hogy több szemszögből mutassa be a Microsoft Internet Explorer család legújabb generációs tagját, az Internet Explorer 9-et és azokat a modern webes technológiákat, amelyeknek az ismerete elengedhetetlen a 21. századi weblapok készítéséhez. A jegyzet első nagy fejezete felhasználói szemmel mutatja be az új böngésző működését, funkcióit és szolgáltatásait. A második nagy fejezet rávilágít arra, hogy miért ideális választás az Internet Explorer 9 a vállalatok számára, legyen szó akár kis-, közép-, vagy nagyvállalatról. A jegyzet harmadik fejezete pedig azokba a modern webes technológiákba nyújt betekintést, amelyek webfejlesztőként szükségszerűek a szabványos, modern weblapok készítéséhez és az Internet Explorer 9-ben rejlő lehetőségek kihasználásához.

Mind a jegyzet, mind pedig a hozzátartozó programozási mintapéldák a weben is elérhetők és letölthetőek a **Devportal.hu** oldaláról.

Kopacz Botond

Pécs, 2011. április

1. fejezet: Az Internet Explorer 9 felhasználói szemmel

Az első nagy fejezet célja, hogy felhasználói szemmel mutassa be az Internet Explorer legfrissebb változatát, az Internet Explorer 9-et. Az első fejezet betekintést nyújt azokba az új-, vagy továbbfejlesztett funkciókba, amelyeknek köszönhetően az Internet Explorer nagyszerű választás azoknak a számítógép felhasználóknak, akik szeretnének egy gyors, egyszerűen kezelhető és biztonságos böngészővel internetezni. Részletesen bemutatja azt az újjászületett felhasználói felületet, amely minden szempontból a böngészés élményét kívánja fokozni, valamint azokat a biztonsági funkciókat is, amelyek révén kényelmesen, hátradőlve böngészhetünk anélkül, hogy az egyre veszélyesebbé váló internet világában valaki kárt okozna nekünk.

Az internet térnyerésnek hajnalán az internet mint fogalom és technológia a nagyközönség szemében csupán egy kuriózumnak számított. Az akkor korszerűnek tekintett, ámde valójában kezdetleges távközlési technológia csupán egy szűk réteg számára biztosította az internetezés lehetőségét. Az idő múlásával és a technológia fejlődésével folyamatosan szélesedett azoknak a felhasználóknak a köre, akik rendelkeztek internet hozzáféréssel. Ennek köszönhetően a webes világ is interaktívabb lett, és az újdonság szerepét átvette a funkcionalitás. A felhasználók többsége már olyan konkrét feladatok elvégzésére használta az internetet, mint például az egymással való kapcsolattartás, vagy a személyes és vállalati kommunikáció. Azonban manapság az internet fogalma ennél sokkal többet jelent. Az interneten élünk közösségi életet, az internet segítségével szórakoztatjuk magunkat videó megosztó portálokon vagy online játékokkal, és az internetet használjuk mindennapi életünkhöz szükséges feladataink elvégzésére is. Ahogy egyre bővül az internet felhasználásának köre, úgy bővül azoknak a webes alkalmazásoknak és weboldalaknak a komplexitása is, amelyek a legkülönfélébb szolgáltatásokat nyújtják.

Ezeknek a modern lehetőségeknek a kihasználáshoz szükség van egy olyan korszerű böngészőre, amely képes gyorsan feldolgozni és megjeleníteni még a legkomplexebb webes alkalmazásokat is, hiszen mit sem érne az a sok hasznos szolgáltatás, ha az időnket a tényleges információ feldolgozás helyett azzal kellene töltenünk, hogy várunk a böngészőre.

A megújult Internet Explorer legfontosabb célkitűzései között szerepel a böngésző sebességének növelése, a böngészés közbeni felhasználói élmény fokozása, az egyszerű és könnyen elsajátítható kezelhetőség, valamint a biztonságos internetezés lehetőségének megteremtése.

Ám a fenti célok eléréséhez, a képességek kihasználásához, nem csak gyors és egyszerűen kezelhető böngészőre van szükségünk, hanem olyanra, amely támogatja a modern webes szabványokat. Hiszen a webes világban minden a szabványok köré épül: a szabványok írják le kedvenc weboldalaink felépítését és szerkezetét, a szabványok határozzák meg azt, hogy milyen elemekkel, vezérlőkkel találkozhatunk a weboldalainkon. Viszont ha a böngészőnk nem támogatja megfelelően a webes szabványokat, akkor a kedvenc weboldalaink nem fognak helyen működni, üzemelni és kedvenc webes szolgáltatásainkat is csak megkötésekkel használhatjuk. Ezért készült úgy az Internet Explorer 9, hogy minden korábbi verziójánál pontosabban támogassa a webes szabványokat, a HTML5-öt, valamint a köré épülő technológiákat.

1.1. Teljesítmény

Bármilyen alkalmazást is kell használni, akár személyes célra, akár a mindennapi munkához, a program egyik legfontosabb jellemzője az, hogy milyen gyorsan képes elvégezni az általunk elvárt feladatot. Miközben egyre több időt töltünk a weben, egyre fontosabbá válik a böngészőnknek azon jellemzője, hogy mekkora sebességgel képes feldogozni egy-egy weblapot, hiszen felhasználóként nem a várakozást jelző homokórára vagyunk kíváncsiak, hanem a tényleges weblap tartalmára. Böngészés közben a hagyományos böngészők többségével a számítógépünk összteljesítményének csupán 10%-át használjuk ki. Ha olyan weboldalt jelenítünk meg, ami számításigényes, lehetséges, hogy akkor is várni kell a weboldal feldolgozására és megjelenítésére, ha van szabad erőforrása a számítógépünknek. Az Internet Explorer 9 használatával ez többé nem jelenthet kellemetlenséget, hiszen úgy készült, hogy a számítógép teljes számítási kapacitását kihasználja. A hardveres gyorsításnak köszönhetően a számítógép teljesítményének 100 %-a felhasználható és képes olyan webes alkalmazások gördülékeny megjelenítésére is, amelyeket korábban csak hagyományos asztali alkalmazásokkal lehetett megvalósítani.

Ha közelebbről is szemügyre vesszük azt, hogy hogyan dolgozzák fel a böngészők a weboldalakat, szembesülhetünk azzal, hogy egy több lépésből álló, összetett folyamatról van szó. Fel kell dolgozni és értelmezni kell többek között a weboldal szerkezeti felépítését (HTML), stílusát (CSS) és a weboldalon elhelyezett parancsfájlokat (JavaScript). Az egyes lépések számításigénye pedig weboldalanként eltérő lehet, ami nagyban megnehezíti a böngésző optimalizálását. Azonban vannak olyan gyakori lépések is, amelyekre mégis érdemes nagy hangsúlyt helyezni. Ilyen például a parancsfájl értelmezés vagy a különböző grafikai műveletek. A továbbiakban e gyorsításokkal és optimalizálásokkal fogunk részletesebben megismerkedni.

Chakra, az új JavaScript motor

Az Internet Explorer 9 Chakra névre hallgató, megújult parancsfájlt feldolgozó motorja felelős azért, hogy a világhálón levő weboldalak többsége a korábbi Internet Explorerekhez képest sokkal gyorsabban jelenjen meg. Az új JavaScript motor lényege abban rejlik, hogy kétprocesszoros architektúrákra optimalizálták. A processzor egyik magján fut maga az Internet Explorer, a másik magon pedig párhuzamosan a gépi kódra lefordított JavaScript parancsfájl.

A lefordított gépi kódnak köszönhetően a számítógép központi feldolgozó egysége egyenletes és folyamatos terhelésnek van kitéve. A többmagos processzor kihasználása mellett a modern webes trendek által használt gyakori fejlesztési minták is alapul szolgáltak az optimalizálás során.

Hardveresen gyorsított grafikus feldolgozás

A modern webes trendeknek köszönhetően az internetes világ egyre közelebb kerül a hagyományos, asztali alkalmazások világához. Ennek a folyamatnak a velejárója, hogy olyan látványos grafikai elemeket jeleníthetünk meg a böngészőben, mint például animált vektorgrafikus alakzatok (SVG) vagy a vászon. A grafikai műveletek rendkívül számításigényes feladatok, és ha a böngészőnk nem képes kihasználni a számítógép grafikus feldolgozóegységének számítási kapacitását, akkor a komplexebb grafikai elemek megjelenítésénél már-már majdnem

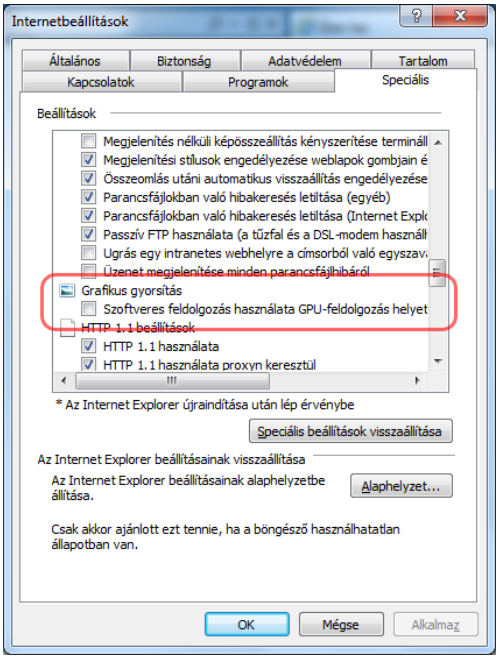
használhatatlanná válhat. Ami a webes világ jövőjét illeti, a látványos grafikai elemek mellett a beágyazott multimédiás tartalmaknak is fontos szerepük lesz az internet „következő generációjában”. Jelenleg azonban a multimédiás tartalmakat a böngészők többsége nem tudja önmaga megjeleníteni, szükségük van valamilyen külső alkalmazásra, hogy ezt megtegyék.

A megújult Internet Explorer sikerül megbirkóznia a fenti kihívásokkal, hiszen teljes mértékben képes igába hajtani a számítógép grafikus processzorát a grafikai elemek megjelenítéséhez. A multimédiás elemek megjelenítését pedig teljes mértékben az Internet Explorer motorja végzi, szintén a grafikus processzor segítésével. Ennek köszönhetően a számítógép központi feldolgozó egysége mentesül a számításigényes feladatok többsége alól, gördülékenyebbé téve a böngészést.

Az Internet Explorer 9 optimalizálásának része még egy olyan speciális funkció is, amely azt határozza meg, hogy a számítógép grafikus feldolgozó egysége (GPU) és a központi feldolgozó egysége (CPU) közül melyiket hatékonyabb használni. Ennek a funkciónak főképp a régebbi számítógépek esetén van nagy jelentősége, ugyanis ezeknek a grafikai teljesítménye elmaradhat a központi feldolgozó egység teljesítményétől. Ennek a számítási sebességnek a függvényében az IE9 az alábbi esetekben kapcsolhatja ki a hardveres gyorsítást:

- A grafikus feldolgozó egység és a hozzá tartozó illesztőprogram lassabban jeleníti meg a grafikus elemeket, mint a központi feldolgozó egység
- A grafikus feldolgozó egység illesztőprogramja instabil, a népszerű weboldalak megjelenítésekor összeomlik
- A grafikus feldolgozó egység vagy az illesztőprogramja nem megfelelően jeleníti meg a kívánt tartalmat

A grafikus gyorsítás azonban kézzel is kikapcsolható, mégpedig az Internet Explorer 9 *„Eszközök (Alt + X)-> Internetbeállítások -> Speciális -> Grafikus gyorsítás”* felületén keresztül.



1. ábra: Hardveres gyorsítás kikapcsolása

Amennyiben ez az opció inaktív (szürke), az Internet Explorer nem támogatja az adott számítógépen a hardveres gyorsítást a fenti három ok valamelyikéből adódóan. Ebben az esetben a problémát nagy valószínűséggel megoldja az, hogyha frissítjük a grafikus vezérlő illesztőprogramját a lehető legfrissebb verzióra (ezt a Windows Update-en keresztül, vagy a gyártó oldaláról tehetjük meg).

Még több optimalizálás

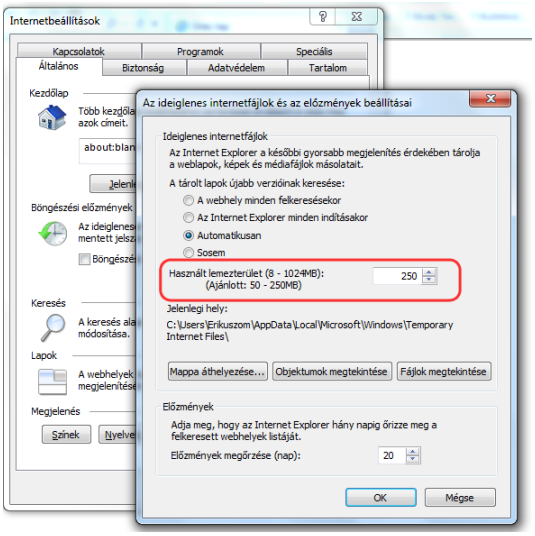
A hardveres gyorsítás az Internet Explorer 9 egyik legfontosabb része és újdonsága. Azonban ezen felül még számos olyan beépített funkcióval is rendelkezik, ami a weblapok feldolgozási sebességét hivatott gyorsítani. A böngészés sebességét alapvetően két tényező határozza meg: a hálózat sebessége, azaz, hogy milyen gyorsan töltődik le a kiszolgálóról az adat, illetve az, hogy a böngésző milyen gyorsan képes feldolgozni ezt az adatot. Összességében elmondható, hogy egy weboldal megjelenítésekor a várakozással töltött idő több, mint fele azzal telik el, hogy a kiszolgálóra várunk.

A hálózati sebesség optimális kihasználásának érdekében az Internet Explorer 9 több fontos újítással is rendelkezik. Az egyik ilyen újítás a továbbfejlesztett névfeloldási (DNS) rendszere. Amikor meglátogatunk egy weboldalt, begépelünk a címsorba egy webcímet, akkor a böngészőnek ebből a begépelt címből meg kell határoznia a szerver fizikai címét. Erre azért van szükség, mert sokkal könnyebb megjegyezni például azt, hogy „www.bing.com”, mint azt, hogy „217.156.169.184”. Ezeket a párosításokat úgynevezett DNS (*Domain Name System*) szervereken

tárolják, és a böngészőknek ezekhez a szerverekhez kell fordulniuk, amikor meghatározzák az egyes webcímek fizikai címét. Ez a folyamat szervertől függően pár ezredmásodperctől több másodpercig is tarthat. Az Internet Explorer 9 pedig ezt az időt csökkenti le azáltal, hogy gyorsítótárazza a meglátogatott webhelyek fizikai címét, így az adott weboldal első letöltését követően nincs szükség arra, hogy a továbbiakban újból a DNS szerverhez forduljon.

A másik fontos újítás a weboldalak gyorsítótárazásának továbbfejlesztése. Amikor egy weboldal letöltődik az internetről, a böngészők többsége ezt a letöltött tartalmat (a képekkel és más elemekkel együtt) elhelyezi a gyorsítótárba. A weboldal újbóli meglátogatásakor a már korábban elmentett elemek nem a kiszolgálóról töltődnek le, hanem a helyi számítógépen található gyorsítótárból. Az Internet Explorer korábbi verzióiban a gyorsítótár alapértelmezett mérete a teljes lemezterület 1/32 részét foglalta el (maximum 50 megabájtot).

Azért nem többet, mert a nagyobb gyorsítótárból való adat visszakeresése majdhogynem több időt vett igénybe, mint a kiszolgálóról való letöltés. Az IE9-ben viszont ezt a „kellemetlenséget” orvosolták, aminek köszönhetően a nagyobb méretű gyorsítótárakkal is hatékonyan elboldogul. Immáron ennek a tárnak az alapértelmezett mérete a merevelemez 1/256-része (és maximum 250 megabájt), így a kisebb tárolókapacitással rendelkező számítógépek (például netbookok) háttértárán kevesebb, míg az átlagos számítógépek esetében nagyobb helyet foglal el, mint korábban. Természetesen ez a méret kézzel is módosítható, méghozzá az *Eszközök (Alt + X)-> Internetbeállítások -> Általános -> Böngészési előzmények -> Beállítások* menüpont alatt.

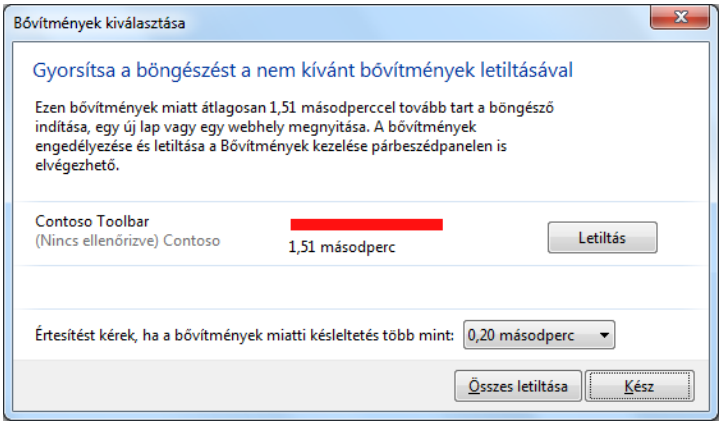


2. ábra: Gyorsítótár méretének növelése

Azonban nem célszerű ezt a méretet nagyon megnövelni, mivel az IE9 maximum 60.000 eltárolt objektumot képes kezelni, és hiába emeljük meg több gigabájtra a gyorsítótár méretét, a tár azelőtt eléri a maximális objektumszámát, hogy kihasználná a megnövelt méretet.

Bővítményteljesítmény-tanácsadó

A különféle bővítményeknek az a szerepük, hogy hasznosabbá, okosabbá tegyék a böngészőt azáltal, hogy kibővítik új funkciókkal. Bővítményekkel pedig bizonyára már mindenki találkozott, hiszen ilyen bővítmény például az oly népszerű SilverLight vagy Flash plug-in. Azon felül, hogy kiterjesztik a böngészőnk funkcióinak palettáját, és okosabbá teszik azt, hátrányuk is van. Számos olyan bővítmény akad, amely lelassíthatja a böngésző betöltését, elindulását, egy-egy új fül megnyitását, vagy magát a böngészést is. Az Internet Explorer 9 folyamatosan figyeli a betöltött bővítmények betöltési idejét, azok sebességét, és akárhányszor megnyitunk egy új fület, vagy egy weboldalra navigálunk, eltárolja ezeket az adatokat.



4. ábra: Bővítmények letiltása

Ezen adatok alapján kiszámolja a bővítmények átlagsebességét, majd ha ez a sebesség elér egy bizonyos értéket, aktivizálja a bővítményteljesítmény-tanácsadót. A bővítményteljesítmény-tanácsadónak ez a határértéke alapértelmezés szerint 0.2 másodperc, ugyanis ha ennél tovább tart egy lap megnyitása, az már hosszú időnek tűnik egy felhasználó szemébe. Amennyiben aktivizálódott a bővítményteljesítmény-tanácsadó funkció, az értesítő sávban megjelenik az az üzenet, hogy „Gyorsítsa a böngészést a bővítmények letiltásával."

3. ábra: Bővítményteljesítmény-tanácsadó működése

Ekkor két opció közül választhatunk: azonnal letilthatjuk a bővítményeket, vagy elhalaszthatjuk későbbre a döntést. Amennyiben ez utóbbit választjuk, a böngésző újraindításakor ismét meg fog jelenni az értesítés. Ha a bővítmények kiválasztása mellett döntünk, akkor felugrik egy olyan ablak, ami tartalmazza a böngésző működését lassító bővítményeket.

A párbeszédpanelen keresztül egyből letilthatjuk azokat a bővítményeket, amelyek a lassulást okozzák. Sőt, ezen felül azt is megtekinthetjük, hogy mennyi idő alatt sikerült betöltenie a böngészőnek az adott bővítményt, illetve bővítményeket. Az alapértelmezés szerinti 0,2

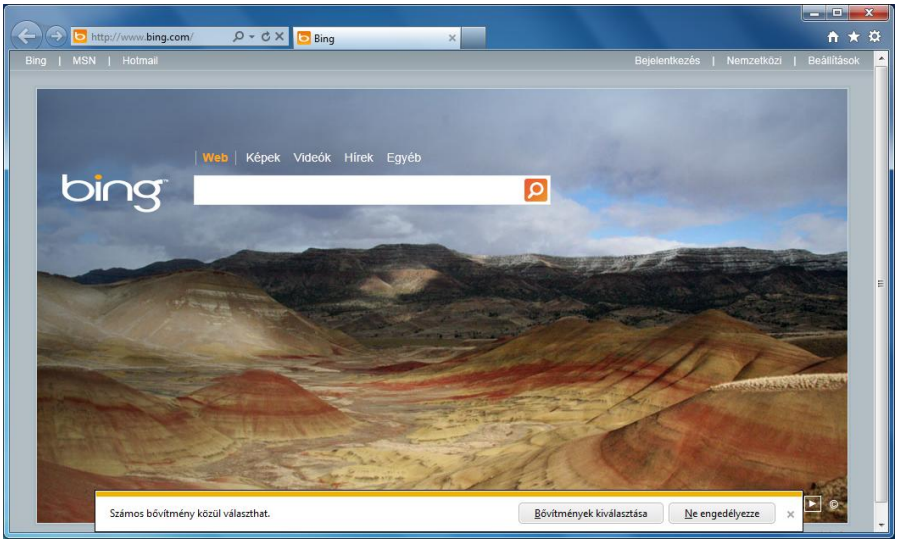
másodperces betöltési határérték módosítását szintén ezen a párbeszédpanelen keresztül tehetjük meg. Ez a lehetőség különösen fontos lehet lassabb számítógépek esetében. Ugyanis ha nem rendelkezünk kellőképpen gyors számítógéppel, és mégis sok bővítményt töltetünk be az IE9-el, akkor bármilyen új bővítményt telepítünk fel, a böngésző újraindításakor meg fog jelenni ez az értesítés, ez pedig zavaró, illetve megtévesztő lehet.

1.2. Megjelenés és használat

Az Internet Explorer 9 újításai nem csupán sebességben mutatkoznak meg, hanem megjelenésben is, hiszen a korábbi verzióihoz képest teljes mértékben megújult felhasználói felülettel rendelkezik. Az új felület – a többi böngészővel ellentétben – teljes egészében a webhelyekre, weblapokra, illetve azok tartalmára helyezi a hangsúlyt. Az átlag felhasználók böngészési szokásait figyelembe véve a megújult felület az olyan gyakran használt funkciókat emeli ki, mint például a navigálás az előző oldalra, több böngésző fül párhuzamos használata vagy a címsorból való navigáció. A gyakran használt funkciók előtérbe helyezésének és összevonásának eredményeképpen egy olyan kezelőszerv jött létre, amelyben csak azok az elemek látszanak, amelyek a modern böngészéshez szükségesek. Emellett rendkívül nagy területet foglal el a böngészőkeretből az a rész, ahol a felkeresett webhely tartalma megjelenítődik, így felhasználóként az igazán lényeges dolgokra, azaz a megnyitott webhelyre koncentrálhatunk böngészés közben. Az első ránézésére szembetűnő főképernyő változásain túl még számos olyan felületi újítással is rendelkezik az IE9, ilyen például az új értesítési sáv vagy a megújult új lap megnyitása funkció.

Webhely központú főképernyő

Az 5. ábrán látható főképernyő jól szemléleti a legfontosabb újdonságokat: a megvékonyított böngészési eszköztárat, azt a rendkívül nagy területet, ahol a webhely foglal helyet és az új értesítési sávot, amely nem zavarja a felhasználót böngészés közben.

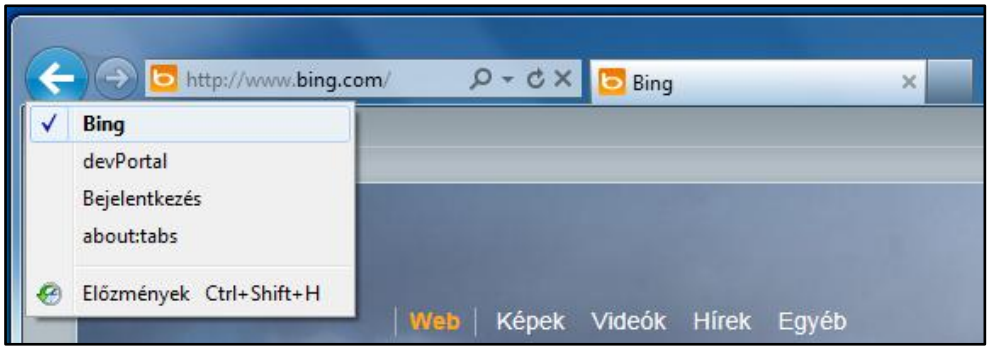


5. ábra: Internet Explorer 9 főképernyője

Navigációs gombok

A megújult navigációs sávban található vezérlőgombok, menüpontok, címsor és lapok összevonásának a helytakarékos megoldáson felül a legfontosabb szerepe az, hogy a felhasználók számára megkönnyítsék, leegyszerűsítsék és felgyorsítsák a leggyakrabban használt funkciók elérését.

Böngészés közben az egyik leggyakoribb művelet az előző oldalra való navigálás, ezért a böngésző eszköztárában a többi gombhoz képest kiemelt szerepet kapott, amit a mérete is jól tükröz. Hosszanti megnyomásával nem csak az előző weboldalra navigálhatunk vissza, hanem valamennyi korábban megnyitott webhelyre is. Ezen felül a böngészési előzmények is elérhetőek egy kattintással. Ha hozzászoktunk a gyorsbillentyűk használatához, akkor hasznos lehet, ha tudjuk, hogy az *Alt + jobbra / balra* billentyűkombinációk megnyomásával is navigálhatunk előre, illetve vissza.



6. ábra: Navigálás az előző oldalra

Intelligens címsor

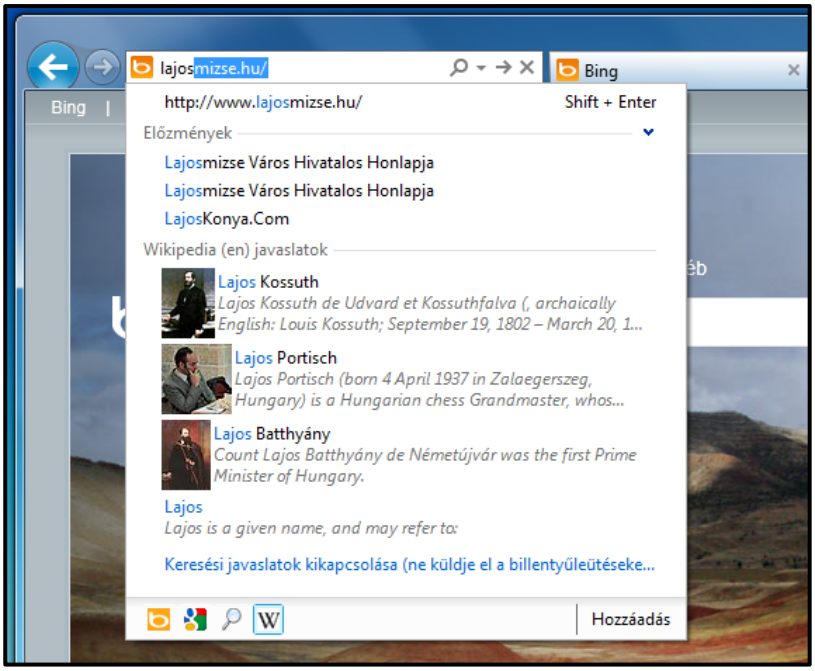
A korábbi böngészőkben a címsornak csupán egy feladata volt, mégpedig az, hogy elnavigáljon bennünket a kívánt webhelyre azáltal, hogy begépeltük a webhely címét. Manapság azonban ez a vezérlő egyre okosabbá vált, és a böngészők többségében a navigáción túl fontos szerepe van még a világhálón való keresésben is. Az Internet Explorer 9 újjátervezett címsora viszont ennél sokkal több: számos hasznos funkciót rejt magában, és ezt a széleskörű funkcionalitást ötvözi a letisztult megjelenéssel és a könnyű kezelhetőséggel.

Az IE9 címsorával (*OneBox)* a hagyományos navigáción túl lehetőségünk van háromféle keresésre is: kereshetünk a meglátogatott webhelyek címeiben, a böngészési előzményeink között, valamint a különböző keresési szolgáltatók révén kutathatunk szövegesen és vizuálisan is a világhálón. Gépelés közben a címsorból legördül egy több részre bontott lista. A lista első felében a böngésző a korábban meglátogatott webhelyek címei között keres (fontos tudni, hogy a keresés csak a fő szervercímekben történik, tehát ha meglátogatjuk a www.alma.com/piros és a www.alma.com/zöld weboldalakat, akkor a találati listában a www.alma.com webcímet fogjuk

látni). Az itt megjelenített első találatot egyből meg is nyithatjuk a Shift + Enter billentyűkombináció lenyomásával. A találati lista második felében a böngészési előzményeinkben történik a keresés. Itt alapértelmezés szerint csak az első három eleme látszik a találati listának, de lehetőségünk van arra is, hogy az összes találatot megjelenítsük. A találati lista harmadik részében pedig a keresési javaslatokat láthatjuk, azaz a kiválasztott keresési szolgáltató által, a világhálón történő keresés eredményei jelennek meg szövegesen vagy vizuálisan (amennyiben ez a szolgáltatás engedélyezve van).

A világhálón való keresésnek olyannyira kiemelkedő szerepe van, hogy amikor elkezdjük gépelni a karaktereket, majd lenyomjuk az *Enter* gombot, akkor automatikusan a kiválasztott keresési szolgáltató találati listájára fog minket navigálni az IE9. Adatvédelmi okok miatt azonban alapértelmezés szerint a keresési javaslatok ki vannak kapcsolva, de bekapcsolhatjuk azt a keresés közben megjelenő legördülő listából vagy az *„Eszközök -> Bővítmények kezelése -> Keresési szolgáltatók”* menüpontból.

A keresésszolgáltatók olyan Internet Explorer 9 bővítmények, amelyeket különféle internetes szolgáltatók készítettek keresési funkcióikhoz, szolgáltatásaikhoz azzal a céllal, hogy a felhasználók képesek legyenek keresni adatbázisaikban a böngészőn keresztül. Ilyen keresőszolgáltató bővítményeket készített már például a Google, a Bing, a Wikipédia, az Amazon és az eBay is. A keresésszolgáltató bővítmények többsége szövegesen jeleníti meg a keresés eredményét, viszont léteznek olyanok is, amelyek grafikusan. Ez annyit tesz, hogy a találati listában formázott szöveggel egybekötött képeket is tartalmazhatnak a keresések találatai (7. ábra). Amennyiben nem lennének elegendőek az alapértelmezetten telepített keresésszolgáltatók, a http://www.ieaddons.com/hu webhelyről többek között számos magyar keresésszolgáltató is letölthető.



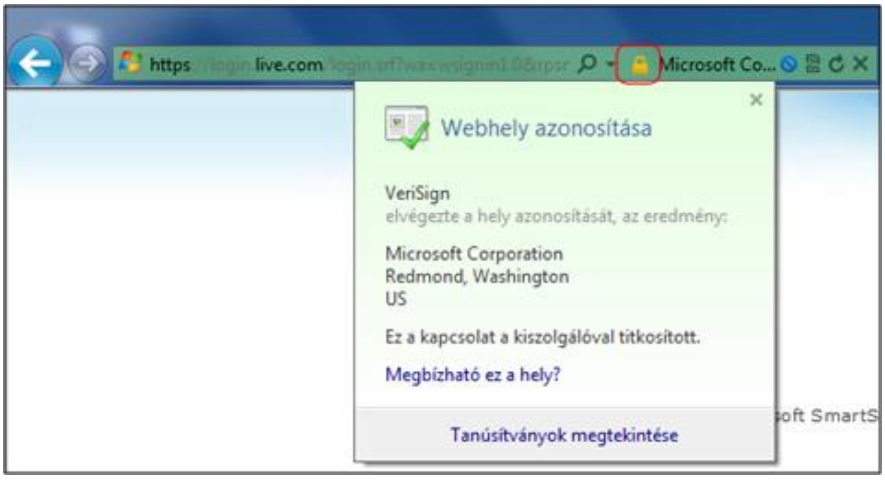
7. ábra: Címsor és keresés

A navigáción túl a címsorban kaptak helyet az olyan kevésbé gyakran használt navigációs gombok is, mint a webhely *frissítése*, vagy a webhely letöltésének *leállítása* (8. ábra).

Azért kerültek oda, mert közvetlen kapcsolatban állnak a címsorban található webhellyel (nem úgy, mint a *Vissza* vagy az *Előre* gombok, amelyek egy másik webhelyre navigálnak), valamint jelentőségük sem olyan számottevő, mint az előre / vissza navigáció.

8. ábra: Kiegészítő navigációs gombok

Szintén a címsorban találhatók azok az ikonok (illetve gombok) is, amelyek a betöltött webhely bizonyos állapotaira utalnak. Ezeknek az ikonoknak a segítségével informálódhatunk arról, hogy milyen a kapcsolatunk a számítógép és a szerver között, arról, hogy a megtekintett oldal mennyire kompatibilis az IE9-el, továbbá, hogy fennakadt-e az oldal bizonyos tartalma a követésvédelem, illetve az ActiveX szűrőkön.



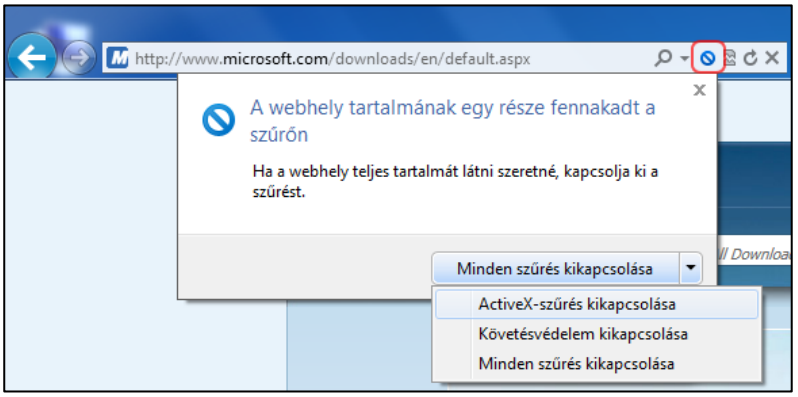
9. ábra: Biztonsági tanúsítvány

A számítógép és a szerver közötti kapcsolat állapotát a *Biztonsági jelentés* ikon jelzi (9. ábra). Amennyiben ez az ikon nem látható, úgy a webhely és a számítógépünk közötti kommunikáció hagyományos HTTP kapcsolaton keresztül történik. Azoknál a webhelyeknél, ahol a kommunikáció titkosított, biztonságos csatornán keresztül zajlik, erre az ikonra kattintva megtekinthetjük a hitelesítést végző szervezet nevét, és megnézhetjük a szervezet által kiállított biztonsági tanúsítványt is. Ha valóban megbízható a webhely, akkor a címsor háttere zöld színű, viszont ha érvénytelen a tanúsítvány, akkor piros lesz, ezzel is jelezve azt, hogy nem biztonságos folytatni a böngészést (10. ábra).

10. ábra: Érvénytelen biztonsági tanúsítvány

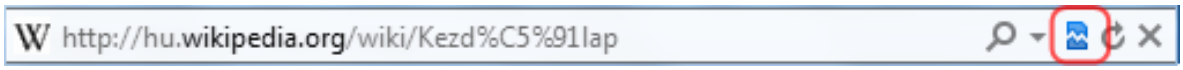
Amennyiben aktiváltuk az Internet Explorer 9 *Követésvédelmi* és/vagy *ActiveX-szűrési* funkcióját, úgy adódhatnak olyan webhelyek, amelyeknek bizonyos részei fennakadnak ezeken a szűrőkön (11. ábra).

Ilyenkor nem a teljes webhely tartalmát láthatjuk a böngészőben, hanem csak bizonyos részeit, aminek következtében a megszokotthoz képest eltérően jelenhet meg az oldal. A szűrőkön fennakadt tartalmat jelző ikon funkciója pedig az, hogy ezt az információt közölje velünk, felhasználókkal, és lehetőséget adjon ezeknek a szűrőknek a kikapcsolására, így a teljes tartalom megtekintésére. A megjelenő kis ablakon keresztül külön is kikapcsolhatók a szűrések, így például beállíthatjuk azt, hogy az adott webhelyen megjelenjenek az ActiveX tartalmak, miközben a követésvédelem továbbra is aktív marad. Fontos megjegyezni, hogy amikor kikapcsoljuk ezeket a szűrőket, a kikapcsolás csak az adott webhelyet érinti, egyéb webhelyek meglátogatása esetén ezek a szűrők továbbra is aktívak.



11. ábra: Követésvédelem, illetve ActiveX-szűrő

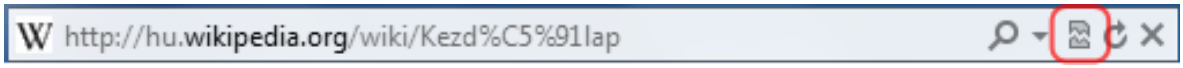
A böngészőipar és webes szabványok fejlődésével elkerülhetetlen, hogy néhány, a korábbi webes szabványoknak teljes mértékben megfelelő webhely az új szabványok szerint másképp jelenjen meg, mint korábban. Ennek az az oka, hogy a webhelyet alkotó dokumentum szerkezetének leírásához használt néhány elem jelentése az új szabvány szerint más, mint a régi szerint. Mivel a régebbi böngészők a korábbi webes szabványokat támogatják, a régi böngészőkön „jól” jelenik meg a „régebben” elkészített webhely, viszont ugyanaz a webhely a modern webes szabványokat támogató böngészőkön megjelenítve másképp nézhet ki. A fenti kompatibilitási problémákból adódó hibák megoldásához az Internet Explorer 9 egy olyan funkcióval rendelkezik, amely képes kompatibilitási üzemmódban megjeleníteni a webhelyeket. A probléma megoldása úgy történik, hogy a megjelenítendő webhely címét összeveti egy folyamatosan frissülő adatbázissal, ami azokat a webhelyeket tárolja, amelyek nem kompatibilisek a modern szabványokkal.



12. ábra: Webhely megtekintés kompatibilitás üzemmódban

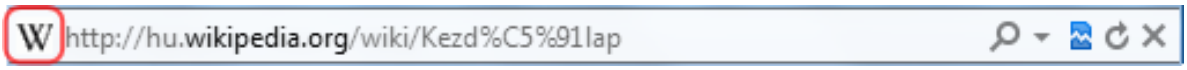
Ha a megjelenítendő oldal szerepel ebben az adatbázisban, akkor kompatibilitás módban jeleníti azt meg, amiről *Kompatibilitási nézet* ikonon keresztül értesíti a felhasználót. A *Kompatibilitási nézet* ikon csak akkor látszik a címsorban, ha a megjeleníteni kívánt webhelyet az IE9 kompatibilitási módban jelenítette meg (12. ábra).

Az ikonra kattintva azonban lehetőségünk van arra is, hogy megpróbáljuk normál üzemmódban megjeleníteni a webhelyet (ilyenkor szintén látszani fog az ikon, azonban nem kéken, hanem szürkén, 13. ábra).



13. ábra: Webhely megtekintés normál üzemmódban

Végül, de nem utolsó sorban a címsor fontos szerepet tölt be az Internet Explorer 9 rögzíthető webhelyek funkciójában is (amelyről majd a későbbiekben szó lesz). A webhelyek többségéhez tartozik egy *favico*-nak nevezett kis ikon, ami a böngészők címsorában, a webhely címe előtt látható, és a webhely ikonját szimbolizálja. (Az ikon neve onnan ered, hogy az Internet Explorer 7-es verziója előtt csak akkor volt látható, ha a weboldalt elmentettük a kedvencekhez, ennek az angol megfelelője favorites). Többek között az ikonnak a segítségével is rögzíthetjük az adott webhelyet a Windows 7 tálcánkon. Ehhez csupán rá kell húzni a tálcára ezt az ikont, és máris rögzült a webhely (14. ábra).

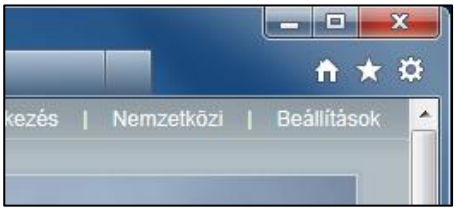


14. ábra: Webhely ikonja

A modern web világában (sajnos) egyre népszerűbbek az olyan adatlopási technikák, amelyek a felhasználók megtévesztésével érik el céljukat, azaz a személyes adatok (felhasználónevek, jelszavak, bankszámla adatok) megszerzését. Ezekben az esetekben a támadók olyan weboldalakat készítenek, amelyek külalakja teljes mértékben megegyezik a megszokott weboldalainkkal. Azért, hogy ezeket az áloldalakat fel is keressék a felhasználók, többnyire olyan kiszolgálókon helyezik el, amelyeknek a címe is hasonlít az eredeti webhely címéhez. Például ha egy támadó el szeretné lopni a Windows Live azonosítónkat, akkor készíthet egy olyan weboldalt, amely külalakra megegyezik a Windows Live bejelentkezési felületével, viszont nem *https://login.live.com* címe van, hanem például *http://login.livemessenger.com*. Ebben az esetben a felhasználók egy részének biztosan nem fog feltűnni, hogy valójában teljesen máshol barangol, mint ahogy azt ő hiszi, és megadja a felhasználónevét / jelszavát. Az IE9 megújult címsora azonban kiemeli a szerver címéből a tényleges szerver címét, tehát a *livemessenger.com* részt, aminek köszönhetően a felhasználó könnyebben észlelheti, hogy a felkeresett webhely nem az, mint aminek látszik.

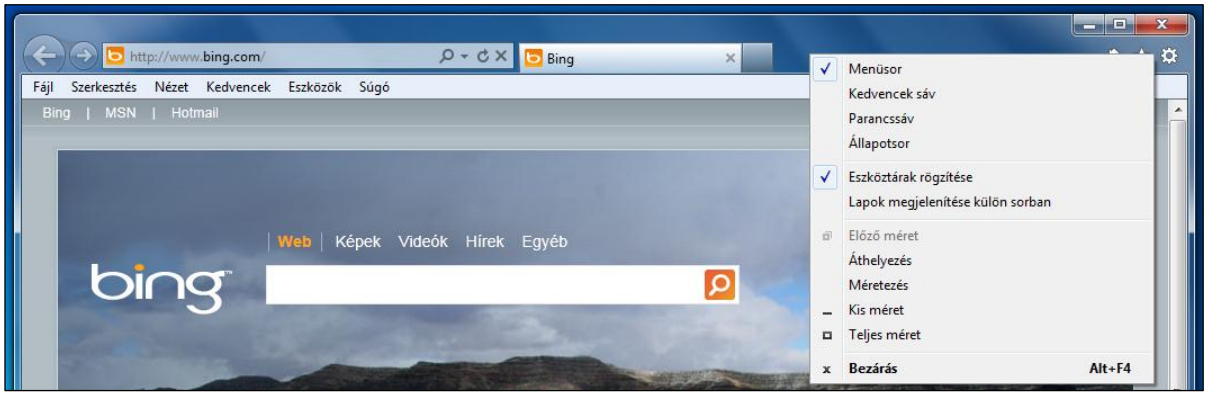
Összevont menüpontok

Az alkalmazások többsége – legyen szó akár böngészőről, akár más alkalmazásról – többnyire rendelkezik menüsávval, amelyből elérhetőek az adott alkalmazás funkciói és szolgáltatásai. A menüsor által nyújtott funkciók között találhatunk olyanokat, amelyeket gyakran használunk, de akadnak olyanok is, amelyeket ritkábban. Egy modern böngészőtől elvárt követelmény, hogy rendkívül széleskörű funkciótárral rendelkezzen. A sok funkció és szolgáltatás pedig sok menüponttal jár, függetlenül attól, hogy átlagos felhasználóként mennyire sűrűn használjuk ezeket. Az Internet Explorer 9 úgy igyekszik könnyen kezelhetővé tenni a böngészőt, hogy a leggyakrabban használt funkciókhoz tartozó menüpontokat kiemeli a navigációs sávban, a ritkábban használt szolgáltatásokat rejtő menüpontokat pedig elrejti a felhasználó elől. Így a gyakori funkciókat gyorsabban elérhetjük, és a többi alkalmazáshoz, böngészőhöz képest sokkal több értékes terület szabadul fel a webhely megjelenítéséhez. Három fő szolgáltatást emel ki a főképernyő: a *Kezdőlapot*, a *Kedvenceket* és az *Eszközöket* (15. ábra).



15. ábra: Gyakran használt menüpontok

Természetesen a böngésző többi szolgáltatása is elérhető a menüsorból, amelyet ideiglenesen az *Alt* gomb lenyomásával, véglegesen pedig jobb egérgombbal a böngészőkeret fejlécében -> *Menüsor* menüponttal tudunk megjeleníteni (16. ábra)



16. ábra: Menüsor megjelenítése

Böngészés során az egyik leggyakrabban használt (és egyben a legegyszerűbb) művelet a kezdőlapra való navigáció, ezért az IE9 főképernyőjén is kitüntetett szerepet kapott. Azok a felhasználók, akik a gyorsbillentyűket részesítik előnyben, az *Alt + Home* billentyűzet kombinációval érhetik el ezt a funkciót.

Kitüntetett szerepet kapott a *Kedvencek* menüpont is az IE9 főképernyőjén, hiszen a jól bevált böngészési szokások közé tartozik a kedvenc webhelyek eltárolása. Már a világháló terjedésének hajnalán, az első böngészők is rendelkeztek ezzel a funkcióval, amelynek népszerűsége azóta is töretlen. Az Internet Explorer 9 *Kedvencek központja* azonban nem csak egy egyszerű eszköz arra, hogy a felhasználók rendezhessék, karbantarthassák kedvenceiket, hanem ennél sokkal több. A kedvenc weboldalak megtekintése mellett egy kattintással megnézhetjük azokat a hírcsatornákat is, amelyekre feliratkoztunk, valamint megtekinthetjük a korábban meglátogatott webhelyeinket, azaz a böngészési előzményeket is (17. ábra).

A *Kedvencek központ Javasolt webhelyek* szolgáltatása pedig megpróbál az érdeklődési körünknek megfelelő webhelyeket javasolni a korábban felkeresett webhelyeink alapján. Például ha meglátogatjuk a www.facebook.com weboldalt, akkor a szolgáltatás - a többi IE felhasználó előzményei alapján - felajánlja, hogy megtekintsük többek között a hi5.com, a myspace.com és a youtube.com webhelyeket. Mivel a szolgáltatás a személyes adatainkon alapul, adatvédelmi okok miatt alapértelmezés szerint ki van kapcsolva, és ha ki szeretnék próbálni, kézzel kell bekapcsolnunk. Ezt megtehetjük a bővítmény-karbantartó felületről.

17. ábra: Kedvencek központ

Az Internet Explorer 9 által nyújtott leghasznosabb funkciókat az *Eszközök* menüpont foglalja össze és teszi gyorsan elérhetővé. Az *Eszközök* menüpont segítségével egy kattintással elérhetők az olyan fontos szolgáltatások, mint például a nyomtatás, a letöltéskezelő, a nagyítás, a bővítménykezelés vagy a beállítások, és az olyan biztonsági funkciók is, mint például a böngészési előzmények törlése, az InPrivate-böngészés, a SmartScreen-szűrő, a Követésvédelem vagy az ActiveX-szűrés. Az összevont *Eszközök* menüpontnak köszönhetően úgy érhetjük el ezeket a fontos és gyakran használt funkciókat, hogy nem kell végigböngésznünk az összes menüpontot, ezzel pedig időt takaríthatunk meg.

Megnövelt böngészési terület

Az elsődleges ok, amiért felhasználóként elindítjuk a böngészőt az, hogy megjelenítsük kedvenc webhelyeinket, híreket olvassunk, közösségi életet éljünk, szórakozzunk vagy információkat szerezzünk az általunk érdekesnek tartott dolgokról. Senki sem azért indítja el a kedvenc böngészőjét, hogy annak felhasználói felületében gyönyörködjön, hanem azért, hogy az általa felkeresett webhely tartalmát megjelenítse. Az Internet Explorer 9 ezt az igényt maximálisan szem előtt tartja azáltal, hogy a böngészőkeretből hihetetlen nagy területet foglal el az a rész, ahol a webhely tartalma megjelenítődik. Az Internet Explorer 9 a teljes böngészőablak 86%-át használja arra, hogy megjelenítse a webhelyek tartalmát, míg a konkurens böngészők átlagosan csupán a 78%-át (1. táblázat).

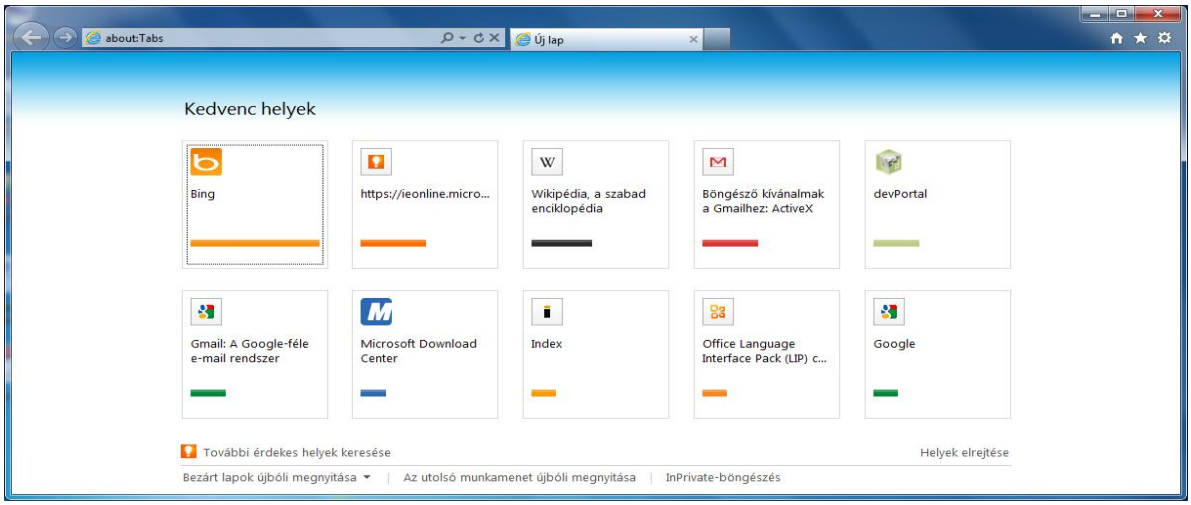
Böngésző	Böngésző ablak	Fejléc + lábléc	**Tényleges tartalom**
Internet Explorer 9 (9.0.8112)	100%	14%	**86%**
Mozilla Firefox 4	100%	21%	**79%**
Google Chrome 10 (10.0.648)	100%	20%	**80%**
Opera 11 (11.01.1190)	100%	25%	**75%**

1. táblázat: Böngészők hasznos területének alakulása

A tényleges méret, amit ez a körülbelül 8%-nyi különbség lefed egyenesen arányos a böngésző ablakunk nagyságával és a számítógépünk felbontásával. Már egy átlagos felbontású, teljes ablakméretre maximalizált böngésző esetében is sok értékes információt jelenthet ez a terület.

Új lap megnyitása

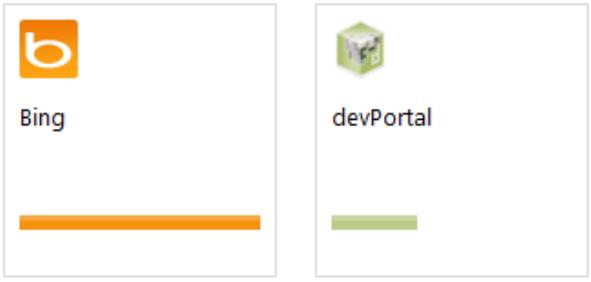
Az Internet Explorer 9-es verziójában található, a korábbi verziókhoz képest teljesen átszabott többlapos böngészés az egyik leggyakrabban használt és leghasznosabb funkciója a böngészőnek. Az átlagos felhasználók böngészési szokásait figyelembe véve a többlapos böngészés az egyik legnépszerűbb böngészési szokás, és ennek fényében az IE9 kiemelt figyelmet fordít az új lap megnyitásának folyamatára is.



18. ábra: Új lap megnyitása

Új lap megnyitásakor egy üres képernyő helyett az Internet Explorer 9 automatikusan felajánlja az általunk leginkább kedvelt webhelyek megnyitását, ezzel is rendkívül gyorssá és egyszerűvé téve a kedvenc webhelyeink közötti navigációt (18. ábra). A kedvenc webhelyek megjelenítése mellett egy kattintással újranyithatjuk a legutóbb bezárt weblapjainkat, vagy akár a teljes munkamenetünket is. Ezen felül szintén egy kattintással elindítható az InPrivate-böngészés is.

Új lap megnyitásakor az általunk legkedveltebb tíz webhely belekerül egy kis dobozba, és ez a tíz doboz egy öt oszlopos és két soros elrendezést vesz fel, úgy hogy a leggyakrabban látogatott webhelyeink kerülnek előre, és a kevésbé látogatottak hátra. Az egyes webhelyekhez tartozó dobozokban fel vannak tüntetve a webhelyek nevei, a hozzájuk tartozó ikonok, valamint az, hogy mennyire gyakran látogatjuk az adott webhelyeket (a webhely gyakoriságát egy színes állapotjelző sáv mutatja, a színt pedig a webhely ikonjából képezték). A sorba rendezett dobozoknak és az aktivitást jelző állapotjelző sávnak köszönhetően nagyon könnyen megtalálhatjuk és elérhetjük azokat a webhelyeket, amelyeket nagy valószínűséggel meg szeretnénk látogatni (19. ábra).



19. ábra: Kedvencek webhelyek aktivitásának feltüntetése

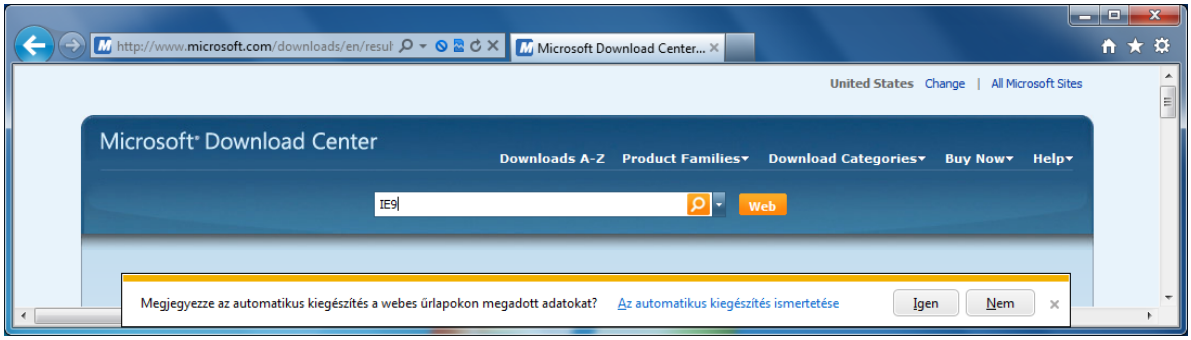
Az utoljára bezárt lapok és az utolsó munkamenet visszaállítása rendkívül felhasználóbarát funkciók. Az utoljára bezárt lapok funkció leginkább akkor lehet hasznos, hogyha véletlenül zárunk

be egy webhelyet, és nem emlékezünk a címére. Természetesen alternatív megoldásként visszakereshetjük a bezárt webhely címét a böngészési előzményekből is, azonban az új lap felületéről indított egy-kattintásos módszer lényegesen egyszerűbb és gyorsabb. Az utolsó munkamenet visszaállítása akkor lehet leginkább nagy segítség, amikor véletlenül záródik be az Internet Explorer, az összes meglátogatott webhellyel együtt. Ezt a funkciót választva újra betölthetjük a korábbi munkamenet során megnyitott weblapjainkat (az éppen aktuálisan megnyitott weblapok mellé).

Az InPrivate-böngészés funkció szintén hasznos lehet, hogy ha az új lapon egy olyan webhelyet szeretnénk megnyitni, ahol nem kívánjuk eltárolni a böngészés előzményeit. Az InPrivate-böngészést választva InPrivate módban elindul egy új példánya az Internet Explorer 9-nek, ami teljesen független lesz az aktuális munkamenettől, és nem tárol semmilyen információt rólunk, illetve a böngészésről.

Értesítési sáv

Manapság számos módja van annak, hogy a böngésző kommunikáljon velünk, felhasználókkal annak érdekében, hogy biztonságosabbá és megbízhatóbbá tegye a böngészést. Az Internet Explorer 9-ben ez a kommunikáció egy teljesen új módon, az értesítési sávon keresztül történik. Annak érdekében, hogy a figyelmünket a lehető legkevésbé zavarja meg böngészés közben, az értesítési sávban megjelenő üzenetek nem feltétlenül igénylik az azonnali beavatkozásunkat, vagyis ideiglenesen akár figyelmen kívül is hagyhatjuk az itt szereplő üzeneteket.

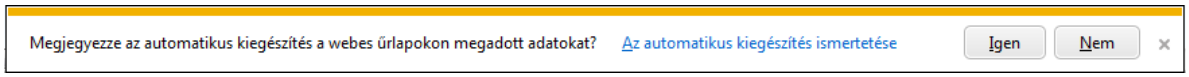


20. ábra: Értesítési sáv

Az Internet Explorer által küldött üzeneteket három nagy csoportba lehet besorolni: léteznek tájékoztató jellegű üzenetek, javaslatok és blokkoló jellegű üzenetek. A három nagy üzenetcsoport közül kettő az értesítési sávon keresztül jelenik meg, amivel nem zavarja a böngészést, és csupán a blokkoló jellegű üzenetek azok, amelyek addig nem engedik folytatni a barangolást, ameddig be nem avatkozunk. A tájékozató jellegű üzenetek olyan üzenetek, amelyek nem igénylik a beavatkozásunkat, csupán információt közölnek arról, hogy valamilyen esemény történt. Ilyen esemény lehet például az, hogy az Internet Explorer sikeresen kiürítette a böngészési előzményeket (21. ábra).

21. ábra: Tájékoztató jellegű üzenetek

A javaslatok olyan üzenetek, amelyekben nem szükséges azonnal döntést hoznunk bizonyos eseményről, hanem későbbre is halaszthatjuk. Ilyen üzenet például az, amikor a böngésző megkérdezi, hogy eltárolja-e a jelszavunkat, vagy bekapcsolja-e az automatikus kiegészítést (22. ábra).



22. ábra: Javaslatok

A blokkoló jellegű üzeneteket olyan események váltják ki, amelyek azonnali beavatkozást igényelnek a felhasználótól. Ilyen üzenet lehet például, amikor egy webhely fennakad a SmartScreen szűrőn és az IE9 figyelmeztet bennünket arra, hogy nem biztonságos tovább folytatni a böngészést az adott webhelyen. A gördülékenyebb böngészés érdekében a korábbi Internet Explorerekben található blokkoló jellegű üzenetek egy részét átalakították tájékoztató jellegű üzenetekké, hogy kevésbé zavarjanak meg bennünket böngészés közben. Például az IE8-as verziójában, amikor bejelentkeztünk egy webhelyre, akkor a böngésző megkérdezte azt, hogy el kívánjuk-e a tárolni a jelszavunkat, és egészen addig nem folytatódhatott a böngészés, ameddig erről nem döntöttünk. Az új értesítési sávnak köszönhetően viszont tovább folytatódik a böngészés anélkül, hogy azonnal be kellene avatkoznunk. A zavartalan böngészést biztosítja továbbá az is, hogy az IE9 alapértelmezés szerint nem ad ki semmilyen hangot, amikor rákattintunk egy hivatkozásra, hiszen más módon is jelzi felénk azt, hogy igen, valóban megtörtént a kattintás (ez a hang visszakapcsolható az *„Eszközök -> Internetbeállítások -> Speciális -> Kisegítő lehetőségek -> Rendszerhangok lejátszása”* menüpontból).

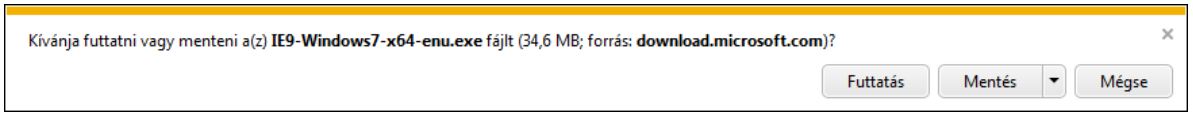
Szintén a zavartalanabb böngészést szolgálja az értesítési sávnak az a tulajdonsága, hogy egyszerre mindig csak a legnagyobb prioritású üzenetet jeleníti meg, hiszen ellenkező esetben elveszítené lényegét azáltal, hogy a felhasználó belezavarodna a sok üzenetbe.

Letöltéskezelő

Az Internet Explorer 9-ben megújult letöltéskezelő több szempontból is egy remek eszköz arra, hogy felhasználóként gyorsabban és biztonságosabban töltsünk le fájlokat a webről és arra, hogy ezeket a letöltéseket később karbantartsuk, rendszerezzük. Az új letöltéskezelés segítségével bármikor felfüggeszthetjük, majd folytathatjuk letöltéseiket, valamint a SmartScreen szűrő használatával megbizonyosodhatunk arról is, hogy a letöltendő fájl ártalmatlan-e vagy sem.

Az IE9 letöltés kezelésének két fő módja van: a már korábban megismert értesítési sávon keresztül biztosítja a letöltésekhez kapcsolódó egyszerűbb műveleteket, és rendelkezik egy különálló letöltés vezérlő felülettel is, az összetettebb, haladóbb műveletek elvégzéséhez.

A modern webes világban két oka lehet annak, hogy valaki letölt egy fájlt az internetről. Letöltheti azért, hogy megtekintse (például egy kisebb dokumentum), vagy azért, hogy eltárolja a számítógépén későbbi felhasználáshoz.



23. ábra: Fájl letöltése

Ha nem csak megtekinteni szeretnénk a letöltendő fájlt, úgy az értesítési sávon keresztül, a *Mentés,* a *Mentés másként* vagy a *Mentés és futtatás / megnyitás* gombok megnyomásával eltárolhatjuk azt. A *Mentés* opciót választva a fájl alapértelmezetten a személyes *Letöltések* mappánkba fog elmentődni, míg a *Mentés másként* opciót választva a számítógép kiválasztott mappájába kerül. A *Mentés és futtatás / megnyitás* opcióval a letöltendő fájl a *Mentés* opcióhoz

hasonlóan, alapértelmezetten *Letöltések* mappánkba tárolódik el, majd letöltés után megnyílik. Mindhárom esetben a letöltött fájlt később megtekinthetjük, illetve megnyithatjuk a letöltésvezérlőn keresztül. Abban az esetben, hogy ha csak megtekinteni szeretnénk a letöltendő fájlt (és csak ideiglenesen szeretnénk letölteni), úgy a *Megnyitás / Futtatás* opciót kell kiválasztanunk. Ilyenkor letöltés után egyből megnyílik a letöltött fájl, viszont nem marad semmilyen nyoma a letöltésvezérlőben, így a későbbiek során nem nyithatjuk meg újból. A *Megnyitás / Futtatás* opciót választva úgy tölthetünk le fájlokat, hogy a letöltés vezérlő nem lesz tele mindenféle ideiglenes fájlokkal, amelyekre a későbbiek során valószínűleg nem lesz szükségünk.

Letöltés közben szintén az értesítési sáv segítségével informálódhatunk arról, hogy hány százaléka töltődött már le a fájlnak, illetve arról is, hogy mennyi idő van még hátra a letöltésből. Szintén az információs sáv segítségével bármikor felfüggeszthetjük, illetve folytathatjuk a letöltéseinket (24. ábra).

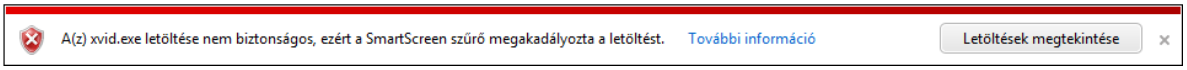
24. ábra: Információs sáv letöltés közben

Sajnálatos módon nem minden letöltött fájl biztonságos. Akad sok olyan is, ami azzal a céllal készült, hogy megfertőzze a számítógépünket és kárt tegyen abban. Számos modern böngészőben találkozhatunk olyan „biztonsági” megoldással, hogy ha a felhasználó futtatható állományt (.exe kiterjesztésű fájlt) tölt le, akkor figyelmezteti, kárt tehet a számítógépében. Azonban ez az információ az esetek többségében félrevezető, hiszen számtalan olyan eset adódhat, amikor futtatható állományt kell letöltenünk a világhálóról (például programok, telepítő fájlok, stb.). Az Internet Explorer 9 ennél sokkal intelligensebb módon figyelmeztet bennünket a veszélyről: ha a letöltendő fájl futtatható állomány, akkor megvizsgálja a meta-adatait, jellemzőit, és amennyiben nincs digitálisan aláírva, akkor figyelmeztetést ad arról, hogy veszélyes lehet.

Ha nincs digitálisan aláírva, valamint nem tartozik a gyakori letöltések közé, akkor a megszokott narancssárga információs sáv átvált piros színűvé, és ezen keresztül figyelmeztet bennünket arra, hogy a letöltendő fájl ártalmas lehet (25. ábra).

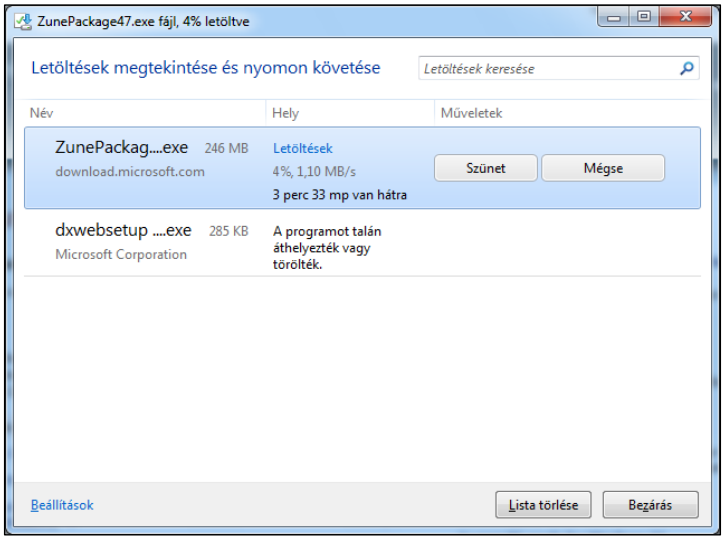
25. ábra: Veszélyes állomány letöltése

Bekapcsolt SmartScreen-szűrővel pedig nem csak figyelmeztet bennünket, hanem egyenesen megakadályozza azoknak a fájloknak a letöltését, amelyek rosszindulatúak. Ebben az esetben nincs mód arra, hogy figyelmen kívül hagyjuk ezt a figyelmeztetést, és mégis letöltsük a fájlt (26. ábra).



26. ábra: Rosszindulatú állomány letöltése

Az információs sáv mellett az új letöltés kezelőfelületen keresztül is elérhetjük, illetve nyomon követhetjük letöltéseinket (27. ábra). A listásan felsorolt letöltésekben bármikor kereshetünk, megtekinthetjük a fájlhoz tartozó legfontosabb információkat (például azt, hogy mekkora a mérete, melyik webhelyről származik vagy azt, hogy hol helyezkedik el a számítógépen), és hasznos műveleteket is végezhetünk rajtuk. Ilyen hasznos művelet lehet az egyesével való eltávolítás (a listából és a számítógépről), a fájl megnyitása, a fájlt tartalmazó mappa megnyitása, a letöltési hivatkozás megnyitása, a biztonsági ellenőrzés megismétlése vagy a fájl bejelentése nem biztonságos állományként.



27. ábra: Letöltéskezelő (Ctrl + J)

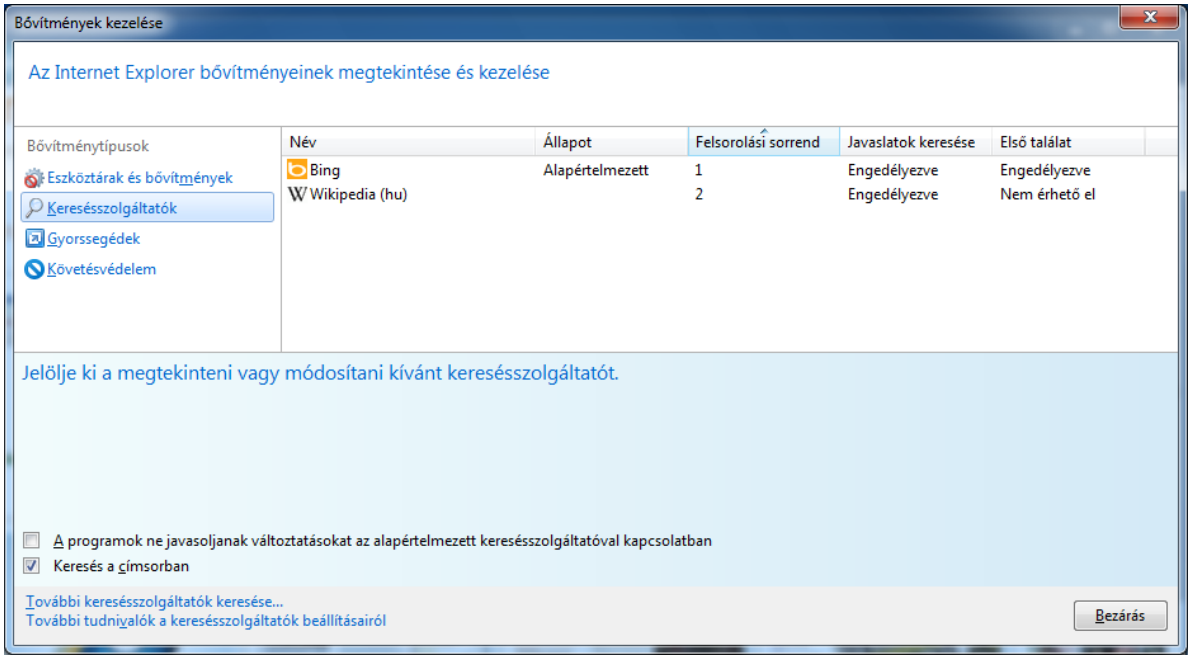
Az új felületen található egy *Beállítások* gomb is, amivel azonnal elérhetjük az IE9 letöltésekre vonatkozó beállításait (alapértelmezett letöltési mappa megváltoztatása, értesítések beállítása). Természetesen a már korábban befejeződött letöltések mellett az éppen aktuális letöltéseket is nyomon követhetjük, hiszen teljesen reális forgatókönyv az, hogy a letöltés közben bezárjuk az értesítési sávot, később mégis informálódni szeretnénk arról, hogy mennyi idő van még hátra a letöltésből.

Bővítménykezelő

A korábbi fejezeteken keresztül megismert bővítményteljesítmény-tanácsadó funkciót az Internet Explorer 9 egy olyan speciális felülettel egészíti ki, amelyen keresztül bármikor informálódhatunk mind az aktuálisan betöltött, mind a rendszerben levő összes, inaktív bővítményeinkről és a hozzájuk tartozó olyan fontos információkról, mint például a bővítmény gyártója, verziója vagy betöltési sebessége. A bővítménykezelő felületen keresztül azonban nem csak informálódhatunk, hanem karban is tarthatjuk a telepített eszköztárainkat, keresésszolgáltatóinkat, gyorssegédeinket, valamint a követésvédelmi listáinkat is. (28. ábra)

Az *Eszköztárak és bővítmények* szekcióban lehet megtekinteni, letiltani illetve engedélyezni az olyan eszköztárakat, mint például a *LinkedIn Toolbar*, az *MSN Toolbar*, valamint újakat letölteni. Fontos tudni, hogy elsősorban ezek azok a bővítmények, amelyek leginkább lelassíthatják a böngészést, illetve a böngésző működését, így óvatosan kell bánnunk velük.

A felület *Keresésszolgáltatók* részében lehet megtekinteni azokat a böngésző kiegészítőket, amelyek segítségével keresni tudunk a címsoron keresztül. Ebben a részben lehet be, illetve kikapcsolni azt, hogy a címsorba gépelésnél az IE9 megjelenítse-e a keresési javaslatokat, vagy egyáltalán keressen-e bármilyen formában. Természetesen lehetőséget biztosít arra is, hogy eltávolíthassuk az egyes keresésszolgáltatókat, beállítsuk az alapértelmezett keresési szolgáltatót, vagy újakat töltsünk le az internetről.



28. ábra: Bővítménykezelő

Az Internet Explorer 8-as verziójából ismerhető gyorssegédek (gyorssegéd-szolgáltatók) segítik a weblapon található szövegekkel végzett munkát, így lehetőséget adnak például a weblapon található címek térképen való megtekintésére, szavak definíciójának vagy fordításának megtekintésére és további szolgáltatások elérésére. Ezeknek a kiegészítőknek a karbantartása szintén bővítménykezelő felületen történik, méghozzá a *Gyorssegédek* részében, ahol engedélyezhetjük, letilthatjuk, illetve törölhetjük telepített gyorssegédeinket.

A bővítménykezelő *Követésvédelem* részében tekinthetjük meg, állíthatjuk be, tilthatjuk le illetve távolíthatjuk el az IE9 követésvédelmi funkciója által használt listákat. Ezeknek a listáknak a tartalmát, funkcióját később, a biztonsági funkciók részben fejtjük ki.

1.3. Windows 7 együttműködés

A 2009 októberében debütáló Windows 7 rendszer számos új és felhasználóbarát funkcióval örvendeztette meg felhasználóit. Az új felhasználói felület amellett, hogy a korábbi verziókhoz képest szebb megjelenést kapott, számos olyan funkcióval is rendelkezik, amelyek a számítógéppel végzett mindennapi munkát hivatottak megkönnyíteni és felgyorsítani. Ezeknek a funkcióknak a sorából kiemelendő – legalább is az Internet Explorer nézőpontjából – a Windows 7-ben megújult tálca és a *Snap* funkció, amelyek az operációs rendszer megjelenése óta hihetetlen népszerűségnek örvendenek.

Az új tálcáról elérhető ugrólisták (Jump List menük) segítségével a felhasználók gyorsan és egyszerűen érhetik el a leggyakrabban használt dokumentumaikat, képeiket és zenéiket. A tálca rögzítés funkcióját kihasználva pedig a felhasználók játszi könnyedséggel rögzíthetik kedvenc alkalmazásaikat és dokumentumaikat a tálcára, a megnyitott ablakok miniatűr előképeivel pedig hihetetlen módon leegyszerűsödik a megnyitott alkalmazások áttekinthetősége (főleg sok ablak megnyitása mellett). A Windows 7 Snap funkciója azoknak a felhasználóknak lehet rendkívül fontos, akik tipikusan sok megnyitott ablakkal dolgoznak egyszerre, de mégsem rendelkeznek több monitoros megjelenítéssel. A funkció mögött az az ötlet áll, hogy egyszerűen fel lehessen osztani a képernyőt két egyenlő részre, úgy, hogy benne két ablak párhuzamosan elférjen egymás mellett, ezzel is megkönnyítve a munkavégzést.

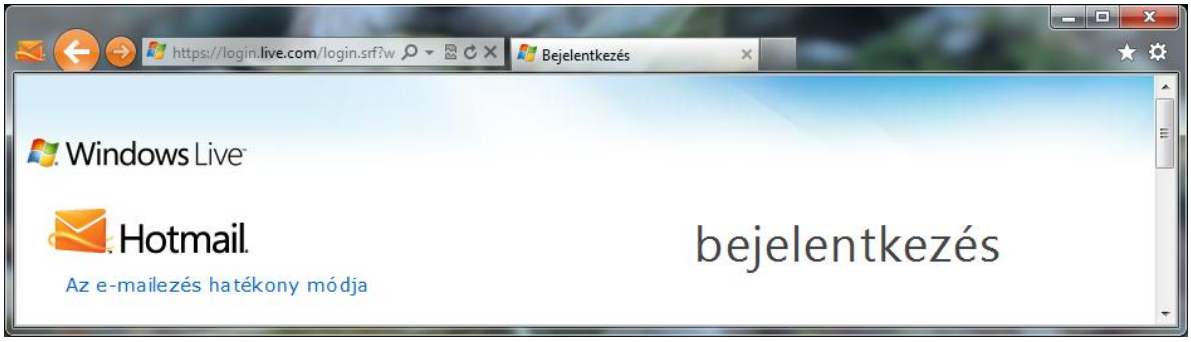
Az Internet Explorer 9 igyekszik a lehető legteljesebb mértékben kihasználni a fent említett Windows 7 szolgáltatásokat annak érdekében, hogy úgy kezelhessük kedvenc webes alkalmazásainkat, webhelyeinket, mint hogyha azok a megszokott asztali alkalmazásaink lennének (29. ábra). Amellett, hogy lényegesen egyszerűbbé és gyorsabbá teszi a webhelyek elérését abban is segít, hogy hatékonyabban és élvezetesebben használjuk ki a világhálón való szörfözéssel töltött időnket.

29. ábra: Tálcán rögzített webhelyek

És hogy ez mit is jelent valójában? IE9 felhasználóként a hagyományos alkalmazásokhoz hasonlóan kedvenc webhelyeinkhez parancsikonokat készíthetünk, amelyeket a tálcán vagy Start menüben rögzíthetünk. Ezeknek a parancsikonoknak a hagyományos alkalmazásokhoz hasonlóan lehet egyedik ikonjuk, rendelkezhetnek saját, egyedi ugrólistákkal, valamint a webhely miniatűr előképének megjelenése is testreszabható. Például ha a kedvenc levelező portálunk fel van készítve az Internet Explorer 9 szolgáltatásaira, és rögzítjük ezt a webhelyet a tálcán, akkor később úgy tekinthetjük meg kimenő, illetve bejövő leveleinket, hogy nincs szükségünk arra, hogy előzetesen megnyissuk a böngészőt, majd azon belül keresgéljük a különböző menüpontokat. Csupán a webhelyhez tartozó ikon ugrólistájából kell kiválasztanunk a kívánt „mappát", és az Internet Explorer 9 meg fogja nyitni nekünk azt a weblapot, ami az adott mappát tartalmazza. A webhelyek miniatűr előnézeti képein megjelenő vezérlők testreszabásából származó előnyöket pedig az olyan webhelyek rögzítésekor élvezhetjük ki igazán, amelyek rendelkeznek multimédiás lejátszó felülettel (például videó lejátszóval), mert ilyenkor a lejátszó vezérelhető akár az előnézeti képből is anélkül, hogy a webhely ablakát ténylegesen aktívvá kellett volna tennünk (pl. Windows Media Player).

Rögzített webhelyek

A rögzített webhelyek funkcióval kedvenc webhelyeinket elérhetjük közvetlenül a Windows tálcánkról, illetve a Start menünkből anélkül, hogy előbb meg kellene nyitnunk a böngészőt. A webhely rögzítésével a böngésző helyett a webhely kerül a felhasználói felület középpontjába. Amikor a tálcáról vagy a Start menüből nyitunk meg egy webhelyet, akkor a böngészőkeret, a böngésző kezelőszervei felveszik a webhely ikonját, illetve annak elsődleges színét, így a böngésző a webhelyre szabottan jelenik meg (30. ábra).



30. ábra: Webhely-központú böngészőkeret és kezelőszervek

A webhelyek tálcán, illetve Start menüben való rögzítésének folyamata nagyon egyszerű és többféleképpen is elvégezhető. Rögzíthetjük a böngésző címsorának elején levő, a webhelyhez tartozó ikon vagy a webhelyet megjelenítő lap fejlécének segítségével is. Mindkét esetben csupán annyi a teendőnk, hogy az ikont, illetve a lap fejlécét rá kell húznunk a Windows tálcára és máris rögzült a webhely. A Start menüben való rögzítése sem bonyolultabb, csupán rá kell kattintanunk az *„Eszközök -> Fájl -> Weblap felvétele a Start menübe"* menüpontra. A rögzítés feloldásához

pedig – a hagyományos asztali alkalmazások ikonjához hasonlóan – jobb egér gombbal rá kell kattintatnunk az ikonra, majd ki kell választanunk *„A program rögzítésének feloldása a tálcán”* menüpontot.

Ugrólisták

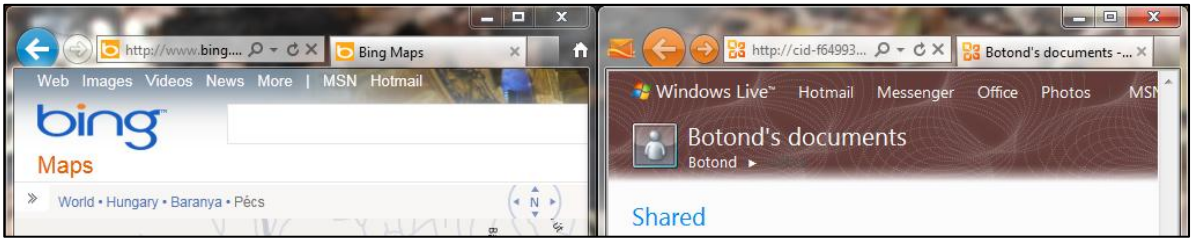
Amennyiben kedvenc webhelyünket felkészítették a Windows 7 együttműködésre, akkor a webhelyet a Windows 7 tálcán, illetve a Start menübe rögzítve úgy érhetjük el a webhelyet, és webhely szolgáltatásait, mint hogyha egy hagyományos asztali alkalmazás volna. A webhelyhez tartozó ugrólista segítségével anélkül érhetjük el a webhely szolgáltatásait, hogy először el kellene navigálunk magára a webhelyre, majd onnan az adott szolgáltatásra. Így a kedvenc webhelyünk számunkra érdekes szolgáltatásait egy kattintással elérhetjük, amivel rendkívül sok időt takaríthatunk meg magunknak. Az ugrólisták tartalmazhatnak csoportosításokat, előzményeket, vagy egyéb alkalmazáshasználatot könnyítő funkciókat is.

31. ábra: Ugrólisták

Windows 7 Snap funkció

Az Internet Explorer 9 számos *Új lap megnyitása* funkciója közül talán a leghasznosabb és legérdekesebb az, hogy a lapokat ötvözhetjük a Windows 7 Snap funkciójával. Már korábban láthattuk azt, hogy böngészés során a felhasználók többsége (és talán mi magunk is) egyszerre több lapot, illetve webhelyet használ egyszerre. Ilyen forgatókönyv például az, ha meg kell terveznünk a másnapi hegyi túra útvonalát, ám az ehhez szükséges információk a beérkezett leveleink között találhatóak. Ebben az esetben meg kell nyitnunk mind az útvonaltervezéshez szükséges webhelyet, mind pedig azt a webhelyet, ahol a levelezésünket elérhetjük. Egy hagyományos böngészővel csak nehézkesen tudjuk mindkét webhely tartalmát egyszerre megtekinteni, hiszen be kell zárni kézzel a második lapot, meg kell nyitni egy újabb példányát a böngészőnek, majd az új böngészőablakban újra meg kell nyitni a bezárt lapot. Az Internet Explorer 9 és a Windows Snap funkció párosításával

ezt a műveletet sokkal egyszerűbben elvégezhetjük: nem kell mást tenni, mint az egyik lap fejlécét a képernyő egyik oldalára húzni, a másik lap fejlécét a másik oldalára.



32. ábra: Windows 7 Snap funkció

1.4. Biztonsági funkciók

A modernkori, 21. századi számítógépünk rendszeres használat mellett számos veszélynek és fenyegetésnek van kitéve. A különböző vírusoktól, férgektől az adathalász eszközökön át számtalan rosszindulatú szoftver igyekszik beférkőzni számítógépeinkbe azért, hogy később átvegye felette az irányírást, bosszúságot okozzon nekünk vagy ellopja személyes adatainkat. Ezeknek a rosszindulatú szoftvereknek vagy támadásoknak a legnagyobb része az internetet és a böngészőnket igyekszik felhasználni arra, hogy közelebb kerüljön hozzánk.

Az Internet Explorer család biztonságossága rendkívül megosztó, hiszen sokan a legveszélyesebb böngészőnek tartják, míg mások a teljes mértékben megbíznak benne. A 2001-ben debütáló Internet Explorer 6-os verziója a világ legnépszerűbb és legelterjedtebb böngészője lett, és még ma, tíz évvel a megjelenését követően is nagy népszerűségnek örvend. Az akkori webes piac egyeduralkodó szereplőjének tekinthető böngésző azonban nem csak a hagyományos felhasználók körében örvendett nagy sikernek, hanem az internetes bűnözők körében is. Az internetes bűnözők pedig nagyon kreatív módokon igyekeztek kihasználni az akkor innovatívnak számító böngészőben rejlő hibákat. Ezekből a biztonsági hibákból, résekből és hiányosságokból adódó kellemetlenségeknek sajnos minden esetben mi, felhasználók láttuk kárát. Természetesen az IE6-ot igyekeztek szorgosan javítgatni és foltozgatni, amitől idővel ugyan biztonságosabb és stabilabb lett, azonban az igazi megoldás az Internet Explorer 7-es verziója hozta, amely 2006-ban jelent meg.

Az IE7-et mérföldkőnek tekinthetjük az Internet Explorerek családjában, hiszem számos olyan újítást, funkciót rejtett magában, ami megbízhatóvá és biztonságossá tette a böngészést, és amely képességek fényében eltörölhette az IE6 hírnevét. A következő generációs Internet Explorerek pedig tovább folytatták a biztonságra és megbízhatóságra épülő tervezést oly annyira, hogy az IE8-ban debütáló SmartScreen technológia 2009. márciusi megjelenése óta több mint egymilliárd támadást védett ki, illetve hiúsított meg.

Természetesen az Internet Explorer 9-es változata sem marad el biztonság téren elődjeitől, hiszen az oly sikeresnek bizonyult SmartScreen technológia tovább fejlesztéséve, az ActiveX-szűréssel, a követésvédelemmel, a helyközi, parancsfájlt alkalmazó támadások elleni szűrőjével és az InPrivate-böngészés funkciójával méltán nevezhető napjaink legbiztonságosabb böngészőjének.

SmartScreen-szűrő

A SmartScreen-szűrő segítségével az Internet Explorer 9 egyaránt segít megóvni személyes adatainkat és számítógépünket az interneten leselkedő fenyegetésektől és veszélyektől. Az adathalászati eszközök egy olyan „népszerű” eszközök a modern internetes bűnözők körében, amelyek a megtévesztés erejével próbálják megszerezni, ellopni személyes adatainkat. Ez a folyamat az esetek többségében úgy történik, hogy kapunk egy e-mail üzenetet, amely látszólag valós webhelyről, banktól vagy közösségi oldalról származik, és arra kér bennünket, hogy kattintsunk rá az e-mailben szereplő hivatkozásra majd végezzünk el valamilyen műveletet (például erősítsük meg e-mail címünket, változtassuk meg lejárt jelszavunkat vagy tekintsünk meg valamilyen értesítést). Amikor rákattintunk a levélben található hivatkozásra, akkor a böngésző egy olyan oldalra továbbít bennünket, amely látszólag teljes mértékben megegyezik az adott oldal bejelentkezési felületével, de valójában egy teljesen más kiszolgálón található oldalt nyitottunk meg. Itt bejelentkezés után továbbít bennünket a valós webhelyre, és látszólag minden a megszokott ütemben történik. Nagy valószínűséggel észre sem vettük azt, hogy valójában nem a kívánt webhelyet látogattuk meg, és azt, hogy a háttérben valaki ellopta a személyes adatainkat (e-mail cím, bejelentkezési név, jelszó). Az ilyen támadások leginkább olyan webhelyek nevében történnek, mint például az elektronikus bankok, az eBay vagy PayPal, hiszen ha valakinek sikerül megszereznie ezekhez a webhelyekhez a belépési adatainkat, akkor ténylegesen is hozzáférhet a pénzünkhöz. Azonban egyre gyakrabban érkeznek hamis e-mailek az olyan népszerű közösségi oldalakról is, mint például a facebook vagy a twitter, hiszen ezek a webhelyek nagy népszerűségnek örvendenek, és minél nagyobb az oldal népszerűsége, annál nagyobb valószínűséggel akadnak olyan felhasználók, akik bedőlnek a megtévesztésnek.

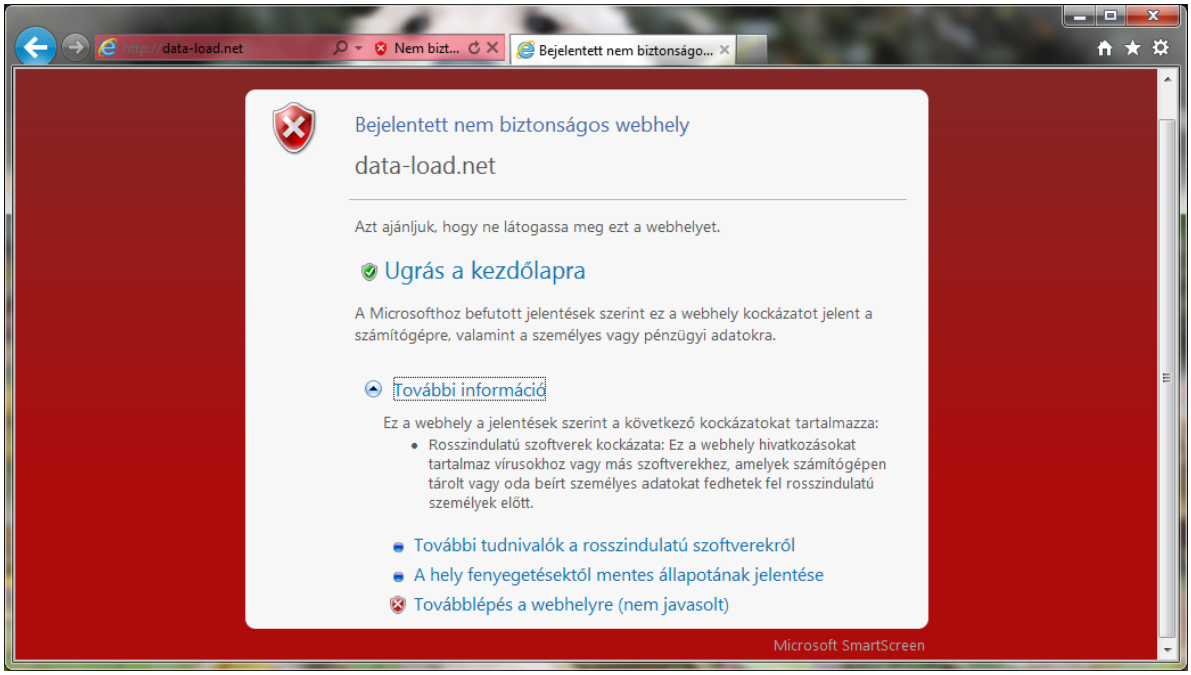
Az csalásnak azért estünk áldozatává, mert a kapott e-mail és az ál-webhely is teljes mértékben hasonlított az eredeti e-mail formátumra, illetve webhelyre. Régebben erre az átverésre csupán akkor jöhettünk rá, ha közelebbről is megvizsgáltuk annak a hivatkozásnak a címét, amire kattintottunk vagy feltűnt az, hogy a böngésző címsorában szereplő cím valójában nem az, amit megszoktunk. Ha a támadó azonban még ügyesebb, akkor olyan címet is adhat az ál-weboldalának, mint például a *http:// 84.0.107.230/login.facebook.com*. Ebben az esetben a cím is nagyon megtévesztő lehet, és nem informatikusként nagy valószínűséggel fel sem fog tűnni nekünk az, hogy a fenti webhely címe valójában a *http://84.0.107.230* kiszolgálóra mutat, és nem a *login.facebook.com* címre (ebben az esetben a login.facebook.com csupán egy mappa vagy webhely az ál-weboldal kiszolgálóján).

Az Internet Explorer 9 SmartScreen-szűrőjének segítségével kiszűrhetjük a fenti ál-webhelyeket. Ha meglátogatunk egy ilyen ál-webhelyet, akkor az IE9 figyelmeztet bennünket, hogy minden bizonnyal megtévesztés áldozatai vagyunk és nem biztonságos az adott webhely további böngészése.

A SmartScreen-szűrő többféle módon is segíti azt, hogy még véletlenül se legyünk valamilyen trükk áldozatai. A felkeresendő webhely címét először összeveti egy, a számítógépünkön tárolt címlistával, ami azoknak a gyakori webhelyeknek a címét tartalmazza, amelyek az IE csapata szerint biztonságosak. Ha a webhely nem szerepel ebben a listában, akkor összeveti egy olyan folyamatosan frissülő online listával, amely a kártékony, illetve támadó webhelyek címeit

tartalmazza. A személyes adataink védelmének érdekében a webhely címe (valamint IP címünk és SmartScreen verziónk) biztonságos SSL csatornán keresztül továbbítódik a Microsoft felé, és az IE adatvédelmi szabályzata szerint csupán arra szolgál, hogy hatékonyabbá tegyék a SmartScreen funkciót.

Ha a meglátogatandó webhely szerepel a veszélyesnek tartott webhelyek listájában, akkor a webhely helyett egy piros, figyelmeztető lapot jelenít meg az IE9, ahol ismerteti velünk azt, hogy veszélyes lehet a böngészés folytatása és lap megnyitása (33. ábra). Amennyiben biztosak vagyunk abban, hogy a webhely mégis biztonságos, akkor figyelmen kívül hagyhatjuk ezt a figyelmeztetést és tovább folytathatjuk a böngészést. Sőt, lehetőségünk van arra is, hogy bejelentsük a webhelyet biztonságos webhelyként is. A veszélyes webhelyek listája ugyan folyamatosan frissül, azonban mégis adódhatnak olyan veszélyes webhelyek, amelyeket még nem tartalmaz a lista. Ebben az esetben lehetőségünk van bejelenteni az adott webhelyet, így később nem fog veszélyt jelent sem ránk, sem felhasználótársainkra.

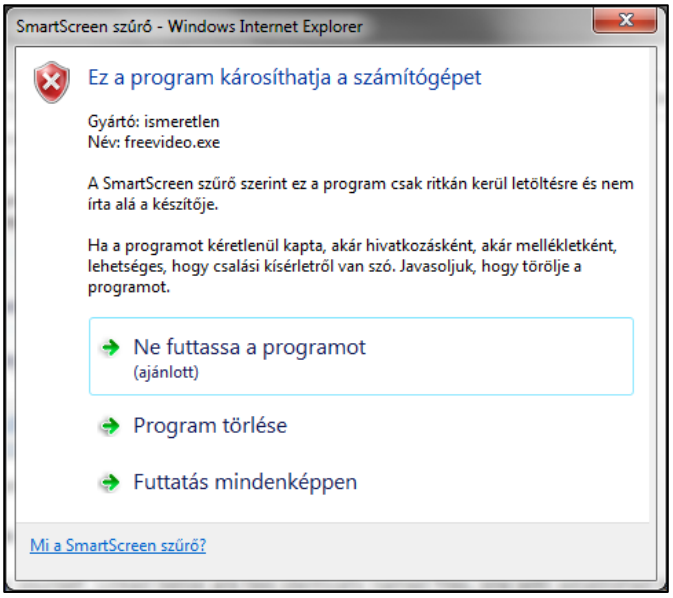


33. ábra: Nem biztonságos webhely megjelenítése

A SmartScreen technológiával felvértezett Internet Explorer 9 nem csupán az ál-webhelyek kiszűrésével és az adathalászat megakadályozásával próbálja biztonságosabbá tenni számunkra a böngészést. Hiszen nem csak személyes adatainak vannak veszélyben böngészés közben, hanem a számítógépünk is.

A vírusok, férgek, kémprogramok és társaik manapság leginkább a böngészőnket felhasználva telepednek meg a gépünkön, mégpedig azáltal, hogy letöltünk egy fájlt (többnyire „ingyenes” zene vagy videó letöltő oldalakról vagy illegális szoftvereket kínáló webhelyekről). Egy-egy ilyen

rosszindulatú fájlt álcázhatnak akár képként, zeneszámként vagy dokumentumként is, ám minden esetben valójában egy olyan alkalmazást töltünk le, amely később megtelepszik a számítógépünkön, és vagy sikerül kiirtanunk, vagy nem. A modern böngészők többsége képes figyelmeztetni a felhasználót, hogy ha alkalmazást tölt le, akkor a letöltés veszélyforrást hordoz magában. Ám ahogy azt már korábban említettem, ezeket a figyelmeztetéseket az esetek többségében figyelmen kívül kell hagynunk, hiszen nagy valószínűséggel szándékosan töltjük le az adott fájlt, és tisztában vagyunk vele, hogy biztonságos. Éppen ezért, ha véletlenül mégis rosszindulatú alkalmazást töltünk le, akkor a megszokás miatt automatikusan figyelmen kívül hagyjuk ezt az üzenetet. Az Internet Explorer 9 esetében viszont teljesen más a helyzet: csak akkor figyelmeztet bennünket, ha valóban fenyegetést jelent a letöltendő fájl.



34. ábra: Gyanús alkalmazás futtatása

A letöltött fájl vizsgálata szintén a SmartScreen technológia segítségével történik, mégpedig a fájl „hírnevét” felhasználva. Első hallásra furcsának tűnhet az, hogy hogyan lehet egy alkalmazásnak, fájlnak hírneve, viszont ha közelebbről szemügyre vesszük, az interneten található állományokkal kapcsolatban számos dolgot megvizsgálhatunk. Megvizsgálhatjuk azt, hogy az alkalmazás rendelkezik-e digitális aláírással, ismert-e a gyártója vagy azt, hogy hányan töltötték már le a világhálóról, stb.

Az IE9 SmartScreen szűrője a fenti információk alapján dönti el, hogy a letöltendő alkalmazás biztonságos-e vagy sem. Amikor letöltünk egy futtatható állományt a számítógépünkre, akkor az IE9 megvizsgálja, hogy rendelkezik-e digitális aláírással, és ha nem, akkor összeveti két online adatbázissal is. Először egy olyan adatbázissal, amely a rosszindulatú fájlokat tartalmazza, és ha az általunk letöltött fájl szerepel ebben az adatbázisban, akkor az automatikusan törlődik a számítógépről. Kapunk egy figyelmeztetést, amiben az áll, hogy biztonsági okokból le lett tiltva az elérése. Amennyiben nem szerepel ebben az adatbázisban, úgy összeveti a második adatbázissal is,

ami az Internet Explorer felhasználói által letöltött fájlokat tartalmazza. Ennek alapján megvizsgálja azt, hogy a fájl milyen gyakran lett letöltve más felhasználók által. Amennyiben nem tartozik a gyakori letöltések körébe, akkor egy olyan figyelmeztetést kapunk, amiben az áll, hogy a letöltött fájl veszélyes lehet (viszont nem ebben az esetben nem törlődik automatikusan).

Természetesen nem minden ritkán letöltött fájl veszélyes. A figyelmezetést csupán azért kapjuk, mert azok a fájlok, amelyeket ritkábban töltöttek le, nagyobb biztonsági kockázatot rejtenek magukban.

ActiveX-szűrő

Manapság a legtöbb weboldal annak érdekében, hogy interaktív tartalmat vagy videót jelenítsen meg, olyan bővítményeket használ fel, mint például a SilverLight vagy a Flash. Az ActiveX pedig azt a technológiát jelenti, amin keresztül ezek a bővítmények képesek az Internet Explorerben futni és működni. Maga a technológia még 1996-ban jelent meg azzal a fő céllal, hogy különböző alkalmazások szabványos interfészen keresztül közöljenek információt egymással, és az Internet Explorer 6 megjelenésével vált nagyon népszerűvé. Az internet népszerűségének terjedésével pedig egyre inkább hozzájárult a webes felhasználói élmény fokozásához.

Azonban nem csak előnyökkel jár az ActiveX vezérlők használata, hanem hátrányuk is van, hiszen ezek a bővítmények mind egy külső gyártótól származó, az Internet Explorertől „független" alkalmazások. A legtöbb esetben a legnagyobb bosszúságot az okozza, hogy az ActiveX vezérlők által futtatott kis programok nagymértékben lelassítják a böngészőt, valamint veszélyeztethetik a személyes adataink biztonságát is.

Az Internet Explorer 9 ActiveX-szűrőjének segítségével saját magunk dönthetünk arról, hogy melyek azok a webhelyek, ahol valóban meg kívánjuk tekinteni az ActiveX tartalmakat (például videó megosztó portál), és melyek azok, ahol nem (például számos reklámmal ellátott webhely).

Az ActiveX-szűrő alapértelmezés szerint kikapcsolt állapotban van, annak érdekében, hogy kedvenc webhelyeink a megszokott formában jelenjenek meg, és azért, hogy egy adott webhely böngészési élménye a webhelyek készítőin múljon, ne az Internet Exploreren. Azonban ha teljes mértékben az irányításunk alatt szeretnénk tartani a böngészőnk által megjelenített tartalmakat, bármikor bekapcsolhatjuk az *„Eszközök -> Biztonság -> ActiveX-szűrés"* menüpont alatt.

Bekapcsolás után az összes webhelyen automatikusan letiltódnak az ActiveX tartalmak. Ha egy olyan webhelyet látogatunk meg, amely ActiveX tartalmakat szeretne megjeleníteni, zavaró felugró ablakok helyett a már korábban bemutatott címsoron keresztül értesülhetünk arról, hogy a webhely bizonyos része fennakadt a szűrőn. A címsorban helyet foglaló állapotjelző ikonnak a segítségével azonban bizonyos webhelyeket hozzáadhatunk a kivételek listához, így ezeken a webhelyeken megjelennek az ActiveX tartalmak.

Követésvédelem

Mint ahogy azt már korábban olvashattuk, a webes technológiák fejlődésének számos pozitív, illetve negatív hatása van. Szórakozhatunk a weben, intézhetjük üzleti ügyeinket, illetve munkánkat és hozzáférhetünk egy elképzelhetetlenül nagy információs adatbázishoz, amiből szinten mindent

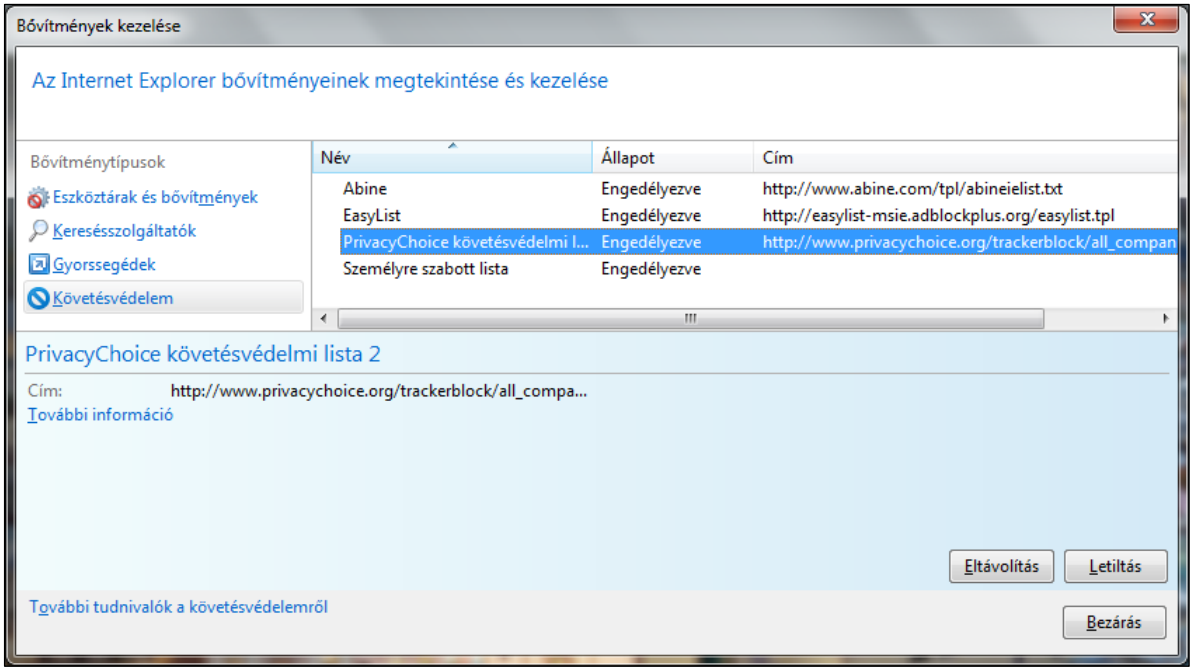
megtudhatunk. Eközben számos veszélynek is ki vagyunk téve, az egyre nagyobb számú online bűnözők arra törekednek, hogy hozzáférjenek személyes adatainkhoz és számítógépünkhöz. Azonban a negatív és a pozitív véglet között olyan semleges felhasználási réteget találhatunk, mint például a jóindulatú adathalászat. A jóindulatú adathalászat lényege az, hogy bizonyos szervezetek összegyűjtsék és feldolgozzák a böngészési szokásainkat, statisztikákat készítsenek, majd a statisztikák alapján célzott formában reklámokat juttathassanak el felénk. A web ilyen irányú felhasználására komoly üzleti vállalkozások jöttek létre, amelyek nyereségüket azzal szerzik, hogy reklámokat küldenek nekünk.

Természetesen a reklámok eljuttatása a felhasználókhoz azon túl, hogy zavaró lehet, önmagában még nem egy elítélendő, rossz tevékenység. Az eszköz, ahogy ezt megteszik azonban némiképp aggályos: tudtunkon kívül és beleegyezésünk nélkül nem csak a böngészési szokásainkat figyeli, hanem bizonyos személyes adatainkat el is tárolhatják, majd később felhasználhatják.

Mindemellett manapság a webes fejlesztések körében egyre elterjedtebbé vált, hogy az oldalakat külső forrásokkal, tartalommal lássák el (pl. reklámblokkok, külső webhelyről származó képek és videók beágyazása a weboldalakba, google analytics szolgáltatás). Ez a trend amellett, hogy gyakorlati szempontból kényelmes (hiszen bizonyos adatokat, tartalmakat nem a saját webkiszolgálóikon kell tárolniuk), adatvédelmi szempontból megkérdőjelezhető: lehetőséget nyújt arra, hogy a weboldal megtekintésekor egy külső szervezet adatokat szerezzen be rólunk, amiből természetesen mit sem veszünk észre.

Az Internet Explorer 9 követésvédelmi funkciója azzal a fő céllal készült, hogy a fent említett adatvédelmi réseket befoltozza. Egy olyan új biztonsági funkcióról van szó, aminek segítségével beavatkozhatunk abba, hogy honnan és milyen tartalmat tekintsünk meg: egy adott weboldalra navigálva kiszűrhetjük az idegen helyről származó tartalmakat (például a reklámokat, vagy a szokásait elemző külső programrészeket). Valójában azzal, hogy teljes mértékig irányításunk alatt tarthatjuk azt, hogy milyen tartalmat jelenítünk meg, a böngészés rendkívül biztonságossá válik (és a biztonság mellett még az idegesítő reklámoktól is megszabadulhatunk).

Az IE9 követésvédelmi funkciójának működése úgynevezett követési listákon alapul. Ezek a követésvédelmi listák tartalmazzák azoknak a webkiszolgálóknak a címeit, amelyekről engedélyezett, illetve tiltott a tartalmak letöltése (35. ábra). A követésvédelmi listákat többnyire olyan szervezetek készítik, amelyeknek fő célkitűzése a személyes adataink védelme. Ilyen szervezet például a *PrivacyChoice* is, aminek a weboldaláról letölthetünk egy olyan listát, amivel részlegesen, illetve teljesen blokkolhatjuk a legnagyobb és legnépszerűbb felhasználói szokásokat tároló és elemző webcímeket, szervezeteket. További követésvédelmi listákat pedig a *http://ieaddons.com/hu* címről tudunk letölteni. A letöltött listákat az IE9 megadott időközönként automatikusan frissíti, hogy mindig napra kész legyen.



35. ábra: Követésvédelmi listák karbantartása

A követésvédelem funkció aktiválása (*„Eszközök -> Biztonság -> Követésvédelem”* menüpont) után az egyes webhelyek letöltése közben az Internet Explorer megvizsgálja a webhely tartalmát, és olyan elemeket keres, amelyek valamilyen külső webhelyen tárolt adatokra mutatnak (például Flash animációk, képek, parancsfájlok). Ha talál ilyen elemet, akkor összeveti a hivatkozott kiszolgáló címét az általunk telepített követésvédelmi listákban szereplő címekkel. Ha a külső hivatkozásnak a címe szerepel a tiltott címek listájában, akkor a webhely az a része nem fog letöltődni, illetve megjelenni és a címsorban aktiválódik az az ikon, amit azt jelzi, hogy a webhely bizonyos része fennakadt a szűrőn.

Az ActiveX szűrőhöz hasonlóan, a követésvédelem esetében is beállíthatjuk azt, hogy bizonyos webhelyek kivételt képezzenek a szűrés alól, így ezeknek a webhelyeknek a tartalma teljes mértékben megjelenjen.

Szűrő helyközi, parancsfájlt alkalmazó támadások ellen

A helyközi, parancsfájlt alkalmazó támadás (XSS) egy speciális módja annak, hogy rosszindulatú támadóink megszerezzék személyes adatainkat. Az ilyen támadások a kedvenc webhelyeink biztonsági réseit kihasználva érik el a céljukat azáltal, hogy olyan parancsfájlokat szúrnak be a webhely tartalmába, amelyek eltárolják és továbbítják személyes adatainkat a támadó felé. Ebben az esetben nem megtévesztésről van szó, hiszen valós az általunk felkeresett webhely, csupán rosszindulatú parancsfájlokat tartalmaz.

Képzeljük el azt, hogy az internetet böngészve rátalálunk egy olyan webhelyre, amely egy érdekesnek ígérkező cikket tartalmaz, ami tegyük fel, hogy egy új közösségi oldalról szól. A cikk alján pedig szerepel egy olyan hivatkozás, ami arra hívja fel figyelmünket, hogy kattintsunk rá, ha ki szeretnénk próbálni. A hivatkozást megnyitva látszólag nem történik semmilyen érdekes dolog, hiszen valóban annak a webhelye nyílik meg, amit ki szeretnénk próbálni (éppen ezért a SmartScreen szűrő sem fog figyelmeztetni semmire). Ami viszont elkerülte a figyelmünket az az, hogy a hivatkozás nem csupán a webhely címét tartalmazta, hanem még sok minden mást. Ennek a „sok minden másnak" a segítségével elhelyeztek egy olyan parancsfájlt az általunk megjelenített webhelyen, ami továbbítja a bevitt adatainkat a támadó felé, aki máris megszerezte a bejelentkezéshez szükséges adatainkat. Természetesen a fenti példánál sokkalta kifinomultabb módszereket és trükköket használnak támadóink, viszont nagyon leegyszerűsítve így zajlik egy helyközi, parancsfájlt alkalmazó támadás.

Az Internet Explorer 9 helyközi, parancsfájlt alkalmazó támadások elleni szűrőjével biztosak lehetünk abban, hogy nem leszünk a fentiekhez hasonló támadások áldozatai sem. Személyes adataink védelmének érdekében ez a szűrő alapértelmezés szerint mindig aktív, és nem engedélyezi futtatni a nem biztonságos parancsfájlokat, ezzel megakadályozva a támadást.

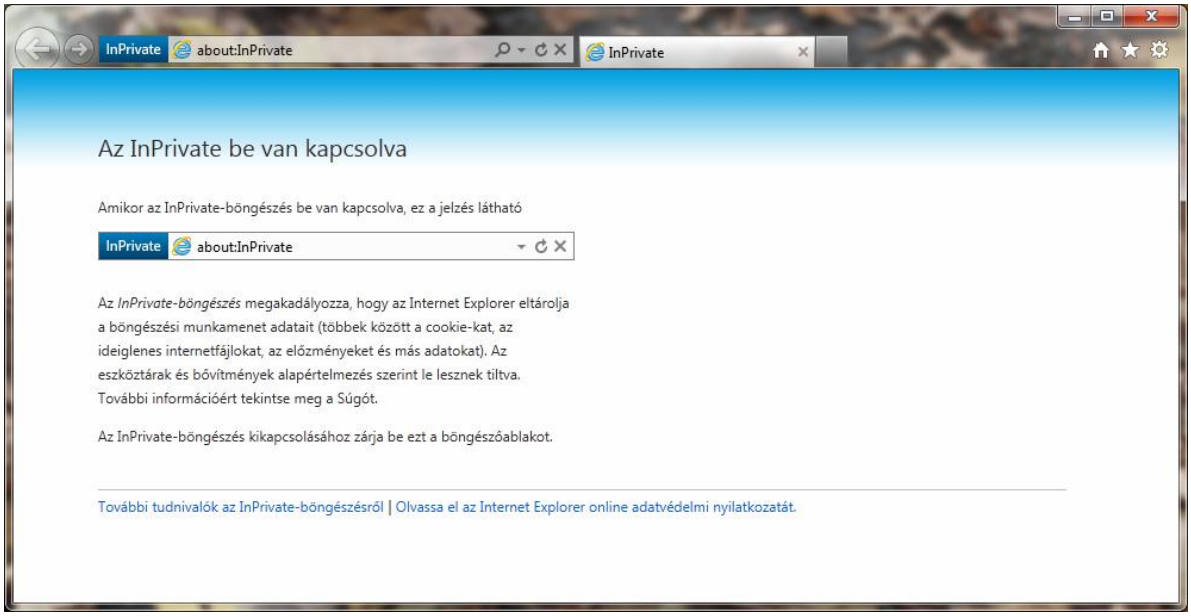
Akadhatnak azonban olyan webes szolgáltatások is, amelyek ugyan biztonságosak, mégsem képesek együttműködni a szűrővel. Ha ilyet tapasztalunk, minden esetben győződjünk meg arról, hogy valóban biztonságos-e a webhely (például azáltal, hogy felvesszük a kapcsolatot a webhely üzemeltetőivel), és nincs veszély, akkor kapcsoljuk ki ideiglenesen a szűrőt, majd ha megtekintettük a webhelyet, kapcsoljuk vissza. A szűrőt ki, illetve bekapcsolni az Internet Explorer *„Eszközök -> Internetbeállítások -> Biztonság fül -> Egyéni szint -> Parancsfájlkezelés -> XSS-szűrő engedélyezése"* menüpontból lehet.

InPrivate-böngészés

A modern böngészők egyik legfontosabb újdonsága az anonim vagy privát böngészési üzemmód, aminek segítségével eltüntethetünk magunk mögött minden nyomot, már ami a böngészési munkamenetünket illeti. Ebben az üzemmódban a böngészők csak minimális mennyiségű adatot tárolnak a munkamenetről, és kilépés után még ezt a kevés adatot is felejtik.

Nem kivétel ez alól az Internet Explorer 9 sem, hiszen rendelkezik egy speciális, InPrivate-böngészésnek nevezett üzemmóddal, amelynek célja az anonim böngészés biztosítása. Az InPrivate üzemmód segít megakadályozni, hogy a számítógép többi felhasználója megtekintse a böngészési előzményeinket. Például ha munkahelyünkön olyan webhelyeket tekintünk meg, aminek nem örülnének feletteseink, vagy ajándékot keresünk szeretteinknek, akkor böngészés végét követően kézzel kell megkeresnünk a releváns előzményeket, majd törölnünk azokat. Az InPrivate-böngészés segítségével többek között ezt a körülményes műveletet is leegyszerűsíthetjük. Csak úgy, mint a böngészési előzmények törlésénél, az InPrivate módban meglátogatott webhelyek eltűnnek a böngészési előzményekből, a címsorból és a webhelyek beviteli mezőnek tartalmát sem fogja megjegyezni az automatikus kiegészítés szolgáltatás.

Az InPrivate-böngészési módot több helyről is elindíthatjuk: a tálcán rögzített IE9 ikon ugrólistájának segítségével, az új lap megnyitásának felületéről valamint az *„Eszközök -> Biztonság -> InPrivate-böngészés”* menüpontból is.



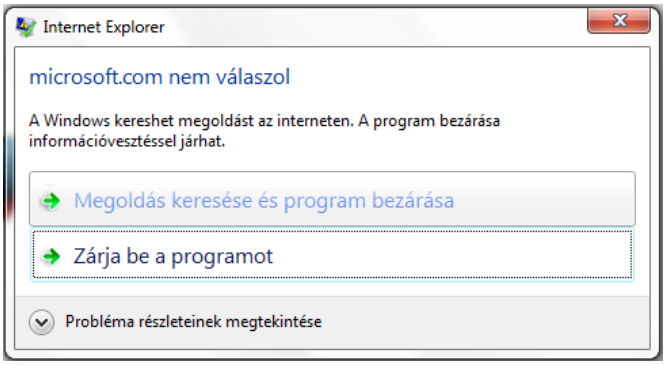
36. ábra: InPrivate-böngészés

Elindításakor az Internet Explorer megnyit egy új ablakot (36. ábra), és az anonim böngészés csak erre az újonnan megnyitott ablakra, és az új ablakban megnyitott valamennyi lapra lesz érvényes. Egy másik böngészőablak megnyitásakor azonban az újonnan megnyitott ablakra az InPrivate üzemmód által nyújtott anonimitás és védelem már nem lesz érvényes.

Azonban nem csak valós, élő emberek kíváncsiskodhatnak a böngészési szokásaink után, hanem – mint ahogy azt már korábban is olvashattuk – az általunk meglátogatott webhelyek is. Ezek a webhelyek eltárolják azt, hogy náluk jártunk, majd felhasználják vagy a segítségünkre, vagy ellenünk. A követésvédelmi és ActiveX-szűréssel karöltve biztosak lehetünk abban, hogy böngészési előzményeinkben valóban nem kutakodhat sem élő ember, sem egy külső webhely, szolgáltatás.

Lapok elszigetelése és helyreállítása

Azt már láthattuk, hogy a modern böngészők egyik legalapvetőbb tulajdonsága, hogy támogatják a többlapos böngészést. De mi a helyzet abban az esetben, ha egyszerre több megnyitott lappal dolgozunk, és az egyik lap lefagy? Akkor elveszítettük az összes többi megnyitott lapunkat is a munkánkkal együtt? Az Internet Explorer 9 használata esetén a válasz: nem.



37. ábra: Műveletek lefagyott lap esetén

Az új Internet Explorer 9 kifinomult lapkezelési technológiájának köszönhetően a megnyitott lapok egymástól elszigetelten és függetlenül jelenítik meg az adott webhely tartalmát, így ha az egyik lefagyna, akkor nem veszik el a teljes munkamenet, a többi megnyitott lap maradéktalanul képes megjeleníteni saját tartalmát. Ha egy weblap több mint 10 másodpercig nem válaszol, akkor az IE9 figyelmeztet bennünk erre és lehetőséget ad a lap azonnali bezárására és visszaállítására (38. ábra).

38. ábra: Lefagyott weblap figyelmeztetés

Amennyiben az Internet Explorernek nem sikerülne visszaállítania a lapot, és „erőszakkal” kell bezárnunk a böngészőt, úgy sincs minden elveszve, hiszen a böngésző újraindítását követően, az új lap felületről újranyithatjuk a legutóbbi munkamenetünket, amelynek visszakapjuk a hiba előtti állapotát.

2. fejezet: Az Internet Explorer 9 rendszerüzemeltetői szemmel

Az Internet Explorer böngészők vállalati alkalmazása már az internet térnyerésének hajnalán is nagy szépszerűségnek örvendett, főleg az Internet Explorer 6 megjelenésével. Ez a népszerűség azóta is töretlen maradt, mindamellett, hogy a többi böngészőgyártók is felzárkóztak a böngésző-ipar kiélezett versenyébe. Sőt valójában a felhasználók körében egyre inkább népszerűek kezdtek lenni a nem Internet Explorer alapú böngészők, a vállalati szférában azonban mi sem változott. Olyannyira nem módosult a vállalatok hozzáállása, hogy a tizenegy évvel ezelőtt megjelent IE6 piaci részesedése még mindig rendkívül nagy, 11,6%. Egyébként a Microsoft már kampányt is indított *„The Internet Explorer 6 Countdown”* néven az IE6 visszaszorítására.

Nem véletlen, hogy ekkora sikernek örvendenek a különböző Internet Explorerek vállalati környezetben, hiszen mások a prioritások vállalati mint felhasználói szemmel. A külcsín és a használhatóságnál sokkal fontosabb a központi felügyelet, és az, hogy a vállalat rendszergazdái távolról képesek legyen beállítani és módosítani a böngésző tulajdonságait. És ez az oka is, amiért a többi böngésző nem képes felvenni a versenyt e téren az Internet Explorer családdal, hiszen sem a Mozilla Firefox, sem a Google Chrome nem rendelkezik olyan eszközzel, amellyel központilag menedzselhető lenne (maximum külső fejlesztők által készített komponensekkel valósítható meg a központi felügyeletet).

2.1. Kilenc ok, amiért az IE9 a legjobb böngésző üzleti felhasználók számára

Bevezetésként egy kisebb összefoglaló keretében ismerkedjünk meg az Internet Explorer 9 azon általános jellemzőivel, amitől a legjobb választás lehet az üzleti felhasználók számára.

Teljesítményközpontú kialakítás

Az Internet Explorer 9 már a kezdetektől fogja azzal a céllal készült, hogy a világ leggyorsabb böngészője legyen. Ezt a tényt olyan valós webes statisztikák igazolják, amelyek minden felhasználó számára lényegesek. Az IE9 JavaScript motorja, a Chakra a népszerű WebKit SunSpider teljesítményteszt szerint a jelenlegi leggyorsabb JavaScript motor, így gyorsabb böngészéssel örvendeztetheti meg a felhasználókat. Bár a böngészés szempontjából az oldalakon elhelyezett parancsfájlok teljesítménye lényeges, ez csak egyike azoknak a teljesítményt befolyásoló tényezőknek, amiket az IE9 fejlesztésekor szem előtt tartottak.

Hardveres gyorsítás

Az Internet Explorer 9 az egyetlen hardveres gyorsítást alkalmazó Windows alapú böngésző, amely a számítógép egész teljesítményét kihasználja. A grafikus processzor (GPU) révén a böngészőben megjelenő valamennyi grafikus elem, videó és szöveg megjelenítését hardveres gyorsítás segíti. A hardveres gyorsítás hatékonyságának kihasználásával a vállalatok olyan webes alkalmazásokat készíthetnek, amelyek működésüket tekintve közelebb állnak a natív Windows-alkalmazásokhoz, mivel a CPU és GPU együttes használatával a nagyobb számításigényű feladatok ellátása sem megy a teljesítmény kárára.

Webhelyközpontú kialakítás

A felhasználók azért indítják el a böngészőt, hogy hozzáférhessenek kedvenc webhelyeikhez vagy webes alkalmazásaikhoz. Ennek érdekében a böngészőt új, letisztult kezelőfelülettel látták el, amely a fő hangsúlyt a webhelyekre teszi. Az Internet Explorer 9 áramvonalasabb kialakításának révén a böngészőablak valamennyi böngészőnél nagyobb képernyőfelülettel áll a felhasználók rendelkezésére. A felhasználók értesítésének módja is megújult és letisztultabbá vált, aminek köszönhetően kevésbé zavarják a böngészést. A felhasználóbarátabb megjelenésnek és a kevesebb zavaró tényezőnek köszönhetően a vállalat alkalmazottai hatékonyabban használhatják fel a böngészésre fordított időt.

A Windows szerves része

Az új böngészőben a kedvenc webes alkalmazások és webhelyek hasonlóképp működhetnek, mint a számítógép natív alkalmazásai. Az Internet Explorer 9 hatékonyan és gördülékenyen működik együtt a Windows 7 rendszerrel, így a Windows 7-ben megszokott vezérlőelemekkel, például Ugrólista menükkel, gyors méretezéssel (Snap) fokozhatja a hatékonyságot. Munkatársaik rögzíthetik kedvenc webhelyeiket a Windows 7 tálcáján, azon keresztül pedig értesítéseket kaphatnak, és megtekinthetik munkájuk, illetve személyes webhelyeik és alkalmazásaik miniatűr képét.

Biztonságosabb böngészés

Az IE9 az Internet Explorer 8 kiváló és díjnyertes biztonsági funkcióira épült. Az IE9 olyan jól bevált technikákkal és funkciókkal teszi biztonságosabbá a böngészést és nyújt védelmet a vállalkozásokat érintő “drive-by” típusú és a felhasználók megtévesztésére épülő támadásokkal, valamint a meglátogatott webhelyeket fenyegető kockázatokkal szemben, mint például a SmartScreen szűrő, a védett üzemmód, az adatvégrehajtás megakadályozása és az XSS (cross site scripting) szűrő. Mostanában pedig a felhasználók megtévesztését kihasználó kártevők jelentik az egyik legnagyobb biztonsági fenyegetést. Az IE9 rendelkezik jelenleg a legmagasabb blokkolási aránnyal a böngészők közül a SmartScreen technológiának köszönhetően. A technológia az Internet Explorer 8-ban jelent meg 2009 márciusában, és azóta több mint másfél milliárd kártékony programot blokkolt. A kártékony programokkal való fertőzés valószínűségének csökkenésével csökken a kényszerű

leállások száma, és csökken a kártevők támadásai utáni helyreállítások költsége is. Ha a vállalkozás az adatvédelemre is kiemelt figyelmet fordít, az Internet Explorer 9 követésvédelem funkciójával szabályozhatja a böngésző kommunikációját a külső webhelyekkel, így megvédheti bizalmas adatait az online nyomkövető programokkal szemben.

Könnyen bevezethető

Az Internet Explorer 9 olyan jól bevált és támogatott bevezetési lehetőségekkel rendelkezik, amelyek más böngészőkben nem találhatóak meg, és amelyek lehetővé teszik a gyors és egyszerű szervezeti bevezetést. Az IE9 a Windows Update, a WSUS szolgáltatás, a System Center Configuration Manager, csoportházirend és hálózati mappa segítségével is beállítható, sőt a már meglévő Windows-lemezképekbe is gyorsan beépíthető. Az Internet Explorer Administration Kit (IEAK) segítségével percek alatt olyan testreszabott IE9 telepítőcsomagok készíthetőek, amelyek megkönnyítik és leegyszerűsítik a böngésző saját, céges verziójának telepítését.

Átfogó felügyeleti eszköz

Az IE9 átfogó felügyeleti eszközöket kínál mind a kis-, mind a nagyvállalatok számára. A nagyobb felhasználók több mint 1500 csoportházirendet alkalmazhatnak bármilyen igény szerinti konfiguráció érvényesítésére, ami lehetővé teszi az IE9 beállításainak teljeskörű, részletes szabályozását. Az egyszerű szabályozást igénylő vállalkozások számára olyan eszközök állnak rendelkezésre, mint például a Security Compilance Manager vagy az IEAK “Easy to Deploy” (Könnyen telepíthető) szakasza, amellyel az adott környezetre szabott, teljes körű biztonsági és felügyeleti konfigurációk alakíthatók ki.

Kompatibilitás és az áttérés támogatása

Az Internet Explorer az egyetlen olyan böngésző, amely hivatalosan is támogatja a korábbi böngésző- és dokumentum-üzemmódokkal való kompatibilitást és többek között még az IE6 egyes régi funkcióinak emulálására is képes. Az IE9-ben a felhasználók a megfelelő korábbi kompatibilitási üzemmód kiválasztásával könnyedén megjeleníthetik a kívánt webhelyeket, a rendszergazdák pedig a csoportházirendek segítségével programozottan ültethetnek át webhelyeket kompatibilitási üzemmódokba. Az összes böngésző közül az IE biztosítja a leghosszabb támogatási életciklust, így a vállalat egyéni ütemezése szerint frissíthető újabb verziókra.

Modern webes szabványok támogatása

A modern webes alkalmazások a legújabb webes szabványok támogatását igénylik. Az Internet Explorer 9 minden eddiginél több szabványnak felel meg, többek között a HTML5, a CSS3, az SVG 1.1 és az ECMAScript5 szabványoknak. A W3C-vel és a többi szabványügyi testületekkel való szoros együttműködés révén az IE9 aktív szerepet tölt be a modern webes szabványok fejlesztésében és piacra kerülésében. Az IE9 kiváló szabványtámogatása azt eredményezi, hogy a modern webes szabványok szerint kifejlesztett webhelyek az ezeket a szabványokat támogató más böngészőkkel is

működni fognak. A HTML5 és az IE9 együttesen ideális alapot biztosítanak a Windows rendszerre történő webfejlesztéshez, ma és a jövőben egyaránt.

2.2. Internet Explorer Administration Kit (IEAK)

Az Internet Explorer Administration Kit (IEAK) segítségével leegyszerűsíthetjük a vállalat igényeire szabott Internet Explorerek készítését, telepítését és felügyeletét. Egy testreszabott böngészővel fokozhatjuk a termelékenységet, a kezelhetőséget, a megbízhatóságot és gondoskodhatunk arról is, hogy a vállalat valamennyi böngészője azonos szolgáltatásokkal bírjon.

Egyszerűen beállíthatjuk azt, hogy

- hogyan kívánjuk terjeszteni az Internet Explorert (terméket tartalmazó CD, illetve DVD segítségével vagy hálózaton keresztül)
- telepítés közben milyen szinten igényeljen felhasználói beavatkozást
- mi legyen az alapértelmezett kezdőlap
- milyen keresési szolgáltatókkal bírjon
- hova irányítsa a felhasználót alkalmazás támogatásért
- milyen webhelyek szerepeljenek a Kedvencek, a Hírcsatornák és a Hivatkozások között
- milyen ActiveX vezérlőket futtathat az Internet Explorer

Az IEAK telepítésekor ki kell választanunk azt, hogy milyen szerepkörben kívánjuk személyre szabni az Internet Explorert: internet szolgáltatóként (ISP), internetes tartalomszolgáltatóként (ICP) vagy vállalati rendszergazdaként. Internetszolgáltatóként a felhasználóinknak, tartalomszolgáltatóként az általunk készített szoftvert vagy hardvert felhasználóknak, vállalati rendszergazdaként pedig a vállalat munkatársainak készíthetünk testreszabott Internet Explorer csomagokat. Az eltérő szerepkörök eltérő beállításokat tesznek lehetővé (2. táblázat).

Funkció	Internetszolgáltató	Tartalomszolgáltató	Vállalati rendszergazda
Egyedi bővítmények	Igen	Igen	Igen
Testreszabott címsor	Igen	Igen	Igen
Kedvencek	Egy mappa tetszőleges hivatkozással	Egy mappa tetszőleges hivatkozással	Tetszőleges számú mappa tetszőleges számú hivatkozással
Online támogatási hivatkozás	Igen	Igen	Igen
Webszeletek	Maximum 5	Maximum 5	Tetszőleges számú
Gyorssegédek	Megkötésekkel	Megkötésekkel	Tetszőleges számú, bármilyen típusú gyorssegéd
Kezdőlapok	Maximum 3	Maximum 3	Tetszőleges számú

Első indítás varázsló	Nem távolítható el	Nem távolítható el	Eltávolítható
Hírcsatornák	Egy mappa tetszőleges hivatkozással	Egy mappa tetszőleges hivatkozással	Tetszőleges számú mappa tetszőleges számú hivatkozással
Böngészőbeállítások személyreszabása	Nem	Nem	Igen
Biztonsági és adatvédelmi beállítások	Nem	Nem	Igen
Vállalati beállítások	Nem	Nem	Igen
Kompatibilitási nézet	Igen	Igen	Igen
Egyedi kapcsolat beállítások	Igen	Nem	Igen

2. táblázat: Szerepkörök és beállítási lehetőségek

A megfelelő szerepkör kiválasztása és a telepítést követően a személyreszabás egyszerűen, egy varázsló segítségével történik. A varázslóban ki kell választanunk a cél operációs rendszert és a platformot, az Internet Explorer nyelvét majd azt, hogy hogyan kívánjuk eljuttatni a telepítőt a felhasználóknak és azt, hogy milyen személyreszabásokat kívánunk elvégezni.

A tesztreszabások elvégzését követően a varázsló elkészíti a vállalatunk igényeihez optimalizált telepítőcsomagot, amit már csak el kell juttatnunk felhasználóinkhoz.

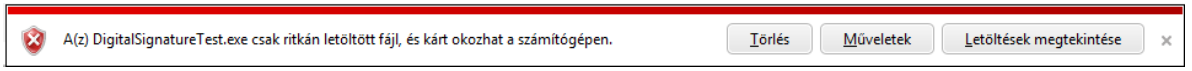
2.3. Alkalmazásunk ellátása digitális aláírással

Mint ahogy azt már a korábbi fejezetekben olvashattuk, az Internet Explorer 9 SmartScreen szűrője megvizsgálja a felhasználók által letöltött futtatható állományokat, és ha azok nem rendelkeznek megfelelő digitális aláírással, akkor figyelmeztet arra, hogy nem biztonságos a letöltés.

Mindamellett, hogy a SmartScreen szűrőnek e funkciója megóvja a felhasználók számítógépét a káros és veszélyes alkalmazásoktól, bennünket is arra ösztönöz, hogy az interneten közzétett alkalmazásainkat ellássuk digitális aláírással (amennyiben ezt még nem tettük volna meg).

A digitális aláíráshoz két dologra van szükségünk: a Microsoft .NET Framework SDK-ra, valamint egy digitális tanúsítványra, ami tartalmazza a vállalatunk nevét, címét, email címét, a tanúsítvány lejáratának dátumát és egy speciális aláírást. Megbízható tanúsítványt azonban nem készíthetünk saját magunk, hiszen akkor elveszítené lényegét: egy tanúsítványokkal foglalkozó szervezettől kell igényelnünk, az eljárás azonban nem ingyenes. Természetesen mi magunk is készíthetünk tanúsítványt (amit meg is fogunk tenni), viszont ennek a tanúsítványnak az eredetét nem fogja tudni ellenőrizni az IE9 SmartScreen-szűrője.

A következőkben meg fogjuk nézni azt, hogyan lássuk el digitális aláírással alkalmazásainkat annak érdekében, hogy ne akadjon fent a fájl a SmartScreen szűrőn. Első lépésként vegyünk egy hagyományos futtatható állományt, amit feltöltünk a világhálóra. Amikor ezt a fájl le szeretnénk tölteni, akkor a már jól megismert biztonsági figyelmeztetés fog fogadni bennünket: a fájl csak ritkán letöltött fájl és kárt okozhat a számítógépen (39. ábra).



39. ábra: Biztonsági figyelmeztetés digitális aláírással nem rendelkező fájl letöltésekor

Második lépésben készíteni fogunk magunknak egy teszt tanúsítványt. Ehhez meg kell nyitnunk a parancssort (cmd.exe), méghozzá abban a mappában, ahova Framework SDK-t telepítettük (például *C:\Program Files (x86)\Microsoft SDKs\Windows\v7.0A\Bin*). A parancssorba be kell gépelni azt az utasítást, amivel a tanúsítványt létrehozzuk:

```
# makecert.exe -sv <PrivateKeyName>.pvk -n "CN=<CompanyName>" <CertificateName.cer>
# Példa:
makecert.exe -sv C:\work\Test.pvk -n "CN=Almat-keresek Kft." C:\work\Test.cer
```

Ha a **test.pvk** fájl még nem létezik, akkor meg kell adni a tanúsítványhoz tartozó jelszót. A folyamat végén pedig lesz egy **.cer** és egy **.pvk** kiterjesztésű fájlunk. Ez után a következő lépés a tanúsítvány átkonvertálása SPC (Software Publisher Certificate) formátumra.

```
# cert2spc.exe <CertificateName>.cer <CertificateName>.spc
# Példa:
cert2spc.exe C:\work\Test.cer C:\work\Test.spc
```

Ebben a lépésben szükségünk lesz az előzőleg megadott jelszóra, amit remélhetőleg még nem felejtettünk el☺. A továbbiakban, a fájl aláírásához csak a **.pvk** és a **.spc** fájlokra van szükségünk. A következő lépés ennek a két fájlnak az egyesítése egyetlen PFX fájllá.

```
# pvk2pfx.exe
#    -pvk <CertificateName>.pvk -pi <password>
#    -spc <CertificateName>.spc
#    -pfx <CertificateName>.pfx -po <password>
# Példa:
pvk2pfx.exe
       -pvk C:\work\test.pvk -pi alma
       -spc C:\work\test.spc
       -pfx C:\work\test.pfx -po alma
```

Ha ez első lépésekben nem adtunk meg jelszót a tanúsítványhoz, akkor ebben a lépésben mindenképp meg kell adni legalább a PFX fájlhoz tartozó jelszót, különben az aláírás nem fog megfelelően működni, sőt leginkább sehogy sem fog. Ha ezzel a lépéssel is végeztünk, akkor végre hozzákezdhetünk a fájl digitális aláírásához.

```
#signtool.exe sign
#      /f <CertificateName>.pfx /p <password>
#      /t <TimestampURL>
#      /v "<File>"
```

A TimestampURL paraméter egy „időbélyegző szerver” címét várja, és bármelyik lehet az alábbiak közül:

- http://timestamp.verisign.com/scripts/timstamp.dll
- http://timestamp.globalsign.com/scripts/timstamp.dll
- http://timestamp.comodoca.com/authenticode

```
signtool.exe sign
       /f C:\work\test.pfx /p alma
       /t http://timestamp.verisign.com/scripts/timstamp.dll
       /v "C:\work\DigitalSignatureTest2.exe"
```

Ha ezt, a már digitális aláírt fájlt is feltöltjük a világhálóra, akkor már nem a megszokott biztonsági figyelmeztetéssel fogunk találkozni, hanem azzal, hogy nem sikerült ellenőrizni a fájl gyártóját (40. ábra). Azonban ha egy valódi tanúsítvánnyal végeztük volna el a műveletet, akkor semmilyen üzenetet nem kapnánk letöltéskor.

40. ábra: Biztonsági figyelmeztetés érvénytelen digitális aláírással rendelkező fájl letöltésekor

2.4. Egyedi keresési szolgáltató létrehozása

Az Internet Explorer 9 címsorába beépülő keresésszolgáltatók segítségével – mint ahogy azt már korábban is láthattuk – a felhasználók a népszerű webes keresőkön felül elérhetik az olyan webhelyek keresési szolgáltatását is, amelyek nem a világhálón keresnek, hanem saját adatbázisukban (például Wikipédia). Ez a funkció azon túl, hogy felhasználói szemmel hasznos és szórakoztató, vállalati szemmel nézve ennél sokkal több. Lehetőséget ad arra, hogy meglévő ügyfeleink, felhasználóink egyenesen a saját adatbázisunkban keressenek, amivel csökkenthetik a felesleges kattintások számát és így időt spórolhatnak meg, amitől nőhet az ügyfél-elégedettség. Emellett egy rendkívül hatékony promóciós eszköz lehet, hiszen kvázi „státusz szimbólumként” is felfogató, ha rendelkezünk saját keresésszolgáltatóval.

Mielőtt elkészítenénk saját keresésszolgáltatónkat, készítenünk kell egy olyan XML fájlt (OpenSearch Description), amely leírja a keresési szolgáltatásunk tulajdonságait.

```
<?xml version="1.0" encoding="UTF-8"?>
<OpenSearchDescription
    xmlns=http://a9.com/-/spec/opensearch/1.1/
    xmlns:ie="http://schemas.microsoft.com/Search/2008/">
    <ShortName>Saját keresésem</ShortName>
    <Image
        height="16"
        width="16"
        type="image/icon">
        http://pelda.hu/example.ico
```

```
    </Image>
    <Url
        type="text/html"
        template="http://pelda.com/search.aspx?q={searchTerms}&amp;source=IE"/>
    <Url
       type="application/x-suggestions+json"
       template="http://pelda.example.com/search.aspx?q={searchTerms}"/>
    <Url
        type="application/x-suggestions+xml"
        template="http://pelda.example.com/search.aspx?q={searchTerms}"/>
</OpenSearchDescription>
```

Minden OSD fájlnak tartalmaznia kell legalább a keresésszolgáltató nevét (**ShortName**), valamint a keresési címet (**Url type="text/html"**). A keresés címében szereplő **{searchTerms}** rész lesz helyettesítve azzal a szöveggel, amit a felhasználó begépel.

Annak érdekében, hogy az elkészített keresésszolgáltató bővítményünket a felhasználók használni tudják, készítenünk kell egy olyan linket, amire kattintva fel tudják venni a saját bővítményeik közé.

```
<a href="#"
    onclick="window.external.AddSearchProvider('http://www.pelda.com/provider.xml')">
    Keresésszolgáltató felvétele</a>
```

Ha végeztünk a gyorssegéd beállításával, és a hivatkozás kihelyezésével, nincs más dolgunk, mint felkészíteni a webhelyünket arra, hogy visszaadja a keresési javaslatokat. Mint ahogy az OSD fájlból is láthatjuk, kétféle formában is visszaadhatjuk a javaslatok listáját: a JavaScriptben divatos JSON formátumban és XML formátumban is. A JSON formátumba visszaadott eredményeket csak szövegesen tudja megjeleníteni a böngésző, míg az XML formátumban visszakapott eredményeket grafikusan is. Az alábbi XML dokumentum azt példázza, hogy milyen felépítésűnek kell lennie annak az XML fájlnak, amit meg tud jeleníteni az Internet Explorer.

```
<?xml version="1.0"?>
<SearchSuggestion xmlns ="http://schemas.microsoft.com/Search/2008/suggestions">
<Query>időjárás pécs</Query>
<Section>
    <Separator title="Grafikus javaslataim"/>
    <Item>
        <Text>Pécs, Magyarország 25° - Napos idő</Text>
        <Description>Délelött enyhén felhős, délután napsütéses idő várható</Description>
        <Image source=http://pelda.com/sunny.jpg
               alt="Napos idő" width="75" height="75"/>
        <Url>http://pelda.com/idojaras.aspx?v="Pecs"</Url>
    </Item>
    <Separator title="Szöveges javaslataim"/>
    <Item>
       <Text>Xbox 360</Text>
    </Item>
    <Item>
       <Text>Pécs, 3 napos előrejelzés</Text>
       <Description>Három napos előrejelzés</Description>
```

```
      <Url>http://pelda.com/elorejelzes.aspx?v=pecs&p=3</Url>
    </Item>
    <Item>
      <Text>Pécs, 15 napos előrejelzés</Text>
      <Description>Tizenöt napos előrejelzés</Description>
      <Url>http://pelda.com/elorejelzes.aspx?v=pecs&p=15</Url>
    </Item>
    <Separator />
      <Item>
            <Text>Pécs időjárás…</Text>
      </Item>
      <Item>
            <Text>Előrejelzések..</Text>
      </Item>
    </Section>
</SearchSuggestion>
```

Egyedi keresésszolgáltatók készítése saját kereső szolgáltatásunkhoz nem igényel nagy többlet energiát, és mégis nagyban segíthet az ügyfeleink igényeinek maximális kielégítésében. A ráadás pedig az, hogy nem szükséges a többi böngészőre is külön-külön elkészíteni a keresésszolgáltatót, hiszen az OpenSearch Description formátum szabványos, így a többi böngészővel is képes hatékonyan együttmükődni (kisebb módosításokkal).

2.5. Egyedi követésvédelmi lista készítése

Az első fejezetben megismerkedhettünk többek között az Internet Explorer 9 követésvédelmi funkciójával és láthattuk azt, hogy milyen hatékony eszköz lehet személyes adataink védelmére és a nem kívánt tartalmak kiszűrésére. Azonban nem csak otthoni környezetben, hagyományos felhasználóként tud rendkívül hasznos eszköz lenni, hanem vállalati környezetben is. Gondoljunk csak bele abba, hogy a vállalat alkalmazottainak mindennapi munkájához elengedhetetlen lehet az internet használata (főleg, ha a szellemi tőke felhasználásával termelnek profitot, például IT cégek). Korlátozhatjuk ugyan azt, hogy milyen webhelyeket tekinthetnek meg, azonban sokkal célszerűbb, ha azt korlátozzuk, hogy melyek legyenek azok, amelyeket nem (tipikusan ilyen például a facebook, ami nagyon időrabló tud lenni ☺), hiszen ha a munka során kutakodni kell az interneten, akkor képtelenség számon tartani az összes potenciális webhelyet, amire szükség lehet. Ha tehát adott az, hogy a munkatársaink a weboldalak 99,99%-át megtekinthetik, akkor gondoskodnunk kell arról, hogy a szükséges információk megszerzése során semmilyen bizalmas adat ne kerüljön ki a vállalat számítógépeiről vagy munkatársairól. De abba is belegondolhatunk, hogy böngészés során egy-egy webhely letöltéséhez felhasznált adatmennyiség nagy része valóban a tényleges webhely megjelenítéséhez szükséges, azonban a fennmaradó részt a webhelyeken található reklámok és egyéb külső tartalmak teszik ki. Ez ugyan kis adatmennyiség, de nem elhanyagolható, hiszen ha megszorozzuk ennek az adatnak a méretét a vállalat számítógépeink számával, akkor az eredmény megdöbbentő lehet. És ha a vállalat internet sávszélessége nem végtelen (már pedig nem szokott

az lenni), akkor a követésvédelmi listákkal és az ActiveX-szűréssel lényegesen csökkenthetjük a felesleges adatforgalmat.

Már számos szervezet készített követésvédelmi listákat, amelyeket bármikor letölthetünk a világhálóról, azonban ezek a követésvédelmi listák nem blokkolják a hazai webhelyeken megjelenő reklámok többségét. Éppen ezért kiegészíthetjük a letöltött követésvédelmi listákat a saját, vállalat igényeire szabott listával. Ehhez nem kell mást tenni, mint készíteni kell egy egyszerű szöveges fájlt és azt el kell helyezni a vállalat intranetes webhelyén, amit később, a távoli konfigurációs eszközzel automatikusan hozzáadhatunk a vállalat számítógépein található Internet Explorerekhez.

A TPL (követésvédelmi lista) fájl felépítése a következő:

```
msFilterList
# Az első sor a fejléc, ez jelenti azt, hogy követésvédelmi listáról van szó
# A ’#’ karakterrel kezdődő sorok pedig a megjegyzések, ezeket figyelmen kívül hagyja a
# böngésző.
:Expires=5
# Ez a sor jelenti azt, hogy milyen időközönként frissítse az IE9 a listát.
# Esetünkben például 5 naponta.

+ vállalatom.com
# Ezzel a sorral engedélyezzük azt, hogy a vállalatom.com domainról minden tartalom
# megjeleníthető

- példa.com
# Ezzal a sorral tilthatjuk az összes példa.com domainra mutató külső tartalmat

+ pédla.com
- tracking.példa.com
# A fenti szabállyal engedélyeztük az összes példa.com helyre mutató tartalmat,
# kivéve azt, amelyik a tracking aldomainre mutat

-d példa.com file
-d példa.com file.html
-d példa.com html
-d tracking.példa
# A fenti szabályokkal letiltjuk az összes olyan tartalmat, ami a példa.com helyre mutat,
# és a hivatkozás URL-je tartalmazza a ’file’ vagy ’html’ vagy ’file.html’ szövegrészt.
# A többi tartalom engedélyezett, csupán a tracking.példa.com-ról származó bármilyen
# tartalom tiltott.

-d p*a
# A fenti szabállyal az összes olyan tartalmat blokkoljuk, amely illik az adott
# szövegrészre. Például a ’példa.com’ vagy ’pálmafa.com’

# A blokkoló szabályokhoz hasonlóan működnek az engedélyező szabályok is,
# annyi különbséggel, hogy a ’-’ karakter helyett a ’+’ karaktert kell használnunk.
```

Ha kész vagyunk a fenti TPL fájl létrehozásával, akkor már csak annyi van hátra, hogy el kell helyezni egy kiszolgálónkon és a távoli konfigurációs eszközzel hozzá kell adni a követésvédelmi listát a böngészőkhöz. Az automatikus telepítés mellett meg is oszthatjuk ezt a TPL fájlt például a

vállalat webhelyén keresztül. Ehhez csupán egy speciális hivatkozást kell elhelyezni a weboldalon, és a hivatkozásra kattintva bárki telepítheti az Internet Explorer 9-ébe.

```
<a href="javascript:
   window.external.msAddTrackingProtectionList('https://home/CustomTPL.txt',
     'Belső követésvédelmi lista')"> Kattins ide a lista felvételéhez </a>
```

3. fejezet: Az Internet Explorer 9 fejlesztői szemmel

A korábbi fejezetekben láthattuk, hogy akár vállalati, akár hagyományos felhasználó szemmel tekintünk az új Internet Explorerre, ideális böngésző választás lehet, hiszen minden szempontból kielégítheti igényeinket. Azonban ez nem minden. A modern webes szabványok teljes mértékű támogatása ideális hátteret biztosít a modern webes alkalmazásaink kiszolgálására. Az Internet Explorer 6 sokat kritizált nem szabványos HTML támogatása már rég a múlté, és az Internet Explorer család a 9-es változatának készítése során a böngésző gyártói aktívan közreműködtek a webes szabványokat koordináló konzorciummal annak érdekében, hogy elősegítsék a teljes mértékű szabványos HTML feldolgozást és segítsenek a modern webes szabványok fejlesztésében.

Ebben a harmadik és utolsó fejezetben egy rövid, összefoglaló betekintést fogunk tenni az Internet Explorer 9 által támogatott webes szabványokba, technológiákba, majd különböző példákon keresztül meg fogunk ismerkedni az említett technológiákkal és azok újdonságaival.

A példák elsősorban azokat a kíváncsi és lelkes internetezőket célozza meg, akik foglalkoztak már valamilyen szinten webfejlesztéssel és bepillantást szeretnének nyerni abba, hogy hogyan is lehet kihasználni az Internet Explorer 9 képességeit. A példa során csak olyan ingyenes fejlesztői eszközöket fogunk latba vetni, amelyek bárki számára elérhetőek.

3.1. Támogatott technológiák

Fejlesztés során azt szeretnénk, hogy az általunk elkészített weboldal egyformán jelenjen meg a piacon lévő összes böngészőn. Azonban ez sajnos jelenleg nem ennyire egyértelmű, hiszen ugyanazt a HTML kódot, CSS stílust és JavaScriptet eltérően értelmezik a böngészők. Az Internet Explorer 9 a mai legmodernebb szabványokat követi, ami fejlesztőként nem jelent mást, mint azt, hogy nyugodtan készíthetünk weboldalakat anélkül, hogy azon kellene aggódnunk, hogy a többi, szabványokat követő böngészőkben eltérően fog megjelenni és működni, mint ahogy azt terveztük.

Az Internet Explorer az alábbi modern szabványokat támogatja:

- HTML5
- CSS3
- SVG
- ECMAScript 5
- DOM L2 és L3
- GeoLocation
- ICC színprofilok
- Webes betűtípusok (WOFF)

HTML5 támogatás

Az Internet Explorer 9 támogatja az olyan népszerű HTML5 elemeket, mint például az audió és a video elem, amelyek segítségével hardveres támogatás mellett tudunk videó és audió tartalmakat beágyazni webhelyeinkbe, anélkül, hogy bármilyen külső bővítményre szükségünk lenne. Ennek a támogatásnak köszönhetően olyan egyszerűen helyezhetünk el multimédiás tartalmakat a weboldalainkon, mintha azok képek lennének. Ezen felül, a canvas (vászon) elem támogatásával egyszerűen elláthatjuk weboldalainkat látványos grafikai elemekkel és animációkkal. Az új szemantikus elemeknek segítségével pedig magunk és mások számára is könnyen áttekinthetővé tehetjük weboldalaink szerkezetét és felépítését.

CSS3 támogatás

Az Internet Explorer 9 a teljes körű CSS 2.1 támogatás mellett támogatja azokat a CSS3 modulokat is, amelyek már végleges verzióban vannak. Ennek köszönhetően a webfejlesztőként vagy webes arculattervezőként sokkal kényelmesebben és egyszerűbben adhatunk nagyon látványos megjelenést weboldalainknak. A támogatott CSS modulok listájában olyanok szerepelnek, mint a Backgrounds & Borders modul, a Color modul, a Fonts modul, a Media Queries modul, a Namespaces modul, a Selectors modul vagy a Values and Units modul.

ECMAScript 5 támogatás

A legújabb ECMAScript szabványban leírt funkciók hozzájárulnak ahhoz, hogy az Internet Explorer 9 alá írt parancsfájlok gond nélkül működjenek a többi HTML5 kompatibilis böngészőben is. Emellett a szabvány számtalan olyan újdonságot tartalmaz, ami leegyszerűsíti, megkönnyíti és felgyorsítja a JavaScript-el végzett programozást.

SVG támogatás

A beépített, natív SVG támogatásnak köszönhetően rendkívül látványos és részletgazdag vektorgrafikus elemekkel láthatjuk el webhelyeinket anélkül, hogy bármilyen külső bővítményre szükségünk volna. A többi HTML elemhez hasonlóan az SVG megjelenítés is hardveres gyorsítással van megvalósítva.

Geolocation támogatás

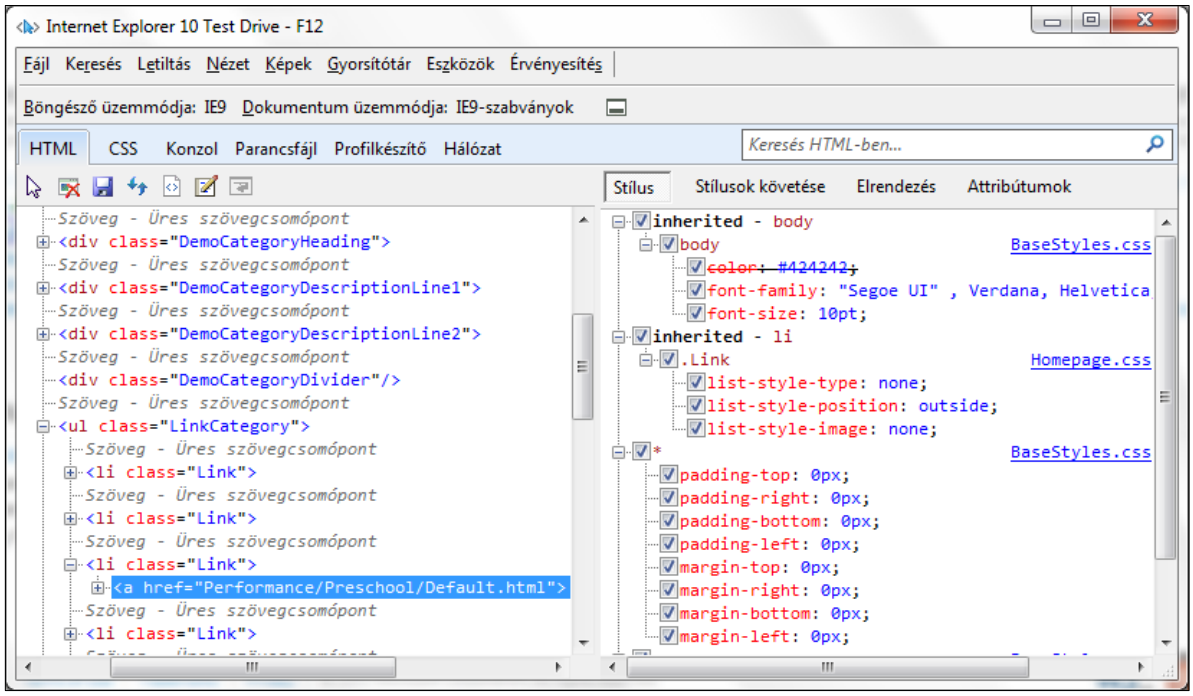
A Geolocation modul segítségével hozzáférhetünk a webhelyet megjelenítő eszköz földrajzi koordinátáihoz, amit később számtalan módon felhasználhatunk. A Geolocation modul előnyeit kihasználva a releváns webhelyeken (például időjárás előrejelzés) nagyban növelhetjük felhasználóink elégedettségét.

WebM támogatás

Az Internet Explorer 9 alapértelmezetten a H.264 tömörítésű videófájlokat képes lejátszani, azonban megfelelő videó-kodek (VP8) megléte esetén képes lejátszani a népszerű webes videóformátumú, WebM videókat is.

3.2. Beépített fejlesztő eszközök

Az Internet Explorer beépített fejlesztői eszközei rendkívül sok téren leegyszerűsíthetik webes alkalmazásainak fejlesztését és tesztelését. A fejlesztői eszközök segítségével módosíthatjuk a böngészőnk által megjelenített webhely tartalmát, stílusát, így még azelőtt kipróbálhatjuk a változtatásainkat, hogy valóban implementáltuk volna őket. A fejlesztői eszközök segítségével egyszerűen átláthatjuk a webhelyeink stílusát és azt, hogy az egyes stílus szabályok milyen elemeken keresztül öröklődnek. Megtekinthetjük a webhely DOM felépítését és szerkezetét, így könnyen informálódhatunk egy-egy megjelenített HTML elem tulajdonságairól (például szélessége, magassága, a dokumentum-fán belül betöltött helye, stb.). Emellett egyszerűen végezhetünk hibakeresést a parancsfájlokon is, megtekinthetjük és módosíthatjuk a változóink tartalmát, töréspontokat helyezhetünk el és lépésenként végigkövethetjük azt, hogy mi történik a parancsfájlban. A fejlesztő eszközök segítésével a weboldal és a kiszolgáló közötti kommunikációt is felügyelhetjük, valamint egyszerűen megváltoztathatjuk az Internet Explorer kompatibilitási nézetét, aminek köszönhetően egyből megvizsgálhatjuk, hogy weboldalunk hogyan fog kinézni az IE korábbi változataival.



41. ábra: Beépített fejlesztői eszközök

A szóban forgó fejlesztő eszközöket a billentyűzet F12-es gombjával érhetjük el. Az eszköz aktiválás után meg fog jelenni a böngésző alján, azonban módosíthatjuk ezt a pozíciót úgy is, hogy egy teljesen független ablak legyen.

3.3. HTML5 a gyakorlatban

Az Internet Explorer rövid fejlesztői áttekintése után most már tisztában vagyunk azzal, hogy milyen technológiákat fogunk alkalmazni a fejezet bevezetésében ígért példák elkészítésekor. A fejezet további oldalain pedig megnézzük azt, hogy milyen technológiákra van szükségünk, röviden áttekintjük a fejlesztő eszközt, amivel dolgozni fogunk és alfejezetekre bontva megismerkedünk majd az egyes HTML5 és CSS3 elemekkel.

Áttekintés

Azzal már bizonyára tisztában vagyunk, hogy egy weboldal felépítése során három fő technológia csoportot használhatunk: a HTML leíró nyelvet, a JavaScript programozási nyelvet és a CSS stílusleíró nyelvet. A HTML leíró nyelv segítségével írjuk le a weboldalaink szerkezetét, felépítését és elemeit, a JavaScript nyelvvel programozhatjuk és tehetjük azt interaktívvá. A CSS stílusleíró nyelv segítségével pedig látványos megjelenést biztosíthatunk weboldalainknak.

Mindhárom nyelv nagy múltra tekint vissza, és már nagyon sok éve szolgálják azt a célt, hogy weboldalakat készíthessünk. Ebből kifolyólag mindhárom nyelvből több verzió is létezik, közülük a legelterjedtebb és leggyakrabban használt a HTML 4 és 4.1, a CSS 2 és 2.1, valamint JavaScript 1.8.1 és 1.8.2. A példákon keresztül azonban mindhárom nyelvnek a legújabb változatát fogjuk használni, mégpedig a HTML5-öt, a CSS3-at és a JavaScript 1.8.5-öt (amelyet az ECMAScript 5 specifikáció ír le).

A példák készítése során használhatjuk például a Microsoft Visual Studio 2010 fejlesztői környezetét (hasonló képességekkel bír az ingyenes Microsoft Visual Web Developer 2010 Express is), hiszen rengeteg fejlesztést megkönnyítő eszközzel rendelkezik. Annak érdekében, hogy rendelkezzünk HTML5 gyorskiegészítéssel (IntelliSense), fel kell telepítenünk a fejlesztőkörnyezet legfrissebb javítócsomagját (írásomkor az SP1-et). A fejlesztői környezet telepítése után fel kell telepítenünk az Internet Information Services webkiszolgálót, ugyanis a Visual Studio beépített webkiszolgálója nem boldogul el a HTML5 videók által használt MIME-típusokkal. Ha mindezzel végeztünk, akkor neki is láthatunk a fejlesztésnek.

A Visual Studio termékcsaládon felül használhatjuk még az ingyenes Microsoft WebMatrix fejlesztői környezetet is, amely elsősorban a beépített webszerverének és a rendkívül könnyű kezelhetőségének köszönhetően lehet jó választás.

Előkészítés

A példák elkészítésének első lépése a fejlesztői környezet feltelepítése. Ha ezzel végeztünk a „*File -> New Project -> ASP.NET Emptry Web Application*” sablont kiválasztva neki is kezdhetünk a webhelyünk kialakításának. Miután a Visual Studio elkészítette a projektünket, a projekt tulajdonságainál (*„Project -> Properties -> Web -> Servers”)* át kell állítanunk azt, hogy milyen kiszolgáló alatt fog futni az alkalmazás (*„Use Local IIS Webserver”).*

A következő lépés a webhely struktúrájának kialakítása. Létre kell hoznunk egy mappát a JavaScript fájloknak, egy mappát a stílusfájloknak, valamint egy mappát, ahol a webhelyen megjelenítendő tartalmakat (képeket, videókat, zenét) tárolni fogjuk. Ezt követően pedig a projektben található web.config fájlba kell állítanunk azt, hogy a kiszolgálónk megfelelően kezelje a HTML5 MIME-típusokat. Végeredményként hasonlóan kell kinéznie a web.config fájlunknak:

```
<?xml version="1.0"?>
<configuration>
  <system.webServer>
    <staticContent>
      <mimeMap fileExtension=".mp4"  mimeType="video/mp4" />
      <mimeMap fileExtension=".m4v"  mimeType="video/m4v" />
      <mimeMap fileExtension=".ogv"  mimeType="video/ogg" />
      <mimeMap fileExtension=".webm" mimeType="video/webm" />
      <mimeMap fileExtension=".m4a"  mimeType="audio/mp4" />
      <mimeMap fileExtension=".oga"  mimeType="audio/ogg" />
      <mimeMap fileExtension=".ogg"  mimeType="audio/ogg" />
      <mimeMap fileExtension=".spx"  mimeType="audio/ogg" />
      <remove  fileExtension=".svg"/>
      <mimeMap fileExtension=".svg"  mimeType="images/svg+xml" />
      <mimeMap fileExtension=".svgz" mimeType="images/svg+xml" />
      <remove  fileExtension=".eot" />
      <mimeMap fileExtension=".eot"  mimeType="application/vnd.ms-fontobject" />
      <mimeMap fileExtension=".otf"  mimeType="font/otf" />
      <mimeMap fileExtension=".woff" mimeType="font/x-woff" />
    </staticContent>
  </system.webServer>
</configuration>
```

HTML5 szemantikus elemek

Az után, hogy minden szükséges telepítést és beállítást elvégeztünk, neki is láthatunk az első, HTML5 alapú weblapunk létrehozásához. A Visual Studio nem rendelkezik HTML5 weblap szabvánnyal, így a hagyományos weboldal sablont kell használnunk erre a célra. Az újonnan létrehozott HTML fájl alapértelmezés szerint tartalmazni fogja a weblap nélkülözhetetlen elemeit: a fejlécet és a törzset. A fejlécből azonban ki kell törölnünk a nem oda illő részeket, hiszen a HTML5 szabvány egyik újdonsága, hogy nagyon röviden megadhatjuk a dokumentum, weblap típusát.

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
```

A következő lépésben ki fogjuk egészíteni ezt a fájlt néhány új szemantikus elemmel. A szemantikus elemek is részei a HTML5 újdonságainak és fő céljuk az, hogy mind a böngészők, mind a fejlesztők számára átláthatóbbá tegyék a dokumentumot, mind tartalmi és mind szerkezeti szinten. Viselkedés szempontjából nem különböznek a már jól ismert **div** elemektől, nem rendelkeznek semmilyen speciális funkcióval, és nem bírnak semmilyen speciális tulajdonsággal sem, csupán a nevükben különböznek. Ugyan ez csekély különbségnek tűnhet, de egy nagyobb weblap esetében nagyban megkönnyíti a dokumentum szerkezetén belüli navigációt, ha ilyen megkülönböztető elemeket használnunk, és nem az általános célú **div** elemeket.

```
<!DOCTYPE html>
<html>
<head>
    <title>Gipsz Jakab weboldala - Kezdőlap</title>
</head>
<body>
    <hgroup>
        <h2>Személyes weboldalam</h2>
        <h3>Kezdőlap</h3>
        <p><time datetime="2011-01-26">January 26, 2011</time></p>
    </hgroup>
    <nav>
        <ul>
            <li><a href="#">Kezdőlap</a></li>
            <li><a href="#">Nyaralásaim</a></li>
            <li><a href="#">Kedvenc videóim</a></li>
            <li><a href="#">Kedvenc zenéim</a></li>
        </ul>
    </nav>
    <article>
        <header><h4>Üdvözöllek a weboldalamon!</h4></header>
        <section>
            <p> Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
            Maecenas porttitor congue massa. Fusce posuere, magna sed pulvinar ultricies,
            purus lectus malesuada libero, sit amet commodo magna eros quis urna. </p>
            <figure>
                <figcaption>Ezen a képen én vagyok =)</figcaption>
                <img width="240" height="320" alt="Kép rólam "
                     src="Media/Images/meandmyselft.png" />
            </figure>
        </section>
    </article>
    <footer>
        <p>Készítette: Gipsz Jakab</p>
    </footer>
</body>
</html>
```

A fenti példában jól megfigyelhetjük, hogy első ránézésére egyértelmű, az általunk felhasznált HTML elemek a weboldal mely szerkezeti részéhez tartoznak.

Bár a szemantikus elemek révén nőtt a felhasználható elemek száma, azonban nem árt tudni, hogy számos, népszerű és kevésbé népszerű elemet már nem használhatunk weboldalaink készítése során. Ilyen elem például a **<big>**, a ** ,** a **<font>,** a **<frame>** vagy a **<frameset>.**

HTML5 multimédiás vezérlő elemek

Az internetet böngészve gyakran találkozhatunk olyan weboldalakkal, amelyekbe valamilyen multimédiás tartalom van beágyazva. Ennek oka az, hogy az utóbbi időben egyre elterjedtebbé vált az efféle beágyazott tartalom (legfőképp a videó), ami nem meglepő, hiszen ki ne osztaná meg kisfilmjét, saját készítésű klipjét vagy a kiránduláson készített felvételeket a barátaival, ismerőseivel. Az egyszerűen használható Flash és SilverLight bővítmények elterjedésével és az internet sávszélesség növekedésével minden akadály elhárult a multimédiás tartalom felhasználás elől. A HTML5 szabvány pedig épít erre a népszerűségre, így az új nyelvben két multimédiás elemet is találhatunk: az **<audio>** és **a <video>** elemet. Értelemszerűen az **<audio>** audió tartalmakat játszhatunk le, a **<video>** elemmel pedig videó tartalmakat. Azonban fontos tudnunk, hogy az eltérő böngészők eltérő formátumú médiákat támogatnak, ezért lehetőség szerint több forrással is lássuk el a multimédiás elemeinket. Az Internet Explorer 9 esetében a támogatott formátumok közé tartoznak a WAV, illetve MP3 audió tartalmak és a H.264 tömörítésű MPEG 4 videó tartalmak (megfelelő kodek megléte esetén a WebM videó formátumot is támogatja).

```
<article>
    <header><h4>Kedvenc videóim és számaim!</h4></header>
    <video id="myvideo" controls="controls" loop="loop" preload="metadata" width="640"
           height="360" poster="Media/Images/VideoPoster.png">
        <source src="Media/Videos/sample.mp4" />
        <source src="Media/Videos/sample.ogv" />
        Nem támogatott böngésző!
    </video>
    <audio controls="controls" loop="loop">
        <source src="Media/Audios/sample.mp3" />
        <source src="Media/Audios/sample.ogg" />
        Nem támogatott böngésző!
    </audio>
</article>
```

A fenti példából láthatjuk, hogy az új multimédiás elemek számos hasznos tulajdonságokkal bírnak. A `controls` tulajdonság segítségével vezérelhetjük azt, hogy a böngésző által megjelenített lejátszó rendelkezzen-e vezérlőgombokkal vagy sem. A `loop` tulajdonsággal vezérelhetjük azt, hogy a lejátszás végét követően megismétlődjön-e a lejátszás vagy sem. A `preload` tulajdonság vezérli azt, hogy az oldal letöltése után a videó mely része töltődjön le automatikusan: választhatunk az `auto`, `metadata`, illetve `none` értékek közül. Az `auto` érték jelenti azt, hogy a teljes tartalom letöltődik, a `metadata` érték azt, hogy csak a média tulajdonságai, a **none** pedig értelemszerűen semmilyen automatikus letöltést nem engedélyez. Annak érdekében, hogy a nem HTML5 kompatibilis böngészőkben utaljunk arra, hogy a böngésző nem támogatott, szöveget is elhelyezhetünk a multimédiás elemek belsejében. Azok a böngészők, amelyek támogatják a HTML5 szabványt, ezt a szöveget figyelmen kívül fogják hagyni.

42. ábra: Beépített multimédiás lejátszó

Mindamellett, hogy a multimédiás vezérlő elemek használata, kezelése roppant kényelmes, és valójában olyan egyszerűen használhatjuk őket, mintha csak képek lennének, rendkívüli JavaScript támogatással is bírnak. A böngészőbe épített lejátszó minden tulajdonságát, funkcióját illetve eseményét elérhetjük JavaScriptből is. Ennek köszönhetően egyszerűen adhatunk egyedi, személyreszabott megjelenést lejátszónknak csupán pár sornyi JavaScript és CSS felhasználásával.

```
<article>
    <header><h4>Saját videólejátszóm!</h4></header>
    <video id="myvideo" loop="loop" preload="metadata" width="640" height="360"
           poster="Media/Images/VideoPoster.png">
        <source src="Media/Videos/sample.mp4" />
        <source src="Media/Videos/sample.ogv" />
        Nem támogatott böngésző!
    </video>
    <br />
    <span id="btnPlay">Lejátszás</span>
    <br />
    <span id="btnPause">Szüneteltetés</span>

    <script type="text/javascript">
        var myvideo = document.getElementById('myvideo');
        var btnPlay = document.getElementById('btnPlay');
        var btnPause = document.getElementById('btnPause');

        btnPlay.onclick = function () {
            if (!myvideo.paused || myvideo.ended) {
                alert('Nincs megállítva a videó!');
                return;
            }

            myvideo.play();
        }

        btnPause.onclick = function () {
            myvideo.pause();
        }
```

```
    </script>
</article>
```

A HTML5 vászna

A HTML5 szabvány egyik legizgalmasabb új vezérlőeleme a vászon (<canvas>), amelynek segítségével dinamikusan, programozható módon jeleníthetünk meg kétdimenziós grafikákat, képeket és alakzatokat a böngészőnkben. Ez az új elem hivatott átvenni az eddig oly népszerűvé vált Flash technológia helyét, hiszen az egyszerűbb képek és alakzatok kirajzolásától az összetettebb animációk programozására is felhasználható, valamint lehetőséget ad a videókkal való munkavégzésre is. Megjelenésével a webes multimédiás alkalmazások egy új dimenziója tárulhat ki előttünk: a népszerű böngészőben futó fizetős webes játékok újjászületése mellett készíthetünk egyszerűbb animációkat valamint reklámokat is. Üzleti célokra is felhasználhatjuk, hiszen látványos grafikonokat és diagramokat készíthetünk csupán a böngészőnk segítségével. A vászon további előnye még az is, hogy ugyanazt az animációt rövidebb programkóddal is elérhetjük, mint a Flash vagy SilverLight esetében. Azonban a sok előnye mellett hátránya is akad, hiszen ha animációt készítünk, vagy videót játszunk vissza, kézzel kell gondoskodni a vászon újrarajzolásáról.

A vászon használata a gyakorlatban három lépésből áll: első lépésként el kell helyezni egy **<canvas>** elemet a weboldalon, majd inicializálni kell a rajzfelületét. A harmadik lépés pedig a vászon tényleges programozása a rajzfelületen keresztül.

```
<article>
   <header><h4>Látványos weblapom.</h4></header>
   <canvas id="myCanvas" width="640" height="380">
       Nem támogatott böngésző!
   </canvas>

   <script type="text/javascript">
      var myCanvas = document.getElementById('myCanvas');
      var myContext = myCanvas.getContext("2d");
   </script>
</article>
```

A fenti programrésszel ugyan megadtuk azt, hogy mekkora legyen a vásznunk, azonban ha megtekintjük a böngészőben, nem fogunk semmit sem látni, hiszen alapértelmezetten egy üres, átlátszó téglalapot kapunk. Annak érdekében, hogy „kézzel fogható” legyen a vásznunk, rajzolnunk kell rá valamit.

A legegyszerűbb alakzat, amit a vászonra rajzolhatunk a téglalap. Csakúgy, mint a többi geometriai alakzat esetében rajzolás alatt két műveletet érthetünk: az alakzat kitöltését vagy az alakzat körvonalának megrajzolását. Ennek értelmében a téglalapunk megrajzolásához két függvényt használhatunk: a **stokeRect(posX, posY, width, height)** és a **fillRect(posX, posY, width, height)** függvényeket. Sőt, a téglalappal végezhető műveletek körébe tartozik még egy olyan speciális függvény is, amelynek segítségével a vászon tartalmát törölhetjük. Ez a függvény pedig nem más, mint a **clearRect(posX, posY, width, height)**. Animációk készítése során a vászon újrarajzolása előtt célszerű használni ezt a függvényt, annak érdekében, hogy az előzőleg kirajzolt tartalomtól fájó búcsút vegyünk.

```
var myCanvas = document.getElementById('myCanvas');
var myContext = myCanvas.getContext("2d");

// Vászon teljes tartalmának törlése
myContext.clearRect(0, 0, myCanvas.clientWidth, myCanvas.clientHeight);

// Téglalap alakú körvonal rajzolása a vászon 20;20 koordinátáiból
myContext.strokeRect(20, 20, 100, 100);

// Kitöltött téglalap rajzolása a vászon a 140;20 koordinátáiból
myContext.fillRect(140, 20, 100, 100);
```

Természetesen nem csak téglalapot rajzolhatunk a vászonra, hanem olyan egyszerűbb geometriai alakzatokat is, mint például a vonal vagy a kör. Viszont ezeket a geometriai alakzatokat a vászon másképp kezeli, mint a téglalapot, és ahhoz, hogy megfelelően használni tudjuk őket, tisztában kell lennünk a vászon alakzatkezelési módszerével.

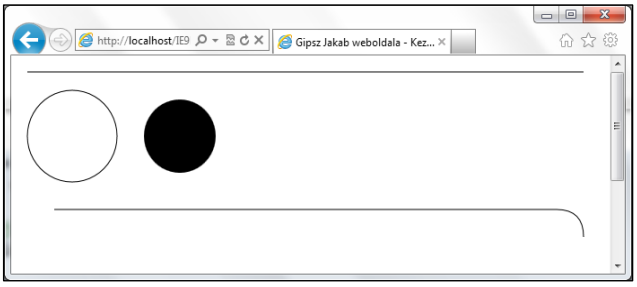
Amikor egyszerűbb geometriai alakzatokat rajzolunk, akkor azt a vászon valójában egyetlen komplex alakzatként értelmezi. Hogyha egyszerre több alakzatot is rajzolni szeretnénk, akkor ezt közölnünk kell a vászonnal is. Annak érdekében, hogy valóban azt a végeredményt kapjuk, mint amit szeretnénk, egy-egy alakzat rajzolásának kezdetekor és végekor meg kell hívnunk a vászon **`beginPath()`**, illetve **`closePath()`** függvényét. Továbbá fontos tudnunk azt is, hogy amikor alakzatokat rajzolunk, akkor csak egy virtuális ecsettel dolgozunk, és mindaddig nem fogjuk látni a végeredményt, ameddig meg nem hívjuk a vászon **`stroke()`** és, illetve vagy **`fill()`** függvényét. Ezeknek az alapozó ismereteknek a birtokában el is kezdhetünk különböző alakzatokat rajzolni a vászonra.

```
var myCanvas = document.getElementById('myCanvas');
var myContext = myCanvas.getContext("2d");

// Alakzat rajzolás elkezdése
myContext.beginPath();

// Virtuális ecset elmozgatása az adott koordinátákba
myContext.moveTo(10, 10);

// Vonal rajzolása az adott koordinátákba (az ecset aktuális pozíciójától)
myContext.lineTo(myCanvas.clientWidth - 10, 10);

// Körvonal megrajzolása
myContext.stroke();

// Alakzat rajzolás vége
myContext.closePath();

myContext.beginPath();
// 50 pixel sugarú virtuális kör rajzolása, melynek középpontja az 60x60-as pont
// A szög kiinduló pontját 0 radiánra állítottuk, végpontját pedig 2PI-re, amivel
// így írja le a teljes kört.
myContext.arc(60, 80, 50, 0, Math.PI * 2, false);
myContext.stroke();
myContext.closePath();
```

```
myContext.beginPath();
// Egy másik kör rajzolása a vászon egy másik pontjába
myContext.arc(180, 80, 40, 0, Math.PI * 2, false);

// Alakzat kitöltése
myContext.fill();
myContext.closePath();
```

Az egyenes vonalak rajzolásán túl a vászon arra is lehetőséget ad nekünk, hogy harmad- és negyedfokú görbéket rajzoljunk. Ennek akkor vehetjük igazán hasznát, amikor ívelt alakzatokat szeretnénk rajzolni, például lekerekített téglalapot.

```
// Lekerekített görbe rajzolása
myContext.beginPath();
myContext.moveTo(40, 160);
myContext.lineTo(myCanvas.clientWidth - 40, 160);

// Görbe rajzolása
myContext.quadraticCurveTo(myCanvas.clientWidth - 10, 160,
    myCanvas.clientWidth - 10, 190);

myContext.stroke();
myContext.closePath();
```



43. ábra: Vászon tartalma

Az előző példákban mondhatni sokat használtuk a vászon körvonalrajzoló, illetve kitöltő függvényét. Bizonyára már azt is észrevettük, hogy mind a körvonal, mind pedig a kitöltés színe fekete. Természetesen nem ez az egyetlen szín, amit rajzolás során használhatunk, hiszen alkalmazhatjuk a CSS3 keretében használható összes színt, és az egyszerű színeken felül színátmeneteket és mintákat is használhatunk.

Annak érdekében, hogy módosítsuk a vászon alapértelmezett fekete színét, két tulajdonságát kell beállítanunk. A **`strokeStyle`** tulajdonsággal a körvonalak színét módosíthatjuk, míg a **`fillStyle`** tulajdonsággal pedig a kitöltését.

Ha csupán új színt szeretnénk beállítani, akkor egyszerű dolgunk van. A CSS3-ban megszokott színmegadási módokkal tudunk új színt meghatározni.

```
// szöveges színmegadási mód
myContext.strokeStyle = "red";
// hexadecimális színmegadási mód
```

```
myContext.strokeStyle = "#FF0000";
// RGB (Red, Green, Blue) modellel történő színmegadás
myContext.strokeStyle = "rgb(255,0,0)";
// RGB modell & átlátszósággal történő színmegadás
myContext.fillStyle = "rgba(255,0,0,0.5)";
// HLS (Hue, Saturation, Lightness) modellel történő színmegadás
myContext.fillStyle = "hsl(0, 100%, 50%)";
// HLS modell & átlátszósággal történő színmegadás
myContext.fillStyle = "hsla(0, 100%, 50%, 0.5)";
```

Színátmenetes stílus készítésekor kétféle színátmenet közül választhatunk: az egyenes színátmenet és a sugaras színátmenet közül. Mind a két esetben a vászon **fillStyle** illetve **strokeStyle** tulajdonságnak egy **CanvasGradient** objektumot kell átadnunk.
Egyenes színátmenet definiáláshoz a vászon **createLinearGradient(x0, y0, x1, y1)** függvényével hozhatjuk létre a szükséges **CanvasGradient** objektumot, sugaras színátmenet definiáláshoz pedig a **createRadialGradient(x0, y0, r0, x1, y1, r1)** függvényével. Az első esetben a vászon egy virtuális téglalap, a második esetben pedig két virtuális kör segítségével rajzolja meg a színátmenetet. A színátmenetek csak a megadott virtuális alakzatok koordinátáin belül láthatóak, azon kívül nem. Tehát ha definiálunk egy kisméretű egyenes színátmenetet, majd egy nagyobb téglalapot töltünk ki vele, a megrajzolt téglalapnak csak egy része lesz kitöltve a színátmenettel. A **CanvasGradient** objektum létrehozását követően már csak hozzá kell rendelni a szín-végpontokat a színátmenethez és készen is vagyunk. Színvégpontot a **CanvasGradient** objektum **addColorStop(offset, color)** metódusával adhatunk hozzá a színátmenethez, ami paraméterként a 0 és 1 tartomány közé eső pont helyét és a végpont színét várja.

```
var gradient = myContext.createLinearGradient(0, 0, 300, 300);
gradient.addColorStop(0.0, "rgba(0,0,0,0.0)");
gradient.addColorStop(0.5, "rgba(0,0,0,0.8)");
gradient.addColorStop(1.0, "rgba(0,0,0,1.0)");

myContext.fillStyle = gradient;
myContext.fillRect(0, 0, 300, 300);

// myContext.createRadialGradient(sx, sy, sr, dx, dy, dr)
// s = kiindulási(start) kör
// d = befejezési(destination) kör
var radGrad = myContext.createRadialGradient(200, 200, 10, 200, 200, 150);
radGrad.addColorStop(0.0, "rgba(255,0,0,0.0)");
radGrad.addColorStop(0.5, "rgba(0,255,0,0.8)");
radGrad.addColorStop(1.0, "rgba(0,0,255,1.0)");
```

Mintával való kitöltéskor sincs sokkal összetettebb dolgunk, csupán létre kell hozni egy CanvasPattern objektumot a vászon **createPattern(image, repetition)** függvényével, amit később, a fentiekhez hasonló módon kell beállítani vászonnak. Első paraméterként magát a mintát kell meghatározni: átadhatunk képet, egy másik vászon objektumot és videót is. Második paraméterként pedig az ismétlődés módját kell beállítanunk (a CSS-ben használt repeat, **no-repeat**, **repeat-x** és **repeat-y** értékek valamelyikével).

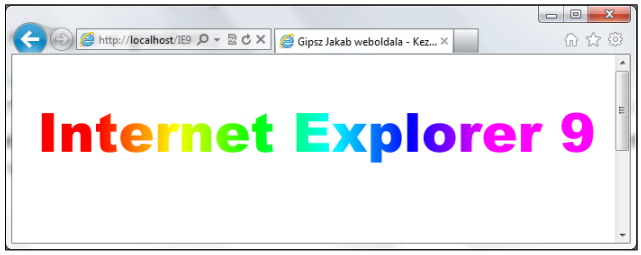
```
// Mintaként használt kép inicializálása
var patternImage = new Image();
```

```
patternImage.src = "logo.png";

// Csak a kép letöltődése után állítjuk be mintaként a képet,
// ellenkező esetben előfordulhat, hogy a minta üres lesz
patternImage.onload = function () {

    var pattern = myContext.createPattern(patternImage, "repeat");
    myContext.fillStyle = pattern;
    myContext.fillRect(0, 0, 500, 200);
}
```

Most, hogy már tudunk színeket használni, körvonalakat rajzolni, megismerkedhetünk azzal is, hogy miként lehet szöveget írni a vászonra. A szöveg kiírásának, avagy rajzolásának módja – a vászon előbbi függvényeihez hasonlóan – egyszerű műveletnek tekinthető. A vászon **fillText(text, posX, posY)** és **strokeText(text, posX, posY)** függvényeit felhasználva tudjuk ellátni körvonallal, illetve kitöltéssel kirajzolandó szövegeinket. A betűk színén felül meghatározhatjuk azt a betűtípust is, amivel rajzolni szeretnénk, méghozzá a vászon **font** tulajdonságán keresztül.

```
var myCanvas = document.getElementById('myCanvas');
var myContext = myCanvas.getContext("2d");

// Szöveg stílusának meghatározása
myContext.font = "60px bold, Arial Black";

// Szöveg méretének (az aktuális stílust alkalmazva) lekérdezése
var textWidth = myContext.measureText("Internet Explorer 9").width;

var gradient = myContext.createLinearGradient(0, 0, textWidth, 0);
gradient.addColorStop(0.14, "rgb(255,0,0)");
gradient.addColorStop(0.28, "rgb(255,255,0)");
gradient.addColorStop(0.42, "rgb(0,255,0)");
gradient.addColorStop(0.56, "rgb(0,255,255)");
gradient.addColorStop(0.70, "rgb(0,0,255)");
gradient.addColorStop(0.84, "rgb(255,0,255)");
gradient.addColorStop(0.10, "rgb(255,0,0 )");

myContext.fillStyle = gradient;

// Szöveg kirajzolása a megfelelő X,Y pontokba
myContext.fillText("Internet Explorer 9", 20, 100);
```

A fenti példával egy látványos színátmenet segítségével kirajzoltuk az „Internet Explorer 9” szöveget a vászon 20, 100-as koordinátáiba (44. ábra).



44. ábra: Szöveg rajzolása

Már láthattuk azt, hogy hogyan tudunk alakzatokat és szöveget rajzolni valamint kitölteni, azonban a vászon számos érdekes funkciója közül kiemelendő még a képmanipulációs képessége is. Tehát a vászonra nem csak vonalakat rajzolhatunk, hanem képeket, videókat és más vásznakat is. A rajzolást követően pedig érdekesebbnél érdekesebb dolgokat művelhetünk a betöltött képekkel. A legérdekesebb művelet talán az, hogy a betöltött képeknek és videóknak módosíthatjuk, manipulálhatjuk a képpontjait. A képpontok manipulálásával pedig olyan látványos vizuális hatásokat érhetünk el, mint például a Gauss-elmosás, színek manipulálása (világosítás, élénkítés). Sőt, a kedvenc képszerkesztő alkalmazások összes szűrőjét, művészi effektusát is megvalósíthatjuk a kellő ismeretek birtokában. Mindezeknek a hatásoknak alkalmazása egy valós időben lejátszott videón pedig lenyűgözően tud mutatni, főleg akkor, ha nem felejtjük el azt, hogy mindezt böngészőben tettük meg, némi JavaScript felhasználásával.

A képek betöltése egyértelmű lehet számunkra, hiszen nemrég láttunk arra példát, hogy hogyan kell mintaként beállítani egy képet. Természetesen ebben az esetben sem bonyolultabb az eljárás: létre kell hozni egy JavaScript kép objektumot, majd ezt át kell adni a vászon megfelelő függvényének. De mi a helyzet akkor, hogyha egy videót vagy egy másik vásznat szeretnénk rajzolni? A dolgunk ebben az esetben sem sokkal bonyolultabb, hiszen a vászon úgy kezeli a videókat vagy más vásznakat, mint hogyha csak képek lennének. Hiszen a videó esetében a tartalom képkockákból épül fel, és mindig meg lehet határozni azt, hogy adott időpontban a videó éppen melyik képkockáját látjuk. A vászon esetében is hasonló a helyzet, ugyanis hiába van tele mindenféle látványos animációkkal az a vászon, amivel dolgozni szeretnénk, mindig lesz aktuális képkockája, azaz mindig meg tudjuk határozni azt, hogy éppen mit lát a felhasználó.

A kép és videó (illetve vászon) rajzolásakor a vászon **drawImage** függvénye lesz a segítségünkre, amelyet háromféleképpen paraméterezéssel is meghívhatunk. A **drawImage(img,posX,poxY)** paraméterezéssel egyszerűen beillesztjük a képet a vászon adott koordinátáiba; **drawImage(img,posX,posY,width,height)** paraméterezéssel pedig át is méretezzük azt. Sőt, a **drawImage(img,clipX,clipY,clipWidth,clipHeight,posX,posY,width,heigth)** paraméterezéssel még körbe is vághatjuk igényeink szerint.

```
<canvas id="myCanvas" width="640" height="380">
    Nem támogatott böngésző!
</canvas>
<video id="myVideo" autoplay="autoplay" width="640" height="380" style="display: none">
    <source src="Media/Videos/sample.mp4" />
</video>
```

```
<script type="text/javascript">
    var myCanvas = document.getElementById('myCanvas');
    var myContext = myCanvas.getContext("2d");

    var myVideo = document.getElementById('myVideo');

    // Kép rajzolása
    var imgCanvas = new Image();
    imgCanvas.src = 'Media/Images/meandmyselft.png'
       imgCanvas.onload = function () {
        myContext.drawImage(imgCanvas, 10, 10, 200, 200);
    }

    // Felíratkozunk a videó lejátszás eseményére,
    // annak érdekében, hogy akkor rajzoljuk ki a videó, amikor
    // ténylegesen elindult a lejátszása
    myVideo.onplay = function () {
        myContext.drawImage(myVideo, 220, 10, 200, 200);
    }
</script>
```

A fenti példából láthatjuk, hogy a videó kirajzolásához szükségünk van egy HTML5 videó elemre, amit később JavaScriptben megkeresünk és átadunk a **drawImage** függvénynek. Annak érdekében, hogy egyszerre ne két videót lássunk, hanem csak a vászon által kirajzoltat, el kell rejtenünk az eredeti videó elemet. Azzal, hogy beállítottuk a videónak az **autoplay** tulajdonságot, gondoskodtunk arról, hogy automatikusan induljon el a videó. Így a vásznon megjelenő videónak nem csak képe lesz, hanem hangja is. Azonban ha a fenti kódot megtekintjük a böngészőben, valószínűleg semmit nem fogunk látni a videóból. Ennek az oka az, hogy a vászon nem rajzolja újra önmagát, így amikor kirajzolta a videó első képkockáját, be is fejezte teendőit. Annak érdekében, hogy valóban megjelenjen a vásznon a videó, folyamatosan frissíteni kell a vásznat.

```
// Szintén csak akkor kezdjük el kirajzolni a videót, amikor elkezdődik a lejátszása
myVideo.onplay = function () {
    drawFrames();
}

// Ez a függvény fog azért felelni, hogy a vászon adott területét frissítse.
// Rekurzív saját magát hívja, ameddig véget nem ér a videó lejátszása.
// Annak érdekében, hogy kevesebbet kelljen dolgozni a böngészőnek,
// a vászon csak azon területét töröljük, ahol a videó található.
function drawFrames() {
    if (myVideo.ended || myVideo.paused) {
        return;
    }

    // Vászon törlése
    myContext.clearRect(220, 10, 300, 200);

    // Videó rajzolása
    myContext.drawImage(myVideo, 220, 10, 300, 200);

    // Vászon újrarajzolása megadott időközönként (25FPS)
```

```
    window.setTimeout(function () {
       drawFrames();
    }, 1000 / 25);
}
```

Az egyszerű alapműveletek után nézzük meg azt, hogy hogyan tudjuk a vászon képpontjait manipulálni. Ehhez három függvényt használhatunk: **createImageData, getImageData és a putImageData**. A **createImageData(width,height)** függvény segítségével egy adott méretű üres képpont halmazt hozhatunk létre, a **getImageData(posX,posY,width,height)** függvénnyel a vászon képpontjaiból ragadhatunk ki egy darabot. A **putImageData(data, posX,posY)** függvény pedig arra szolgál, hogy a vászon bizonyos képpontjait felülírjuk.

```
// Kép rajzolása
var imgCanvas = new Image();
imgCanvas.src = 'Media/Images/meandmyselft.png'
imgCanvas.onload = function () {
   myContext.drawImage(imgCanvas, 0, 0);
   // Kiragadjuk a vászon egyik felének képpontjait
   var pixels = myContext.getImageData(0,0,myCanvas.clientWidth/2,myCanvas.clientHeight);

   // Végigszaladunk a képpontokon, négyesével, mert egy képpont négy helynek felel meg
   // a tömbben (három alapszín + átlátszóság)
   for (var i = 0; i < pixels.data.length; i += 4) {
       // A képpont piros komponensének kinyerés
       var red = pixels.data[i];
       // A képpont zöld komponensének kinyerés
       var green = pixels.data[i + 1];
       // A képpont kék komponensének kinyerés
       var blue = pixels.data[i + 2];
       // A képpont átlátszóságának kinyerés
       var alpha = pixels.data[i + 3];
       // Színek súlyozott átlagolása => szürkített képpont
       var avg = 0.3 * red + 0.59 * green + 0.11 * blue;

       pixels.data[i] = avg;
       pixels.data[i + 1] = avg;
       pixels.data[i + 2] = avg;
       pixels.data[i + 3] = alpha;
   }

   // Vászon képpontjainak felülírása a módosított képpontokkal
   myContext.putImageData(pixels, 0, 0);
}
```

Azonban fontos megjegyezni, hogy a fent használt **getImageData** és **putImageData** függvényeknek nagy a számításigénye, ezért csak akkor használjuk őket, ha valóban indokolt. Használatuk során tartsuk szem előtt azt is, hogy minél kisebb területen kell képpontokat módosítanunk, annál gyorsabban végez vele a böngésző. Amennyiben Internet Explorer 9 alatt próbálgatjuk és használjuk a képpontok manipulálást, célszerű, ha kikapcsoljuk a hibakeresést, ellenkező esetben többszörösére is megnőhet a műveletek számításigénye.

Már csak egyetlen dolog maradt hátra, amit elvégezhetünk a vásznunk képpontjaival, az pedig nem más, mint a mentés. A vászon **toDataURL()** függvényével elmenthetjük a teljes tartalmát képként, méghozzá alapértelmezetten PNG formátumban. Ez a funkció akkor lehet nagyon hasznos, ha például a vásznon történt animációt, vagy rajzolást el szeretnénk menteni. Mentéskor azonban nem egy konkrét fájlként mentődik el az a kép, hanem a HTML5 újdonságai között szereplő **dataURL** formátumban. Ez a formátum egy speciális szöveges formátum, amit ha közelebbről megvizsgálunk, csupa számunkra értelmezhetetlen karakterből áll. Viszont ezek az értelmezhetetlen karakterek írják le a fájlunk bináris alakját. Szöveges mivoltának köszönhetően pedig egyszerűen elküldhetjük a kiszolgálónknak webes szolgáltatásokon keresztül.

```
// Kép rajzolása
var imgCanvas = new Image();
imgCanvas.src = 'Media/Images/meandmyselft.png'
imgCanvas.onload = function () {
    myContext.drawImage(imgCanvas, 0, 0);

    // Vászon mentése PNG formátumban.
    // FONTOS: Itt nem a context objektumot kell használni, hanem a vásznat
    var pngData = myCanvas.toDataURL();

    // Vászon mentése JPG formátumban
    // Szintén a vászon objektumot kell használni, nem a context objektumot
    var jpegData = myCanvas.toDataURL("image/jpeg")

    // A képek tartalmát leíró dataURL megtekintése új ablakokban
    var newWndPNG = window.open('', '', 'height=150,width=300');
    newWndPNG.document.write(pngData);
    var newWndJPEG = window.open('', '', 'height=150,width=300');
    newWndJPEG.document.write(jpegData);
}
```

A vászonnal végezhető rajzműveletek után most nézzük meg a vászon egyik legérdekesebb funkcióját, a geometriai transzformációkat. A kétdimenziós transzformációk segítségével például elforgathatjuk, eltolhatjuk vagy átméretezhetjük a vászon tartalmát. Azonban mielőtt még közelebbről is megismerkednénk a vászon transzformációs képességeivel, nézzük meg azt, hogy hogyan kezeli a vászon a transzformációkat.

A vászon inicializálásakor a háttérben létrejön egy alapértelmezett transzformációs mátrix. Majd amikor különböző transzformációikat hajtunk végre (például forgatás), akkor ezek a műveletek ezt az alapértelmezett transzformációs mátrixot fogják megszorozni a művelet mátrixával. Majd az után, hogy minden transzformációt elvégeztünk, a vászon megjeleníti a tartalmát úgy, hogy alkalmazza rá a módosított transzformációs mátrixot. Nyilvánvalóan amennyiben nem végeztünk egyetlen transzformációt sem, úgy az alapértelmezett mátrixot végrehajtva a végeredményben semmilyen változást nem fogunk tapasztalni. Azonban ha egymás után több transzformációt is alkalmazunk, akkor ügyelnünk kell az egyes műveletek sorrendjére, hiszen az egyes műveletek mátrixszorzást jelentenek (és mint tudjuk, a mátrixszorzás nem kommutatív művelet). Természetesen egymástól függetlenül is hajthatunk végre különböző transzformációkat, ebben az esetben viszont az egyes műveletcsoportok között el kell tárolnunk a vászon alapértelmezett mátrixát. Például ha át szeretnénk méretezni egy téglalapot, majd el szeretnénk forgatni egy

másikat, akkor a két művelet között el kell tárolni a vászon állapotát, különben mindkét téglalapon végre lesz hajtva mind a forgatás, mind pedig az átméretezés. Ezt az állapotmentést a vászon **save()** függvényével tudjuk megtenni, viszont tisztában kell lennünk azzal, hogy nem csak a transzformációs mátrix tartozik ehhez az állapothoz, hanem még számos más tulajdonság is (például **fillStyle**, **strokeStyle**, **lineWidth**, **font**, stb.). Az állapot visszatöltése pedig a **restore()** függvénnyel történik, ami visszaállítja azt az állapotot, amit a **save()** metódus során eltároltunk.

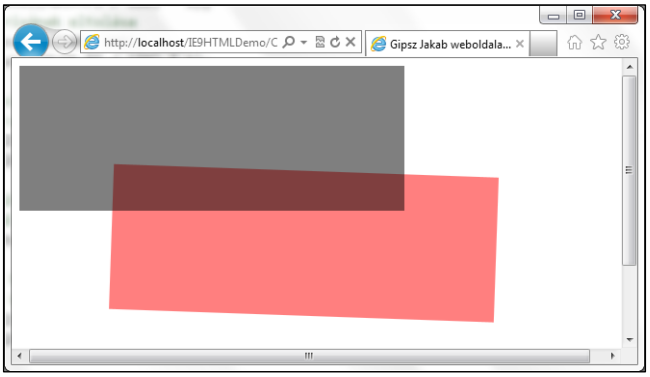
A vászon „legegyszerűbb" transzformációs függvénye a **transform(a,b,c,d,e,f)** metódus. Ez a függvény nem csinál mást, mint az alapértelmezett transzformációs mátrixot megszorozza a paraméterként átadott mátrixszal. A paramétert a következőképp foghatjuk fel:

$$\begin{matrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{matrix}$$

Tulajdonképpen ez az a függvény, aminek a segítségével bármilyen egyedi transzformációt alkalmazhatunk a vászonra. Egy másik, ehhez nagyon hasonló függvény a vászon **setTransform(a,b,c,d,e,f)** függvénye, ami az előbbi szorzással ellentétben egyszerűen csak felülírja a mátrixot. Szerencsénkre azonban nem csak ezeket az „egyszerű" függvényeket használhatjuk a különböző transzformációk során, és a gyakori transzformációk végrehajtásához szükséges mátrixot a vászon beépített függvényeivel is előállíthatjuk. Így például a forgatás, eltolás és átméretezés során használhatjuk a **rotat(angle)**, **translate(posX, posY)** és a **scale(width, heigth)** függvényeket is (mazochisták saját mátrixot is készíthetnek ☺).

```
var myCanvas = document.getElementById('myCanvas');
var myContext = myCanvas.getContext("2d");

// Vászon eredeti állapotának eltárolása
myContext.save();

// Vászon mátrixának elforgatása 1 fokkal
myContext.rotate((Math.PI / 180) * 1);
// Vászon mátrixának eltolása
myContext.translate(100, 100);
myContext.rotate((Math.PI / 180) * 1);

// Mivel rajzolás során módosítottuk a mátrixot,
// így a 0;0 koordinátákból kiinduló téglalap valójában a vászon közepére került
myContext.fillStyle = 'rgba(255,0,0,0.5)';
myContext.fillRect(0, 0, 400, 150);

// Ha nem állítanánk vissza az elmentett állapotot, akkor
// a következő téglalap szintén a vászon közepén foglalna helyet.
myContext.restore();

// Láthatjuk, hogy ennek a téglalapnak a koordinátái megegyeznek az elözővel,
// viszont ez került a "jó" helyre, míg a másik nem
myContext.fillStyle = 'rgba(0,0,0,0.5)';
myContext.fillRect(0, 0, 400, 150);
```

A fenti példában kirajzolt két téglalap során láthattuk, hogy bár mindkét téglalapot ugyanazokba a koordinátákba rajzoltuk, mégsem fedik el egymást (45. ábra), mégpedig azért, mert a két alakzat rajzolása között elmentettük a vászon állapotát.



45. ábra: Transzformációk

Mint ahogy azt már korábban is említettem, a vásznat úgy kell elképzelni, mint egy valódi rajz-vásznat, egy rajzoló felületet, amit a festő megfest. Ebből adódik az, hogy sajnos nem rendelkezik olyan egyszerűen és könnyen használható animációs eszközökkel, mint a Flash vagy a SilverLight, és az animálása egy kicsit összetett. Míg a fejlettebb technológiák esetében használhatunk időalapú-, vagy eseményvezérelt animációkat, addig a vászon esetében az animáció minden elemét kézzel kell programozni és újrarajzolni. Azonban mi nagyon lelkesek vagyunk, és ilyen apróságok nem veszik el a kedvünket az animálástól. Hiszen igazán látványos reklámokat, vagy játékokat csak úgy készíthetünk, ha megbarátkozunk a vászon animálásával. És igazság szerint ezt a barátkozást már akkor elkezdtük, amikor megismerkedtünk a vászon videókezelési képességeivel. A helyzet ebben az esetben is nagyon hasonló. Kell készítenünk egy olyan függvényt, ami folyamatosan frissíti a vászon tartalmát, és ha már van egy folyamatosan frissülő felületünk, akkor az animálás sem nehéz.

```
var myCanvas = document.getElementById('myCanvas');
var myContext = myCanvas.getContext("2d");

var currentRotation = 0; var rotationSpeed = 20;
window.onload = drawFrames;

function drawFrames() {
    myContext.clearRect(0, 0, myCanvas.clientWidth, myCanvas.clientHeight);
    drawLine();
    window.setTimeout(drawFrames, 1000 / 60);
}

// Forgó négyszög rajzolása
function drawLine() {
    myContext.save();

    // Mátrix eltolása a vászon közepébe
    myContext.translate(myCanvas.clientWidth / 2, myCanvas.clientHeight / 2);
```

```
    // Tartalom forgatása x fokkal
    myContext.rotate((Math.PI / 180) * currentRotation);

    // Téglalap rajzolása. A negatív koordinátákra azért van szükség,
    // hogy a transzformáció alkalmazása során a négyszög saját közepe körül forogjon
    myContext.fillRect(-100, -100, 200, 200);

    myContext.restore();
    currentRotation += rotationSpeed;
}
```

SVG támogatás

Az előző pár oldalon keresztül megismerkedhettünk a HTML5 egyik legizgalmasabb elemével, a vászonnal. Azt ugyan láthattuk, hogy segítségével nagyon látványos vizuális hatásokat készíthetünk, viszont van egy olyan tulajdonsága is, hogy az általa megjelenített kép pixelgrafikus. Ez természetesen nem baj, viszont néha szükségszerűen olyan ábrákat kell megjelenítenünk, használnunk, amelyeket kedvünk szerint átméretezhetünk úgy, hogy közben nem romlik a kép minősége.

Az SVG (Scalable Vector Graphics) a számítástechnikában egy gyakran alkalmazott leíró nyelv, aminek segítségével egyszerűen leírhatjuk a vektorgrafikus képeinket. Az SVG a HTML-hez hasonlóan egy XML alapú leíró nyelv, így a használata nem igényel semmilyen speciális technológiai tudást, csupán tisztában kell lennünk azzal, hogy hogyan épül fel, milyen elemei vannak.

```
<article>
    <svg width="100%" height="100%">
        <!--Egyszerű vonal-->
        <line x1="0" x2="10" y1="100" y2="100" />
        <!--Egyszerű téglalap-->
        <rect class="rect" x="20" y="20" width="200" height="60" />
        <!--Lekerekített téglalap-->
        <rect x="20" y="200" width="300" height="40" rx="10" ry="10" />
        <!--Kör-->
        <circle cx="250" cy="50" r="20" />
        <!--Ellipszis-->
        <ellipse cx="100" cy="150" rx="80" ry="10" />
        <!--Szöveg-->
        <text x="20" y="120" style="font-size:1cm; fill:red;">Internet Explorer 9</text>
    </svg>
</article>
```

Azonban nem csak egyszerűbb alakzatokat írhatunk le az SVG segítségével, hanem összetett vonalakat, ábrákat is.

```
<polyline fill="none" stroke="blue" stroke-width="10" points="
        050,375 150,375 150,325 250,325 250,375 350,375
        350,250 450,250 450,375 550,375 550,175 650,175 650,375" />

<polygon fill="lime" stroke="blue" stroke-width="10" points="
        850,75  958,137.5 958,262.5 850,325 742,262.6 742,137.5" />
```

Természetesen egy komplexebb ábra rajzolása során az ábránkat lehetetlen lenne kézzel elkészíteni, ezért számtalan vektorgrafikus képszerkesztő program létezik a piacon, és közülük szinte az összes képes a rajzolt képet SVG formátumba exportálni.
Maga a nyelv olyannyira hasonlít a HTML-re, hogy egy-egy elem rendelkezik saját **style** és **class** tulajdonsággal, ami megfelel a HTML-beli **style** és **class** tulajdonságoknak, tehát még CSS stílusokat is alkalmazhatunk az egyes alakzatainkra (itt a CSS stílus kicsit másképp értelmezhető, mint hagyományos HTML környezetben, hiszen SVG specifikus „stílusokat" is használhatunk).
Mint ahogy azt a vásznon is megtehettük, az SVG segítségével is megtehetjük, hogy nem csak színekkel tölthetjük ki az alakzatainkat, hanem színátmenetekkel.

```
<svg width="50%" height="50%">
    <defs>
        <lineargradient id="MyGradient" x1="0%" y1="0%" x2="0%" y2="100%">
            <stop offset="0%" style="stop-color: rgb(255,0,0); stop-opacity: 1" />
            <stop offset="100%" style="stop-color: rgb(255,255,0); stop-opacity: 1" />
        </lineargradient>
    </defs>
    <rect x="10" y="10" width="300" height="100" style="fill: url(#MyGradient)">
    </rect>
</svg>
```

Egy másik nagy különbség a vászon és az SVG között, hogy az SVG elemeit könnyedén elérhetjük JavaScriptből, így a különböző animációk programozása rendkívül egyszerű.

```
<svg width="50%" height="50%">
    <a onclick="javascript:alert('Internet Explorer 9');">
       <rect x="20" y="20" width="250" height="100"
             style="fill: blue; cursor: pointer; opacity: 0.9">
       </rect>
    </a>
</svg>
```

Természetesen az SVG leíró nyelv segítségével nagyon komplex grafikákat is leírhatunk, sőt vektorgrafikus játékokat és alkalmazásokat is készíthetünk. Azonban egy összetett nyelv révén külön a teljes körű bemutatását a fejezet terjedelmi korlátai nem teszik lehetővé.

CSS3 támogatás

Mint ahogy azt már említettem, a CSS a webes fejlesztés egyik alapvető pillére, hiszen CSS segítségével adhatunk megjelenést és külalakot weboldalainknak. Az Internet Explorer 9 által is támogatott CSS3 legnagyobb előnye az, hogy modulokból áll, és ezeket a modulokat egymástól függetlenül fejlesztik. Így ha a szabvány készítői befejeznek egy modult, a böngészők csomagban alkalmazhatják azt, aminek köszönhetően sokkal hamarabb eljuthat a fejlesztőkhöz és a felhasználókhoz, mint hogyha meg kellene várni, míg a szabvány teljes egészében elkészülne. Ilyen modul például a **Background & Borders**, a **Colors**, a **Selectors** modul és még számos egyéb.
Csakúgy, mint a HTML5 szabvány készítésekor, a CSS3 moduljainak készítésekor is szem előtt tartották a webfejlesztők igényeit, és éppen ezért olyan funkciók kerültek bele az új CSS nyelvbe, amelyek megkönnyítik, leegyszerűsítik a dolgunkat.
Azonban még mielőtt tovább mennénk, nézzük meg, hogy hogyan épül fel a CSS nyelv. Két fő alkotóeleme van: azok a szabályok, amelyeknek alapján megkereshetjük a dokumentumaink

bizonyos elemeit (CSS kiválasztók, szelektorok), és azok a stílusok, amelyeket alkalmazhatunk a megtalált elemekre.
Az egyes elemek megkeresésére és kiválasztására használt szabályok között léteznek egyszerűbbek és összetettebbek is. Az egyszerűbb kiválasztási módszerek közé sorolhatóak például azok a szabályok, amelyek egy-egy elem típusa, azonosítója vagy attribútumai alapján keresik meg a kívánt elemet, míg az összetettek közé a pszeudó kiválasztók és a pszeudó elemek.
A leggyakrabban használt egyszerű kiválasztó az osztály kiválasztó. Szinten nincs olyan weblap, amely ne alkalmazná őket. A népszerűségének oka az, hogy segítségével olyan „osztályba", vagy csokorba foglalhatunk össze tetszőleges számú stílust, aminek többnyire beszédes neve van (például **pirosDoboz**). És attól kezdve, hogy van neve, a HTML elemek leírásakor könnyen meg tudjuk címezni az így definiált „osztályokat". A népszerűségének másik oka, hogy ha külső szemlélőként vizsgáljuk meg az oldal felépítését, akkor nagy segítségünkre lehetnek a tájékozódásban is, hiszen hogyha az elemek beszédes CSS osztálynevekkel rendelkeznek, akkor abból tudunk következtetni az elem funkciójára is.

```
<head>
    <title>>Gipsz Jakab weboldala - CSS</title>
    <style>
        .kepFeltoltes        {
            background-color: Red;
            color: White;
        }
    </style>
</head>
<body>
    <article>
        <section class="kepFeltoltes">
            .... <input type="file" /> ...
        </section>
    </article>
</body>
```

Az azonosító alapú kiválasztók segítségével az egyes elemek azonosítóit tudjuk latba vetni annak érdekében, hogy megtaláljuk őket. Az azonosító alapú kiválasztók alkalmazása is egy rendkívül széles körben elterjedt és jól bevált dolog, hiszen ameddig nem voltak HTML5 szemantikus vezérlő elemek, addig a fejlesztők a weboldal egyes szerkezeti elemeit azonosítók alapján különítették el egymástól. Például ha a weboldal fejlécét leíró elemei egy ** ... ** elem között foglalnak helyet, akkor nem készítünk egy külön CSS osztályt a fejlécnek, hiszen már adott az azonosítója.

```
<head>
    <title>>Gipsz Jakab weboldala - CSS</title>
    <style>
        .header      { background-color: rgba(0,0,0,0.6); }
        .mainContent { background-color: rgba(0,0,0,0.2); }
        .footer      { background-color: rgba(0,0,0,0.4); }
    </style>
</head>
<body>
```

```
    <section id="header">...</section>
    <section id="mainContent"></section>
    <section id="footer"></section>
</body>
```

Szintén a legnépszerűbb kiválasztók sorába sorolhatjuk azokat a kiválasztókat, amelyek megvizsgálják az elemek típusait, és annak alapján döntik el, hogy alkalmazni kell-e a stílusokat vagy sem. Így például globálisan beállíthatjuk azt, hogy webhely bekezdéseinek milyen legyen a betűszíne, vagy azt, hogy a szöveges elemeknek milyen legyen a betűtípusa.

```
<style>
    /* A dokumentum összes <span> elemére alkalmazzuk a stílust */
    span
    {
        color: color: rgba(0,0,0,0.2);
        font-style: italic;
    }
    /* A dokumentum összes <p> elemére alkalmazzuk a stílust */
    p
    {
        padding-left: 10px;
        font-size: 11pt;
    }
</style>
```

Az attribútum alapú kiválasztók már egy kicsit bonyolultabbak az előzőeknél. Hiszen többféle szabályt is megadhatunk az elemek kereséséhez. A legegyszerűbb eset az, amikor azt szeretnénk megvizsgálni, hogy az adott elem rendelkezik-e valamilyen tulajdonsággal. Ezen felül vizsgálhatjuk még azt is, hogy az adott tulajdonság milyen értékkel végződik, kezdődik, valamint milyen értéket tartalmaz. Ezt a típusú kiválasztót célszerű az előző kiválasztók valamelyikével kombinálni, hiszen akkor vehetjük igazán a hasznát.

```
 <style>
    /* Különböző beviteli mezőkre más-más stílust alkalmazunk */
    input[type="checkbox"] { border: solid 1px gray; }
    input[type="text"]     { border: solid 1px gray; }
    input[type="file"]     { border: solid 1px gray; }

    /* Azokra az "IMG" elemekre érvényes, amelyek "contenttype"
    tulajdonsága "alma" értékkel kezdődik */
    img[src^="alma"] { border:solid 1px rgb(255,255,0)  }

    /* Azokra az "IMG" elemekre érvényes, amelyek "contenttype"
    tulajdonsága "jpeg"-el végződik */
    img[src$="jpg"]  { border:solid 1px rgb(0,255,0); }

    /* Azokra az "IMG" elemekre érvényes, amelyek "contenttype"
    tulajdonsága tartalmazza a "png"-t */
    img[src*="png"]  { border:solid 1px rgb(0,255,255); }
</style>
```

A CSS3 legizgalmasabb újdonságai azonban nem az egyszerű kiválasztók körében fedezhetőek fel, hanem az összetettekében: a pszeudó kiválasztókban és osztályokban. Mint ahogy nevük is

mutatja, ezek „ál” kiválasztó szabályok, és nem az elemek konkrét tulajdonságait vizsgálják, hanem például a dokumentum fában betöltött szerepeit vagy bizonyos állapotait. Például annak függvényében tudunk különböző stílust alkalmazni egy hivatkozásra, hogy a kurzor a hivatkozás felett van vagy sem, de vizsgálhatjuk pl. azt is, hogy egy **checkbox** vezérlő elem be van-e pipálva vagy sem. Továbbá olyan izgalmas kiválasztókat is találunk a pszeudó kiválasztók körében, mint például a negáló és az n-edik elem kiválasztó.
Mint ahogy azt már tudjuk, a HTML lehetőséget biztosít arra, hogy az egyes elemeket azonosítóval lássuk el. Az azonosítók számos dolog miatt hasznosak, többek között azért, mert a weboldal címén keresztül közölhetjük azt a böngészővel, hogy ugorjon a dokumentum bizonyos részeire. Például ha van tíz bekezdésünk az oldalon, és mindegyik bekezdésnek van egyedi azonosítója, akkor **http://www.valami.com/pelda.htm#bekezdes03** cím beírásával a böngésző a **pelda.htm** dokumentum **„bekezdes03”** azonosítójával rendelkező elemére fog ugrani. A CSS3 **:target** kiválasztójának segítségével pedig azonosíthatjuk azt, hogy a webhely elemeire történt-e ilyen módú hivatkozás vagy sem.

```
<head>
    <style>
        P:target
        {
            font-size: 200%;
            color: Red;
        }
    </style>
</head>
<body>
    <article>
        <p id="chapter1">
            Internet Explorer 9</p>
        <p id="chapter2">
            Internet Explorer 9</p>
        <p id="chapter3">
            Internet Explorer 9</p>
    </article>
</body>
```

Például ha a fenti weboldalt megnyitjuk **#chapter2** hivatkozással, akkor a második bekezdésünk szövegének színe piros lesz, és a mérete pedig kétszer akkora lesz, mint a többi bekezdésnek.
Azonban az egyes elemek állapotát felhasználva is tudunk szabályokat alkalmazni. Például elrejthetjük azokat az elemeket, amelyek le vannak tiltva.

```
<head>
    <style>
        input[type="text"] { background-color: Red; }

        input[type="text"]:disabled { opacity: 0.1; }
    </style>
</head>
<body>
    <article>
        <input type="text" value="Internet Explorer" />
```

```
        <!--Letiltott elem-->
        <input type="text" disabled="disabled" value="Internet Explorer" />
    </article>
</body>
```

A CSS3 legizgalmasabb, leghasznosabb és legkomplexebb kiválasztói az n-edik elem kiválasztók. Ezeknek a kiválasztóknak a segítségével az elemek dokumentumfán belül betöltött elhelyezkedése alapján tudunk kiválasztási szabályokat létrehozni. Például megkereshetjük egy táblázat első hat sorát, vagy páros, illetve páratlan sorait.
Az n-edik elem kiválasztók körébe négy kiválasztó tartozik:

- **:nth-child(an+b)**
- **:nth-last-child(an+b)**
- **:nth-of-type(an+b)**
- **:nth-last-of-type(an+b)**

Először is nézzük meg azt, hogy mit jelent a zárójelbe írt kifejezés, az **an+b**. Ha a kifejezés például **3n+0**, azaz **3n**, akkor a kiválasztó minden harmadik elemet fogja megtalálni (azt hogy mihez képest, később meglátjuk). Ha a kifejezés **0n+3**, azaz **3**, akkor a kiválasztó csak a harmadik elemet fogja megtalálni. Azonban ha a kifejezés például **3n+1**, akkor a szabály az első, és az azt követő minden harmadik elemet fogja megtalálni (első, negyedik, hetedik). A **2n+1** kifejezés például a páratlan elemeket, míg a **2n+0** a páros elemeket keresi meg. A páratlan, illetve páros elemeket helyettesíthetjük az **odd**, illetve **even** kifejezésekkel is.

És most nézzük, meg mi a különbség a négy kiválasztó között. Az első két szabály az adott elem szülőeleméhez képest az n-edik elemeket választja ki, míg a második két függvény az adott szülőeleméhez képest csak azokat az n-edik elemeket, amelyeknek a típusa megegyezik az adott elem típusával.

```
<style>
    /* A kiválasztott elem: <span>első szöveg</span> */
    span:nth-child(2n)   { color:Red; }

    /* A kiválasztott elem: <span>második szöveg</span> */
    span:nth-of-type(2n) { color:Green; }
</style>
...
<body>
    <a>első hivatkozás </a>
    <span>első szöveg</span>
    <span>második szöveg</span>
</body>
```

Az **nth-child(an+b)** és az **nth-of-type(an+b)** az adott elem szülőelemének első elemétől kezdi a kiválasztást, míg az **nth-last-child(an+b)** és az **nth-last-of-type(an+b)** a szülőelem utolsó

```
<style>
    /* Első elem kiválasztása.
       A kiválasztott elem: <span>első szöveg</span> */
    span:nth-child(1)      { color:Red; }

    /* Első elem kiválasztása (szülő elem utolsó gyerekétől kezdve).
```

```
       A kiválasztott elem: <span>harmadik szöveg</span> */
    span:nth-last-child(1) { color:Green; }
</style>
...
<body>
    <span>első szöveg</span>
    <span>második szöveg</span>
    <span>harmadik szöveg</span>
</body>
```

Első ránézésre nem tűnik egyszerűnek a használata, azonban ha sikerül megértenünk a lényegét, akkor már sokkal több értelmet nyer a kifejezés. Az alábbi példa jól szemlélteti, hogy mennyire egyszerű megformázni egy táblázat páros, illetve páratlan sorait.

```
<style>
    /* Táblázat páratlan sorainak formázása (2n+1) */
    tr:nth-of-type(odd)      { background-color: rgba(0,0,0,0.2); }

    /* Táblázat páros sorainak formázása (2n+0)    */
    tr:nth-of-type(even)     { background-color: rgba(0,0,0,0.1); }
</style>
...
<body>
    <table>
        <tr><td>Teszt #1</td></tr>
        <tr><td>Teszt #2</td></tr>
        <tr><td>Teszt #3</td></tr>
        <tr><td>Teszt #4</td></tr>
    </table>
</body>
```

Természetesen az egyszerűbb n-edik elem szabályokra van önálló beépített kiválasztó is. Például egy elem szülőelemének első gyerek elemét kiválaszthatjuk az **:nth-child(1)** szabállyal, de használhatjuk helyette a **:first-child** kiválasztót is.

Bizonyos esetekben szükség lehet arra, hogy meghatározzuk azokat az elemeket, amelyek nem tartalmaznak semmilyen más gyerek elemet, vagy szöveget. Ezeket az elemeket az **:empty** kiválasztóval tudjuk megkeresni,

```
<style>
    div       {  height:30px; width:200px; }
    div:empty {  background-color:red;     }
</style>
...
<body>
    <!--Ennek a doboznak a háttere piros lesz.-->
    <div></div>
    <!--Ennek a pedig áttetsző.-->
    <div>Szöveg</div>
</body>
```

A CSS3 legizgalmasabb kiválasztói között szerepel még a negáló szabály is. Ennek a kiválasztónak a segítségével nem azt vizsgáljuk, hogy egy elemre alkalmazható-e a kiválasztási szabály, hanem azt,

hogy mely elemekre nem alkalmazható. A fenti példánál maradva könnyen megérthetjük működését.

```
<style>
    div             {  height:30px; width:200px; }
    div:empty       {  background-color:red;     }
    div:not(:empty) {  background-color:green;   }
</style>
...
<body>
    <!--Ennek a doboznak a háttere PIROS lesz.-->
    <div></div>
     <!--Ennek a doboznak a háttere ZÖLD lesz.-->
    <div>Szöveg</div>
</body>
```

Abban az esetben is hasznos lehet, hogy ha a megkeresendő elemet csak nagyon bonyolult szabállyal tudjuk megtalálni.

```
/* Meg szeretnénk találni a dokumentum összes fejlécét,
   kivéve az elsőt és az utolsót. */

/* Működőképes megoldás, viszont átláthatatlan */
h3:nth-of-type(n+2):nth-last-of-type(n+2) { background-color: Red; }

/* Lényegesebben átláthatóbb, de ugyan az a végeredmény */
h3:not(:first-of-type):not(:last-of-type) { background-color: Lime; }
```

A CSS3-ban léteznek olyan speciális kiválasztó szabályok is, amelyek eltérnek a korábban bemutatottaktól, hiszen nincsenek semmilyen összefüggésben a dokumentum szerkezeti felépítésével. Ilyen kiválasztó például a **`::first-line`** és a **`::first-letter.`** Ezzel a két kiválasztóval az adott elem első sorát, illetve első betűjét tudjuk kiválasztani. Attól függően pedig, hogy mekkora a böngészőablak mérete, a **`::first-line`** kiválasztóval megtalált elem, azaz inkább szöveg folyamatosan változhat, és ahogy csökkentjük például az ablak méretét, úgy csökken azoknak a szavaknak a száma, amit a szabály megtalál.

```
<style>
    p               { color: red; font-size: 12pt; }
    p::first-letter { color: green; font-size: 200%; }
    p::first-line   { color: blue; }
</style>
...
<body>
    <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Maecenas porttitor
congue massa. Fusce posuere, magna sed pulvinar ultricies, purus lectus malesuada libero,
sit amet commodo magna eros quis urna. Nunc viverra imperdiet enim. Fusce est. Vivamusa
tellus. </p>
</body>
```

Elsőre nagyon meglephet bennünket az, hogy a CSS3 segítségével képesek vagyunk dinamikusan tartalmat beszúrni az egyes elemeink elé, illetve mögé. A meglepetést még az is fokozhatja, hogy mindezt speciális kiválasztók segítségével tehetjük meg. Ez a két speciális kiválasztó pedig nem más, mint a **`:before`** és az **`:after`**.

```
<style>
    /* A H1 elem kirajzolása során növeljük a főcímhez tartozó számlálót,
       és lenullázzuk az alcímekhez tartozó számlálót */
    H1  { counter-increment: chapter; counter-reset : section;}

    /* A H1 elem elé beszúrjuk a főcímhez tartozó számlálót és
       egy kis statikus szöveget */
    H1:before { content: counter(chapter) '. fejezet: ';  }

    /* A H2 elem kirajzolásakor növeljük az alcímhez tartozó számlálót */
    H2 { counter-increment: section; }

    /* A H2 elem elé beszúrjuk a főcím és az alcím számlálóját is */
    H2:before {content: counter(chapter) "." counter(section) " "; }
</style>
...
<h1>Internet Explorer 9 </h1>        <!--1. fejezet: Internet Explorer 9-->
    <h2>Hardveres gyorsítás</h2>     <!--1.1 Hardveres gyorsítás-->
    <h2>Új felhasználó felület</h2>  <!--1.2 Új felhasználó felület-->
    <h2>Új biztonsági funkciók</h2>  <!--1.3 Új biztonsági funkciók-->
<h1>HTML5</h1>                       <!--2. fejezet: HTML5-->
    <h2>HTML leíró nyelv</h2>        <!--2.1 HTML leíró nyelv-->
    <h2>CSS stílusleíró nyelv</h2>   <!--2.2 CSS stílusleíró nyelv-->
    <h2>JavaScript</h2>              <!--2.3 JavaScript-->
```

A CSS3 újdonságai azonban nem merülnek ki a kiválasztókban. Azt már láthattuk, hogy hogyan lehet különféle szabályok segítségével stílusokat definiálni. De mi a helyzet akkor, hogy ha nem a dokumentum elemeire szeretnénk szabályokat definiálni, hanem azt szeretnénk meghatározni, hogy hogyan nézzen ki a weboldal például mobil megjelenítőn, táblagépen vagy netbookon? A megoldást a CSS3 **Media Queries** modulja szolgáltatja, aminek keretében a weboldalt megjelenítő eszköz tulajdonságainak alapján tudjuk szabályozni a weboldal megjelenését. Például beállíthatjuk azt, hogy más stílus fájl töltődjön be akkor, ha a megjelenítő eszköz felbontása kisebb, mint egy adott érték. Továbbá azt is meghatározhatjuk, hogy hogyan nézzen ki a nyomtatás során. És mindehhez nem szükséges újra felépítenünk az oldal szerkezetét, csupán készítenünk kell egy másik CSS fájlt, ami a kisfelbontású megjelenítőkre lett optimalizálva.

```
<head>
    <!--Ez a stílus fájl minden esetben betöltődik -->
    <link rel="stylesheet" type="text/css" media="all" href="Styles/Default.css">

    <!--Ez a stílus fájl pedig akkor, ha kinyomtatjuk az oldalt-->
    <link rel="stylesheet" type="text/css" media="print" href="Styles/Print.css">

    <!--Ez a stílus fájl pedig csak akkor, ha 16:9-es a képaránya a megjelenítőnek-->
    <link rel="stylesheet" media="device-aspect-ratio: 16/9"
          href="Sytyles/WideScreen.css" />

    <!--Ez pedig akkor, ha a szélessége nem nagyobb, mint 600 pixel-->
    <link rel="stylesheet" media="max-width: 600px" href="Sytyles/SmallScreen.css" />
</head>
```

Természetesen nem csak a betöltendő stílusfájlokra adhatunk meg ilyen speciális feltételeket, hanem a CSS fájlokon, vagy stílusokon belül is.

```
<head>
    <style type="text/css">
        /* Ez a stílus minden esetben alkalmazva lesz */
        .doboz
        {
            width: 400px;
            height: 200px;
            border: solid 1px gray;
        }

        /* Csak akkor lesz piros a doboz, ha a böngésző szélessége
           600 és 900 pixel között van. Természetesen ha átméretezzük a böngészőt,
           akkor az IE újból leellenőrzi, hogy a megfelelő méretek között van-e
           az ablak */
        @media screen and (min-width: 600px) and (max-width: 900px)
        {
            .doboz
            {
                background: red;
            }
        }

        /* Csak akkor lesz zöld a doboz, ha a megjelenítő eszköz maximális
           szélessége 480 pixel */
        @media screen and (max-device-width: 480px)
        {
            .doboz
            {
                width: 200px; height:100px;
                background: green;
            }
        }
    </style>
</head>
<body>
    <div class="doboz">
    </div>
</body>
```

A webes arculattervezés egyik legnagyobb ellensége a különböző felbontású megjelenítők, azonban ez az „ellenség” kellő körültekintéssel alkalmazott **Media Queries** modul segítségével legyőzhető.

Azonban a webes arculattervezést még az is nehezíti, hogy a látványosra sikerült tervet át kell ültetni a gyakorlatba. Ehhez pedig a terven szereplő minden színt át kell alakítani olyan formátumúra, amellyel képes elboldogulni a böngésző. A CSS3 **Colors** modulja pedig ezt a műveletet egyszerűsíti le azáltal, hogy a korábbi kezdetleges színmegadási formákat kiegészíti a népszerű színmegadási módokkal. Ilyen forma például az **RGB**, illetve **RGBA** modell ahol a szín

megadása a vörös, zöld és kék (illetve áttetszőség) komponensek segítségével történik, vagy a **HLS** (Hue, Saturation, Lightness), illetve a **HLSA** modell.

```
<style>
    .Sarga
    {
        background-color: Yellow;
        background-color: #FFF500;
        background-color: rgb(255, 245,0);
        background-color: rgba(255, 245, 0, 1.0);
        background-color: hsl(58, 100%, 50%);
        background-color: hsla(58, 100%, 95%, 1.0);
    }
</style>
```

A CSS segítségével ugyan nagyon látványos megjelenésű weboldalakat készíthetünk, ám a stílusleíró nyelv korábbi változatai több szempontból is gátolták az alkotói kreativitást. Az egyik ilyen szempont az az, hogy hiába készítünk látványos arculattervet, lehetnek olyan elemei is, amit egyszerűen nem lehet megvalósítani a CSS segítségével. Ilyen elemek lehetnek például az egyedi betűtípusok. A CSS3 elődeivel ugyanis csak olyan betűtípussal jeleníthettünk meg szövegeket, amelyek telepítve voltak a felhasználó számítógépén, így a használható betűtípusok száma nagy mértékben korlátozódott. A CSS3 **Fonts** modulja viszont megadja nekünk azt a lehetőséget, hogy betűtípusokat ágyazzunk be a weboldalainkba, és ennek révén egyedi betűtípusokkal jelenítsük meg weboldalainkat. Többek között olyan népszerű formátumokat használhatunk, mint például az **EOT** (**Embedded OpenType**), a **TTF** (**TrueType Font**) vagy a **WOFF** (**Web Open Font Format**).

```
<head>
    <style>
        @font-face {
            font-family: 'Sample1';
            src: url('Media/Fonts/sample1.ttf');
        }
        @font-face {
            font-family: 'Sample2';
            src: url('Media/Fonts/sample2.woff');
        }

        p               { font-size: 150%;}
        p:nth-of-type(1) { font-family: Sample1; }
        p:nth-of-type(2) { font-family: Sample2; }
    </style>
</head>
<body>
    <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
       Maecenas porttitor congue massa. </p>
    <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
       Maecenas porttitor congue massa. </p>
</body>
```

A CSS3 moduljainak sorába tarozik egy **Background & Borders** modul is. Mint ahogy a neve is sejteti, ez a modul írja le a háttérkezeléshez és a körvonalhoz használható stílusokat. Azonban

ennél sokkal többet is rejt magában. Legyen szó akár webes alkalmazásról, weboldalról, vagy hagyományos asztali alkalmazásról, a megjelenést illetően egyre divatosabbak a lekerekített sarkú ablakok, szövegdobozok és egyéb elemek. Ugyan a CSS3 előtt is elláthattuk lekerekített dobozokkal a weboldalainkat, azonban ehhez képekkel kellett trükköznünk. A CSS3-ban viszont ezt egyszerűen megtehetjük.

```
<style>
    .addRoundedCorners
    {
        /* Lekerekítés alkalmazása mind a négy sarokra */
        border-radius: 10px;
            /* Lekerekítés alkalmazása mind a négy sarokra */
        border-radius: 5px 5px 5px 5px;

        /* Lekerekítés alkalmazása a bal-felső sarokra */
        border-top-left-radius: 20px 20px;
    }
</style>
```

Azonban ez még nem minden, ugyanis a CSS3 révén árnyékokkal is elláthatjuk az elemeinket, méghozzá nagyon egyszerűen.

```
<style>
    .addShadow
    {
        /* Árnyék elmosás nélkül. Az első két paraméter az eltolás mértéke,
            a harmadik pedig a színe */
        box-shadow: 10px 10px #888;

        /* Az első két paraméter az árnyék eltolásának mértéke az eredeti objektumtól.
            A harmadik paraméter az elmosás mértéke, a negyedik pedig az árnyék színe  */
        box-shadow: 10px 10px 10px #888;

        /* Árnyék elmosással. Az 5px 1px értékek határozzák meg az elmosás jellegét  */
        box-shadow: 0 0 5px 1px #888;

        /* Belső árnyék, elmosással */
        box-shadow: 10px 10px 10px #888 inset;
    }
</style>
```

Továbbá a **Backgrounds & Borders** modul még azt is leírja, hogy hogyan lehet képet beállítani a keret hátttereként (**border-image-source**).

Azt már láthattuk, hogy a vászon és az SVG segítségével miképp alkalmazhatunk különböző kétdimenziós transzformációkat bizonyos elemekre. És azt is tudjuk, hogy mire való a CSS leíró nyelv: alapvetően stílusok megadására. A CSS3 készítői azonban a két funkciót egybefésültek, aminek révén stílusokon keresztül is elforgathatjuk, eltolhatjuk, megnyújthatjuk és átméretezhetjük az elemeinket. Csakúgy, mint a vászon esetében, itt is fontos az egyes transzformációk sorrendje, és ha felcseréljük őket, akkor nagyon megváltozhat a végeredmény.

```
<style>
        .trasform
        {
```

```
            /* Objektum elforgatása, átméretezése, eltolása és elferdítése.
               Sajnos a böngészők csak előtaggal támogatják. */
            -ms-transform: rotate(10deg)
                           scale(1.75, 1.75)
                           translate(300px, 100px)
                           skewX(20deg);

            /* Szabványos stílus definiálás */
            transform: rotate(10deg) scale(1.75, 1.75)
                       translate(300px, 100px) skewX(20deg);
        }
</style>
```

A fent bemutatott példákon és modulokon felül még számos érdekességet tartogat a CSS3, azonban ezek az érdekességek jelenleg még csak ötletek, vagy félig vannak kidolgozva. Például némelyikkel nem csak kettő-, de háromdimenziós transzformációkat is megvalósíthatunk, készíthetünk animációkat és áttűnéseket, valamint kialakíthatjuk az oldalunk szerkezetét CSS „sablonok” segítségével is.

ECMAScript 5 újdonságok

Az Internet Explorer 9 ECMAScript támogatását egy mondatban talán úgy lehet összefoglalni, hogy a szabvány támogatásának révén böngészőfüggetlen parancsfájlokat készíthetünk. Ez az egy mondat önállóan is nagy jelentőséggel bír, hiszen aki már foglalkozott HTML fejlesztéssel, bizonyára tudja, hogy mennyire nem egyszerű olyan parancsfájlokat készíteni, amelyek minden böngészővel kompatibilisek. Ezen felül fejlődött maga a JavaScript nyelv is és kiegészült számtalan olyan hasznos funkcióval, amelyre már régóta igény volt.

A legérdekesebb újdonságok közé sorolható a JavaScript **`Object`** konstruktor kibővítése, amelynek eredményeképp egyszerűen tudunk saját objektumokat létrehozni, másolni, valamint a saját objektumunk tulajdonságait is egyszerűbben kezelhetjük az új **`getter`** és **`setter`** funkciókkal.

```
<script type="text/javascript">
    // Ember objektum létrehozása
    var ember = {
        nev: "Ismeretlen ember",
        lakCim: "Ismeretlen lakcím",
        szuletesiDatum: new Date()
    };
    // Eredmeny: tesztelek.name = "Ismeretlen ember"
    var tesztelek = Object.create(ember);

    // Ember objektum kibővítése -> Férfi objektum létrehozása.
    // Az újonnan létrehozott objektum "nem" tulajdonsága alapértelmezetten férfi lesz.
    var ferfi = Object.create(ember);
    Object.defineProperty(ferfi, "nem", { value: "férfi" });

    // tesztbenedek.nem = "Férfi"
    var tesztbenedek = Object.create(ferfi);
</script>
```

A JavaScript nyelvben nagy újdonságnak számító **getter** és **setter** segítségével objektumorientáltabb módon tudunk tulajdonságokat hozzárendelni az objektumainkhoz, ezáltal „okosabb” objektumokat hozhatunk létre.

```
// Ember objektum kibővítése -> Férfi objektum létrehozása.
// Az újonnan létrehozott objektum "nem" tulajdonsága alapértelmezetten férfi lesz.
var ferfi = Object.create(ember);
Object.defineProperty(ferfi, "nem",
{
   // A tulajdonság mindig a "férfi" értéket adja vissza
   get: function () { return "férfi" },

   // Ha módosítani próbálnánk a tulajdonságot, akkor a JavaScript hibaüzenetet dob
   set: function (value) { alert('Ezt a tulajdonságot nem lehet módosítani'); }
});
// ferfi objektum példányosítása
var tesztpista = Object.create(ferfi);

// "nem" tulajdonság felülírása. A böngésző feldobja a hibaüzenetet és nem fog történni
// semmi, a "nem" tulajdonság továbbra is "ferfi" marad
tesztpista.nem = "nő"
```

Azonban az új JavaScript nyelv objektumkezelési képességein túl is találunk érdekes és hasznos dolgokat, függvényeket. Ilyen hasznos függvények például azok, amelyeknek a segítségével megkereshetjük elemeinket a dokumentumfában. A **document.getElementsByClassName** akkor lesz segítségünkre, amikor a CSS osztálynevek alapján szeretnénk bizonyos elemeket megkeresni, a **document.querySelector** és a **document.querySelectorAll** függvények pedig akkor, ha CSS szabályok alapján szeretnénk keresni.

```
<article>
    <header id="MainHeader">
       <h1 class="head">Cím</h1>
    </header>
    <p>Első bekezdés. <span class="bold">Félkövér szöveg.</span></p>
    <p class="bold">Második bekezdés.</p>
</article>
<script type="text/javascript">

    // Eredmény: "Cím"
    var firstHeaderText = document.getElementsByClassName("head")[0].innerText;

    // Eredmény: "Második bekezdés"
    var secondParagraph = document.querySelector("p:nth-of-type(2)").innerText;

    // Eredmény: "Félkövér szöveg"
    var boldText = document.querySelector("span.bold").innerText;

    // Eredmény: "<h1 class="head">Cím</h1>"
    var header = document.querySelector("#MainHeader").innerHTML;
</script>
```

A JavaScript területet érintő hasznos újítások sorából érdemes még kiemelni az új tömb funkciókat is, amelyek az adatfeldolgozást hivatottak megkönnyíteni, leegyszerűsíteni. Ezeknek a

függvényeknek a sorába tartozik például az **Array.every(fn)**, az **Array.some(fn)**, az **Array.forEach(fn)**, az **Array.map(fn)**, az **Array.filter(fn)** vagy az **Array.reduce(fn)**.

```
<script type="text/javascript">
    function isEven(element) {
        return element % 2 == 0;
    }
    function printArray(element) {
        document.write(element + "<br />");
    }
    function negateArray(element) {
        return 0 - element;
    }
    function sumArray(prevElement, nextElement) {
        return prevElement + nextElement;
    }

    var sampleNumberArray = [0, 9, 2, 8, 1, 5, 7, 3, 4];

    // Megnézzük, hogy a tömb összes eleme páros-e
    // A visszatérési érték: false, mert van közöttük páratlan elem
    sampleNumberArray.every(isEven);

    // Megvizsgáljuk, hogy a tömb elemei között találunk-e párosat.
    // A visszatérési érték: true, mert van közöttük páros elem
    sampleNumberArray.some(isEven);

    // A tömb összes elemét kiírjuk a dokumentumra
    sampleNumberArray.forEach(printArray);

    // A tömb összes elemét negáljuk, majd a függvény visszatérési értéke egy
    // új tömb lesz, a következő elemekkel: 0, -9, -2, -8, -1, -5, -7, -3, -4
    sampleNumberArray.map(negateArray);

    // Megkeressük a tömb összes páros tagját és egy új tömbben visszatérünk vele,
    // így tehát az eredmény: 0, 2, 8, 4
    sampleNumberArray.filter(isEven);

    // Tömb elemeinek összeadása. Eredmény: 39
    sampleNumberArray.reduce(sumArray);
</script>
```

Rögzíthető webhelyek

Mint ahogy azt már korábban láthattuk, az Internet Explorer 9 remekül együttműködik a Windows 7 rendszerrel. A rögzíthető webhelyek funkcióval a felhasználók úgy érhetik el weboldalunkat, mint hogyha azok hagyományos asztali alkalmazások lennének.

Természetesen bármilyen webhely rögzíthető a Windows 7 tálcára vagy Start menübe, utólagos módosítások nélkül, azonban csak akkor használhatjuk ki igazán a funkciót, ha felkészítjük erre az oldalunkat. Ez a felkészítés pedig egyáltalán nem bonyolult vagy nehézkes, pár sorral megoldható.

```
<head>
    <title></title>
```

```
    <meta name="application-name" content="Példa Webhely" />
    <meta name="msapplication-tooltip" content="Példa Webhely megnyitása" />
    <!--Webhely címe-->
    <meta name="msapplication-starturl" content="http://localhost/IE9/index.htm" />
    <!--Böngészőablak mérete-->
    <meta name="msapplication-window" content="width=800;height=600" />

    <!--Webhely ikonjának megadása-->
    <link rel="shortcut icon" type="image/x-icon"
          href="http://localhost/IE9/favicon.ico" />
    <link rel="icon" type="image/ico" href="./favicon.ico" />
</head>
```

A fenti példában ugyan felkészítettük a webhelyet a Windows 7-et való együttműködésre, azonban nem definiáltunk ugrólistákat, de az alábbi módon megtehetjük ezt.

```
<!--Statikus ugrólisták-->
<meta name="msapplication-task" content="name=Kezdőlap;
    action-uri=http://localhost/IE9/Index.html;icon-uri=./Media/Images/home.ico" />

<meta name="msapplication-task" content="name=Rólam;
    action-uri=http://localhost/IE9/AboutMe.html;icon-uri=./Media/Images/about.ico" />

<meta name="msapplication-task" content="name=Videóim;
    action-uri=http://localhost/IE9/Videos.html;  icon-uri=./Media/Images/videos.ico" />
```

Az ugrólistáinkat azonban egyedi csoportokba is foglalhatjuk némi JavaScript segítségével.

```
<script type="text/javascript">
    // Saját ugrólista törlése
    window.external.msSiteModeClearJumplist();

    // Egyedi kategória létrehozása az ugrólistába
    window.external.msSiteModeCreateJumplist('Pécs - Időjárás');

    // Kategória elemeinek felvétele (fordított sorrendben fognak megjelenni)
    window.external.msSiteModeAddJumpListItem('Holnap',
        './Pecs/3', './Media/Images/rain.ico');
    window.external.msSiteModeAddJumpListItem('Ma este',
        './Pecs/2','./Media/Images/sunny.ico');
    window.external.msSiteModeAddJumpListItem('Ma délután',
        './Pecs/1', './Media/Images/sunny.ico');


    // Kategória megjelenítése
    window.external.msSiteModeShowJumplist();
</script>
```

Készíthetünk olyan webhelyeket is, amelyek interaktívak és középpontjukban a felhasználói aktivitás áll. Ilyenkor szükség lehet arra, hogy figyelmeztessük a felhasználót például arra, ha üzenetet kapott, vagy valami egyéb történt.

```
var figyelmeztetes = true;

// ... Szükséges műveletek elvégzése
```

```
if (figyelmeztetes) {

    // Állapotjelző ikon rajzolása a webhely ikonjára
    window.external.msSiteModeSetIconOverlay('./Media/Images/warning.ico', 'Warning');

    // Ablak aktiválásának kérése (villogó tálca)
    window.external.msSiteModeActivate()
}
```

WebStorage modul

A modern, 21. századi számítástechnika világa egy olyan irány felé kezd eltolódni, ahol kiemelt fontosságú szerepet kap a számítási felhő. A számítási felhőnek az a fő célja, hogy az adataink, a kedvenc alkalmazásaink nem helyi számítógépeken legyenek tárolva, hanem egy központi helyen, a felhőben. Azonban számítási felhő ide vagy oda, nem küszöbölhetjük ki azt, hogy (legalább ideiglenesen) adatot tároljunk a felhasználó számítógépén. A korábbi webes szabványok értelmében a webes felhasználó oldali adattárolást sütiken keresztül valósíthattuk meg. Azonban a sütiknek több hátrányuk is van. Az egyik talán az, hogy nagyon kis mennyiségű adatot tárolhatunk bennük. A másik hátránya pedig az, hogy minden szerverkérésnél a kérés fejlécében továbbítva van a süti, ami felesleges adatforgalmat eredményezhet. . A HTML5 Web Storage modul kétféle megoldást kínál az adataink tárolására. Az egyik alternatíva a **Session Storage,** a másik pedig **Local Storage**. Mind a Session Storage, mind a Local Storage egy közös Storage API-ból származik, ezért a két megoldás implementálásának módja megegyezik (ugyanazokat a függvényeket és tulajdonságokat használhatjuk), a különbség csupán az adatok felhasználásnak módjában jelentkezik. Mind a Session Storage, mind a Local Storage megoldás esetében az adatokat kulcs – érték páros formájában lehet eltárolni és visszakeresni. A kulcs nem más, mint egy szöveges objektum, az érték pedig bármilyen típusú objektum lehet, amit a böngésző támogat.
A Session Storage objektum szolgáltat megoldást a sütik kiváltására, hiszen működése nagyban hasonlít a sütik működéséhez. Az objektum tartalma ugyan úgy, mint a sütik esetében, továbbítva van a kiszolgáló felé. Az első nagy különbség pedig az adatok továbbításában jelentkezik. Ugyanis az adat nem minden szerver-kérésnél továbbítódik a kiszolgáló fele, hanem csak akkor, hogyha változás történt benne. Ezen felül pedig több százszor akkora adatot tárolhatunk bennünk, mint a sütikben, és ezeknek az adatoknak a kinyerése is lényegesen egyszerűbb. Továbbá fontos tudnunk, hogy a Session Storage objektum csak az éppen aktuális ablakra érvényes, így ha eltárolunk bizonyos adatokat ebben az objektumban, majd bezárjuk az ablakot és újra megnyitjuk, az objektumunk üres lesz.

A Local Storage egy olyan speciális JavaScript objektum, ami arra szolgál, hogy nagymennyiségű adatokat tárolhassunk benne. A Session Storage objektumtól eltérően, itt, ha valamilyen adatot beleteszünk ebbe az objektumba, akkor az a böngésző bezárása, majd újranyitása után is elérhető. Kiválóan alkalmas például off-line webes alkalmazások kialakításához, amikor nem biztosított a folyamatos internet elérés (például levelező portál).

```
<script type="text/javascript">
    // Létrehoztunk két változót, az egyik a LocalStorage, a másik a SessionStorage
    // tárolónak felel meg.
```

```
    var localStorage = window['localStorage'];
    var sessionStorage = window['sessionStorage'];

    // Adat elhelyezése a LocalStorage tárolóba
    localStorage.setItem("test", "LocalStorage test");

    // Adat lekérése a LocalStorage tárolóból. Eredmény: "LocalStorage test
    var value = localStorage.getItem("test");

    // LocalStorage tároló ürítése
    localStorage.clear();

    // Újból lekérjük a "test" kulcshoz tartozó adatot a tárolóból.
    // Eredmény: null, hiszen kiürítettük a tárolót
    value = localStorage.getItem("test");
</script>
```