SZAKDOLGOZAT

Dulai József

Debrecen

2009

Debreceni Egyetem

Informatika Kar

AZ ADOBE FLASH TECHNIKA

Témavezető:
Dr. Tornai Róbert
Egyetemi adjunktus

Készítette:
Dulai József
Mérnök Informatikus

Tartalomjegyzék

1. Bevezetés

1996-ban a FutureSplash nevű cég mutatta be a Flash-t [8]. Szakdolgozatom célja, hogy a webes fejlesztést az innovatív Flash technológián keresztül mutassam be, ami egy webes multimédia megoldásként keletkezett, amivel először könnyen, gyorsan letölthető interaktív gombokat készíthettünk az Internetre. Amivel azonban igazán elbűvölt mindenkit, az az, hogy vektorgrafikát lehet vele készíteni. Az Adobe Photoshop segítségével készített pixeles grafikákkal ellentétben a vektor grafika ugyanis méretezhető. Ez azt jelenti, hogy a felhasználó rá tud közelíteni a képre, vagy a kép egy részére anélkül, hogy a kép minősége romlana. Rugalmasságot adhatunk képeinknek ezáltal.

A Macromedia továbbfejlesztette a technikát, és a Flash-t bittérképes, audio támogatással bővítette. A Flash 3-as verziójánál jelent meg az átlátszóság (Transparency) és a morfózis képessége [1]. Valamint a Macromedia a Flash Asset Xtra-val áthidaló kapcsolatot hozott létre a Shockwave és a Flash között.

A Flash nemcsak a világot hódította meg, Magyarországon is óriási publicitásnak örvend: több ezres fejlesztő- és rajongótábort vonultat fel maga mögött, köztük cégeket és iskolákat is. A Flash technológia hosszú évek óta fénykorát éli, de a felhasználók még mindig nem fogytak ki az ötletekből. Naponta kerülnek újabb és újabb megoldások a weboldalakra Flash [10] animációk formájában, de a fejlesztők hihetetlen tömege is ontja a jobbnál jobb, Flash alapú alkalmazásokat világszerte.

Két fontos érv, ami miatt a Flash technológiát egyszerűen nem lehet figyelmen kívül hagyni [11]: a technológia rendkívül gyors elterjedése (szinte minden operációs rendszer támogatja, sőt már a mobiltelefonokon is megtalálható), valamint a Flash lejátszó szinte 100%-os elterjedése (egy 2003 elején elvégzett nagyszabású felmérés alapján a világ számítógépeinek 97%-án használják a Flash lejátszót külön futtatott programként vagy böngésző pluginként), ami vélhetően napjainkban még nagyobb százalékot mutat.

Valamint a Flash fontosságáról tesz tanúbizonyságot, hogy 2005-ben bejelentették a Macromedia és az Adobe házasságát [12], melynek oka a jövőbeli növekedési lehetőségek maximális kihasználása, illetve a hatásosabb küzdelem az egyre fenyegetőbb ellenféllel, a Microsofttal szemben, nem csak a PC-s, hanem a mobil vagy a kézigépes világban is.

A Flash technológiák a weboldalaktól kezdve, az online alkalmazásokon át, a mobil eszközökig, segítik az online tartalom fejlesztését.

A Flashnek köszönhetően látványosabb, interaktívabb és elérhetőbb weboldalakat lehet tervezni bármely ágazat számára, legyen az a szórakoztatóipar, fogyasztási cikkek, közigazgatás vagy oktatás. A Flash révén olyan vektor-animációs eszköz került a fejlesztőkhöz, felhasználókhoz, amely először tette lehetővé az akkor még statikus weben a mozgó grafika egyszerű bevezetését. A későbbiekben az anyagunkhoz hangokat is csatolhattunk.

Eme meghatározó okok, tulajdonságok miatt választottam a Flash bemutatását, amit a következő oldalakon fogok ismertetni.

2. Alapvető fogalmak

A fejezetben ismertetésre kerülnek a multimédiában használatos azon alapfogalmak, amelyek elengedhetetlenek a későbbiekben ismertetendő Adobe Flash használatához.

2.1 Számítógépes grafika

A multimédia rendszerekben használt képeket két nagy csoportra lehet osztani, ezek az alábbiak [13]:

- állóképek vagy számítógépes grafikák,
- mozgóképek vagy videók.

Egy multimédiaalkalmazásban az állóképek számos célra használhatóak fel: Illusztrálható velük szöveg, felhasználhatók az egyes szövegek magyarázataként, de készíthető velük multimédiaalkalmazások számára animáció is. Az állóképeket többféleképpen hozhatjuk létre:

- Az állókép megrajzolása egy rajzolóprogrammal.

  Ez a módszer azonban igen sok időt vesz igénybe, valamint szép rajzok készítése az átlagosnál nagyobb rajzoló tehetséget igényel.

- Meglévő képek, fényképek digitalizálása, szkennelése:

  Ekkor csupán be kell a már meglévő képet (képeket) olvasni egy lapolvasó (scanner) segítségével, majd az adott alkalmazás igényeinek megfelelő alakítás után fel is használhatjuk őket.

- Digitalizálás videoszalagról videodigitalizáló kártya segítségével:

  Ehhez egy videó digitalizáló berendezésre lesz szükség (videó digitalizáló kártya), amely a videoszalagon tárolt képeinket az általunk választott digitális formátumban rögzíti, így azokat be tudjuk illeszteni a kívánt multimédiás alkalmazásba.

Grafika típusai:

- bittérképes grafika
- vektorgrafika

2.1.1 Bittérképes (pixel) grafika

A bittérképes grafikával igen jó minőségű képeket lehet létrehozni. A képek minősége azonban együtt jár a képeket tartalmazó állományok méretének gyarapodásával.

Ezen grafika tulajdonságai:

- Felületet apró képpontokra bontják.
- A pontok síkbeli helyzetét, színét, világosságát tárolják. Ezt numerikus értékekkel lehet kifejezni.:

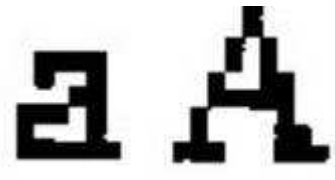
1. ábra: B*ittérképes grafika*

A bittérképes grafika (1. ábra) a képek megjelenítésének legegyszerűbb módja. A képet függőleges és vízszintes irányban pontokra osztja fel, és minden egyes pontról tárolja annak szín és fényerősségi információit. A tárolt szín- és fényerősség információk azonban igen sok helyet foglalnak el.

A bittérképes grafika hátrányai:

- Az állományok nagy méretűek.

- Nehéz a képpel úgy műveleteket végezni, hogy maga kép ne tartalmazzon (2. ábra) torzulást (ha egy rajzot például ki szeretnénk nagyítani akkor az egyes körvonalak csipkéssé, szaggatottá válnak).:



2. ábra: *Torzulás*

Használt fájl formátumok:

- BMP
- GIF
- TIFF
- JPEG
- PCX

2.1.2 Vektorgrafika

A vektorgrafika vagy geometriai modellezés a számítógépes grafikában az az eljárás, melynek során geometriai primitíveket (rajzelemeket), mint például pontokat, egyeneseket, görbéket és sokszögeket használunk képek leírására [5].

A vektorgrafika fájlba menti a kép alakzatainak . és színeinek leírását pl. trapéz esetén a területét, határoló vonalainak hosszát, oldal által bezárt szöget

3. ábra: *Vektorgrafika*

Amikor egy rajzolóprogram segítségével egy vektorgrafikával (3. ábra) készült rajzot hozunk létre, akkor a rajzolóprogram egy „láthatatlan” hálóra rajzolja ki az általunk készített grafikát. Ezt a grafikát aztán utasítások halmazaként tárolja a program egy állományba. Az utasítások pontosan leírják az alakzat pozícióját, méretét, színét, alakját és a megjelenítéssel kapcsolatos jellemzőit. Amikor ki szeretnénk rajzoltatni a képernyőre az így tárolt grafikát, akkor a program végrehajtja a grafika állományában található utasításokat, és ezekből építi fel a képernyőn megjelenő rajzot. Tehát a vektorgrafika nem a képet alkotó pontokat tárolja, hanem az azok megjelenítéséhez szükséges utasításokat.

A módszer előnye: egyszerűen lehet a grafikus kép egyes részeivel műveleteket végezni, (forgatni, nagyítani) és nem okoznak nagy torzulást (4. ábra) a kép egészén, továbbá ezek az állományok kis méretűek.

A módszer hátránya: minél összetettebb egy rajz, annál több utasítás szükséges annak leírásához, tehát annál tovább tart egy kép megjelenítése. Fénykép minőségű kép létrehozására nem igazán alkalmas.

Vektorgrafikában kinagyított betűk képe:

4. ábra: *Torzulásmentes nagyítás*

Formátumok:

- PSD
- PS
- EPS
- WMF
- SVF
- Egyéb vektorgrafikus programban készített rajzok (.ai, .cdr)

2.2 Az animáció

„Animálni” azt jelenti „életed adni”. Az animáció valaminek a megmozdítása, ami saját magától nem tud mozdulni [3]. Az animáció a grafikának az idő dimenziót adja, ami sokszorosan növeli az átadható információ mennyiséget. Ahhoz, hogy az animáció szerzője valamit animálhasson, képes kell, hogy legyen arra, hogy közvetlenül, vagy közvetetten definiálja hogyan fog időben és térben mozogni „az” amit animálni szeretne.

A számítógépes animációnak két alapcsoportja létezik:

- Számítógéppel támogatott animáció: *rendszerint a 2D animációnál alkalmazzák. Ebben az esetben az egyedüli algoritmus melyet a számítógép az animáció alkotásához alkalmaz, az interpoláció a kulcs alakzatok között.*

- Számítógéppel létrehozott (generált) animáció:
    - *Alacsony szintű technikák, melyek segítenek az animáció alkotójának a mozgás pontos definiálásában. Ilyenek például az alakzatok közötti interpolációs algoritmusok (in-betweening), melyek a szerzőnek megkönnyítik a mozgás részleteinek kitöltését, miután meghatározta a mozgással kapcsolatos szükséges és elegendő számú információt.*
    - *Magas szintű technikák, melyeket a mozgás általános leírásához használnak.*

Ezek a technikák rendszerint algoritmusok, vagy modellek melyeket a mozgás generálásához szokás alkalmazni megfelelő szabályok, vagy határértékek definiálásával. Az animáció szerzője definiálja a modellhez járó szabályokat, vagy kiválasztja a megfelelő algoritmust definiálva a kezdőértékeket és a határértékeket. Ezután a rendszer (számítógép) szabályozza a mozgást az algoritmus, illetve a modell segítségével. Ezek a technikák sűrűn összetett számításokra vannak alapozva.

2.2.1 Az animáció elemei

Az alkotási folyamatban a **prezentáció** alatt a teljes animáció van értelmezve. A prezentáció alapelemei az epizódok (*acts*). Az **epizód** egy egységet alkot, melyhez egy definiált térbeli díszlet (környezet) van csatolva. A prezentáció rendszerint egy vagy több epizódból áll. Az epizód jelenetekre osztható. A **jelenet** egy nem megszakított akciót definiál. A jelenet egy vagy több beállításra osztható. A **beállítás** egy nem megszakított kamera felvételt jelent. A beállítás individuális képekből áll (*frame*).

Ez alapján, a következő hierarchia állítható fel:

Prezentáció - epizód - jelenet - beállítás - kép.

2.2.2 Az animáció típusai

Az animációk két részből állnak, ezek a háttér és az előtér. A háttér az animáció alapja. A hátteret és az előteret általában számítógépes programokkal lehet létrehozni. Ahhoz, hogy egy multimédia alkalmazásba animációt tartalmazó elemeket építsünk be, két út között választhatunk:

- Animáció szerkesztő program: *segítségével létrehozhatunk önálló animációs állományokat, melyeket lejátszunk a multimédia alkalmazással.*
- Objektum animáció: *segítségével lehet kifejleszteni, és a multimédia alkalmazásba beépíteni animációt tartalmazó elemeket.*

Az animációt tartalmazó elemek lejátszási sebessége függ számítógépünk teljesítményétől és az animáció típusától is. Míg a hagyományos mozgóképek esetén mintegy 25-30 kép másodpercenkénti lejátszása teszi lehetővé a mozgókép illúziójának fenntartását, addig az animációk lejátszási sebessége ennek a fele, másodpercenkénti 14-18 kép. Ez a lejátszási sebesség megfelelő minőséget ad az animáció lejátszásakor, és nem terheli meg a számítógép hardverét különösebben.

1) Állandó előtérrel készülő animáció:

Az alapötlet a régi stílusú pörgethető lapokkal egyezik meg. A lapok általában egy egyszerű rajzot tartalmaznak, például ahogyan egy ló fut. A kép háttere lapról-lapra csak egy kicsit változik, miközben az előtér változatlan marad, így amint gyorsan átpörgetjük a lapokat, a folyamatos mozgás illúzióját kelti bennünk. Az állandó előtérrel készülő animáció közel 60 évvel ezelőtti találmány. Ekkor még az animáció minden egyes lapját kézzel rajzolták meg az alkotók. Napjainkban számítógéppel segítik az ilyen típusú animáció elkészítését is: az egyes lapokon levő objektumokat másolással viszik át a következő lapra, és a változó részeket is sokkal könnyebben szerkesztik.

2) Állandó háttérrel készülő animáció:

Az állandó háttérrel készülő animáció az állandó előtérrel készülő animáció folytatásaként keletkezett, kb. 50 évvel ezelőtt. A szerzők abból indultak ki, hogy lényegesen egyszerűbb az állandó hátteret az animáció minden egyes lapjára felvenni, és utólag rárajzolni a változó előteret, mint fordítva. Az ilyen módon keletkezett animáció lépései megegyeznek az előző pontban említett animáció lépéseivel, csak az előtér és a háttér kezelésének módja különbözik. A számítógép alkalmazása itt is jelentősen leegyszerűsíti a rajzolási folyamatot. Az állandó hátteret csak egyszer kell megrajzolni, és ezután oldalról-oldalra át lehet másolni. Az előtérben keletkező mozgást ábrázoló képek részleteit pedig gyakran a számítógép hozza létre, a felhasználónak csak a kezdő és a végső állapotot kell megrajzolni. Ha például egy, a képernyőn átrepülő madarat szeretnénk

megrajzolni, akkor elegendő a madár kezdő és végső állapotát megrajzolni a képernyőn, a számítógép kiszámítja és berajzolja a közbülső elemeket.

3) Objektum animáció:

Az objektum animáció az egyik legalapvetőbb animációs technikát képviseli. Ahelyett, hogy az előző két pontban említett animációt tartalmazó állományt létrehoznánk, itt létrehozunk egy grafikus objektumot, amelyet egy program segítségével mozgatunk a képernyőn.

Az objektum animáció tehát egy előre elkészített grafikus objektumot mozgat a képernyő előre megadott útvonalán.

Az objektum animáció segítségével, minimális munkával válthatunk ki igen nagy hatást a multimédiás alkalmazás felhasználójából. Mivel az objektum animáció nem igényel külső támogatást, és a mozgatandó rajz is csak egyszer szerepel a multimédia alkalmazás állományában, ezért kevés helyet igényel a memóriában, kevéssé köti le a számítógép erőforrásait, tehát igen szabadon használható.

3. Az Adobe Flash

Az Adobe Flash egy szoftver, amely az Adobe Systems termékcsaládba tartozik [3]. Egy olyan professzionális multimédiafejlesztő-alkalmazás, amelynek segítségével könnyedén fejleszthetünk webes alkalmazásokat, játékokat, mozikat, vagy akár mobil tartalmakat. A Flash hatékonyan ötvözi a vektoros rajzolóprogram és a professzionális animációszervező-program minden előnyét.

3.1 ActionScript

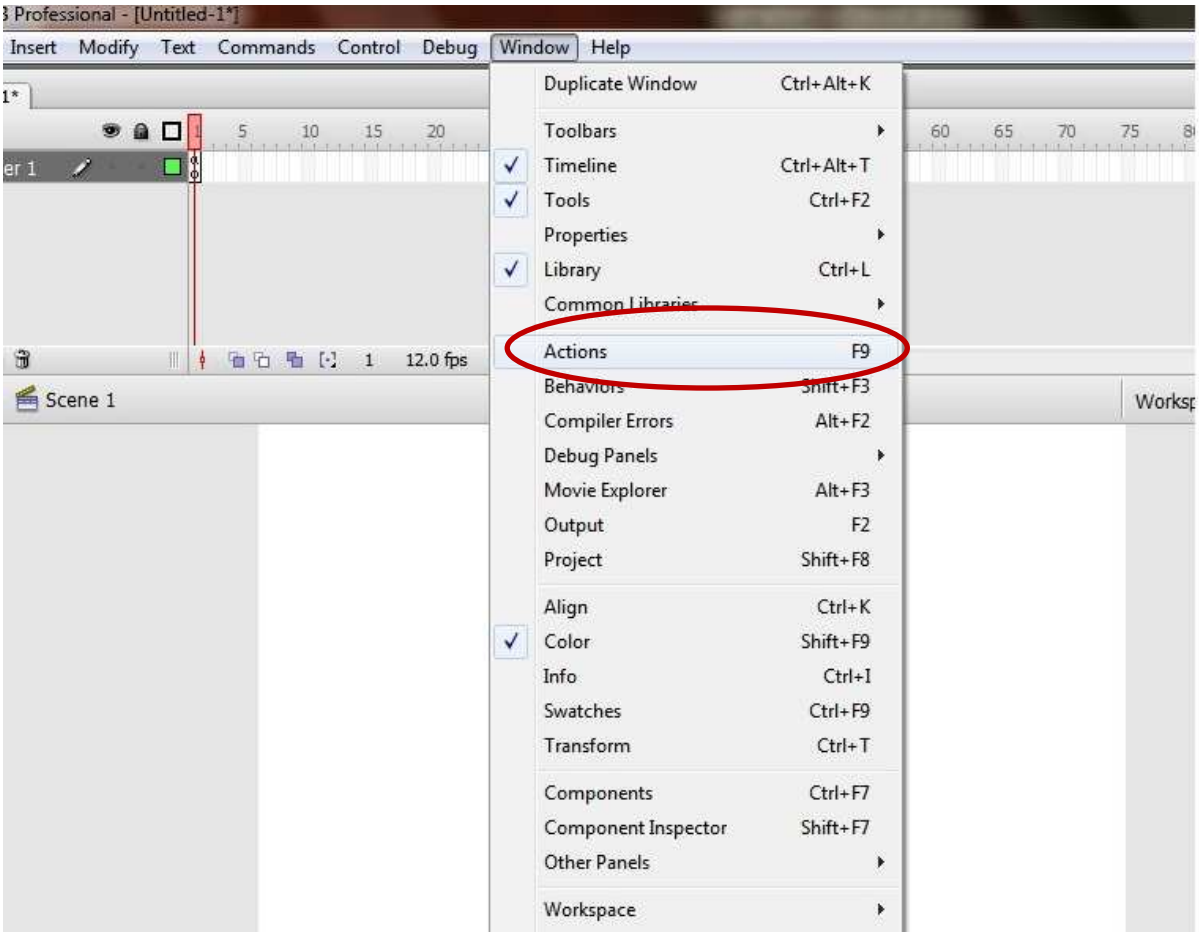
Az ActionScript, pontosabban az ActionScript 3.0 robosztus programozási (szkript) nyelvet használja az alkalmazás funkcionalitásának kiterjesztésére az Adobe Flash CS3, és az Adobe Flash CS4.

Az ActionScript programnyelvvel - a Java nyelvhez hasonlóan - további interaktivitást adhatunk Flash animációinkhoz [1].

Az ActionScript használatához nincs szükség mély programozási tudásra. Általános feladatok esetén lemásolhatjuk a más Flash- felhasználók által megosztott kódokat vagy kódrészleteket.

Eme programnyelv nélkül a film minden egyes alkalommal ugyanúgy játszódik le. Ahhoz viszont már az ActionScript nyelvre lesz szükség, hogy a felhasználónak lehetőséget adjunk például a film megállítására és indítására.

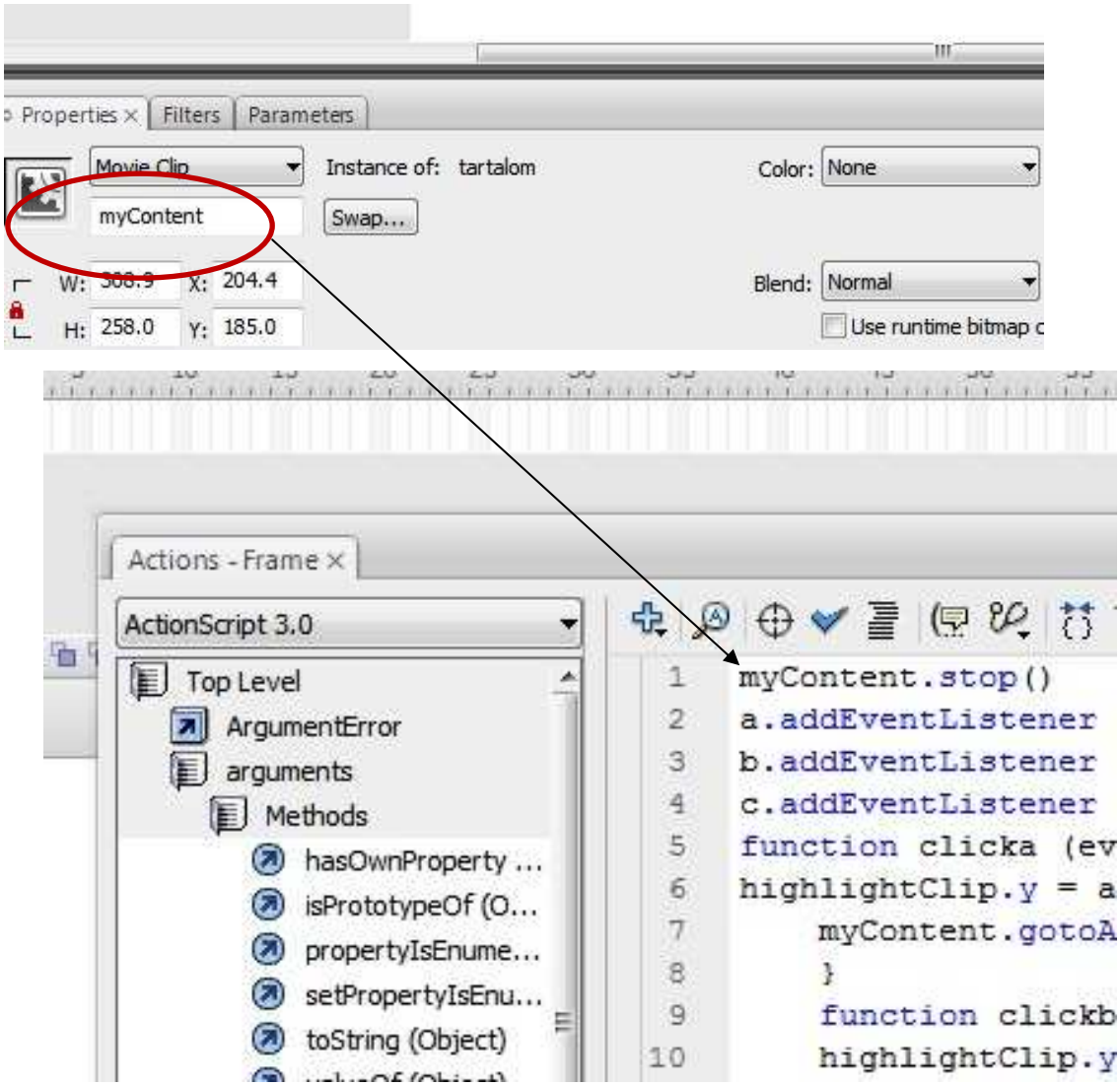
Az ActionScript hosszú utat tett meg, mire utasítások kis csoportjából teljes értékű programozási nyelvvé fejlődött, és igazán hatékonnyá vált. A hatékonyság azonban azt is jelenti, hogy a nyelv ma már sokkal bonyolultabb.



5. ábra: A*ctions menü*

Az ActionScript megnyitása a Flash Window menüpont alatt található Actions paranccsal (5. ábra), vagy simán F9 billentyű lenyomásával, vagy a képkockán a jobb egérgomb Actions menüponttal történik. A kódot mindig az Actions táblába gépeljük be.

Előbb az animációt kell elkészíteni, az egyes szimbólumokat, grafikákat el kell neveznünk, hogy később a programozás alatt hivatkozni tudjunk rá (6. ábra).



6. ábra: *Hivatkozás*

Az Actions tábla főbb részei:

- Toolbox*: az eszközkészletben az összes telepített műveletet megtaláljuk, mappákba rendezve.*
- Script(Kód): *a Kódablakban a műveletek a végrehajtási sorrendjükben jelennek meg.*
- Navigator: *a Navigátor ablakban a filmünk összes kódját megtaláljuk.*
- Current Script (Aktuális kód) fül: *a Jelenlegi kód fül megmutatja, hogy éppen melyik parancsfájlt szerkesztjük.*

- Pin Script (Kód rögzítése): *ezzel a gombbal saját lapot hozhatunk létre egy adott parancsfájlhoz, hogy ne kelljen mindig kiválasztani az objektumot vagy réteget, amihez hozzá szeretnénk adni egy kódrészletet.*
- Script Assist (Kódsegéd): *később lesz róla szó.*
- Options menü: *a beállító menüben további lehetőségek közül választhatunk, például beállíthatjuk a betűtípust.*
- Help gomb: *ez a gomb elektronikus súgót jelenít meg a kijelölt ActionScript-részlethez.*
- Options eszköztár:
    - *Add Statement: Ez a gomb egy menüt jelenít meg, ami ugyanazokat a kódelemeket tartalmazza, mint az Eszközkészlet.*
    - *Find: Ezzel a gombbal kereshetjük ki a kódrészleteket, éppen úgy, mint egy szövegszerkesztőben.*
    - *Insert Target Path: Ezzel a gombbal egy adott objektumot, például egy adott filmet címezhetünk meg. Ez a gomb segít az elérési út meghatározásában.*
    - *Check Syntax: Ezzel a gombbal meggyőződhetünk róla, hogy nyelvtanilag hibátlan-e az ActionScript kódunk, és az eredményt megjeleníthetjük az Output ablakban.*
    - *Auto Format: Ez a gomb rendbe teszi a kódot, és beljebb viszi a sorokat, ahol kell. Ettől a kód sokkal átláthatóbb lesz.*
    - *Show Code Hint: Ez a gomb bekapcsolja a kódkiegészítő segédet, ami egy gyorssúgó buborék formájában nyújt segítséget az ActionScript-íráshoz.*
    - *Debug Options: Ezzel a gombbal töréspontokat adhatunk meg, és törölhetjük őket azokon a helyeken, ahol szándékosan meg szeretnénk állítani a Flash futását egy adott kódsornál, hogy megvizsgálhassuk, mit is csinál éppen.*
    - *Collapse Between Braces and Collapse Selection: Ezekkel a gombokkal összecsukhatjuk a kód egyes részeit, ha a parancsfájl nagyon hosszú, és az adott résszel mát végeztünk.*

- *Expand All: Ezzel bonthatjuk ki gyorsan a korábban összecsukott kódrészeket. A kódok egyébként úgy is összecsukhatjuk, illetve kibonthatjuk, ha a kijelölt kódrészlet mellett a margón megjelenő kis + vagy – jelre kattintunk.*
- *Apply Block Comment: Néha csak feljegyzéseket szeretnénk készíteni a magunk számára, amelyeket a Flash-nek figyelmen kívül kellene hagynia, illetve így szeretnénk ideiglenesen kikapcsolni egyes kódrészeket, amelyeken még dolgozunk. Ezzel a gombbal éppen ezt érthetjük el.*
- *Apply Line Comment: Míg a megjegyzésblokk több sort is tartalmazhat, az egysoros megjegyzések // jellel kezdődnek, és a Flash csak azt az egy sort hagyja figyelmen kívül.*
- *Remove Comment: Ezzel a gombbal gyorsan eltávolíthatjuk a kijelölt megjegyzést.*
- *Show/Hide Toolbox: A megjelenítő/ elrejtő gomb az eszközkészletet lehet elrejteni, illetve megjeleníteni.*

3.1.1 A programozási technológia megismerése

A fejezetben használt szakkifejezések nagy része megegyezik a más programozási nyelvekben használt terminológiával [2].

- **Változó:**

A változó egy azonosító, amely konkrét adatot tartalmaz. Amikor létrehozunk egy változót, adattípust rendelünk hozzá, amely meghatározza, hogy milyen típusú adatot tárolhat a változó. Például: Létre szeretnénk hozni egy változót, amely a filmben lévő képkockákat számolja, majd, amikor a következő filmkockára lépünk, adjon hozzá egyet (1. kód).

```
var frames = 0;
currentframes = frames + 1;
```

1. kód: *Változó növelése*

A változóknak egyedinek kell lenniük és különbséget teszünk a kis- és nagybetűk között.

- **Kulcsszó:**

Az ActionScript programozási nyelvben a kulcsszavakat adott feladatot végrehajtó szavaknak tartjuk fenn. A „var” kulcsszóval például változót hozunk létre. A Flash súgóban a teljes kulcsszólistát megtaláljuk.

- **Paraméterek:**

A paraméterek specifikus részleteket adnak meg, és mindig zárójelben szerepelnek. Például a gotoAndPlay(3); (ugrás a 3. képkockára).

- **Függvény:**

A függvény olyan utasításcsoport, amelyre névvel hivatkozunk. Függvényekkel úgy futtatjuk utasítások sorozatát, hogy nem kell az egész kódot újra begépelni.

- **Osztály:**

Minden objektumot egy osztály definiál, amely az objektum absztrakt megjelenítése. Az osztály az ugyanolyan metódusokkal és tulajdonságokkal rendelkező objektumok kategóriája.

- **Tartomány:**

A tartomány határozza meg a Flash fájl számára azt a területet, amelyben egy változóra hivatkozhatunk. A változó lehet lokális, globális vagy időegyenes-szintű. Függvényben kapcsos zárójelek között deklaráljuk a lokális váltókat.

- **Metódusok:**

  A metódusok műveletet eredményező kulcsszavak. Például Stop() és a gotoAndPlay() vagy a gotoAndStop().

- **Tulajdonságok:**

  A tulajdonságok az osztályban lévő objektumokat írják le.

3.1.2 Az Actions eszköztár kezelése

Az Actions eszköztár az ActionsScript legfontosabb elemeihez nyújt gyors hozzáférést. Kategóriánként csoportosítja az elemeket. Ha valamelyiket hozzá szeretnénk adni a kódhoz, kattintsunk duplán rá az eszköztáron.

Az Actions eszköztáron több kategória is fel van sorolva. A Top Level kategória a leggyakoribb osztályokat listázza ki.

Ha egy adott kódrészletre van szükségünk, de nem vagyunk biztosak abban, hogyan kell formázni, az Index kategóriában, ábécérendben megkereshetjük azt.

3.1.3 A feltételes utasítások

A feltételes utasítások leggyakrabban használt típusa az if utasítás. Ez a parancs a zárójelben levő értéket vagy kifejezést értékeli és ellenőrzi. Ha az érték igaz, az ezt követő utasítás fut le, ha nem, a program a kód végrehajtása nélkül továbblép. `Else` utasítás hozzáadásával a feltétel nem teljesülése esetén végrehajtandó alternatív instrukciókat adhatunk meg.

A következő példában az `if` utasítás meghatározza, hogy a jelszó nulla-e; ha az „`if`” utasítás teljesül, akkor a `gotoAndStop()` metódus hajtódik végre; egyébként, ha a jelszó „Tom”, a `gotoAndPlay` metódus fog végrehajtódni (2. kód):

```
if(password == null) {
gotoAndStop(„Rejection”);
}
else if  (password = „Tom”) {
gotoAndPlay („Acceptance”);
}
```

2. kód: *Utasítás létrehozása*

3.1.4 A Script Assist mód

Egyes műveletek olyan egyszerűek, hogy megadásuk nem jelent különösebb problémát. Egyesek viszont olyanok, amelyekhez paraméterekre van szükség, és ha nem ismerjük eléggé az ActionScript programnyelvet, nem egyszerű megjegyezni, milyen paramétereket használhatunk [2].

A Script Assist nem írja meg helyettünk a kódot, de segít abban, hogy helyesen rakjuk össze a darabokat, és ne kelljen állandóan szintaktikai hibákat bogozva visszatérni a programhoz.

Használatához kattintsunk az Actions panel Script Assist gombjára. Ezt követően kattintsunk duplán az Actions eszköztár valamelyik elemére, hogy a Script mezőhöz adjuk azt. Az Actions panel felső része az adott elemmel elérhető mezőket és lehetőségeket mutatja. Válasszuk ki a kívánt lehetőséget, és adjuk meg neki a megfelelő értékeket.

3.1.5 Java vs Action

Természetesen mindkét nyelvnek vannak előnyei és hátrányai. Sokan tesztelték e két nyelv képességeit. Az hogy ki melyik nyelvet hozza ki győztesnek többek között attól is függ, hogy mire kívánjuk használni.

Én most általánosan fogom összehasonlítani a két nyelvet.

Az ActionsScript előnye [7], ami eléggé fontos a nyelvet tekintve, hogy böngészőbarát. Azaz minden böngészővel kompatibilis. Nem számít, hogy a felhasználó honnan próbálja betölteni az animációját. A 3.0-ás verzió az elődökhöz képest akár tízszer gyorsabban képes a kódot végrehajtani.

A JavaScript egy natív (gépi) programozási nyelv számtalan böngészőre, mleyet viszonylag könnyű megtanulni és megvalósítani. Lényeges pozitív tulajdonság, hogy nincs szükség extra eszközökre a JavaScript íráshoz, bármilyen szöveg vagy HTML szerkesztő képes kezelni, így nem kell drága fejlesztői környezetet vásárolunk kódunk létrehozásához.

Kiemelendő, hogy a GUI- alapú böngészőket is támogatja. A JavaScript olyan DOM objektumokat hoz létre, amelyek elférnek egy HTML dokumentumban. Igaz az ActionScript is natívan támogatja a böngészőket, azonban egyedül a JavaScript teszi lehetővé a felhasználók számára, hogy futási időben szabhassák testre metódusok segítségével a könyvtáraik összetételét.

A két nyelv hátrányairól röviden:

Az Action - és JavaScript nem igazán alkalmas nem webes alkalmazásokhoz. Nem alkalmas alacsony szintű programozásra.

A JavaScript lényeges hátránya a biztonság, mivel a kód végrehajtódik a felhasználó számítógépén, így esetlegesen rosszindulatú célokra is használhatják. Valamint az eltérő böngészők eltérően értelmezhetik a JavaScriptet. Mivel a szerver-

oldali szkriptek mindig ugyanazt a kimenetet állítják elő, a kliens-oldali szkriptek kicsit kiszámíthatatlanok lesznek.

3.2 Az Adobe Flash munkafelülete

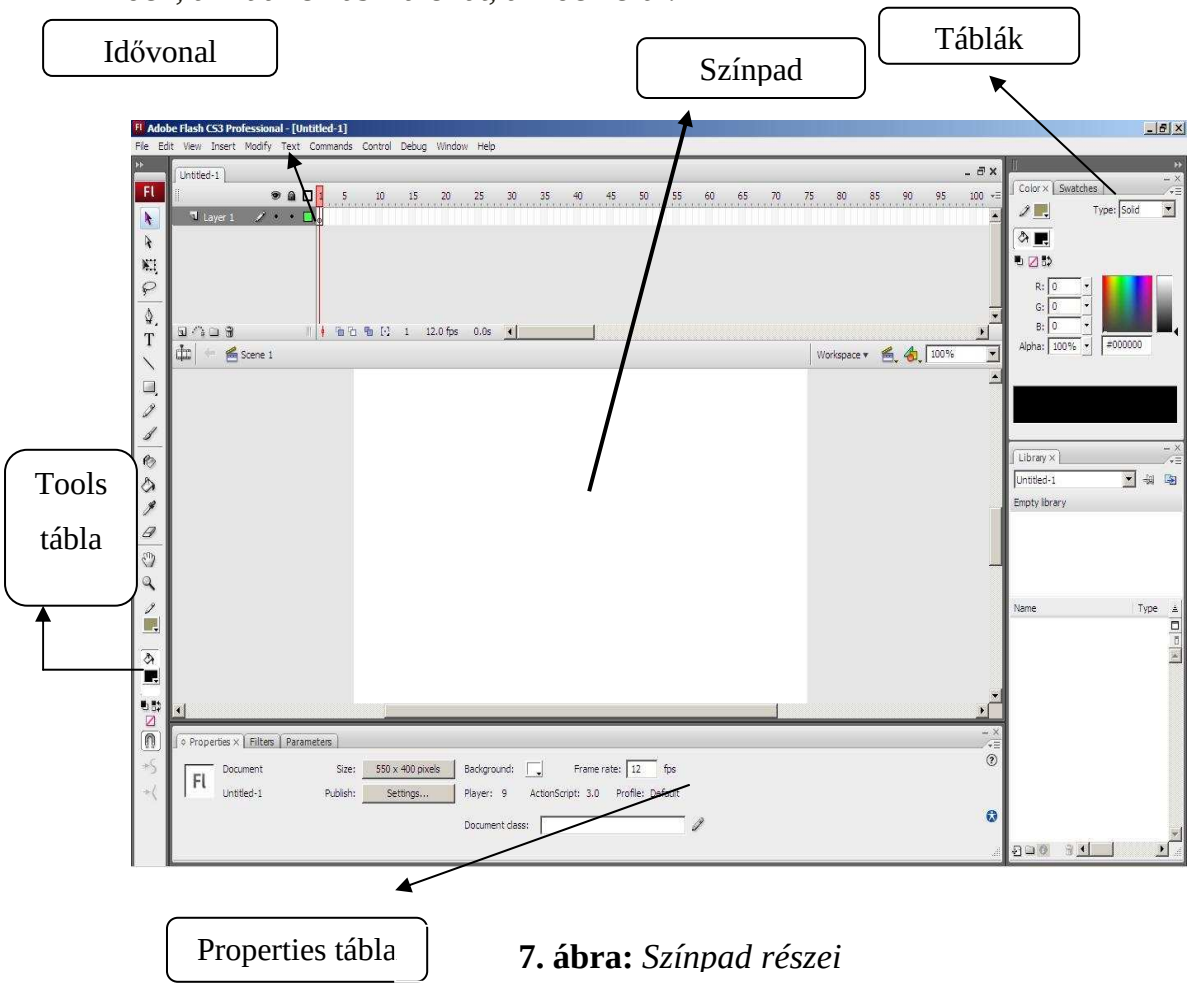
A Flash működését csak akkor érthetjük meg, ha mindig tudjuk, hol vagyunk.

Ezért fontosnak tartom, hogy először, ebben a részben a Flash szerkezetével kezdjem a bemutatást, ami a kezdetben bonyolultnak tűnő felületet próbálja meg barátságosabbá tenni számunkra [9].

Mivel a Flash-ben rengeteg dolgot tudunk kreálni, így ez a fejezet viszonylag nagy lélegzetvételű lesz.

3.2.1 A színpad

A Flash munkafelületének közepén elhelyezkedő nagy, fehér téglalapot színpadnak (Stage) nevezzük (7. ábra). A szöveg, a grafikák, fényképek, vagyis minden, amit a felhasználó lát, az ide kerül.



7. ábra: *Színpad részei*

Két fogalomról szót kell ejteni: a színpad méretéről és a nagyítás mértékéről. Alaphelyzetben a színpad egy 550 képpont széles és 400 képpont magas téglalap, de a képpontban megadott méret nem fontos, mert amikor a Flash filmet a Webre mentjük, meghatározhatjuk, hogy a Flash bármilyen képméretre átméretezze azt.

3.2.2 A Tools tábla

A Tools az a tábla, amely a Flash összes létező rajzeszközét tartalmazza. Bármikor, bármit hozunk létre a színpadon, valamelyik eszközt ki kell jelölni a Tools táblán. Alapállapotban a Flash felület bal oldalán rögzítve helyezkedik el.

Igaz, hogy a Tools tábla elsősorban a színpadon rajzolásra való, de a már megrajzolt elemek szerkesztésére is használható.

1) A Tools rekesz segítségével,

- grafikákat és szövegeket hozhatunk létre: a vonal (Line) és a szöveg (Text) eszközökkel.
- grafikákat szerkeszthetünk: a radír (Eraser) és a festékvödör (Paint Bucket) eszközökkel.
- vagy egyszerűen kijelölhetjük az objektumokat: a kijelölés (Selection), a részkijelölés (Subselect), vagy a lasszó (Lasso) eszközökkel.

2) A két megjelenítőeszköz (Hand és a Zoom) segítségével megváltoztathatjuk a színpad megjelenését.
3) A színeszközökkel a létrehozott objektumok színét befolyásolhatjuk.
4) Végül a beállítások rekesze bizonyos eszközök további beállításaira szolgál. Attól függően, hogy melyik eszköz van kiválasztva, ez a rekesz akár üres is lehet.

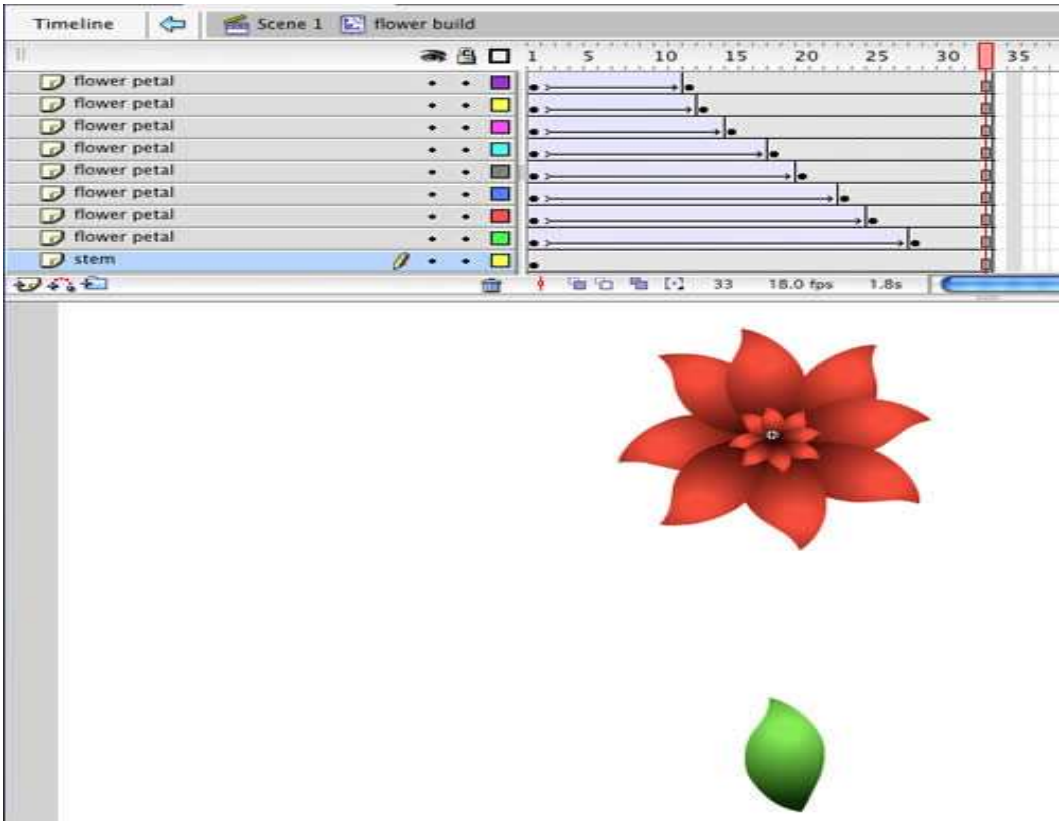
3.2.3 Az idővonal

Az időszalag önálló képek sorozatát tartalmazza, amelyek animációt alkotnak.

Amikor a felhasználó megnézi az animációt, az első képkockán lévő képet látja, azután a második kockát és így tovább.

Amikor elkezdünk animációkat készíteni, az időszalagon számos grafikus jel tűnik fel segítségként. Rögtön láthatjuk például az animáció hosszát, amikor az időszalagra nézünk.

A képkockákon túl az időszalag lehetővé teszi, hogy annyi réteget használjunk az animációnkban, amennyit csak szeretnénk (8. ábra). Mindegyik réteg tartalmazhat egy-egy animációt, így egyidejűleg több animáció is megjelenhet.

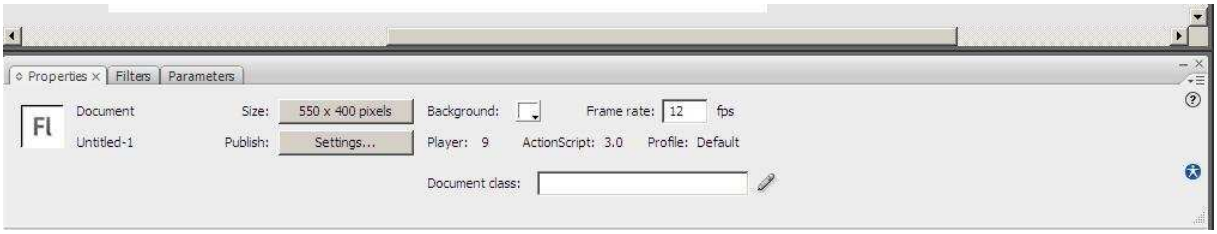


8. ábra: *Rétegek idővonalban*

3.2.4 Properties tábla

A Properties tábla az éppen kijelölt objektum tulajdonságait mutatja (9. ábra), lehetővé téve a beállítások módosítását. Amikor például kijelölünk egy szövegblokkot, a Properties táblán megváltoztathatjuk a betűtípust és a betűméretet, vagy amikor egy zárt alakzatra kattintunk, beállíthatjuk annak kitöltőszínét. E tábla segítségével nem csak a színpadon levő objektumokat módosíthatjuk, hanem az időszalag képkockáit is, illetve a dokumentumtulajdonságokat [9].

Akkor is végrehajthatunk változtatásokat a Properties táblán, amikor semmi nincs kijelölve. Annak ellenére, hogy látszólag nincs semmi hatása, valójában ilyenkor meghatározzuk, mi történjen legközelebb, amikor létrehozunk egy objektumot.



9. ábra: *Properties tábla*

3.3 Rétegek

A rétegeket legkönnyebb úgy lehet elképzelni, mint egy egymáson levő, áttetsző rajzlapokat. Amikor egy új Flash filmet hozunk létre, akkor az csak egy réteget tartalmaz. További rétegek nagyon könnyen hozzáadhatók a filmhez, mindezt a jobb animáció, jobb áttekintés és szervezés érdekében tesszük. Ha az egyik rétegen dolgozunk, pl. rajzolunk, festünk, szerkesztünk stb., akkor az nincs kihatással a többi rétegen levő objektumokkal. Az üres réteg teljesen átlátszó. A rétegek elsődleges céljuk, hogy különálló időszalagokat biztosítsanak, amelyeken külön-külön vezérelhetjük az animációkat. A Flash-ben bármennyi réteget be tudunk illeszteni az Insert-Timelime-Layer menüponttal, vagy egyszerűen az időszalag bal alsó ikonjára kattintva. Az egyes rétegeket egymáshoz képest tudjuk le-és felcsúsztatni a bal

egérgomb lenyomása mellett. Ezzel még inkább áttekinthetőbbé és logikusabbá válik munkánk. Valamint a rétegek elhelyezkedésének fontos szerepe van a maszkolt rétegek készítésénél.

3.3.1 Rétegtulajdonságok

A tulajdonságok segítik a szerkesztésben, hogy könnyebben átláthassuk és ne rontsuk el munkánkat. Például azért, mert rossz rétegen vagyunk.

1) **Rétegnév**: *ennek a tulajdonságnak a segítségével olyan nevet adhatunk a rétegnek, amilyet csak akarunk. Rendszerezettebb marad a munkánk, ha kihasználjuk ezt a szolgáltatást, és ésszerűen nevezzük el a rétegeket.*

2) **Rétegek megjelenítése/elrejtése:** *ez a tulajdonság lehetővé teszi, hogy átmenetileg elrejtsük bármelyik különálló réteg tartalmát, a szem alatti pontra kattintva. Ha a felső szem gombra kattintunk, az összes réteget elrejtjük vagy megjelenítjük.*

3) **Rétegek lezárása/lezárás megszüntetése:** *lehetővé teszi, hogy igény szerint egyesével lezárjuk a rétegeket, vagy feloldjuk a lezárást (egyszerre az összeset is lehet).*

4) **Rétegek megjelenítése körvonalként:** *ezzel a tulajdonsággal körvonalként jeleníthetjük meg a rétegek taralmát, majdnem ugyanúgy, ahogy láthatatlanná is tehetjük, csak nem olyan szélsőséges módon.*

3.3.2 Rétegtípusok

A típusok szerepe, hogy a munkáink során az épp legjobban megfelelő réteget válasszuk ki, mivel különböző hatások eléréséhez különböző rétegtípusokra van szükségünk.

1) **Szokásos réteg:** *ez a rétegtípus az egyszerű lap ikon, szamárfüllel a jobb alsó sarkában. Ez az alapértelmezett rétegtípus.*

2) **Hagyományos vezetőréteg:** *ez a rétegtípus különleges réteg, amelybe azt rajzolhatunk, amit akarunk. A vezetőrétegeken szereplő elemeket a program nem menti, amikor .swf-et hozunk létre, tehát nem jelennek meg a végső fájlban, és nem növelik a fájlméretet.*

3) **Mozgásvezető réteg:** *ez a típus úgy viselkedik, mint egy hagyományos vezetőréteg. A mozgásvezető réteg viszont tartalmaz egy vonala, amelyhez a mozgásátmenet hozzárendeljük. Így érjük el, hogy a mozgásátmenetek egy görbét kövessenek.*

4) **Irányított réteg:** *ez a rétegtípus csak akkor érhető el, amikor a fölötte lévő szomszédos réteg mozgásvezető réteg. Az irányított rétegen létrehozhatunk egy mozgásátmenetet, amely a mozgásvezető rétegen rajzolt görbét követi.*

5) **Maszkréteg:** *segítségével bármilyen alakzatot vagy filmklipszimbólumot elhelyezhetünk, amely meghatározza az alatta lévő maszkolt réteg látható részletét. A maszkrétegen oda rajzolunk, ahol a maszkon lévő lyukakat akarjuk látni.*

6) **Maszkolt réteg:** *ez a réteg csak akkor érhető el, amikor a közvetlenül fölötte lévő réteg maszkréteg. A maszkolt réteg tartalma láthatatlan, kivéve azokat a területeket, ahol a maszkrétegen objektumok szerepelnek, vagyis a maszkréteg fedi fel a maszkolt réteg egyes részeit.*

3.4 Hangok a Flashben

A Flash-ben többféle lehetőség is van a hangok használatára. Felhasználhatjuk a hangokat úgy, hogy folyamatosan halljuk, függetlenül az idősávtól, vagy össze is hangolhatjuk az animációval.

Hanganyagként használhatjuk a megosztott könyvtárban levőket a filmünk készítéséhez vagy importálhattuk hangfájlokat. Továbbá hangokat használhatunk a Sound objektumokban, a lejátszást pedig ActionScript-el vezérelhetjük.

3.4.1 Hangok betöltése

A Flash széleskörűen támogatja a hangokat, de nincs beépített lehetőség hangok felvételére vagy létrehozására. Keresnünk kell egy meglévő hangot, amely rendelkezésünkre áll, vagy hangprogram segítségével kell létrehozni azt.

Támogatott hangformátumok:

- MP3
- WAV
- AIF
- AU

Az AIF és a WAV között nincs alapvető különbség. Egy jó minőségű AIF ugyanolyan, mint egy jó minőségű WAV fájl. Az AU formátum tömörítése csaknem mindig alacsony minőségű, tehát gyakorlatilag elfelejthetjük ezt a formátumot. Az MP3 fájloknak mindig van valamekkora tömörítésük, tehát végső soron nem ezek a legjobbak. Viszont, ha egy MP3 fájl csak nagyon kis mértékben tömörített, a minősége jó marad.

Tehát két nyomos oka lehet az MP3 formátum használatának:

- Az egyetlen forrás egy MP3.

- Úgy tudjuk, hogy a rendelkezésre álló fájl már optimálisan tömörített.

3.4.2 Hangok használata

Most, hogy hangokat töltöttünk be, megvizsgálhatjuk, miképpen lehet elérni, hogy a megfelelő időben szólaljanak meg. Igazából csak egy helyen lehet a Flash-ben hangokat alkalmazni: a kulcskockában. Ha azt akarjuk, hogy egy hang megszólaljon, amikor csak a felhasználó egy gomb felé viszi a mutatót, akkor is egy kulcskockában kell elhelyezni a hangot. Amikor kijelölünk egy kulcskockát, a Properties tábla lehetőséget nyújt annak szabályozására, hogy milyen hang szólaljon meg, amikor elérjük az adott kulcskockát.

Végső soron a Properties tábla a legjobb módja annak, hogy lássuk, melyik hangot melyik kulcskockához adtuk. De ahogy a többi tábla, a Properties tábla is csak a kijelölt kulcskockában használt hangot jeleníti meg. Nagyon könnyű félreértelmezni ezt a táblát, mivel amikor eltávolítjuk a kulcskockák kijelölését, megváltozik.

Amikor a Properties tábla a kívánt kulcskockához tartozó hangot mutatja, eldönthetjük, hogy pontosan hogyan játszódjon le a hang. A legfontosabb, amit a hangok minden használatakor ki kell választani, a Sync (Összehangolás) beállítás. Ez azt szabályozza, hogy a hang adott példánya pontosan hogyan szólaljon meg – a hang és az animáció elemei közötti rangsort.

Összehangolási beállítások:

- **Event (Esemény):** *általában ez nyújtja a legjobb teljesítményt. Amikor az Event beállítás van kiválasztva, a hangok akkor szólalnak meg, amikor elérjük az adott kulcskockát, és addig szólnak, míg véget nem érnek.*
- **Start (Indítás):** *ez a beállítás majdnem ugyanaz, mint az Event, kivéve, hogy ugyanannak a hangnak nem lehet több példánya. Az Event eseteében a hangnak több rétege lehet. Ezzel szemben a Start elindítja a hangot, ha az még nem szól.*

- **Stop (Leállítás):** *ez elég különös beállítás - olyankor használatos, amikor azt akarjuk, hogy egy bizonyos hang elhallgasson.*
- **Stream (Folytonos):** *ennek a beállításnak az eredményeként a hang tökéletes összhangban marad az időszalaggal. A folytonos hangok akkor indulnak, amikor elérjük a kulcskockát. és addig szólnak, amíg van hely az időszalagon. A Strem beállításnak az az előnye, hogy az összhang mindig ugyanolyan marad.*

Tehát, az Event a rövid, hirtelen hangokhoz való. Tulajdonképpen az Event ajánlott minden olyan hanghoz, amelyik nem igényel nagyon pontos összehangolást. A háttérzenékhez, amelyek csak lejátszódnak és ismétlődnek, nincs szükség összehangolásra. Ezért ilyen célra az Event vagy a Start beállítás tökéletesen megfelel.

A Stop összehangolási beállítás nagyon hatásos, de eme módszer kissé cseles lehet, mert kulcskockánként csak egy hangot állít le. Végül a Strem beállítás egyetlen dologra jó: összehangolja a képet a hanggal

3.5 Animációk a Flashben

A Flash legerősebb oldala az animációk készítése. A program segítségével olyan animált objektumokat tudunk készíteni, amelyek a teljes rajzfelületen tudnak mozogni, megjelenni vagy eltűnni, forogni, továbbá méretet, színt és áttetszőséget változtatni. Készíthettük animációt úgy is, hogy megrajzoljuk annak minden képkockáját, de használhatjuk az a lehetőséget is, amikor csak az első és az utolsó képkockát adjuk meg és a Flash kiszámolja/megrajzolja a többit. Az utóbbi az úgynevezett átmeneti vagy tweened animáció.

3.5.1 A nyers animációs eljárás

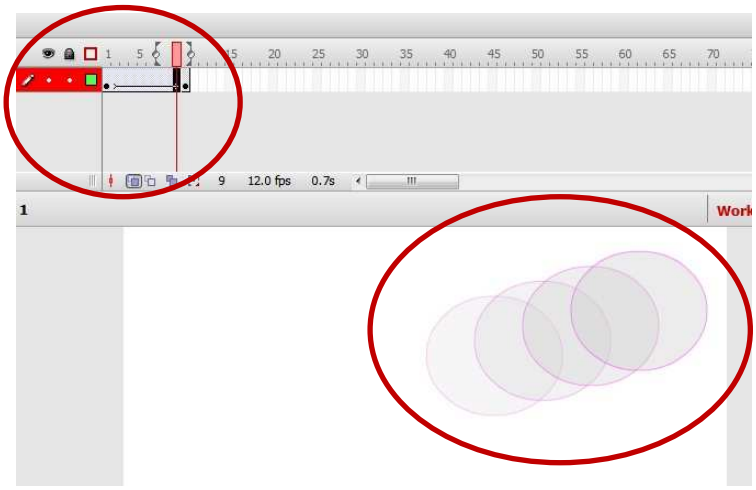
Ha már készítettünk pörgetőskönyvet, akkor tudjuk, hogyan kell létrehozni a képkockáról képkockára animációt. A pörgetőskönyv minden lapja egy kicsit eltérő képet tartalmaz, így amikor átpörgetjük a lapokat, a kép mozog. Tulajdonképpen a

Flashben is ez történik, csak itt az egyes lapokat az időszalag egyes kulcskockái jelentik. Akár egy könyv lapjaira rajzolunk egy képet, akár egy Flash kulcskockára, a művelet nehézkes és körülményes - ezért is hívják nyers eljárásnak.

A képkockáról képkockára animációs eljárás egyszerű. Csak elhelyezünk egy-egy kulcskockát minden képkockában. Minden képkockán teljesen más kép jelenik meg: néha gyökeres a változás, máskor apró a különbség.

A nyers animációs eljárást nagymértékben segíti a hártya használata, amit eredetileg a hagyományos animációban fejlesztettek ki. Amikor a művész kézzel rajzolja meg az egyes képkockákat, valahogyan el kell döntenie, hogy mennyit kell változtatni a képen egyik kockáról a másikra. Másolópapírra rajzolja fel a képkockát, amelyet az előző kocka fölé helyez. Így láthatja az előző képkockát, és ennek megfelelően rajzolhatja meg a következő képet.

A Flashben a hártyának ugyanilyen hatása van. A Flash hártyaszolgáltatása segítségével úgy szerkeszthetünk egy adott képkockát, hogy tetszőleges számú képkockát jeleníthetünk meg az aktuális kocka előtt és után. A hártya bekapcsolásához az időszalag képkockákat megjelenítő ablakában a bal szélső Onion Skin gombra kell kattintanunk és máris szembetűnő változást fogunk tapasztalni animációnkban a színpadon. A képkockák felett megjelenik egy nyitó és záró zárójelre emlékeztető szimbólum, amelyet, ha egymástól távolabbra állítunk, a színpadon több kocka tartalmát fogjuk látni egy időben (10. ábra). Ezzel segítve az aktuális képkocka tartalmának megrajzolását.



10. ábra: *Egyszerre több képkocka megjelenítése*

3.5.2 Animáció mozgásátmenet segítségével

Az átmenetkészítéskor a Flash kitölti a két kulcskocka között található üres képkockákat. A Flash kétfajta átmenettel rendelkezik: mozgás- és alakzatátmenettel. Most a mozgásátmenettel foglalkozunk, amely átalakítja az elhelyezkedés, méret, elforgatás, színezet és áttetszőség kliptulajdonságokat, valamint az alkalmazott szűrőket [9].

Ehhez az időszalagon az első kulcskockában létrehozunk egy tetszőleges alakzatot. Ez lehet kör, téglalap, négyzet, stb. Ezután tetszőleges pontban legyen ez most a 10. képkocka létrehozunk egy kulcskockát és az első kulcskockában lementett szimbólumot elhelyezzük a színpadon lehetőleg másik helyre. Ezt az alakzatot tetszőlegesen megformázhatjuk, például növelhetjük méretét, színét változtathatjuk, elforgathatjuk, stb., majd az első kulcskockára kattintva a Properties tábla Tween-Motion tulajdonságot választva el is készítjük az átmenetet. Lehet pásztázni és látjuk az eredményt. Ha az átmenetkészítéskor a kulcskockát nem a 10. hanem például a 30. kockába helyezzük az átmentetünk finomabban fog mozogni, de ez logikus is, mivel ugyanaz a távolág megtételére több ideje lesz alakzatunknak.

Mozgásátmenetünket finomhangolhatjuk a Properties tábla Edit gombra kattintva, vagy a mellette található Ease érték növelésével, csökkentésével.

A mozgásátmenet szabályai

- A kulcskockákban nem lehet több objektum.
- Az egyetlen meglévő objektumnak nem kell egyszerű alakzatnak lennie. Valójában az a legjobb, ha szimbólumpéldánnyá tesszük, mert így a fájlméret kisebb lesz.

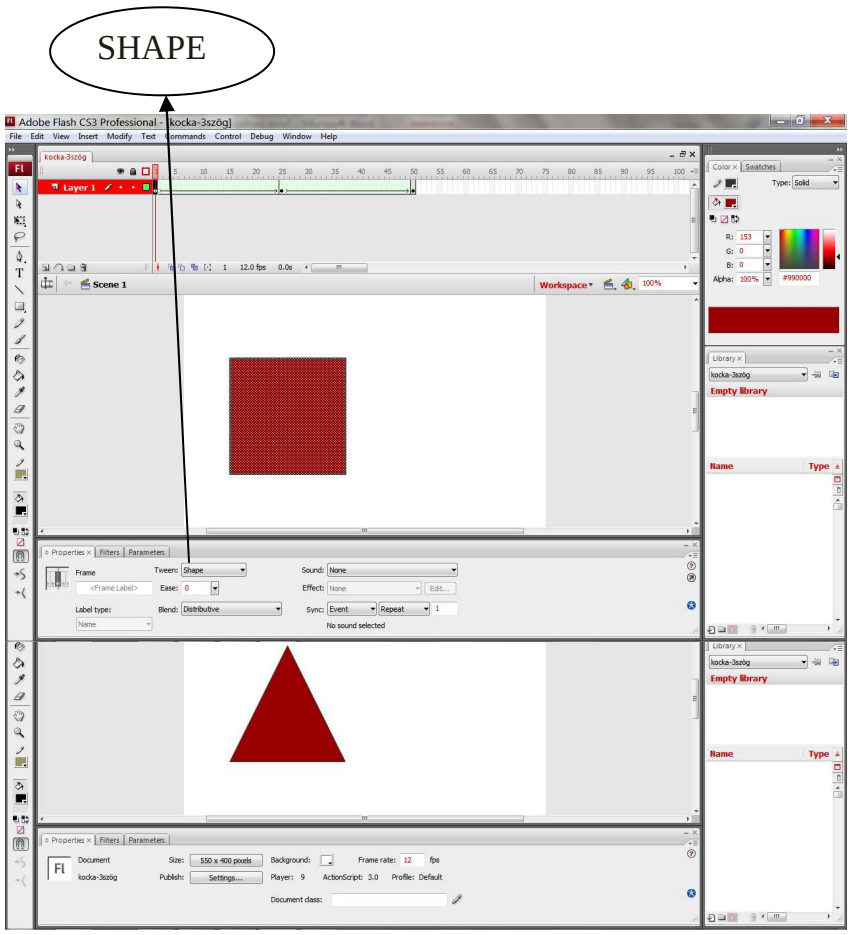
3.5.3 Animáció alakzatátmenettel

Számos lehetőség létezik, amellyel egy Flash fájl kisméretű és gyorsan lejátszható marad. A Könyvtár szimbólumainak újrafelhasználása és a mozgásátmentek készítése a két legjobb módszer. Sajnos az alakzatátmenet az egyik legkevésbé hatékony szolgáltatás, mert megnöveli a fájlméretet. Ennek ellenére az alakzatátmenet nagyon jól néz ki. A Flash-ben nem is lehet máshogy elérni a morph hatást.

Az átváltozás olyan animáció, mely az egyik alakzatot természetes módon átviszi a másikba.

Az alakzatátmenetek szórakoztatnak, mert igazán jól mutatnak, és könnyű őket létrehozni. A mozgásátmenetekhez képest dinamikusabbnak tűnnek, mert minden jellemző átalakul.

Alapvetően csak annyit teszünk, hogy megrajzoljuk a két kulcskockában szereplő alakzatot, és az első kulcskocka átmenet beállítását Shape- re állítjuk (11. ábra).



11. ábra: *Shape funkció*

Az alakzatátmenet szabályai

A Flash könyörtelen, ha nem tarjuk be a szabályait. Szerencsére az alakzatátmenetre vonatkozó szabályok nagyon egyszerűek: nincsenek csoportok és nincsenek alakzatok.

Csak mert az alakzatátmenet szabályai egyszerűek, még nem biztos, hogy könnyű olyan alakzatátmenetet készíteni, ami jól mutat. Számos eljárás van, amellyel a folyamatot könnyebbé és az eredményt jobbá tehetjük.

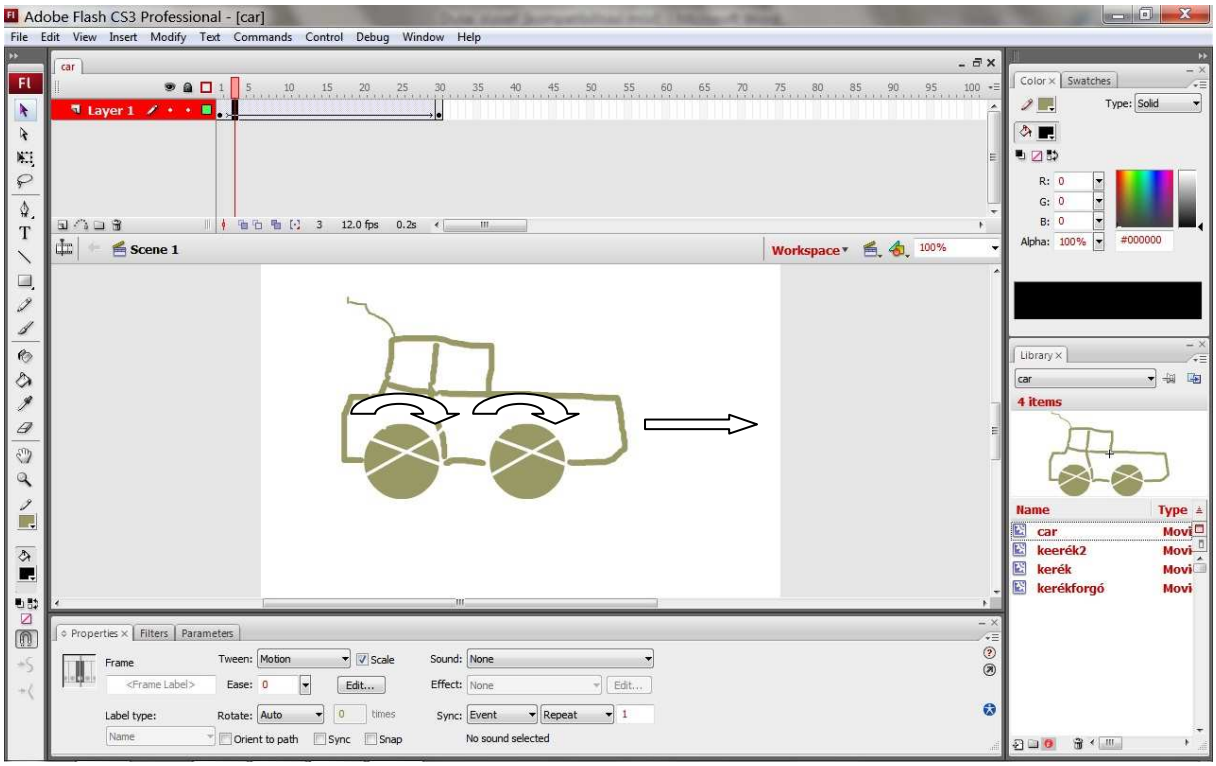
Lehetőségünk van a Flashben az alakzatátmenetek csiszolására és finomhangolására is. Ha az összes jelzésre odafigyelünk, akkor is kaphatunk olyan alakzatátmenetet, amelyik mindenre hasonlít, csak a várt alakzatra nem. A Flashnek van egy kifejezetten az alakzatátmenet-készítéssel kapcsolatos szolgáltatása, amelynek

segítségével megmondhatjuk a programnak, hogy valójában mit akarunk. Ezek az alakzattippek, amelyek a zagyva alakzatátmenetek helyett a várt eredményt kapjuk.

3.6 Animációk beágyazása

Egy filmklip belsejében létrehozhatunk egy animációt, majd animálhatjuk a szóban forgó klip egy példányát (12. ábra). Ez azt jelenti, hogy elkészíthetünk például egy forgó kerék klipjét, majd animálhatjuk a forgó kereket úgy, hogy ne csak forogjon, hanem a képernyőn keresztül is haladjon. Az animációkat filmklip- és garfikaszimbólumokba egyaránt be lehet ágyazni. A legjobb, ha úgy értelmezzük, hogy ha mindig filmklipet használunk, hacsak nincs rá nyomós okunk, hogy grafikaszimbólummal dolgozzunk.

- A beágyazáshoz hozzunk létre egy MovieClip szimbólumot, jelen esetben ez egy kerék.
- Majd ismét mentsük, de ezennel más néven pl.: forgó kerék.
- Az eredeti változatra duplán kattintva a szerkesztősávon láthatjuk, hogy a példányba kerültünk.
- Készítsünk egy mozgásátmenetet a 30. képkockáig, miközben a Properties táblán a Rotate-t óramutató járásával megegyezően 1-szeresre beállítjuk.
- Mennyünk vissza a jelenetben a Sceen1-re kattintva, majd hozzunk létre egy másik kerék példányt másolással és tegyük egymás mellé.
- Ezután rajzoljunk egy karosszériát és alakítsuk az egészet szimbólummá.
- Illesszünk be egy új kulcskockát ahol jónak gondoljuk. Például 40. kockában. Majd helyezzük a színpad jobb oldalára a kocsi szimbólumunkat. és a CTRL+ ENTER lenyomásával kerékforgással áthalad a képernyőn a kocsi.



12. ábra: *Bonyolultabb animáció*

3.6.1 Filmklipszimbólumok és grafikaszimbólumok összehasonlítása

Mind a filmklipek, mind a grafikaszimbólumok állhatnak egy vagy több kockából, de csak a filmklipek lejátszása ismétlődik automatikusan függetlenül attól, hogy hová helyeztük az adott klip példányait. Nem az számít, hogy az eredeti szimbólum grafika-e vagy filmklip. A szimbólumviselkedés csak a Könyvtárból áthúzott szimbólumok példányainak viselkedését befolyásolja. Csak a színpadon lévő példány szimbólumviselkedése számít. Ha áthúzunk egy filmklipet a Könyvtárból, kezdetben Movie Clip lesz a viselkedése, de ezt a Properties tábla segítségével Graphicra változtathatjuk.

A Graphic viselkedésre állított többkockás példányok rendelkeznek néhány egyedi lehetőséggel. A Properties tábla megváltozik, és néhány további lehetőség jelenik meg rajta. Amikor kijelölünk egy Graphic viselkedésű szimbólumot, a Properties táblán meghatározhatjuk, melyik kocka jelenjen meg először. Továbbá a

lenyíló lista más lehetőségei révén a Loop, a Play Once és a Single France beállítás közül választhatunk, amelyek ötvösézésével változtathatjuk, hogy egy Graphic viselkedésű példány pontosan hogyan jelenjen meg.

Ha összehasonlítjuk a Properties tábla azon állapotát, amelyen egy filmklip van kijelölve, azzal az állapottal, amelyen a grafika van kijelölve, észrevehetünk egy látszólag jelentéktelen mezőt a példánynév számára. Ezzel a filmklip-példányokat külön-külön elnevezhetjük a Properties táblán.

A többkockás grafikaszimbólumokat a Flash összehangolja, és illeszkedniük kell annak az időszalagnak a hosszához, amelyen elhelyezzük őket.

A filmklipként viselkedő példányok függetlenek az időszalagjuktól, mindig lejátsszák az összes kockájukat és ismétlődnek.

3.7 Videófájlok használata Flashben

Nem is olyan rég volt, hogy kisebb csodára volt szükség, ahhoz, hogy a Flashben videókat tudjunk megjeleníteni. A Flash videótámogatása viszont mára rendkívül kiforrottá vált. Külső videófájlokat tölthetünk be, vághatjuk és tömöríthetjük őket, és mindezt az On2 egyik csúcstechnikáját képviselő VP6 kodekjének segítségével, ami kis fájlméret mellett is kiváló minőséget biztosít. Habár a legtöbb technika részletet automatikusan kezeli a program, mégis rengeteg módon befolyásolhatjuk a végeredményt.

3.7.1 Videók beágyazása és lejátszásuk

Amikor videót importálunk, eldönthetjük, hogy beágyazzuk, az időegyenesre helyezve azt, így az a Flash dokumentum részévé válik, vagy választhatjuk azt is, hogy kívülről betöltjük, és ekkor különálló fájl marad. A beágyazott videók kiválóan használhatóak kisebb, audiósávok nélküli videóklipek esetén, ha viszont 10

másodpercnél hosszabb videóklipet használunk, töltsünk be inkább külső videófájlokat, és töltsük le a videót fokozatosan vagy folyamatosan Flash Video Server segítségével.

A beágyazás lépései:

1. Válasszuk a File-Import-Import Video fájl menüpontot. Válasszuk ki a megnyitni kívánt fájlunkat, majd kattintsunk a Next gombra. Ekkor megjelenik a Közzététel ablak.
2. Itt válasszuk az Embed (beágyazás) lehetőséget, hogy a tömörített videót .fla fájlba mentsük, és kattintsunk a Next gombra.
3. Következő ablakban különböző beállításokat adhatunk meg. Itt válasszuk az Edit the video first beállítást és, ha beállítottunk minden mást, amit szerettünk volna kattintsunk a Next gombra.
4. Megjelenő Split Video ablakban feldarabolhatjuk a videónkat, majd a plusz gombbal a fájlunkhoz adhatjuk azt.
5. Kattintsunk a Back gombra. Az Embedding ablakba visszakerülve jelöljük be az alul található Embed the entire video gombot, majd Next.
6. Most az Encoding ablakban a videó kódolás minőségét, sávszélességét adhatjuk meg. Ha mindent kellő módon beállítottunk mennyünk tovább, itt ellenőrizhetjük a beállításainak, majd kattintsunk a Finish gombra.

Ezennel beágyaztunk egy videót.

Függetlenül attól, hogy a végső videót beágyazzuk vagy külső fájlként kívánjuk lejátszani, a videót mindig tömörítenünk kell. A forrás formátuma többféle is lehet, de mindig tömörítenünk kell a Flashben rendelkezésre álló két kodek egyikével.

Lényegében annyi, hogy kiválasztjuk a File-Import menüpontot, majd kiválasztjuk a videót. A hangokhoz és a grafikákhoz hasonlóan azonban különféle videóformátumok léteznek, és mindegyiknek megvannak a saját jellemzői. Ahhoz

pedig rengeteg munkára van szükség, hogy elérjük, hogy a videó jól is nézzen ki, tökéletes legyen a lejátszása, és gyorsan és is töltődjön le.

A számítógépre szánt videók kiválasztásánál és elkészítésénél néhány dolgot figyelembe kell vennünk.

- Csak akkor használjunk videót, ha tényleg szükséges.
- A videó tartalmát illetően óvakodjunk az aprócska elemek használatától, amelyek egy kisebb videóablakban nehezen vagy egyáltalán nem láthatóak.
- A nézőpontok, illetve a kamera látószögének váltogatásával mindig jó eredményt érhetünk el, csak ne feledkezzünk el a kisebb képernyőkön nehezebben látható részletekről. Ezen kívül a drámai beállításokat mindig okkal használjuk, ne csak ötletszerűen.
- Bánjunk csínján a különféle trükkökkel, mert mindennek ára van, legyen az a gyengébb teljesítmény, a lassú letöltéshez vezető nagy fájlméret, de a felesleges trükkökbe bele is fáradhat a néző.

A fentieken túl az optimalizálásra is figyelmet kell fordítanunk.

Támogatott videóformátunok:

A Flash különböző formátumú videókat képes tömöríteni, amelyek mindegyike a számítógépen található digitális fájl. Ha csupán egy videószalagunk van, előbb digitalizálnunk kell a filmet [9].

Bármilyen módszert is választunk a digitalizált fájl létrehozására, az alábbi formátumok egyikét kell választanunk:

- QuickTime (.mov)
- Digitális videó (.dv)
- MPEG (.mpg vagy .mpeg)
- Windows Media (.asf vagy wmv)

- Video for Windows (.avi)

Valószínűleg csak az első három egyikére lesz szükségünk.

Videók lejátszása

1) A beágyazott videó lejátszása

A videó képi elemei grafikaszimbólumokként viselkednek. A beágyazott videók persze nem egészen olyanok, mint a grafikák, a gombok vagy a filmklipek, hiszen nem készíthetünk velük átmeneteket, és a színárnyalatukat sem módosíthatjuk úgy, mint a többi szimbólumnál. A videókat azonban a szimbólumtípusok közül mégis a grafikaszimbólumokra hasonlítanak leginkább.

2) Külső videófájlok lejátszása

Először létre kell hoznunk őket. Erre alapvetően 4 módszer létezik:

- Betöltünk egy videót a Flashbe, és az első közzétételi beállítás valamelyikét választjuk. Ennek az az előnye, hogy az .flv fájl létrehozása mellet a Flash egy összetevőt is beállít a számunkra.
- Közvetlenül egy támogatott videószerkesztőben hozzuk létre a filmet egy olyan számítógépen, amelyen telepítve rendelkezésre áll az Adobe Flash Video Encoder.
- A Flashben először beágyazzuk a videót, majd a Könyvtárban levő videóelemből .flv fájlt készítünk. Et eredményezi messze a leggyengébb minőséget, de hasznos lehetőség, ha gyorsan van szükségünk egy .flv fájlra.

Külső .flv fájlok estén a kép és a hang összhangban marad.

A külső fájlok lejátszása esetén: Először is, fel kell töltenünk mind az .swf, mind a .flv állományt. Ha beágyazzuk a videót, csak a videó képkockáiról kapunk előképet. Ha animációt szeretnénk a videóra rajzolni, beágyazott videót kell használnunk. A lényeg tehát, hogy az .flv fájlok lejátszásakor szembe kell nézni néhány korlátozással.

3.7.2 A minőség és fájlméret optimalizálása

A kódolási lehetőségek és trükkök között könnyű szem elől téveszteni a legfontosabb célt: azt, hogy a lehető legjobb minőségű és a lehető leggyorsabban letölthető videót állítsuk elő.

A fájl méretét leginkább befolyásoló két tényező a képkockaszám és a képméret. A kevesebb képkockaszám azonban nem csak kisebb fájlméretet jelent, hanem gyengébb minőséget is, különösen ha a videó sok mozgást tartalmaz.

A legjobb, amit tehetünk, hogy kiragadunk egy kicsi, jellemző részletet a videóból, kipróbáljuk különböző képkockaszám- beállítások mellett, és addig csökkentjük a képkockaszámot, amíg a minőség még elfogadható marad.

A kisebb képméret ugyanígy csökkenti a fájlméretet.

3.8 Flash és a Web

Ebben a részben arról lesz szó, hogy hogyan tudjuk közzétenni Flash fájlunkat a világhálón, hogy ezzel is emeljük weblapunk látványosságát vagy csupán, hogy mások megcsodálhassák.

3.8.1 Egyszerű és beállításokkal történő közzététel

A Flashben található Publish szolgáltatás segítségével gyerekjáték filmünket feltenni a Világhálóra. Ez lényegében annyi, hogy a File- Publish menüpontra kattintva nem csak az .swf fájl, hanem a hozzá tartozó HTML állomány, illetve az esetleg szükséges JavaScript fájlok is elkészülnek.

Egy HTML fájl leírja, hogy hol található a Flash film, és miként kell azt megjeleníteni. Amikor egy .swf fájlra hivatkozunk, akkor megadhatjuk annak szélességét, magasságát, hogy folytonosan ismétlődjön-e, hogy a betöltés után megálljon, és még sok mást is.

A Flash közzétételi szolgáltatása elkészíti számunkra a HTML fájlt, így nem kell megtanulnunk a HTML nyelvet.

Közzététel lépései:

1) Egy új fájlt hozzunk létre és mentsük el.
2) A File-Publish Settings menüpont kiválasztásával nyissuk meg a Publish Settings-et és válasszuk a Formats lapot és ügyeljünk arra, hogy csak Flash és HTML lehetőségek legyenek bejelölve. Ezután kattintsunk az OK-ra.
3) Válasszuk a File-Publish menüpontot. a mentett fájl könyvtárában létrejön többek között egy HTML fájl, ha erre duplán kattintunk, akkor megtekinthetjük a filmünket a böngészőnkben.
4) Végezetül: Ha a .html és .swf fájlokat feltöltjük egy webkiszolgálóra, akkor azt mostantól bárki megtekintheti.

Közzététel beállításokkal:

1) Nyissunk megy egy korábban elkészítet filmet, vagy most hozzunk létre egyet. Feltétel, hogy legyen benne képi váltás.

2) Válasszuk a File-Save As menüpontot, és mentsük a fájlt egy új mappába, amely nem tartalmaz más állományt.
3) Kattintsunk a File- Publish Settings parancsra. Az ezután megjelenő párbeszédablakban történő változtatások a fájllal együtt mentődnek.
4) Váltsunk a Publish Settings ablak Formats lapjára, hogy megadhassuk, mely formátumokban kívánunk menteni.
5) Váltsunk a Flash lapra, és vizsgáljuk meg a Version beállítást. Itt attól függően döntsünk, hogy várhatóan milyen felhasználók fogják megtekinteni a weblapon.
6) Mennyünk a HTML lapra; itt a HTML kódot finomíthatjuk. Ezután a Template listából válasszunk, majd jelöljük be a Detect Flash Version-t.
7) Állítsuk a Dimension beállítást Percent értékre, majd mind a Wiht, mind a Height mezőbe írjunk pl.: 100-at, hogy a film teljesen kitöltse a böngésző ablakát.
8) Ha két kiválasztott formátumhoz tartozó mindkét lapon végeztünk, kattintsunk az OK gombra. Mentsük a Flash fájlt is, majd válasszuk a Publish menüpontot.
9) Végezetül keressük meg a létrejövő HTML fájlt és osszuk meg.

3.8.2 Publikálás mobil eszközökre

A Flsah Lite használatával mobiltelefonokra és más hordozható eszközökre is tehetünk Flash tartalmat. A Flash Lite dokumentum készítéséhez válasszuk ki a File-New menüpontot, azután a Flash File (Mobile) típust, kattintsunk az Ok gombra, majd válasszuk ki az Adobe Device Central alkalmazásban a céleszközöket (13. ábra). A Flash automatikusan a megfelelő beállításokkal nyitja meg a fájlt.

Publikálás előtt tekintsük meg a Device Central segítségével a tartalom előnézetét az adott mobile eszközön.

13.ábra: *Publikálás mobilon*

3.9 Adobe Flash hardveres követelménye

Az Adobe Flash programnak kicsi a rendszerkövetelménye:

Rendszer: Microsoft Windows 2000, XP

Processzor: Intel Pentium III 800MHz vagy nagyobb teljesítményű

Képernyő:1024x768 - 16 bit

Memória:1024 MB

Tárhely:710 MB

Egyéb:

- Internet az aktiváláshoz
- QuickTime

Én egy

14. ábra: *Rendszertulajdonság*

tulajdonságokkal bíró gépet használtam a szakdolgozat írása közben (14. ábra) és saját tapasztalatom szerint elegendő volt.

4. Összefoglalás

A Flash eredetileg egy egyszerű animáció-szerkesztő programnak indult, mára azonban egy komplex platformmá fejlődött, amely egyaránt alkalmas mobil tartalmak, interaktív bemutató vagy akár weboldalak készítésére is [4]. A technológia gyors elterjedésének több oka van, például hogy megjelenésekor úttörőnek bizonyult a dinamikus tartalmak kezelésében. Ma már szinte minden felhasználói számítógépen megtalálható, mert a legtöbb böngészőprogrammal és elterjedt felhasználói operációs rendszerrel kompatibilis. Viszonylagos kis mérete és egységes, megbízható megjelenése a legstabilabb kliens oldali interaktivitást biztosító technológia a webes fejlesztések területén. A webes tartalmak jó része, és számos szolgáltatás ma már elképzelhetetlen lenne Flash (vagy a mostanában megjelenő alternatívái) nélkül.

Záródolgozatomban az Adobe Flash CS3 program segítségével szemléltettem, hogy mi is az a technika, ami a webfejlesztést forradalmasította. Újító jellege miatt tartottam fontosnak, hogy a webfejlesztésről szóló dolgozatom elkészítsem.

Ezzel az áttekintéssel, leírással remélem, hogy a közelebbi betekintést elősegítettem a Flash animációkészítés világába. Próbáltam az alapoktól indulva, egyértelműen bemutatni mindazt, amit fontosnak találok.

A Flash technológia jövője mindenképpen a RIA (Rich Internet Application) megvalósítása. Ez egy olyan webes felület, amely tulajdonképpen webalkalmazásként működik. A jövő mindenképpen a Flash-ről, az Adobe Flex alkalmazásról (Flash alapú fejlesztői környezet) és az egyelőre még fejlesztés alatt álló, következő generációs RIA fejlesztő eszközről, az Apollo alkalmazásról fog szólni.

5. Irodalomjegyzék, források

1. Adobe Flash CS3 Professional (melléklet), 2008, Budapest Perfact- Pro Kft.
2. Flash Mx Stúdió, 2003, Budapest Horváth és Fellner Kft.
3. Kerman, Phillip, Tanuljuk Meg Az Adobe Flash CS3 Professional Használatát 24 Óra Alatt, 2007, Budapest Kiskapu Kft.
4. http://hu.wikipedia.org/wiki/Adobe_Flash
5. http://hu.wikipedia.org/wiki/Vektorgrafika
6. http://pcworld.hu/vizsgalat-alatt-az-adobe-macromedia-ugylet-20050713.html
7. http://positionabsolute.net/blog/2007/04/actionscript-vs-javascript.php
8. http://szoftver.hu/szoftver/Adobe/Flash/CS3+Professional
9. http://www.adobe.com/hu/products/flash/
10. http://www.fokusz.info/index.php?cid=1173942465&sid=1511009955
11. http://www.fokusz.info/index.php?cid=1173942465&sid=1996701914
12. http://www.hwsw.hu/hirek/30310/finisben-az-adobe-es-a-macromedia-osszeolvadasa.html
13. http://www.vassg.hu/pdf/grafika6.pdf

Köszönetnyilvánítás

Szeretném megköszönni témavezetőmnek, Dr. Tornai Róbertnek, hogy segített szakdolgozatom elkészítésében.