Bankovní institut vysoká škola Praha
zahraničná vysoká škola Banská Bystrica

Webové technológie pre realizácie klientsky bohatých aplikácií (CRA)

Bakalárska práca

Lucia Štefániková **september 2013**

Bankovní institut vysoká škola Praha
zahraničná vysoká škola Banská Bystrica

Katedra kvantitatívnych metód a informatiky

Webové technológie pre realizácie klientsky bohatých aplikácií (CRA)

Web technology for the implementation of rich client applications (CRA)

Bakalárska práca

Autor: **Lucia Štefániková**

Informačné technológie

Vedúci práce: **Mgr. René Klaučo, PhD**

Banská Bystrica **september 2013**

Vyhlásenie

Vyhlasujem, že som bakalársku prácu spracovala samostatne a s použitím uvedenej literatúry. Svojím podpisom potvrdzujem, že odovzdaná elektronická verzia práce je identická s jej tlačenou verziou a som oboznámená so skutočnosťou, že sa práca bude archivovať v knižnici BIVŠ a ďalej bude sprístupnená tretím osobám prostredníctvom internej databázy elektronických vysokoškolských prác.

V Nových Zámkoch, dňa 26.08.2013 Lucia Štefániková

Vlastnoručný podpis

Poďakovanie

Rada by som v prvom rade poďakovala vedúcemu mojej bakalárskej práce Mgr. Renému Klaučovi PhD. , za jeho cenné rady, pripomienky a názory pri tvorbe mojej bakalárskej práce. Ďalej by som sa chcela poďakovať Richardovi Strachemu za ústretovosť a pomoc pri tvorbe práce. Osobitné poďakovanie patrí mojim najbližším za ich podporu a pochopenie.

V Nových Zámkoch, dňa 26.08.2013.

ANOTÁCIA

ŠTEFÁNIKOVÁ, Lucia: **Webové technológie pre realizáciu klientsky bohatých aplikácií (CRA).** [Bakalárska práca]. Bankovní institut vysoká škola Praha, zahraničná vysoká škola Banská Bystrica. Katedra kvantitatívnych metód a informatiky. Vedúci práce: Mgr. René Klaučo, PhD. Rok obhajoby: 2013. Počet strán: 52.

Bakalárska práca sa zaoberá webovými technológiami pre klientsky bohaté aplikácie. Prvá kapitola je zameraná na teoretický rozbor a charakteristiku hlavných pojmov v predmetovej oblasti. V druhej analytickej kapitole sa práca zaoberá opisom tvorby aplikácií typu RIA v prostredí Adobe Flash Professional CS5 v jazyku ActionScript 3.0 V uvedenej kapitole sú zadefinované najefektívnejšie typy písania kódu v jazyku ActionScript. Tretia kapitola sa zaoberá vytvorením reálnej animácie v prostredí Adobe Flash Professional CS5 s použitím jazyka ActionScript 3.0 s podrobným popisom tvorby.

Kľúčové slová: RIA aplikácie, webové technológie, AdobeFlash, Flash Professional CS5, ActionScript.

ANNOTATION

ŠTEFÁNIKOVÁ, Lucia: **Web technologies to implement rich client applications (CRA)**. [Bachelor thesis]. Bankovní institut vysoká škola Praha, zahraničná vysoká škola Banská Bystrica. Department of Quantitative Methods and Computer Science. Supervisor: Mgr. René Klaučo, PhD. Year of defense: 2013. Number of pages: 52.

Bachelor thesis deals with web technologies for rich client applications. The first chapter is a theoretical section outlining the terms in the subject area. In the second chapter of the analytical work is concerned with the description of the type of RIA applications in Adobe Flash Professional CS5 in ActionScript 3.0 in that chapter are defined most effective types of writing code in the ActionScript language. The third chapter deals with the creation of real animation in Adobe Flash Professional CS5 using ActionScript 3.0 with a detailed description of.

Keywords: RIA applications, web technologies, AdobeFlash, Flash Professional CS5, ActionScript.

OBSAH

ÚVOD

Cieľom predloženej bakalárskej práce je vysvetlenie a overenie realizácie klientsky bohatých aplikácií. Prvým čiastkovým cieľom je vymedzenie pojmov v predmetovej oblasti. Druhým je opis tvorby aplikácií typu RIA v prostredí AdobeFlash s využitím jazyka ActionScript. Tretím je vytvorenie programu v jazyku ActionScript.

Bohaté internetové aplikácie (angl. *„Rich Internet Applications"*) sú webové aplikácie, ktoré majú vlastnosti grafických desktopových aplikácií. Vďaka súčasným vývojovým nástrojom môžu byť aplikácie založené na RIA rýchlejšie a užitočnejšie. Používateľovi poskytnú kvalitnejší vizuálny zážitok a vyššiu interaktivitu v porovnaní s tradičnými aplikáciami. RIA sú technológie, ktoré umožňujú vytvárať webové stránky a aplikácie, ktoré nielen ponúkajú určité informácie ale taktiež pomerne vysoký podiel funkcionality. V jednoduchosti je možné RIA technológie definovať ako technológie umožňujúce vývoj a prevádzkovanie webových aplikácií, ktorých užívateľské rozhranie prináša vizuálne a funkčné prvky známe z klasických aplikácií.

Predložená bakalárska práca pozostáva z troch hlavných kapitol. V prvej kapitole sa venuje opisu základných pojmov, vznikom RIA aplikácií, Adobe Flash s pomocou ktorého je v tretej kapitole vytvorená animácia s využitím jazyka ActionScript 3.0 a vývojovým prostredím Microsoft Silverlight. Venuje sa opisom vývoja webových aplikácií, ktoré môžu byť rozdelené do dvoch základných oblastí: na strane klienta (client-side), kde ide napríklad o technológie AJAX, JavaScript, Microsoft Silverlight, Adobe Flash a ďalšie alebo na strane servera (server-side), a to sú napríklad ASP.NET, C#.NET, PHP, Java, Perl a podobne. Druhá kapitola sa zaoberá opisom tvorby aplikácií typu RIA v prostredí AdobeFlash s využitím skriptovacieho jazyka ActionScript. Táto kapitola opisuje aj použitie webových technológií od spoločnosti Adobe, v ktorých sa dajú RIA aplikácie vytvárať. V tretej kapitole, ako je už vyššie uvedené je vytvorená animácia v prostredí Adobe Flash Professional CS5 s využitím skriptovacieho jazyka ActionScript 3.0. V tejto kapitole je opísaná tvorba grafickej animácie v jazyku ActionScript 3.0. Animácia spočíva v postupnom formovaní Postavičky v prostredí Adobe Flash CS5, vytváraním animácií a nasledovným napísaním kódu v jazyku ActionScript 3.0.

1 CHARAKTERISTIKA ZÁKLADNÝCH POJMOV V PREDMETNEJ OBLASTI

Pojmom *Client Rich Applications*[1] (CRA) sú označované dynamicky generované webové aplikácie, ktoré svojim kvalitne spracovaným, vysoko interaktívnym ovládaním dosahujú takmer úroveň bežných desktopových aplikácií. V tejto súvislosti sa častejšie stretávame so synonymickým označením RIA[2].

1.1 RIA - Rich Internet Applications

Bohaté internetové aplikácie (angl. *„Rich Internet Applications"*) sú webové aplikácie, ktoré majú vlastnosti niektorých grafických desktopových aplikácií. Vďaka súčasným vývojovým nástrojom môžu byť aplikácie založené na RIA rýchlejšie a užitočnejšie. Používateľovi poskytnú kvalitnejší vizuálny zážitok a vyššiu interaktivitu v porovnaní s tradičnými aplikáciami. RIA väčšinou beží na webovom prehliadači, ale často pre svoje spustenie vyžaduje prítomnosť rozširujúceho plug-inu. Prví používatelia internetu si najčastejšie posielali textové elektronické správy. Postupným vývojom a štandardizáciou technológií HTTP[3], HTML[4], CSS[5], JavaScript[6] a pod., používatelia získali prístup ku graficky pokročilejším webovým aplikáciám.

S použitím súčasných technológií môžu programátori vložiť takmer akúkoľvek funkcionalitu do grafického rozhrania založeného na webových technológiách a zaistiť tak, že vyzerá a správa sa ako tradičný softvér. Vďaka moderným nástrojom vývojári dokážu vytvoriť komplexnú aplikáciu s plnohodnotnou multimediálnou podporou, ako napríklad rôzne fonty, bitmapové a vektorové grafické súbory, on-line konferencie, audio, video, interaktívne prvky 2D a 3D animácie a pod. Tieto aplikácie dnes poskytujú možnosti, ktoré boli pred niekoľkými rokmi v čase obyčajného čítania a surfovania po webe víziou budúcnosti (*Developing with Ext GWT,35s).*

[1] CRA – Client Rich Applications,
[2] RIA – Rich Internet Applications,
[3] HTTP - HyperText Transfer Protocol,
[4] HTML - HyperText Markup Language,
[5] CSS - Cascading Style Sheets,
[6] JavaScript – Programming language – Programovací jazyk,

1.2 Vznik RIA aplikácií

Na začiatku tretieho tisícročia začala vznikať myšlienka tvorby internetových aplikácií s bohatším multimediálnym obsahom a niektorými kľúčovými charakteristikami klasických desktopových aplikácií. Tento myšlienkový prúd vzišiel z niekdajšej spoločnosti FutureSplash, pôvodného tvorcu technológie Flash. Potreba nového prístupu k tvorbe internetových aplikácii vznikla z dôvodu vylepšenia, zrýchlenia a spríjemnenia služieb poskytovaných na internete. Pre pojem RIA neexistuje žiadna presná definícia, ak však tento akronym analyzujeme, odvodíme si niektoré kľúčové aspekty tejto vetvy aplikácií (*RIA v podnikových aplikáciách a .NET RIA sevices, Lukáš Zdechovan, 2010*).

Rich – ako bohaté používateľské rozhranie, ktoré sa neobmedzuje na okruh pár základných prvkov pre zobrazenie dát a interakciu, ktoré ponúka štandard HTML. Aplikácia reaguje na akcie používateľa okamžite, keďže sama beží priamo u klienta. Komunikácia so serverom vo forme požiadaviek a odpovedí prebiehajú asynchrónne (na pozadí), podobne ako v prípade bežného softvéru nainštalovaného v operačnom systém osobného počítača.

Internet – ako internetová aplikácia, ktorej primárnou výhodou je vysoká dostupnosť a rýchly prístup z pohľadu používateľa, ľahšia distribúcia, aktualizácia či správa z pohľadu tvorcu.

Application – ako skutočná aplikácia, ktorej úlohou je poskytnúť istú funkcionalitu, ktorá má byť prínosom pre používateľa. Úlohou teda nie je výhradne šírenie informácií ako u bežných webových stránok či reklamných bannerov.

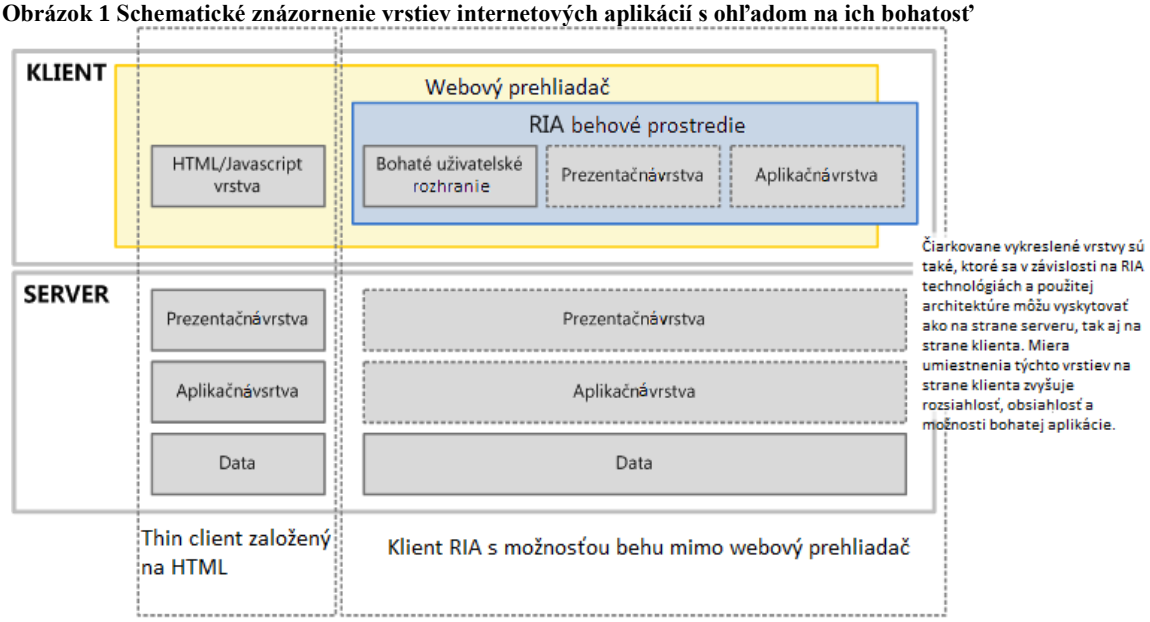
Technológie umožňujúce tvorbu RIA aplikácií vznikali po spoločnosti Macromedia aj u ďalších lídrov na trhu softvérových platforiem. Nad Flash-om sa u spoločnosti Adobe vyvinul framework Flex, Microsoft prišiel s platformou Silverlight, Sun Microsystems začal s vývojom JavaFX.

Alternatívou k RIA, ktorá vznikla u bežných webových aplikácií je technika AJAX, využívajúc skriptovací klientský programovací jazyk Javascript pre asynchrónnu komunikáciu s webovým serverom pomocou XML dokumentov. Bohatšie používateľské rozhranie sa v súčasnosti dosahuje využitím JavaScriptových knižníc na manipuláciu s DOM ako jQuery či mooTools.

1.3 Charakteristika RIA aplikácií

Pojem RIA si musíme predstaviť ako technológie, ktoré umožňujú vytvárať webové stránky a aplikácie, ktoré nielen ponúkajú určité informácie ale taktiež pomerne vysoký podiel funkcionality. Pod touto funkcionalitou si je možné predstaviť predovšetkým funkcie bežne využívané a implementované v klasických desktopových aplikáciách, resp. aplikáciách pre kancelársky softvér, účtovné systémy, grafické a multimediálne nástroje, a ďalšie aplikácie, ktoré internetovú konektivitu využívajú iba ku komunikácii a získavaniu potrebných dát, kde všetky ich ďalšie spracovanie a vizualizáciu rieši klientská časť systému resp. aplikácie. V jednoduchosti je teda možné RIA technológie definovať ako technológie umožňujúce vývoj a prevádzkovanie webových aplikácií, ktorých užívateľské rozhranie prináša vizuálne a funkčné prvky známe z klasických lokálnych aplikácií (napríklad drag and drop, vysoko funkčné formulárové prvky, ponuky, grafické prvky). Behové prostredie týchto aplikácií je webový prehliadač ktorý tvorí rozhranie iba medzi systémom, resp. aplikáciou a užívateľom, prípadne doplnený o zodpovedajúci plug-in zaisťujúci behové prostredie pre konkrétnu RIA technológiu. Ak vychádzame z informácií popisujúcich rôzne typy klientov, potom si môžeme urobiť pomerne presnú predstavu, kam zaradiť RIA technológie, resp. ich aplikácie. Niekedy je možné stretnúť sa s tým, že RIA technológie spadajú pod pojem Web 2.0[7] (*Rýchly vývoj aplikácií v jazyku Visual Basic 2010 pre systém Windows,50s).*

Obrázok 1 Schematické znázornenie vrstiev internetových aplikácií s ohľadom na ich bohatosť

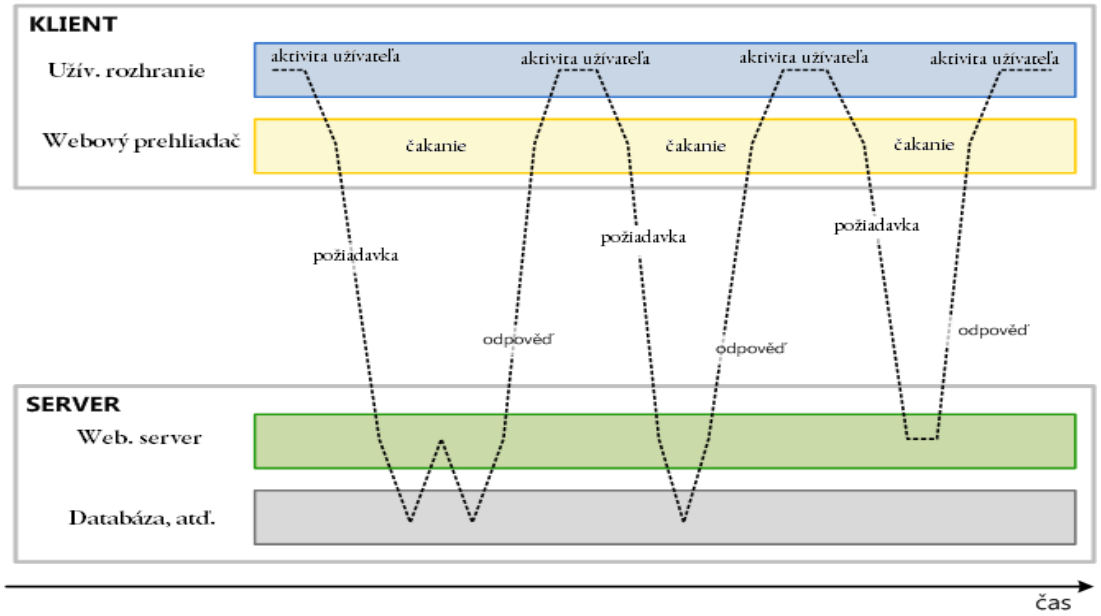


Zdroj: http://silverlight.cs.vsb.cz/01-technologie-a-ria.aspx

[7] Web 2.0 – ustálené označenie pre etapu vývoja webu,

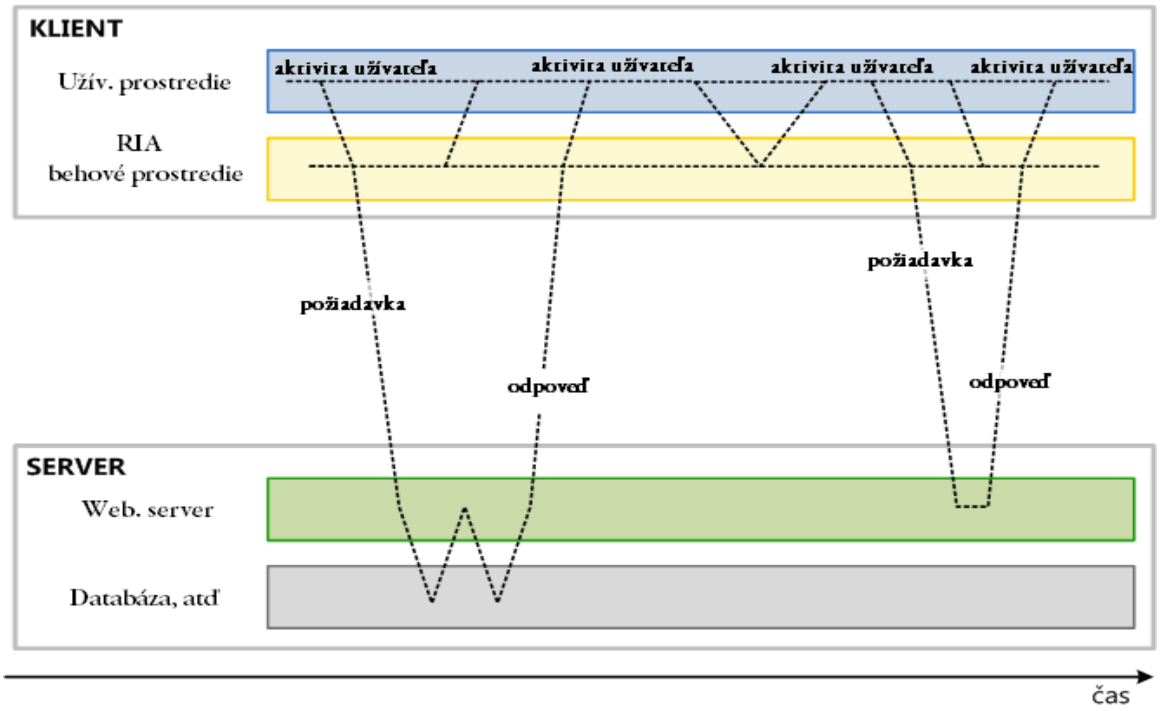
Oproti webovým majú RIA aplikácie jeden zásadný rozdiel, a to v systéme komunikácie klientskej a serverovej časti. Aby bolo možné ponúknuť bohatú funkcionalitu grafického prostredia, nie je možné využívať štandardné synchrónny prístup komunikácie typu požiadaviek a odpovedí, ale je potreba využiť prístup asynchrónny. Aplikácia nekomunikuje so serverom priamo ale využíva behové prostredie, čo znamená, že umožňuje väčšiu flexibilitu a rôznorodosť pri realizácii konkrétnej komunikácie. Táto medzivrstva nezávisle odosiela požiadavky a spracúva odpoveď a následne ju predáva späť aplikácii, a to tak aby aplikácia fungovala asynchrónne s ohľadom na túto komunikáciu *(http://www.itnews.sk/download ,9.4.2011)*.

Obrázok 2 Znázornenie synchrónnej komunikácie klasickej webovej aplikácie



Zdroj: http://silverlight.cs.vsb.cz/01-technologie-a-ria.aspx

Obrázok 3 Znázornenie konceptu fungovania a komunikácie RIA technológií na asynchrónnom princípe



Zdroj: http://silverlight.cs.vsb.cz/01-technologie-a-ria.aspx

RIA aplikácie sa od tradičných webových aplikácií odlišujú predovšetkým v nasledujúcich kľúčových bodoch *(http://adobe.com, 20.2.2013):*

- *Priama interakcia*: V tradičných webových aplikáciách je interakcia s používateľom obmedzená na niekoľko základných ovládacích prvkov: checkboxy, radio buttuny a formulárové polia. To často znemožňuje tvorbu použiteľných a užitočných aplikácií. RIA môžu využívať bohatšie spektrum ovládacích prvkov, ktoré umožňujú vyššiu efektivitu a vyšší komfort pre používateľa. V RIA môžu napr. používatelia priamo editovať prvky stránky alebo meniť pomocou drag-and-drop nástrojov. Taktiež môžu vykonávať operácie ako postupné prechádzanie mapy alebo iného obrázku.
- *Obnovovanie časti stránky*: Štandardné HTML stránky sú zobrazené naraz resp. ako celok. Pokiaľ je na stránke niečo zmenené, musí byť zmena odoslaná na server, ktorý vykoná úpravy, vygeneruje a pošle späť celú stránku. Tradičné webové aplikácie nútia používateľa čakať (kvôli problémom s pripojením siete, obmedzením výpočtového výkonu a iným problémom), pokiaľ sa neaktualizuje celá stránka. Aj so širokopásmovým pripojením môže byť doba čakania príliš dlhá a nepríjemná. RIA aplikácie obsahujú doplnkové technológie, ako real-time streamovanie, vysoko výkonné klientské pracovné stanice a lokálny cacheovací

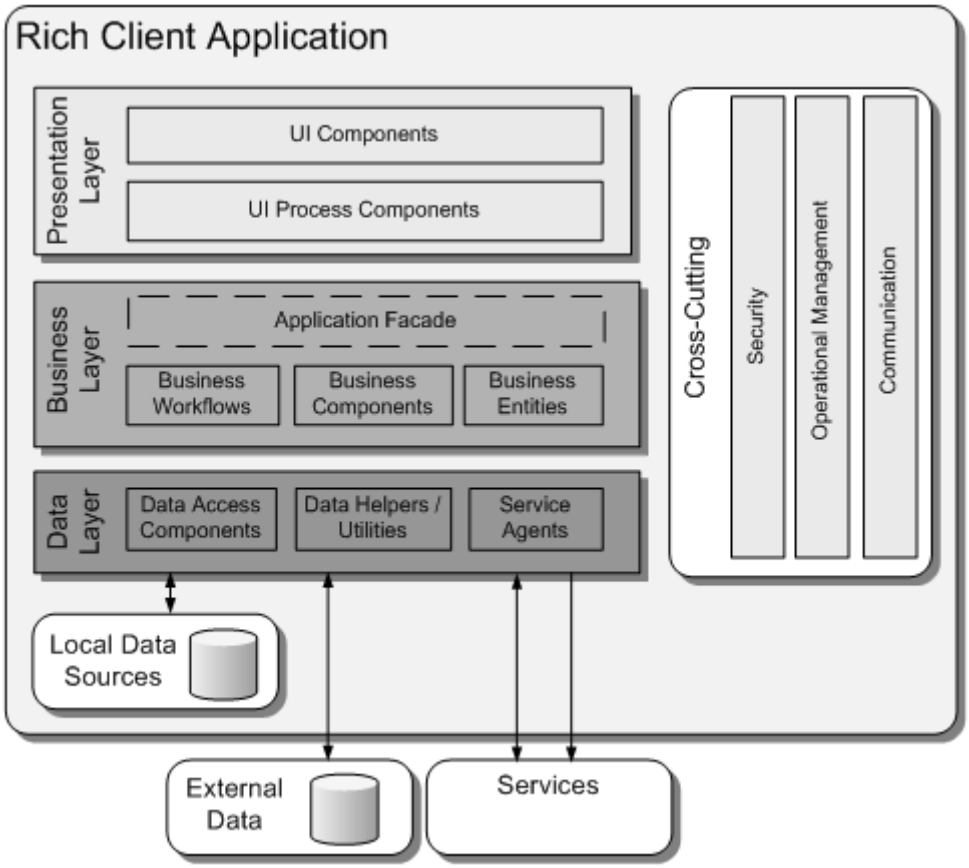
mechanizmus, ktorý dokáže znížiť latenciu (dobu čakania) a zvýšiť rýchlosť odpovede.

- *Lepšia spätná väzba:* Vďaka schopnosti meniť časti stránky bez reloadov môžu RIA používateľovi poskytovať rýchlejšiu a presnejšiu spätnú väzbu, real-time potvrdenie akcií a volieb a taktiež detailné informačné a chybové správy.
- *Súlad vzhľadu a prostredia:* Vďaka RIA je možné efektívnejšie ovládať používateľské rozhranie v rôznych prehliadačoch a operačných systémoch.
- *Offline prevádzka:* Pokiaľ dôjde k výpadku pripojenia internetu, môže byť stále možné využívať aplikácie RIA, pokiaľ sú navrhnuté tak, aby ukladali svoj stav u klienta. Vývoj webových štandardov to umožňuje tiež len pre niektoré tradičné webové aplikácie.
- *Vplyv na výkon:* V závislosti na aplikácii a sieťovom pripojení môžu byť RIA aplikácie výkonnejšie než klasické. Teda pokiaľ sa aplikácia dokáže vyhnúť neustálemu posielaniu požiadaviek na server tým, že ich bude spracovávať na strane klienta, bude znateľne rýchlejší. Spracovaním na strane klienta taktiež zmenší zaťaženie servera. Problémy môžu nastať v mobilných zariadenia, ktoré nemusia disponovať dostatočným výkonom alebo podporou RIA aplikácií.

RIA sú teda webové aplikácie, ktoré majú množstvo charakteristík desktopového aplikačného softvéru, typicky dodávaného formou webového prehliadača s použitím JavaScriptu, ktorý je jeho súčasťou alebo s využitím ďalších rozšírení prehliadača napr. Adobe Flash, JavaFX a Microsoft Silverlight, pričom ide v súčasnosti o tri najbežnejšie platformy. Google trendy však poukazujú, že rozšírenia prehliadačov na báze frameworku sú postupne nahrádzané novou generáciou HTML5 v kombinácii s kaskádovými štýlmi CSS3, JavaScriptom a technológiou AJAX[8]. Webové technológie boli vyvinuté a zosúladené webovými prehliadačmi za účasti organizácie W3C[9] *(http://ltonconfensder1981.blog.cz, 27.03.2013).*

[8] AJAX - Asynchronous JavaScript + XML,
[9] W3C - World Wide Web Consortium,

Obrázok 4 Celkový pohľad na typické bohaté aplikácie klientskej architektúry



Zdroj: http://msdn.microsoft.com

1.4 Vývoj webových aplikácií

Vývoj webových aplikácií je jedným z najrýchlejšie napredujúcich odvetí IT. Medzi vývoj webových aplikácií patrí webový dizajn, tvorba webového obsahu, skriptovanie na klientskej a serverovej strane, webový server, konfigurácia sieťovej bezpečnosti a vývoj e-commerce aplikácií (internetové obchody). Vývoj webových aplikácií môže byť v rozsahu od rozvojových najjednoduchších statických stránok z prostého textu až po najzložitejšie on-line internetové aplikácie, elektronické obchody alebo služby sociálnych sietí. Vývoj sa dostal do novej fázy internetovej komunikácie. Počítačové webové stránky už nie sú len nástroje pre prácu alebo obchod, ale používajú sa hlavne pre komunikáciu. Táto nová forma webovej komunikácie sa taktiež mení na e-commerce prostredie pomocou vysokého počtu zobrazení stránok, online reklamy a internetového marketingu. Vývoj webových aplikácií môže byť rozdelený do dvoch základných oblastí: na strane klienta (client-side) ide napríklad o technológie AJAX, JavaScript, Microsoft Silverlight, Adobe

Flash a ďalšie alebo na strane servera (server-side) a to sú napríklad ASP.NET, C#.NET, PHP, Java, Perl a pod. Internet sa stal hlavnou platformou pre vývoj webových aplikácií, najrôznejších komplexných a zložitých podnikových systémov v niekoľkých oblastiach. Popri ich vlastnej bohatej funkčnosti predstavujú tieto webové aplikácie komplexné riešenia a kladú špecifické požiadavky na ich využiteľnosť, výkon, bezpečnosť, schopnosť rásť a vyvíjať sa *(Flash Development for Android Cookbook, 25 s.).*

1.5 Adobe Flash

Flash bol po prvýkrát vydaný v roku 1996 pod názvom *Future Splash Animation,* vytvorili ho spoluzakladatelia spoločnosti FutureWave a to Jon Gay a Charlie Jackson. Bol to nástroj na tvorbu webovej animácie založený na vektorovej báze. Len o 7 mesiacov neskôr Macromedia FutureWave kúpila a FutureSplash bol premenovaný jednoducho na Flash[10]. V tých rokoch bolo jediným spôsobom ako rozšíriť webový prehliadač o prehrávanie animácie prostredníctvom Javy. Bol naprogramovaný jednoduchý animačný prehrávač, ktorý používal Javu ale bol nadpriemerne pomalý. Na jeseň Netscape uviedli ich plug-in API[11]. V tej dobe sa vyskytla príležitosť na rozšírenie webového prehliadača so slušným výkonom (to bol predchodca Macromedia Flash Player-u). Z jednoduchého webového kreslenia a animácie bolo vyvinuté kompletné multimediálne vývojové prostredie s 500-tisíc vývojármi a Flash využíva viac než 325 miliónov internetových používateľov. Dnes ho poznáme pod názvom Adobe Flash. Medzi technológie, ktoré využívajú Flash patrí Flex, Flash Builder, Flash Catalyst a iné *(http://www.adobe.com/, 2013).*

1.5.1 Adobe Flash Player

Adobe Flash Player je štandardom pre poskytovanie vysoko účinného a bohatého webový obsah. Dizajny, animácie a používateľské rozhrania aplikácií sú nasadené okamžite vo všetkých prehliadačoch a platformách aby bolo prilákaných a zapojených čo najviac používateľov s bohatým webovým zážitkom. Nižšie uvedená tabuľka 1 obsahuje najnovšie verzie Flash Player-a *(Flash Development for Android Cookbook,27 s.).*

[10] Flash – Grafický vektorový program,
[11] API – Application programming interface,

Tabuľka 1 Najnovšie verzie Flash Player na rôznych platformách

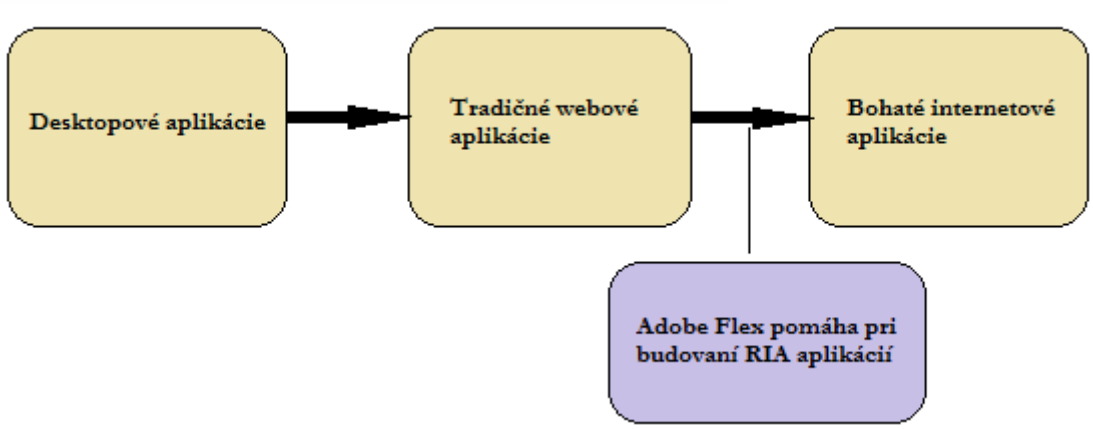
Platformy	Prehliadač	Player verzie
Windows	Internet Explorer (a ďalšie prehliadače, ktoré podporujú Internet Explorer ActiveX ovládacie prvky a plug-ins)	11.6.602.171
Macintosh	Firefox, Opera, Safari	11.6.602.171
OS X	Chrome (Pepper-based Flash Player)	11.6.602.171
Linux	Mozilla, Firefox, SeaMonkey (Flash Player 11.2 je posledná podporovaná verzia Flash Player pre Linux. Adobe bude aj naďalej poskytovať bezpečnostné aktualizácie.)	11.2.202.273
Solaris	Flash Player 11.2.202.223 je posledná podporovaná verzia Flash Player-u pre Solaris.	11.2.202.223

Zdroj: Vlastné spracovanie

1.5.2 Adobe Flex

Flex je vysoko produktívny open source aplikačný framework pre budovanie a udržiavanie webových aplikácií, ktoré sa dajú nasadiť do všetkých prehliadačov, desktopov a iných zariadení. V podstate je to softvér vývojového nástroja (SDK) povolený Adobe Systémami, ktorý umožňuje webovým vývojárom rýchlo a ľahko vytvárať bohaté internetové aplikácie (RIA) na Flash ® Platformách. Adobe Flex sa používa len na strane klienta (client-side) pre výmenu údajov zosynchronizovaných s ľubovoľnou serverovou technológiou podporujúcou http (*Adobe® Flex® 3.0 For Dummies®,2013*).

Obrázok 5 Vývoj aplikácií, ktoré viedli k vývoju bohatých internetových aplikácií frameworku



Zdroj: Vlastné spracovanie

1.5.3 Klady a zápory Flex aplikácií

V súčasnej dobe majú weboví vývojári širokú škálu možností pre rozvoj RIA aplikácií. Jedným z dôvodov je skutočnosť, že Adobe Flex oddeľuje prezentáciu a prístup k dátam vrstvy. Môže použiť akúkoľvek serverovú technológiu, ktorá spĺňa požiadavky - ColdFusion[12], J2EE[13], PHP[14], .NET[15] a pod., pretože Adobe Flex je navrhnutý tak, aby mohol fungovať v XML[16] (aj MXML[17] dokumente), čítať a zapisovať dáta do dátového úložiska. Ďalším hlavným dôvodom je, že Adobe Flex je ideálna technológia ak chceme vytvoriť vysoko interaktívne, expresívne aplikácie webového servera pomocou vizualizácie dát. Adobe Flex umožňuje aby sa z obvyklého vytvárania statickej webovej stránky dali vložiť mini aplikácie bez toho, aby neboli zložité problémy s kompabilitou alternatív ako sú Java applety[18]. Nižšie uvediem niektoré ďalšie dôvody prečo pri tvorbe webových stránok používať Adobe Flex (*Adobe® Flex® 3.0 For Dummies®,2013*):

- pomáha vytvárať robustné aplikácie, ktoré atraktívne zobrazujú komplexné dátové sady,
- je vizuálne zapojený pre návštevníkov webovej stránky,
- pôsobí na všetkých platformách a používateľ si k nim už nemusí inštalovať ďalšie ako napríklad Flash Player,
- audio a video v ňom majú ešte väčšiu interakciu,
- synchronizácia dát umožňuje real – time a dáta môžu byť použité efektívnejšie

1.5.4 Verzie a edície Flexu

Adobe Flex začal s verziou 1.0, postupne sa zlepšoval na 1.5, 2.0, 3.0. Nižšie uvediem komponenty, ktoré sa v súčasnej dobe tvoria Adobe Flex 3.0:

- Adobe Flex 3 SDK – Môžeme vytvárať a nasadzovať Flex aplikácie jedine pomocou Open Source Flex 3 SDK, ktorý obsahuje väčšinou Flex SDK[19] (prekladač, framework, debugger), ale nezahŕňa žiadnu platformu ako Adobe Flash

[12]ColdFusion – Aplikačný server umožňujúci vývojárom rýchlejšie vytvárať, nasadiť a udržiavať Java ™-EE aplikácie pre podnik,
[13] Platforma Java 2, Enterprise Edition,
[14] Hypertext Preprocessor,
[15] NET Framework, prostredie ppotrebné pre beh aplikácií,
[16]XML - eXtensible Markup Language,
[17] MXML – Minimal eXtensible Markup Language,
[18] Applet –je (relatívne) jednoduchá aplikácia, ktorá sa spúšťa z iného programu napr. webového prehliadača,
[19] SDK – Software development kit,

Player, Adobe AIR, alebo ostatné pokročilé knižnice ktoré zahŕňajú kódovanie druhu písma.

- Adobe Flex Builder 3 – Urýchluje vývoj Flex aplikácií, pretože umožňuje inteligentné kódovanie, interaktívne krokovanie v debuggeri a vizuálny dizajn prvkov.

1.5.5 Adobe AIR

Adobe AIR zdieľ

a veľa základných funkcií s Adobe Flash Playerom. Počas vývojového cyklu medzi Flash Playerom 10 a Flash Playerom 10.1, vývojári v Adobe prepísali veľa podkladového kódu, aby položili pevný základ, ktorý nielen ťaží na tradičných webových zážitkov ale môže byť prenesený do mobilných platforiem a televízií. Niet pochýb o tom, že mobilný vývoj aplikácií pomocou Adobe Flash Platforiem sa stala témou zvýšeného záujmu v komunitách Application Developer *(https://get.adobe.com/air).*

1.6 Microsoft Silverlight

Microsoft Silverlight je moderná technológia pre tvorbu dynamického obsahu webových stránok, ktorá umožňuje v rámci kontextu kombinovať klasické textové prvky, vektorovú a rastrovú grafiku, animácie a video. Používa sa pri tvorbe aplikácií, animácií a obrázkov. Podporuje určité typy formátov pre obrázky ako .jpg a .png. Je založený na vektorovej grafike ale ešte nepodporuje 3D a ani vkladanie zvuku do animácií. Podporuje ale .wma, .wmv formáty. Výhodou Silverlightu je výber skriptového jazyka, pričom pri Flash je pevne stanovený ActionScript. Charakteristické je spúšťanie aplikácie v samostatnom okne a nie v prehliadači. Vývoj sústreďuje len pod Windows ale je kompatibilný aj s operačnými systémami MAC OS X a prostredníctvom Moonlightu. Pri uvádzaní novej technológie postupovala firma Microsoft dosť netradične. S finálnou verziou Silverligt 1.0 predstavila aj alfa verziu 1.1. Silverlight 1.0 využíva ako programovací jazyk JavaScript rovnako ako Flash a vo verzii 1.1 je možné v plnej miere využívať .NET jazyky.

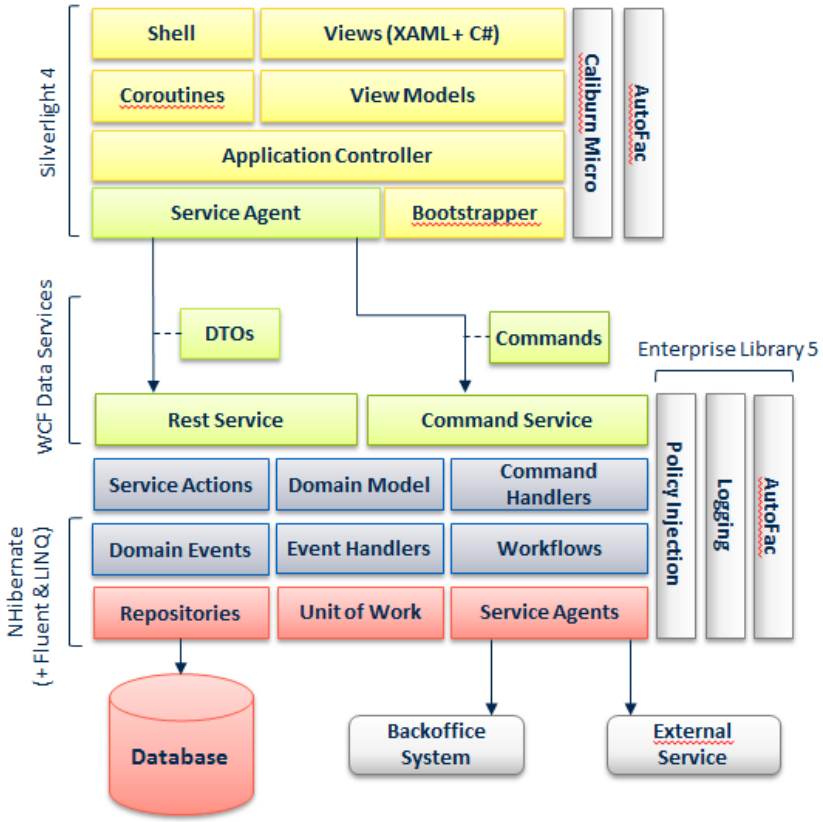
Microsoft Silverlight slúži k interpretácii interaktívnych prezentácií vrátane multimediálnych na klientských počítačoch v prostredí webových prehliadačov na rôznych platformách. V procese vývoja mala táto technológia označenie Windows Presentation

Foundation/Everywhere. Zámer Microsoftu značí „Preniesť čo najviac čŕt nového prezentačného rozhrania WPF (Windows Presentation Foundation), ktoré je súčasťou .NET Framework 3.0“. .NET Framework 3.0 je štandardne predinštalovaný v operačnom systéme Windows Vista. Jedným zo základných pilierov Silverlightu je značkovací jazyk XAML, založený na formáte XML. XAML umožňuje popisovať textový obsah, grafický bitmapový, vektorový a multimediálny obsah. Jazyk XAML slúži pre definovanie prezentačného rozhrania aplikácie. Interaktivita a aplikačná logika, znamená obsluhu udalostí (zavedenie stránky, zatlačenie tlačidla alebo iného ovládacieho prvku, napísanie znaku, pohyb kurzora myši ponad nejaký objekt a iné) je vo verzii Silverlight 1.0 zaistená programovým kódom v jazyku JavaScript. Vo verzii 1.1 je možné aplikačnú logiku programovať už v .NET jazykoch.

V súčasnosti je najnovší Silverlight 4. Jeho predchodca Silverlight 3 je v súčasnosti nainštalovaný na viac než 45% zariadení rôznych platforiem pripojených k internetu, pričom mesačný nárast je 2-3%. Nová beta verzia technológie Silverlight 4 prišla len 4 mesiace po komerčnej dostupnosti verzie 3. Z najvýznamnejších noviniek pre podporu webovej kamery a mikrofónu je multicast streaming, ktorý umožňuje zrýchlené a spomalené prehrávanie videa. Niektoré novinky ako napríklad výstup priamo na tlačiareň, zabudovaný Rich text editor s podporou schránky, podpora pravého tlačidla myši, drag and drop a vloženie HTML stránky sú určené hlavne pre tvorbu takzvaných biznis aplikácií. Symbióza týchto čŕt umožňuje napríklad kópiu zo Silverlight DataGrid tabuľky do Excelu. Nové možnosti poskytuje aj zdieľanie dll knižníc medzi platformami Silverlight a .NET Framework 4.

1.6.1 Architektúra

Obrázok 6 Architektúra Microsoft Silverlight



Zdroj: http://download.codeplex.com/

Platforma Silverlight sa ako celok skladá z dvoch hlavných častí, ktoré doplňujú komponenty pre inštaláciu a aktualizáciu a to:

- *Core presentation framework*, ktorý sa skladá z komponentov a služieb orientovaných na užívateľské rozhranie a interakciu, ktorá zahŕňa užívateľský vstup, prehrávanie médií, správu digitálnych práv (Digital Rights Management – DRM), viazanie dát (data binding) a funkcie pre prezentáciu obsahu ako vektorovú grafiku, text, animácie a obrázky. Taktiež obsahuje jazyk XAML pre špecifikáciu rozloženia obsahu.
- *.NET Framework for Silverlight,* je podmnožinou „veľkého“ *.NET Frameworku,* ktorá obsahuje komponenty a knižnice zahrňujúce integráciu dát, rozšíriteľné ovládacie prvky Windows, prácu so sieťou, knižnice základných tried, *garbage collection* a *common langure runtime* (CLR). Aplikácie môžu vyžadovať ďalšie knižnice, ktoré nie sú v *Framework for Silverlight* obsiahnuté, ale zároveň sa nachádzajú v *Silverlight SDK*. V tomto prípade sú tieto knižnice k aplikácii pribalené a

stiahnuté do prehliadača. Ako príklad môžeme uviesť nové ovládacie prvky užívateľského rozhrania, XLINQ, Syndication (RSS/Atom), serializáciu XML alebo *dynamic language runtime* (DLR).

Ovládacie prvky pre inštaláciu a aktualizáciu majú za úlohu zjednodušiť proces inštalácie pre užívateľa, ktorý aplikáciu spúšťa prvýkrát a následne mu umožniť jednoduché aktualizácie.

1.6.2 Vývojové prostredie pre Microsoft Silverlight

Pre vývoj prezentačného rozhrania založeného na technológii Silverlight slúžia vývojové nástroje z Microsoft Expression. Môžeme využiť komplexný nástroj Microsoft Expression Studio alebo samostatné produkty Microsoft Expression Web pre webové aplikácie, Microsoft Expression Blend na vývoj prezentačnej vrstvy, Microsoft Expression Design na návrh vektorovej a bitmapovej grafiky alebo Microsoft Expression Media k správe multimédií.

1.7 Ostatné aplikácie

Enterprise Content Management (ECM) zahŕňa všetky funkcie pre efektívne riadenie a kontrolu celej spoločnosti vedomosťami. Toto poznanie je v mysliach zamestnancov a najrôznejších obchodných dát. ELO ECM riešenia sú navrhnuté tak, aby zostavovali poradie relevantného obsahu, podnikových procesov a ľudí najmä z hľadiska efektívneho spracovania. „ELOenterprise je sada škálovateľných Java serverov a klientských komponentov pre prostredie Java alebo .NET serverov. Len časť z nich sa dá prevádzkovať na ľubovoľnej platforme , kde je možné púšťať Java aplikácie. Jednotlivé serverové služby ELOenterprise medzi sebou komunikujú len protokolom http a je možné ich rozprestrieť v celkom heterogénnom prostredí, kde je jediná podmienka vzájomná viditeľnosť cez http protokol. Rovnakým spôsobom komunikujú so serverovými službami aj klientské komponenty vrátane pevného klienta (pevný klient je dostupný len pre prostredie MS Windows) *(http://www.eloweb.eu, 2013).*

2 OPIS TVORBY APLIKÁCIÍ TYPU RIA V PROSTREDÍ ADOBE FLASH S VYUŽITÍM JAZYKA ACTIONSCRIPT

Bohaté internetové aplikácie (RIA) nám ponúkajú bohatý vizuálny zážitku, ktorý zlepšuje spokojnosť užívateľov a tým sa zvyšuje aj jeho produktivita. Použitím širokého dosahu na Internet, RIA aplikácie môžu byť nasadené v celej škále prehliadačov. To znamená že môžu byť použité ako internetové alebo desktopové aplikácie. Hlavnými výhodami RIA aplikácií je ich ponuka v osvedčených organizáciách nákladovo- efektívnou cestou k moderným aplikáciám s obchodnými výhodami:

- Ponúknuť užívateľovi bohatší a lepší vizuálny zážitok,
- Udržiavajú tempo s používateľmi a ich rastúcimi požiadavkami,
- Zvyšovanie vernosti u zákazníkov a tým aj zvyšovanie zisku,
- Vplyv existujúceho zamestnanca, procesov a infraštruktúry

2.1 Stavba RIA aplikácií

Rich Internet Applications (RIA) sa dajú budovať pomocou HTML5, CSS3, Javascript a KnockoutJS. S použitím platformy Adobe Flash sa dajú budovať v Adobe Creative Suite, Adobe ColdFusion, Adobe LiveCycle ES, Adobe Scene7 a v prostredí Flash Builder.

2.1.1 Adobe Creative Suite

Creative Cloud je teraz integrovaný s Behance[20], čo tentokrát spojilo vývojárov s tvorivou komunitou ľudí, ktorí sa venujú vývoju v Creative Cloud. Inšpirácia sa hľadá rýchlo. V ProSite je zahrnuté Creative Cloude platené členstvo, je teda ľahšie aktualizovať svoje portfólio a spájať sa s novými ľuďmi s podobnou tvorbou. Je tu veľa príležitostí na zdokonaľovanie svojej tvorby. S Behance sa dá predviesť vlastná tvorba, dá sa získať spätná väzba a celková expozícia pre naše portfólio.

[20] http://www.behance.net

2.1.2 Adobe Scene7

Adobe Scene7 sa využíva napríklad v internetových obchodoch kde umožňuje vidieť tovar v 3D obraze. Väčšinou sa využíva na tvorbu reklamy. Je to hostované riešenie pre správu, posilnenie, publikovanie, isté dynamické marketingové prostriedky a bohatá vizuálna technika obchodovania cez web, mobilné telefóny, email, obrazovky pripojené na internet a tlač.

2.1.3 Adobe LiveCycle ES

Adobe LiveCycle Enterprise Server je platforma, ktorá umožňuje organizáciám automatizovať.

2.2 Prostredie Flash CS5, Flash Builder 4 a Flex

Veľa užívateľov prostredia Flash sú oboznámení o prostriedkoch Adobe Flash Builder a Flex, ale nevedia, ako ich využiť vo svojom vývojovom procese. Prostredie Flash CS5 a Flash Builder sú komerčné aplikácie od spoločnosti Adobe. Flash Builder 4 je nový názov prostredia, ktoré predtým bolo známe ako Flex Builder. V prostredí Flash CS5 alebo Flash Builder 4 môžeme vytvárať súbory SWF pre prehrávač Flash Player a aj samostatné aplikácie platformy Adobe AIR. Ďalšia možnosť pre skúsených programátorov je použiť bezplatnú sadu Flex SDK, ktorá je k dispozícii na stránkach Adobe.

Všetky uvedené programy podporujú celý jazyk ActionScript 3.0. Prostredie Flash Builder sa viac zameriava na pokročilejších programátorov a obsahuje niekoľko funkcií, ktoré sa dajú použiť pri vývoji rozsiahlych RIA aplikácií a projektov riadených dátami. Na druhej strane – prostredie Flash CS5 obsahuje nástroje a rozhrania prispôsobené potrebám návrhárov a tvorcov animácií.

Ak pracujeme na projektoch, ktoré majú okrem veľkého počtu dizajnu, videa, animácií a iného obsahu taktiež interaktívny obsah, ktorý vyžaduje nemalé množstvo zdrojového kódu, mali by sme si zvážiť, či nebudeme vyvíjať svoje projekty v prostredí Flash CS5 aj v prostredí Flash Builder 4. Obe tieto nástroje ponúka balík Adobe CS5 Web Collection a sú veľmi dobre prepojené. Mnoho vývojárov a vývojových tímov bude vytvárať vizuálnu časť aplikácie v prostredí Flash CS5 a potom z neho spustí prostredie Flash Builder 4, v ktorom napíšu svoj zdrojový kód v jazyku ActionScript. To je samozrejme voliteľný krok, pretože zdrojový kód sa dá celý napísať vo Flash CS5.

2.2.1 Flash Professional CS5

Flash Professional CS5 bol vydaný z dielní Adobe 12. apríla 2010 a zahájenie jeho preskúšania a následný predaj 30. apríla roku 2010. Flash Professional CS5 obsahuje podporu publikovania aplikácií pre iPhone. A však 8.4.2010 v Apple zmenili podmienky Licencie určenej na vývoj a účinne zakázali použitie Flashu do iPhone kompilátora. Dňa 20.apríla 2010 firma Adobe oznámila, že už ďalej nebudú robiť žiadne investície spojené s vývojom pre iPhone a iPad v prostredí Flash CS5.

Flash Professional CS5 sa od starších verzií odlišuje vlastnosťami, ktorými sú napríklad nový textový modul (TLF), a ďalšie zlepšovania na inverznej kinematike a už dopredu sú vytvorené fragmenty kódu (Code Snippets). Zdroj:

2.2.2 Flash Builder 4

Softvér Flash Builder 4 je prostredie pre tvorbu pútavých internetových aplikácií s bohatým obsahom pre rôzne platformy. Je určení pre vývojárov, ktorí potrebujú rýchlo vyvíjať bohaté internetové aplikácie (RIA) a obsah pre rôzne platformy, vrátane mobilných s iOS, Androidom a RIM s využitím open source platformy Flex. Obsahuje podporu inteligentného kódovania, odladťovania a vizuálneho navrhovania. Obsahuje aj výkonné nástroje pre testovanie, urýchlený vývoj a zvyšujúci výkon výsledných aplikácií.

2.3 História platformy Flash a jazyka ActionScript

Platforma Flash a jazyk ActionScript sa vyvíjali spoločne, pretože platforma Flash vznikla v roku 1996. V súčasnosti predstavuje kombinácie návrhových a animačných nástrojov prostredia Flash CS5 a pokročilých schopností jazyka ActionScript 3.0 jedno z najmocnejších, najuniverzálnejších a najobľúbenejších vývojových prostredí. Jazyk ActionScript však začínal ako súčasť platformy Flash pomerne skromne.

V prvých troch verziách platformy Flash nebol k dispozícii žiadny programovací nástroj a interaktivita znamenala preťaženie niektorej z niekoľko jednoduchých možností na Akciách (Actions). Tieto akcie umožňovali prechádzať oblasťou Časová os a vytvárať odkazy na URL, ale nič viac.

Platforma Flash 4 bola prvá verzia, ktorá umožňovala vkladať zdrojový kód pomocou jednoduchého skriptovacieho jazyka, vtedy sa začal neformálne nazývať

ActionScript. Pri platforme Flash 5 sa jazyk ActionScript viac vyvinula stal sa z neho oficiálny skriptovací jazyk. S každou ďalšou verziou platformy Flash schopnosti jazyka ActionScript rástli a tento jazyk začal ponúkať interaktívne riadenú animáciu, text, zvuk, video, dát a podobne. V roku 2003 vznikol jazyk ActionScript 2.0 a jeho zručnosti boli porovnateľné s objektovo orientovanými programovacími jazykmi ako sú jazyky Java alebo C#.

Skúsený programátori sa začali viac zaujímať o jazyk ActionScript ako o vývojový nástroj, ale zistili, že aj keď tento jazyk dostačuje ostatným programovacím jazykom v ponuke funkcií, ale nie je efektívny. Bolo to preto, že každá verzia ActionScriptu bola založená na svojej predchádzajúcej verzii, a to až ku svojim jednoduchým koreňom. Prehrávač Flash Player nebol pôvodne navrhnutý pre vytváranie aplikácií s veľkými požiadavkami na výkon hier ale vývojári ich začali používať práve na tieto účely. Bolo skoro úplne jasné, že musí vzniknúť nová verzia jazyka ActionScript, ktorá sa prepíše od základov.

V roku 2006 predstavila spoločnosť Adobe jazyk ActionScript 3.0 s tým, že ponúkal nové funkcie a takisto dramatický nárast efektivity. Prostredie Flash CS3 ako prvé začlenilo túto verziu jazyka. Prostredie Flash CS4 pridalo k jazyku ActionScript ďalšie funkcie vrátane práce s 3D grafikou, nové riadenie animácií a tried jazyka ActionScript pre prácu s platformou Adobe AIR. Prostredie Flash CS5 pokračuje vo vývoji jazyka ActionScript 3.0 a prináša mnoho nových pokročilých funkcií, napríklad vylepšovanie platformy Adobe AIR, prácu z rôznymi zariadeniami na radiči a to vrátane viacdotykových a bežných dotykových zariadení. Prostredie Flash CS5 má taktiež niekoľko nových vlastností, ktoré pomáhajú naučiť sa pracovať s jazykom ActionScript, napríklad panel Fragmenty kódu (Code Snippets), ktorý má umožniť opätovne používať spoločný kód jazyka ActionScript jedným kliknutím myši alebo automatické dokončovanie zdrojového kódu.

2.4 ActionScript

Kód jazyka ActionScript sa bežne umiestňoval do kľúčových snímok v oblasti Časová os (Timeline). V starších verziách platformy Flash bolo možné kód jazyka ActionScript vkladať priamo na objekt napríklad na tlačidlo alebo filmový klip, ale to už v jazyku ActionScript 3.0 nie je možné.

Jazyk ActionScript sa často vyskytuje v prostredí so štandardným orientovaným typom programovania. Jazyk ActionScript 3.0 je založený na štandarde ECMA[21] a vo veľa ohľadoch sa podobá iným programovacím jazykom, napríklad jazykom Java, C# a C++. Jazyk ActionScript je objektovo orientovaný programovací jazyk, čo je vhodné na vytváranie veľkých a zložitých projektov. Alternatívou k umiestneniu zdrojového kódu do oblasti časová os je vytváranie samostatných súborov s kódom jazyka ActionScript v externých súboroch, ktoré je možné používať v ľubovoľnom projekte Flash. V jazyku ActionScript 3.0 je veľmi dôležité aby sme dali názov inštancie každému tlačídlu, filmovému klipu a ďalším objektom, ktoré chceme riadiť pomocou ActionScriptu.

2.4.1 Používanie fragmentov kódu a navigácia v panely Časová os

Panel Časová os (Flash Timeline) je užitočný nástroj pre vytváranie animácií. Takisto sa jedná o prostredie pre nastavovanie webových stránok alebo jednoduché aplikácie, ktoré vyžadujú prechádzanie medzi dvoma rôznymi časťami obsahu. Kód jazyka ActionScript môžeme vytvárať na ľubovoľnej kľúčovej snímke hlavnej časovej osy filmu Flash. Takisto ho môžeme vytvárať v akejkoľvek kľúčovej snímke v symbole filmového klipu. Behom prehrávania zostaveného projektu, Flash spustí kód každej snímky hneď ako sa daná snímka prehrá. Všetok zdrojový kód sa v prostredí Flash zapisuje do panelu Akcie (Actions), ktorý je dostupný z ponuky Okno (Window) alebo stlačením klávesy F9. Okrem toho, že kód môžeme zapísať priamo do panelu Akcie, môžeme ho sem pridať pomocou nového panelu Fragmenty kódu (Code Snippets). Fragmenty kódu sú už dopredu vytvorené časti zdrojového kódu jazyka ActionScript 3.0, ktoré môžeme ľahko pridať do svojich projektov a prispôsobiť svojim požiadavkam. Taktiež predstavujú skvelý spôsob ako začať pracovať s jazykom ActionScript a rozšíriť množstvo úloh, ktoré sme schopný dokončiť.

2.4.2 Metódy gotoAndStop() a gotoAndPlay()

Metódy *gotoAndStop()* a *gotoAndPlay()* sú asi najstaršie funkcie jazyka ActionScript a od verzie 2 prostredia Flash sa takmer vôbec nezmenili. S metódou *gotoAndPlay()* presúvame aktuálnu pozíciu z práve prehranej snímky ku snímke, ktorá sa nachádza v zátvorke a od nej pokračujeme v prehrávaní animácie. Keby sme sa chceli presunúť k určitej snímke a pozastaviť na nej prehrávanie časovej osy, mohli by sme použiť metódu

[21] Medzinárodný štandard na písanie kódu

gotoAndStop(). Tieto metódy nám umožňujú ľahšie meniť obsah časovej osy bez toho aby sme museli zasahovať do zdrojového kódu.

2.4.3 Tvorba premenných

Príklad: *var pocet:Number = 1;*

Kľúčovým slovom *var* oznamujeme jazyku ActionScript, že chceme vytvoriť novú premennú. Názvom tejto novej premennej je napríklad *pocet.* Premenným, ktoré tvoríme môžeme dať ľubovoľné názvy pokiaľ sa držíme nasledovnými pravidlami:

- Názov premennej nesmie obsahovať medzeru,
- Nesmieme používať špeciálne znaky, iba podčiarkovník. Môžeme teda používať iba písmená a čísla,
- Názov premennej nesmie začínať číslom.

Dvojbodka za názvom premennej označuje miesto, za ktorým nasleduje typ dát, ktoré do nej budeme ukladať. V tomto príklade ukladáme do premennej *pocet* číslo. Znamienko (=) označuje miesto, za ktorým nasleduje hodnota, ktorú do premennej ukladáme. Premenným nemusíme priraďovať hodnotu vo chvíli keď ju vytvárame. Často vytvárame premennú len s úmyslom, že do nej neskôr uložíme nejakú informáciu. V predloženom príklade priraďujem premennej *pocet* hodnotu *1*.

2.4.4 Podmienené príkazy

Podmienené príkazy sú hlavným zdrojom sily jazyka ActionScript. S podmienenými príkazmi sa stretneme vo väčšine programovacích jazykoch a fungujú rovnako ako v jazyku ActionScript3.0. Používajú sa tu príkazy *if, else if, else*.

2.4.5 Vytváranie kódu jazyka ActionScript v externých súboroch

Väčšinou sa zdrojový kód jazyka ActionScript vytvára výhradne v snímkach Časovej osy. Jedná sa o efektívny spôsob práce s mnoho vývojárov platformy Flash vytvára všetky svoje projekty iba s kódom na časovej osy. Pri jednoduchých projektoch má tento prístup tú výhodu, že uchovávame celú grafiku a prostriedky v rovnakom súbore ako kód skriptovacieho jazyka. Pri zložitejších projektoch Flash je však ťažkopádne mať stovky riadkov kódu v panely Časová os. Alternatívne riešenie spočíva v ukladaní zdrojového kódu jazyka ActionScript k projektu do jedného alebo viacerých súborov, ktoré obsahujú

iba zdrojový kód. Tieto súbory sa potom dajú spájať s grafikou, animáciou a ďalším obsahom pri tvorbe finálneho projektu. Externé súbory s kódom jazyka ActionScript sú iba súbory s prostým textom uložené pod príponou .as.

Súbory s kódom jazyka ActionScript je možné vytvárať v akomkoľvek programe, ak sa v ňom dajú vytvárať textové súbory, vrátane TextEdit v operačnom systéme Mac a v Poznámkového bloku v operačnom systéme Windows. Samozrejme je lepšie písať kód v programe, ktorý umožňuje zvýrazňovanie kódu alebo upozorňovanie na chyby. Nástroje spoločnosti Adobe, napríklad Flash Professional CS5, CS6, Flash Builder alebo Dreamweaver poskytujú úplnú podporu pre súbory s kódom jazyka ActionScript.

Ak vytvárame filmový klip v prostredí Flash, vytvárame inštanciu triedy *MovieClip*. Ak pracujeme s textovým polom, používame inštanciu triedy *TextField*, pri vytváraní videa je to trieda *Video* a podobne.

2.4.6 Modifikátory prístupu

V ActionScripte sa používajú štyri modifikátory prístupu:

- Public: Nastavením modifikátora *public*, túto triedu špecifikujeme, že je k nej možné pristupovať z ľubovoľného iného súboru,
- Private: Metódy a vlastnosti označené modifikátorom *private* sú k dispozícii iba vnútri súboru danej triedy,
- Protected: Metódy a vlastnosti označené modifikátorom *protected* sú dostupné zo súboru danej triedy a všetkých jej potomkov,
- Internal: Metódy a vlastnosti označené modifikátorom *internal* sú k dispozícii vnútri súboru danej triedy a vo všetkých ostatných súboroch rovnakého balíka (ActionScript 3.0 Oficiálny výukový kurz, str.82).

2.4.7 Parametre

V jazyku ActionScript sú povinné a voliteľné parametre. Ak má funkcia pri parametroch predvolené hodnoty, tak pri ich volaní nemusíme pridávať hodnoty týchto parametrov ako argumenty. Takéto parametre sa nazývajú voliteľné parametre. Ak aj tak pridáme hodnoty voliteľným parametrom, tieto hodnoty prepíšu príslušné predvolené hodnoty. Ak má funkcia parametre bez zadaných predvolených hodnôt, potom musíme tieto hodnoty pridať ako argumenty pri ich volaní. Takéto parametre sa nazývajú povinné.

2.4.8 Udalosti

Pri začiatku kódu píšeme addEventListener, čo znamená pridaj poslucháča udalosti. Za tento príkaz sa píše zátvorka, v ktorej sa nachádza udalosť, napríklad (MouseEvent.CLICK, a sem napíšeme čo po kliknutí chceme aby sa stalo s našou animáciou), ktorú ideme vytvoriť. Potom musíme napísať funkciu (function) a zadať čo má kód po kliknutí myši vykonať.

V ActionScripte používame udalosti myši (MouseEvent) a klávesnice (KeyboardEvent) alebo len udalosť (Event). Udalosti sa pomenúvajú tak, že prvé slovo, ktoré píšeme je, s čím chceme udalosť vykonať: myš alebo klávesnica. Kód v jazyku ActionScript sa píše v anglickom jazyku a ostatné ako názvy funkcií si môžeme napísať v jazyku akom len chceme. Takže udalosti myši a klávesnice sa začínajú Mouse alebo Keyboard, ďalšie čo do kódu píšeme je Event. Napríklad MouseEvent(), KeyboardEvent(), alebo len Event()(*ActionScript oficiálny výukový kurz, str.168*).

3 VYTVORENIE PROGRAMU V JAZYKU ACTIONSCRIPT

V prostredí Flash CS5 môžeme vytvárať súbory SWF pre prehrávač Flash Player aj samostatné aplikácie platformy Adobe AIR. Adobe Flash Profesional podporuje celý jazyk AtionScript 3.0. Prostredie Flash CS5 obsahuje nástroje a rozhrania prispôsobené potrebám návrhárov a tvorcov animácií.

Vytvorila som animáciu v prostredí Adobe Flash Professional CS5 s pomocou jazyka Actionscript 3.0. Adobe Flash Professional CS5. Aplikácia Adobe Flash Professional CS5 je široko rozšírená technológia používaná predovšetkým k tvorbe pútavých projektov zahrňujúcich video, zvuk, grafiku či animáciu.

3.1 Postup tvorby aplikácie v jazyku ActionScript v prostredí Adobe Flash Proffesional CS5

Pri prvom spustení aplikácie Flash uvidíme uvítaciu obrazovku s odkazmi na štandardné šablóny súborov, výukové videá a ďalšie zdroje (viď Obrázok 8). V mojom prípade si vyberiem ActionScript 3.0. Prostredie Adobe vytvorí nový súbor Flash (*.fla).

Cez Súbor (File) si vyberiem príkaz Nový (New) otvorí sa mi dialógové okno Nový dokument (New Document). Animáciu si uložím do priečinku kde si postupne budem ukladať celú animáciu. V priečinku sa mi automaticky vytvorí súbor s názvom Flash Document a spolu s týmto súborom sa vytvorí súbor SWF Movie. Ako prvé sa pri otvorení Flash Professional CS5 otvorí ponuka (viď Obrázok 8). Súbory, ktoré sa mi neskôr ukladajú do priečinka majú príponu .fla a .swf. V záložke Všeobecné (General) si vyberiem možnosť ActionScript 3.0 (jazyk, ktorý animáciám dodáva interaktivitu). Vybraním verzie ActionScript 3.0 vytvorím nový dokument, ktorý bude nakonfigurovaný k prehrávaniu v prehliadači pomocou prehrávača Flash Player. Ďalšie možnosti v tejto záložke potom slúžia k prehrávaniu v alternatívnom prostredí. Napríklad Adobe AIR 2 alebo iPhone OS vytvárajú nové dokumenty nakonfigurované k prehrávaniu v aplikácii AIR na Androide alebo mobilnom zariadení značky Apple.

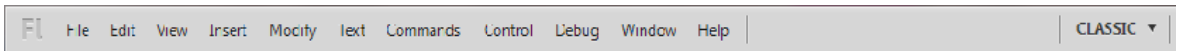
Obrázok 7 Úvodná ponuka

Zdroj: Vlastné spracovanie

3.2 Zoznámenie s pracovnou plochou

V hornej časti pracovnej plochy aplikácie Adobe Flash Professional sa nachádzajú príkazové ponuky a pestrá škála nástrojov a panelov slúžiacim k úprave a pridávaniu prvkov do filmu. Všetky objekty, ktoré budeme ku svojej animácii potrebovať. V pravom hornom rohu si nastavím mód prostredia na Classic. Od tohto nastavenia závisí čo budem mať na obrazovke v prostredí Flash (viď Obrázok 8). Okrem Classic mám v ponuke Animátor, Debug, Dizajnér, Developer, Essentials a Small Screen.

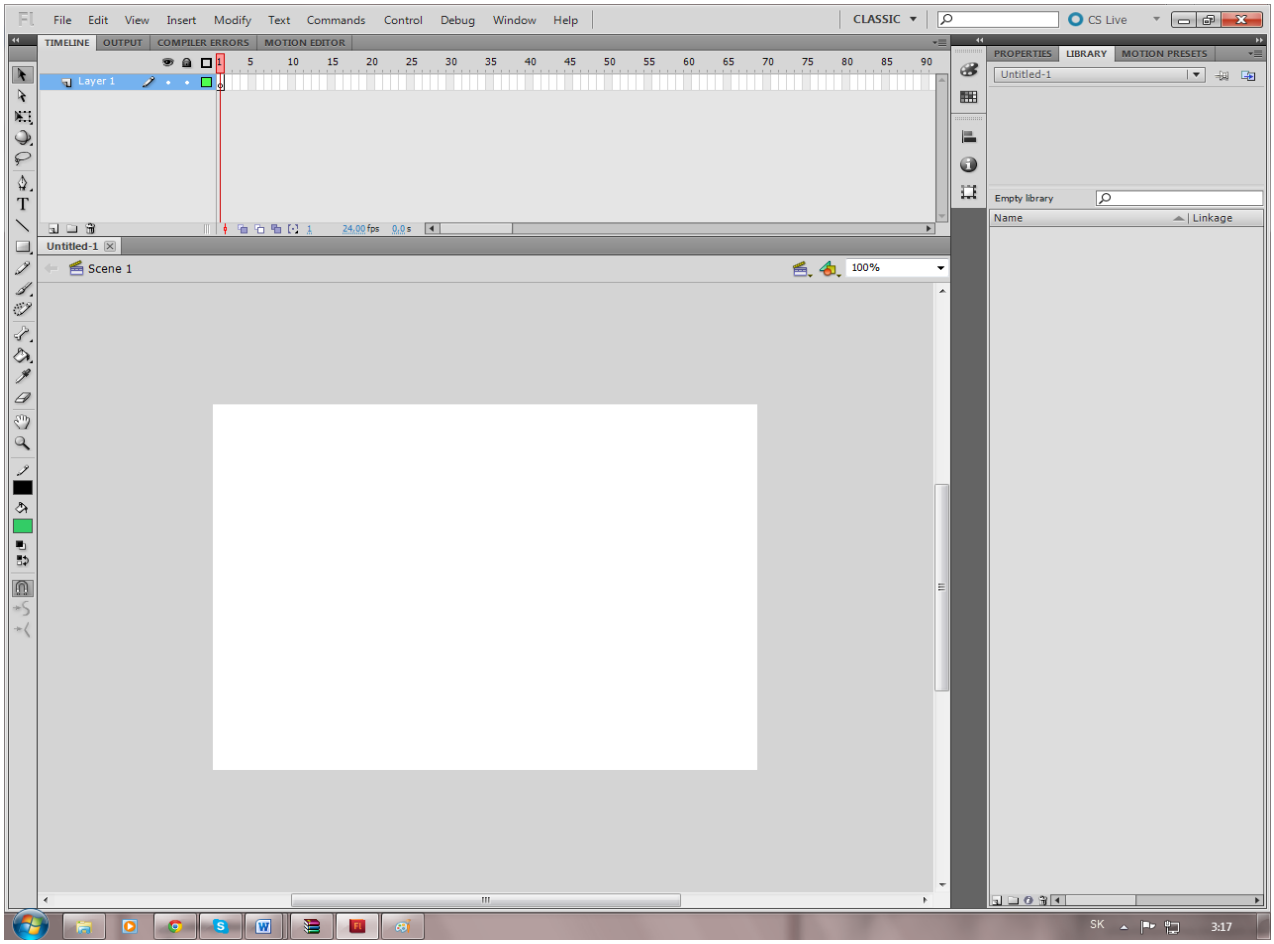
Obrázok 8 Hlavný panel



Zdroj: Vlastné spracovanie

V nastavení Classic sa zobrazuje riadok ponúk, časová os, vymedzená plocha, Panel Nástroje (Tools), Vlastnosti (Properties), Knižnica (Library), Motion Editor. Pri práci vo Flash-i si môžem otvárať, zatvárať, ukotvovať, uvoľňovať a presúvať panely po obrazovke tak, aby mi vyhovovali rozlíšeniu môjho monitora.

Obrázok 9 Pracovná plocha



Zdroj: Vlastné spracovanie

3.3 Tvorba animácie

Začínam s úpravou pozadia. V nástrojoch, ktoré sa na mojej obrazovke nachádzajú na ľavom boku v ponuke nástrojov si vyberiem Obdĺžnikový nástroj (Rectangle Tool), vyberiem celý biely priestor, ktorý mám prednastavený programom a zmením farbu na čiernu vyfarbením s použitím nástroja Plechovka farby (Paint Bucket Tool), ako pomôcka mi stačí stlačiť písmeno (K).

V jazyku ActionScript 3.0 môžeme popisovať farby najrôznejšími spôsobmi, ale najpoužívanejší je použitie 16-kovej hodnoty. Otvorím si položku Okno a vyberiem si ponuku Farba (Color), vyberiem si farbu, zmením si nastavenie farby aby farba nebola pevná (Solid Color). Vyberám si z možností Solid Color, Linear Gradient, Radial Gradient, Bitmap Fill, None. Pri tvorbe animácie som si vybrala Radial Gradient, farba sa znázorňuje v kruhoch od stredu ku krajom (viď Obrázok 10). Pod výberom farby mám okno s posúvaním intenzity farby a sklonu. Dvojitým kliknutím si vyberám farbu na pozadie

tmavú ľavým ukazovateľom, bledú farbu dvojitým kliknutím s pravým ukazovateľom. Celý tento krok môžem začať robiť, ak na začiatku ešte pred vybratím bielej plochy s nástrojom Rectangle Tool musím kliknúť na Layer 1, ktorý sa nachádza na Časovej osy.

Časová os je umiestnená nad vymedzenou plochou. Takisto ako pri filme sa v dokumentoch Flash čas meria v snímkach. Pri prehrávaní filmu sa prehrávacia hlava (znázornená zvislou červenou čiarou) posúva cez jednotlivé snímky na časovej osy, pričom ich obsah môžeme meniť na vymedzenej ploche. Ak chceme zobraziť jednotlivú snímku, stačí na ňu presunúť prehrávaciu hlavu. Na spodnej strane časovej osy sa zobrazuje číslo zvolenej snímky, aktuálny kmitočet snímok (koľko snímok sa prehrá za sekundu) a čas uplynutý od začiatku filmu. Časová os taktiež obsahuje vrstvy, ktoré nám pomáhajú s usporiadaním kresieb v dokumente. Teraz má môj projekt len jednu vrstvu, ktorá sa nazýva Vrstva 1 (Layer 1). Každá vrstva môže obsahovať iný obrázok, ktorý sa objaví na vymedzenej ploche, pričom môžeme kresliť a upravovať objekty v jednej vrstve, bez toho aby to malo vplyv na inú vrstvu. Usporiadanie vrstiev určuje ich prekrývanie. To znamená, že objekty na spodnej vrstve sú na vymedzenej ploche umiestnené úplne na spodku. Jednotlivé vrstvy môžeme skrývať, zamkýnať alebo zobrazovať ich obsah.

Po vybratí farby si Layer 1 – už premenovaný na Pozadie uzamknem a pridám ďalší Layer 2 a premenujem na Podlaha. Znova si v panely Nástroje vyberiem rectangle tool a označím takú plochu na Pozadí aby som vytvorila vrstvu Podlaha. Na Podlahu si z možností výberu farieb vyberiem Linear Gradient, ktorá farby na ploche ukazuje lineárne. Po vybratí odtieňov mi vznikne pozadie mojej animácie (viď Obrázok 10), a môžem vrstvu Podlaha uzamknúť rovnako ako pri Pozadí. Vrstvy sa na Časovej osy blokujú, aby sa ďalšie vrstvy, ktoré budeme pridávať neprekryli s tými, ktoré sú už vytvorené.

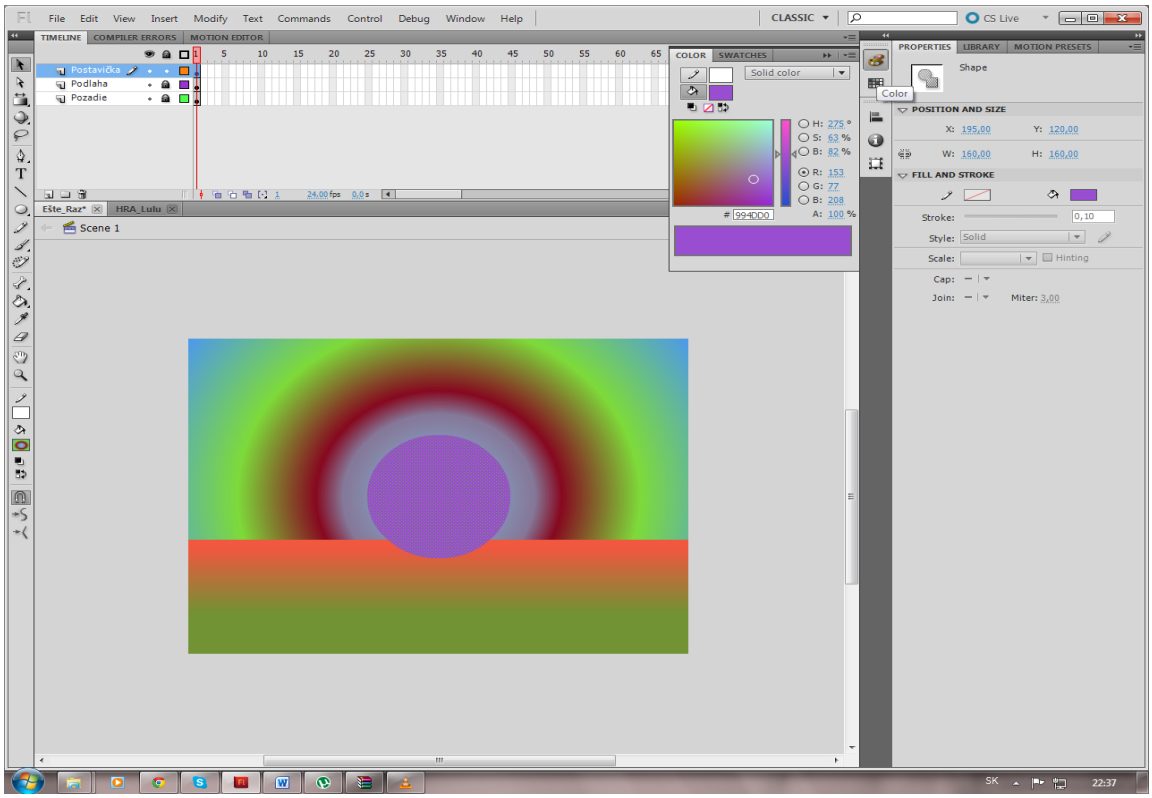
Obrázok 10 Pozadie a Podlaha

Zdroj: Vlastné spracovanie

Vytvorím si novú vrstvu na Časovej osy a pomenujem ju Postavička. Aplikácia Flash ponúka tri druhy kreslenia, ktoré určujú spôsob vzájomnej interakcie objektov na vymedzené pole a spôsob, akým sa dá upravovať. V predvolených nastaveniach používa program Flash zlučovací režim kreslenia, môžeme však aktivovať režim kreslenia objektov alebo použiť nástroj obdĺžnik alebo oval, ktoré využívajú základný režim kreslenia. V zlučovacom režime kreslenia aplikácia Flash zlučuje kreslené tvary, ako sú obdĺžniky a elipsy, v miestach kde sa prekrývajú, takže viac tvarov pôsobí ako jeden. Ak tvar, ktorý bol zlúčený s iným, posuniem alebo zmažem, dôjde k trvalému odstráneniu prekrytej časti. Režim kreslenia objektov kreslené tvary nezlučuje. Zostávajú odlišné a oddelené, a to aj vtedy keď sa navzájom prekrývajú. pre aktiváciu objektového režimu kreslenia vyberiem požadovaný nástroj a potom kliknem na ikonu Kreslenie objektov v oblasti ponúk na panely nástrojov. v režime kreslenia základných objektov používame základné prvky obdĺžnik a oval. Tieto prvky kreslia tvary ako samostatné objekty. Na rozdiel od bežných objektov však môžeme pomocou Vlastností (Properties) meniť pri obdĺžnikoch polomer rohov a pri ovaloch začiatočný, koncový uhol a vnútorný polomer.

Pri vytváraní mojej vrstvy Postavička si vyberiem oválny nástroj (Oval Tool) a vytvorím oválne telo, do ktorého budem postupne vkladať prvky tváre (viď Obrázok 11).

Obrázok 11 Oválny nástroj - Postavička

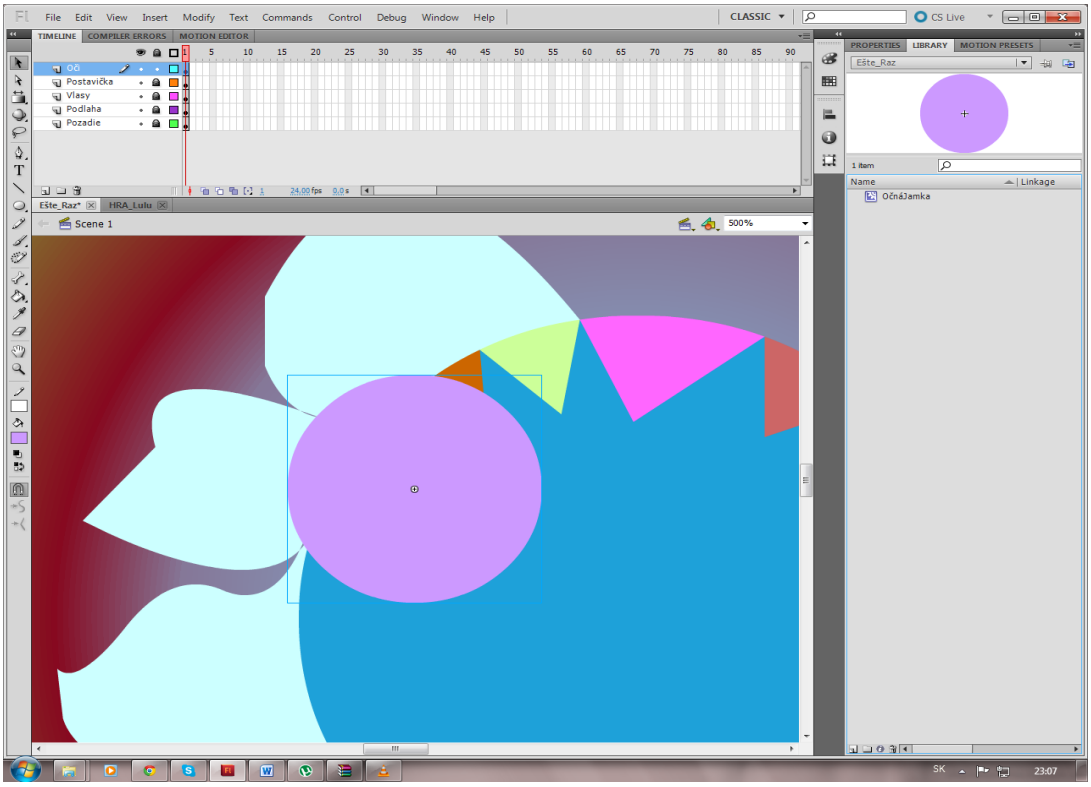


Zdroj: Vlastné spracovanie

Tvorbu postavičky a všetkých prvkov, ktoré budem následne vytvárať nástrojom Oval Tool sa najprv nakreslia okraje oválu, ktoré potom vymazávam pomocou Delete aby mi zostalo len vnútorné vyplnenie. Vyberiem si stred a v panely Vlastnosti (Properties) si zmením veľkosť a následne v možnosti Align moju postavičku vycentrujem vertikálne a horizontálne. Farbu postavičky vytvorím pomocou Solid Color a vrstvu v časovej osy zablokujem. V priebehu tvorenia animácie si ju predbežne ukladám klávesovou skratkou Ctrl+S.

Do časovej osy si pridám novú vrstvu Layer 4 a pomenujem ju Vlasy. Vrstvu môžem začať animovať až keď mám na ňu kliknuté a vidím že je označená sivou farbou. Vlasy vytváram pomocou nástroja Pero (Pen Tool). Vrstvu Vlasy na časovej osy posuniem pod Postavičku aby touto vrstvou boli prekryté. Pridám novú vrstvu Oči, ktorá sa mi na časovej osy automaticky pomenuje Layer5. Zatiaľ mám vytvorenú Postavičku iba do jednej jej polky, postupne túto jednu stranu skopírujem a budem prilepovať, tak aby ostali všetky vlastnosti nezmenené. Mám vybratý nástroj Oval Tool a vytváram oko. Výšku a šírku som si vo Vlastnostiach nastavila na hodnotu 60. Vyberiem si farbu a pokračujem s vybratým kruhom ktorý som si vytvorila. Tento nakreslený kruh si vyberiem tak aby bol označený a po prvýkrát na kruh kliknem pravým tlačidlom myši a v možnostiach si vyberiem Previesť na symbol (Convert to Symbol). V aplikácii Flash sa symboly dajú vytvárať dvoma spôsobmi. Prvý spočíva v tom, že na vybranej ploche nemám vybraté nič, a to zvolením príkazu Vložiť (Insert), Nový symbol (New Symbol). Aplikácia Flash ma prenesie do režimu úpravy symbolov, kde náš symbol môžeme začať kresliť alebo importovať grafiku pre svoj symbol. Druhá možnosť je tá, ktorú používam, a to vybratie nakreslenej plochy a potom následné zvolenie príkazu Zmeniť (Modify), Previesť na symbol (Convert to Symbol) alebo pomocou klávesy F8. Všetko čo je aktuálne vybraté sa mi automaticky premiestni do nového symbolu. Symbol pomenujem OčnáJamka, Typ zvolím Filmový klip (Movie Clip) a registráciu ponechám v strede. Registrácia (Registration) označuje vzťažný bod mojej snímky, stlačím OK a mám vytvorený prvý symbol. Pridávanie symbolov do Knižnice (Library) je dôležitým krokom. Ku každému symbolu neskôr vieme napísať kód alebo s nimi vytvárame pohybujúce animácie. V prostredí Adobe Flash CS5 sa na hlavnej scéne vytvoril symbol, ktorý je teraz označený v modrom štvorci a po dvoj kliknutí naň, OčnáJamka vstúpi do popredia a na obrazovke sa mi objaví nová časová os s prvou vrstvou Layer 1 pre tento symbol, ktorú nazvem Hlavné.

Obrázok 12 Symbol OčnáJamka



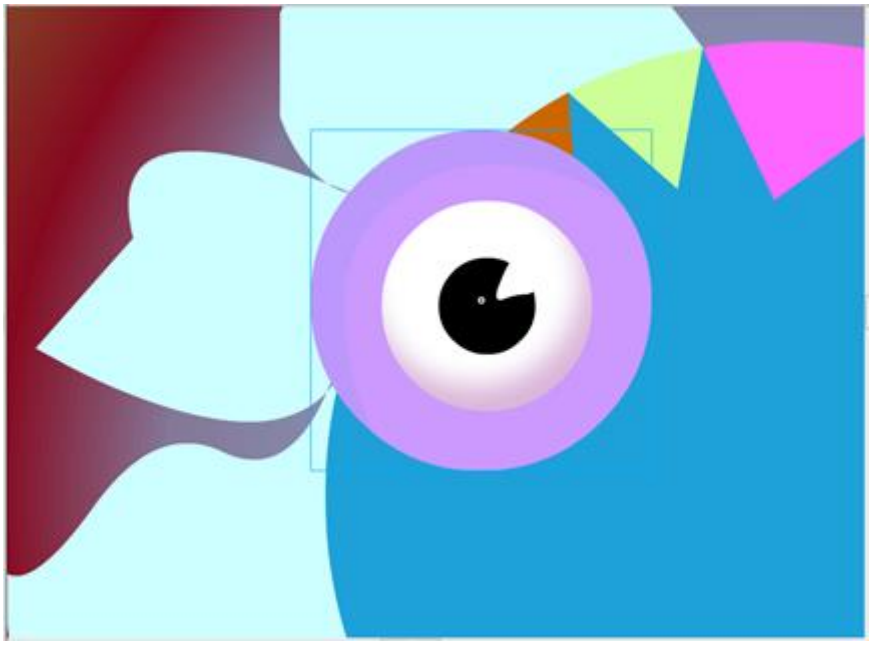
Zdroj: Vlastné spracovanie

Do časovej osy vložím dve nové vrstvy Odraz a Bulva. Po kliknutí na vrstvu Odraz na scéne vytvorím totožný kruh aký už mám vytvorený pomocou Ctrl+C a pravým klikom s vybratím možnosti Vložiť v mieste (Paste in Place). Kliknutím na vrstvu Bulva na scéne vytváram jeden menší biely kruh a pravým kliknutím naň si vyberiem možnosť Convert to Symbol. Tento symbol nazvem Bulva. Po kliknutí na tlačidlo OK sa mi vytvorí ďalšia časová os s novou vrstvou, ktorú nazvem Biele a hneď pridám novú vrstvu Layer 2, ktorú nazývam Šošovka. Na scénu kreslím nový čierny kruh s hodnotou výšky a šírky 17. S vlastných chýb som zistila, že pri vytváraní objektu musím mať vždy kliknuté na vrstvu, ktorej daný objekt patrí. Inak by sa mohlo stať, že sa nastavenia priradia inej vrstve na ktorú mám kliknuté na časovej osy alebo všetky nastavenia zmiznú. S pomocou nástroja Align si šošovku vertikálne aj horizontálne vycentrujem. Po stlačení klávesy Enter pozorujem posun objektu šošovka do stredu vytvoreného oka.

Vo Flash Professional CS5 máme k dispozícii možnosť vytvoriť si vlastné filtre, ktoré môžeme použiť na inštancie filmových klipov. Vo vlastnostiach (Properties) rozbalím časť Filtre a máme k dispozícii niekoľko filtrov (tiene, sklony, škvrny, žiaru, sklon sklonu, sklon žiary a nastavenie farieb), z ktorých každý ponúka rôzne možnosti na doladenie výsledného efektu. Tieto filtre sa dajú nájsť len vo vlastnostiach symbolu

v dolnom rohu ponuky Vlastnosti. Vytvorené oko skopírujem pravým tlačidlo pustím na miesto a pomocou vlastností si obe oči nastavím do rovnakej pozície X a Y.

Obrázok 13 Hotové oko

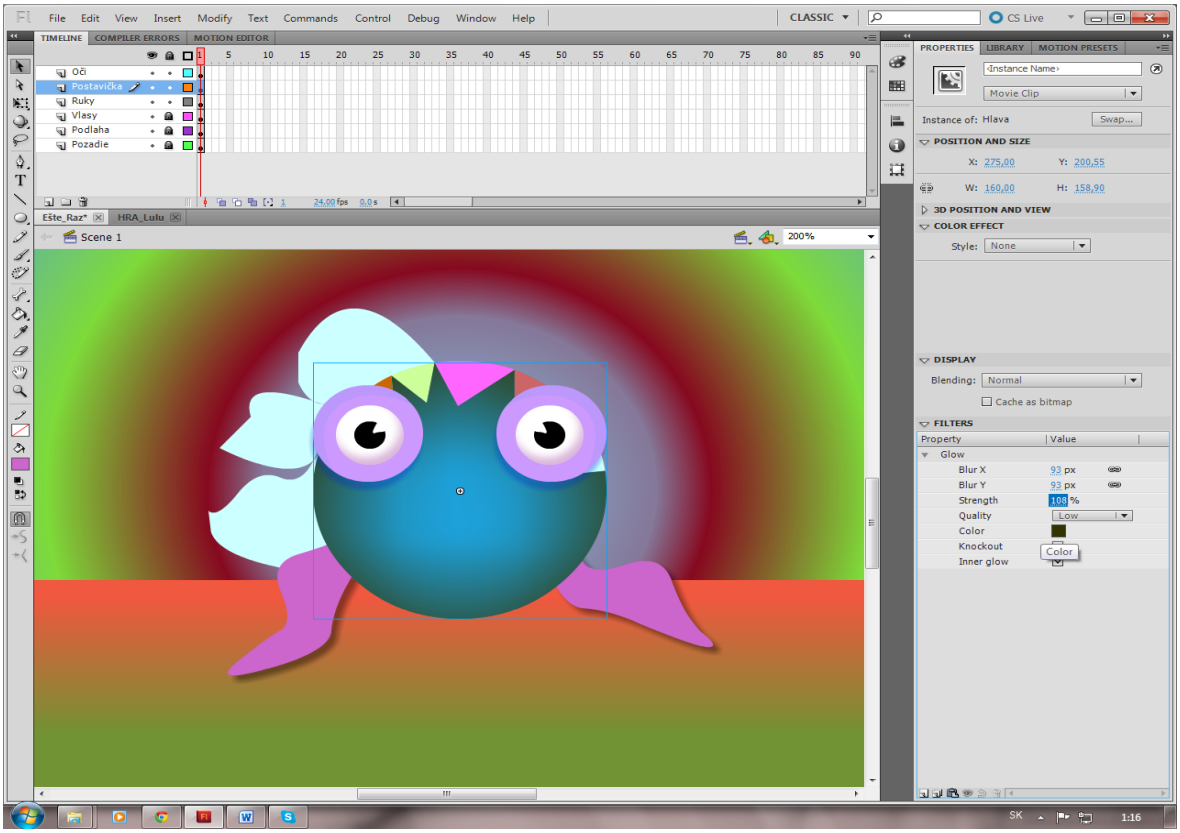


Zdroj: Vlastné spracovanie

Na hlavnej scéne mám pripravený nový Layer, ktorý nazývam Ruky. Okrem tejto vrstvy sa v časovej osy nachádzajú Pozadie, Podlaha, Vlasy, Postavička, Oči a Ruky. Začínam kresliť objekt Ruky. Použijem nástroj Oval Tool a nastavím si ho pomocou Modify – Align (uhol) – Rotate and Skew do vyžadovaného uhla. Znova modelujem len jednu ruku, teda jej ľavú časť, pretože je jednoduchšie si neskôr skopírovať objekt ako vytvárať nový. Keď ju mám hotovú vytvorím si pomocou pravého kliku s vybranou možnosťou Convert to Symbol a začínam dopĺňať potrebné informácie. Keďže ruka je tvaru ovalu priradím jej registračný bod v pravom hornom rohu. Symbol pomenujem Ruka a v možnostiach Typ si môj symbol označím ako Filmový klip (Movie Clip). Po odkliknutí OK Vidím na hlavnej scéne moju ruku označenú v modrom štvorci. Ešte predtým si v panely Vlastnosti nastavím efekt Sklonu. Skopírujem a vytvorím si ďalšiu ruku (viď Obrázok 14). Na Scéne 1 kde sa mi tvoria všetky objekty Časovej osy si odblokujem Vrstvu Postavička, vytvorím si nový symbol Hlava s registráciou na stred a typ Movie Clip. Pomocou efektu Glow v panely Properties prídám postave žiaru. V ponuke zaškrtnem možnosť Vnútorná žiara (Inner Glow) a nastavím si BlurX, BlurY a silu (Strengh). Myšou uchopím obe ruky a na časovú os pridám Layer 7 a nezvem ho Nohy. Kliknutím na túto vrstvu na Scéne pomocou

možnosti Paste In Place si vytvorím objekty nohy. Potom ich otočím do polohy (viď Obrázok 15).

Obrázok 14 Vymodelované Ruky, Hore vytvorená časová os



Zdroj: Vlastné spracovanie

Obrázok 15 Nohy

Zdroj: Vlastné spracovanie

3.4 Pridávanie kľúčových snímok

Animácia je pohyb alebo zmena objektov v čase. Animácia môže byť napríklad aj jednoduchý posun rámov po predvolenej ploche z jednej snímky na druhú. Môže to však byť taktiež oveľa zložitejšie. Animovať sa dá celá rada rôznych vlastností jediného objektu. Môžeme meniť polohu objektu na vymedzenej ploche, jeho farbu, priehľadnosť,

veľkosť, či natočenie alebo dokonca animácia špeciálnymi filtrami. Okrem toho môžeme ovládať taktiež cestu pohybu objektu, či dokonca aj jeho nábeh a dobeh, ktoré určujú zrýchlenie a spomalenie objektu. Základná práca s animáciou v aplikácii Flash vyzerá nasledovne: Pre animovanie objektov najprv vyberieme objekt v predvolenej ploche, klepneme na ňu pravým tlačidlom myši alebo s klávesou Ctrl a potom zvolím príkaz Vytvoriť doplnenie pohybu (Create Motion Tween). Presuniem červenú prehrávaciu hlavu na iné miesto v čase a presuniem objekt do novej polohy. Potom sa o to stará už aplikácia Flash. Technika doplnenia pohybu vytvárania snímok animácie pre zmeny polohy v predvolenej ploche a taktiež pre zmenu veľkosti, farby a ďalších atribútov. Táto technika sa dá aplikovať iba na inštanciách symbolov.

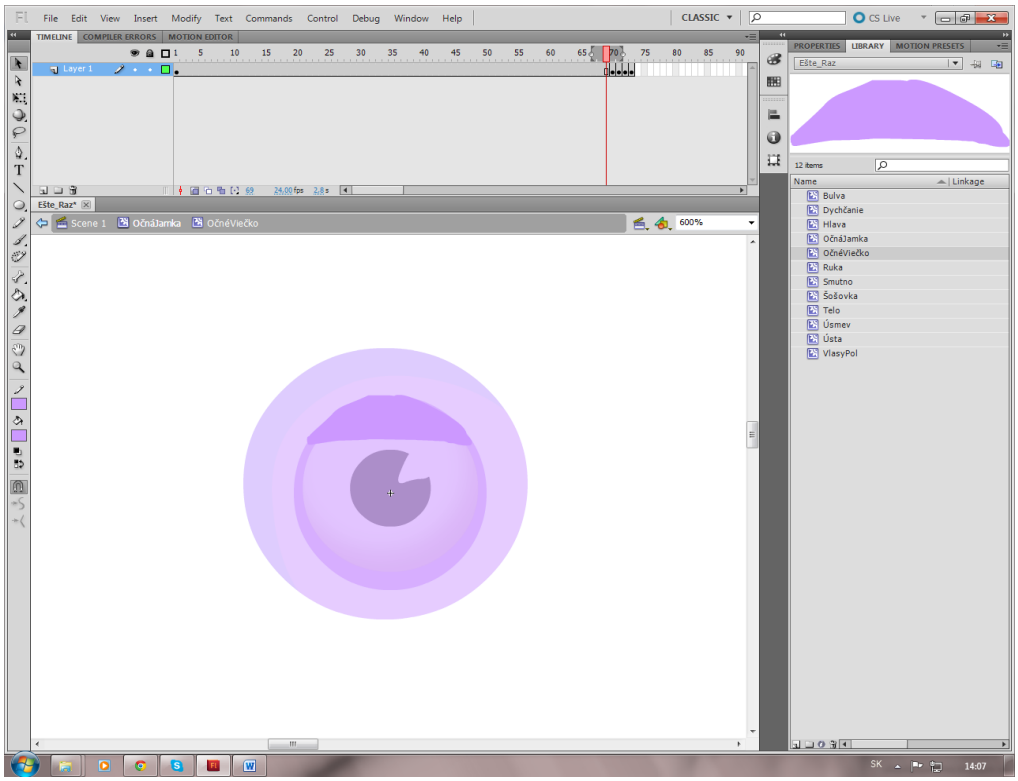
Pre pridávanie Kľúčových snímok používam klávesu F6. Použijem Shape (tvarovanie, klasické alebo pohybové dopĺňanie pre odlišné výsledky).

Prvá vec, ktorá sa musí urobiť je odblokovanie všetkých vrstiev na časovej osy. Kliknutím na vrstvu vlasy si na scéne vytvorím nový symbol pomocou Convert to Symbol a nazvem ho VlasyPol. Dvojitým kliknutím sa dostanem k jeho časovej osy. S hlavného panela si vyberiem Insert a kliknutím na Shape Tween (vytváranie a práca so symbolmi) budem pracovať s časovou osou a jej snímkami. Vyberiem prvú snímku vrstvy Layer 1 a vložím do nej kľúčový snímok Create Shape Tween. Snímka sa zafarbí na zeleno. Taktiež si vyberiem 80 snímok tejto vrstvy a vložím tam ďalšiu kľúčovú snímku (Insert Keyframe). Animácia, ktorá bude hýbať vlasmi sa bude odohrávať na zelených snímkach. Pri Shape Tween animácia začína od prvej snímky a končí na 79. Na snímku 80 si vytvoríme pohyb vlasov. V tomto prípade sa vlasy budú 79 snímok tvoriť do pozície akú som si vytvorila v snímke 80. Keď mám vytvorenú animáciu na jednej strane vlasov skopírujem ich a použijem transformáciu aby som kópiu otočila horizontálne (Flip Horizontal) a konečný efekt je celá hlava pokrytá vlasmi, ktorých pohyb si vieme skontrolovať použitím klávesovej skratky Ctrl+Enter. Vo videu, ktoré sa nám týmto príkazom zapne vidíme pohyblivú animáciu. V nasledujúcom kroku spojím vrstvu Ruky, Vlasy, Postavičku a Oči do jedného symbolu Telo. Zablokujem si Pozadie a Podlahu a uchopením týchto objektov na Scéne si vytvorím nový symbol Telo. Všetky tieto vrstvy sa spojili do jednej. Začínam s animáciou jednotlivých častí tela. Teraz keď na hlavnej časovej osy kliknem myšou označí sa mi do modrého štvorca celé telo okrem nôh. Vrstvu Vlasy si premenujem na Telo. Po dvojitom kliknutí na Postavičku, označím si obe ruky a pravým kliknutím vyberiem možnosť Distribute Layers. Na časovej osy sa mi vytvoria

dve nové vrstvy, ktoré si pomenujem Pravá a Ľavá Ruka. Teraz mám v objekte Telo na jeho časovej osy vytvorené vrstvy: Oči, Postavička, Pravá Ruka, Ľavá Ruka, Vlasy.

Vo vrstve Ľavá Ruka si kliknem na jej prvý snímok a pravým kliknem na možnosť Create Motion Tween. Na scéne zostanú iba nohy a jedna ľavá ruka, ktorú budem animovať. Pohyb rúk dotiahnem až k snímku 100. S použitím F6 pridám novú kľúčovú snímku. V stej snímke ľavej ruky kliknem pravým tlačidlom myši v možnostiach si nájdem Insert Keyframes a kliknem na All. Teraz moja postavička hýbe aj ľavou rukou. Podobným spôsobom vytvorím animáciu pravej ruke. S pohybom rúk som skončila a nasleduje vytvorenie troch symbolov úst Úsmev, Smutno a Dychčanie. Tieto symboly vytváram pomocou New Symbol v hlavnej ponuke cez Insert. V novom symbole ústa na časovej osy vytvorím tri nové Keyframes. Do prvého vložím symbol Úsmev, do druhého symbol Smutno a do tretieho symbol Dychčanie. Pri spustení videa na skontrolovanie animácie pomocou Ctrl+Enter sa mi tieto tri snímky striedajú. Tento problém sa dá odstrániť vložením zdrojového kódu, ktorý budem vkladať v nasledujúcej kapitole. Nasleduje animácia Očí postavičky. V tejto animácii vytvorím efekt zatvárajúcich sa a otvárajúcich očí s použitím pomôcky z časovej osy, a to Onion Skin.

Obrázok 16 Vytváranie blikania očí s pomocou Onion Skin



Zdroj: Vlastné spracovanie

Pri začiatku tvorby každej z mojich animácií sa vždy odrážam od Hlavnej scény a postupným klikaním na jednotlivé objekty sa dostávam k ich časovým osám. Tento krát si ale v Knižnici nájdem symbol OčnéJamky. Po dvojkliknutí na tento symbol si na ich časovú os pridám novú vrstvu a nazvem ju OčnéViečko. S použitím nástroja Oval Tool si na objekt oka nakreslím ďalší oval aby mi prekryl oko a preto z neho odoberiem jeho originálnu farbu aby viečko vyzeralo ako jeden celok s okom postavičky. Z novo vytvoreného viečka s použitím Convert to Symbol a dvojkliknutím na tento symbol sa dostávam k novej časovej osy. Toto bliknutie oka na časovej osy presuniem na 70 snímok vrstvy Layer 1 (drag and drop). Postupne si vytvorím štyri nové kľúčové snímky a do každej nakreslím postupné zatváranie oka. Za tieto štyri okopírujem tri prvé keyframes ale v opačnom poradí aby vytvárali efekt zatvorenia a otvorenia viečka. Po stlačení Ctrl+Enter zistím, že viečko funguje tak ako má. Teraz nasleduje vpísanie kódu do jednotlivých snímok časových osí.

3.5 Kódovanie

Vďaka jazyku ActionScript, ktorý je podobný jazyku JavaScript, môžeme do svojich animácií Flash pridať viac interaktívnych prvkov.

3.5.1 Terminológia skriptovacích jazykov

Mnohé pojmy, ktoré sa používajú na popis jazyka ActionScript, sú podobné termínom používaným v oblasti skriptovacích jazykov. V dokumentácii k jazyku ActionScript sa najčastejšie stretneme s nasledujúcimi pojmami:

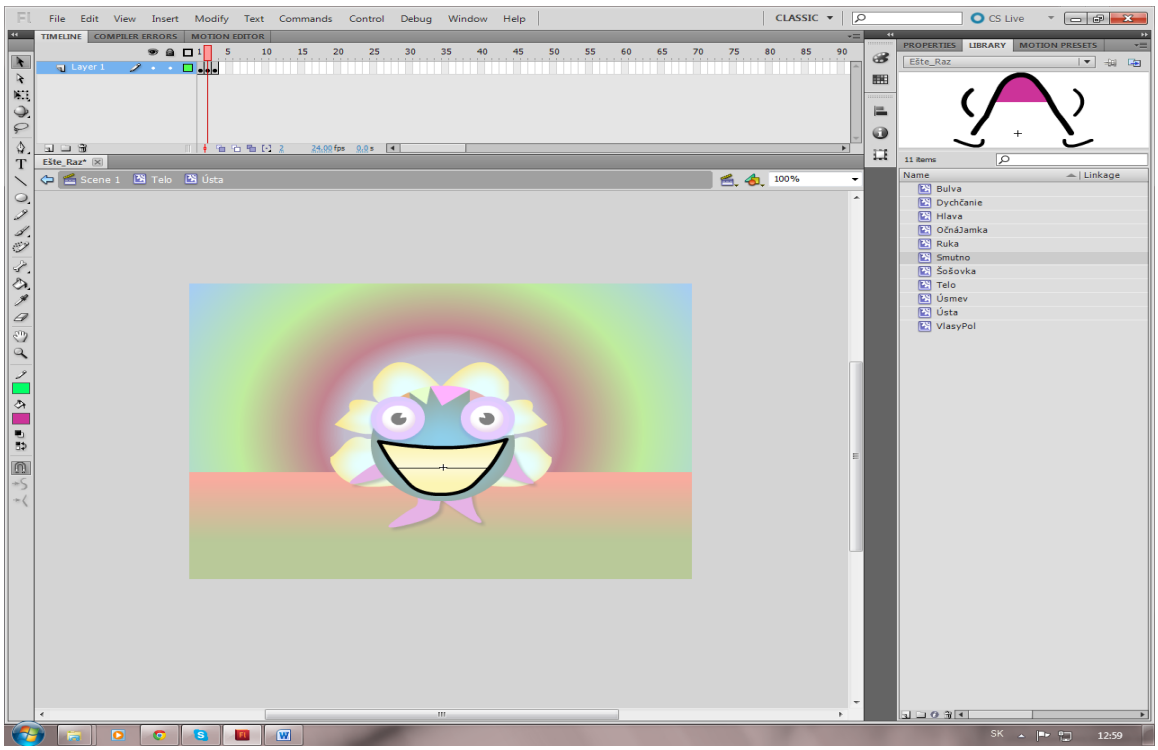
- Premenná - predstavuje určitú časť dát, ktorá môže či nemusí byť konštantná. Pri vytváraní alebo deklarácii premennej jej pridelím typ, ktorý určuje aký druh dát môže daná premenná reprezentovať.
- Kľúčové slovo – je v prostredí jazyka ActionScript vyhradené slovo, ktoré sa používa k prevádzaniu určitej úlohy. Napríklad pomocou kľúčového slova var sa vytvára premenná.
- Argumenty – ďalej špecifikujú príkaz, ku ktorému sa vzťahujú a na riadku kódu sa uvádzajú medzi zátvorkami ().
- Funkcie – sú skupina príkazov, ku ktorým sa môžeme odkazovať názvom. Pomocou funkcií sa dajú spúšťať rovnaké sady príkazov, bez toho aby bolo nutné ich stále dopisovať do scriptu.

- Objekty – V jazyku ActionScript 3.0 pracujeme s objektmi, ktoré sú abstraktné typy dát, ktoré nám pomáhajú prevádzať určité úlohy. Každý objekt by sa mal nejako nazývať. Na objekt, ktorý má meno, sa dá odkazovať v kóde jazyka ActionScript a taktiež sa kódom ovláda.
- Metódy – sú príkazy, ktorých výsledkom je nejaká akcia. Metódy sú pracanti ActionScriptu, pričom každý objekt má svoju vlastnú sadu metód.
- Vlastnosti – popisujú objekt. Napríklad medzi vlastnosťami filmového klipu patrí výška a šírka, súradnice x a y alebo horizontálna a vertikálna mierka.

Vrátim sa naspäť na Scénu 1, a postupne budem animácii, jej jednotlivým vrstvám vdychovať život. Zmením vzhľad úst, po kliknutí myšou na úsmev, očiam napíšem kód aby sledovali všetky pohyby myšou. Hviezdu v pozadí naprogramujem tak, aby sa hýbala po pozadí a každým kliknutím sa otočila o 10 stupňov.

V knižnici si nájdem symbol Ústa a dvojitým kliknutím sa dostanem dovnútra symbolu. Na časovej osy symbolu Ústa mám vytvorené tri keyframes.

Obrázok 17 Symbol Ústa, Layer 1 s troma kľúčovými rámami



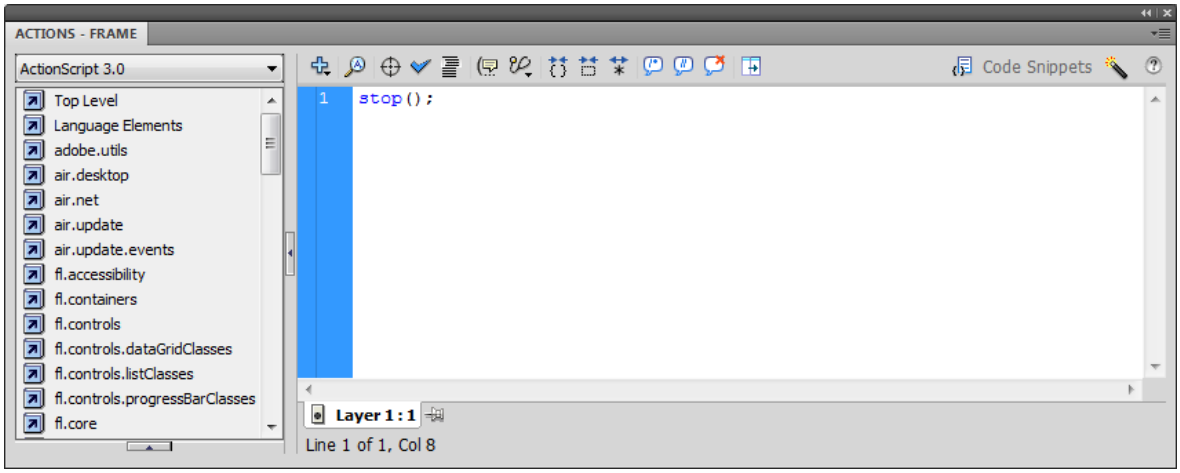
Zdroj: Vlastné spracovanie

Keď chceme začať písať kód musíme si vytvoriť vždy novú vrstvu kde do prvej snímky budeme písať kód. Pridám sa Vrstvu (Layer 2) a nazvem ju Script. Pomocou klávesy F9 si na prvej snímke Script otvorím panel Akcie (Actions).

Panel Akcie sú miestom, kde sa zapisuje všetok kód. Tento panel je rozdelený na niekoľko čiastkových podokien. V ľavom hornom rohu sa nachádza panel nástrojov Akcie, ktorý obsahuje niekoľko kategórií, v ktorých je usporiadaný celý kód jazyka ActionScript. V hornej časti panela nástrojov Akcie je rozbalovacia ponuka, ktorá zobrazuje iba kód pre zvolenú verziu jazyka ActionScript.

Do prvého riadku kódu napíšem stop();. Je to funkcia, ktorá má v zátvorke informácie, v mojom prípade informácie o snímkach objektov úsmev, smutno, dychčanie. Kód sa objaví v podokne Skript a v prvej kľúčovej snímke vrstvy Script sa objaví malinké písmeno „a“, ktoré signalizuje, že daná kľúčová snímka obsahuje kód jazyka ActionScript.

Obrázok 18 Objekty tváre sa zastavia na prvej snímke



Zdroj: Vlastné spracovanie

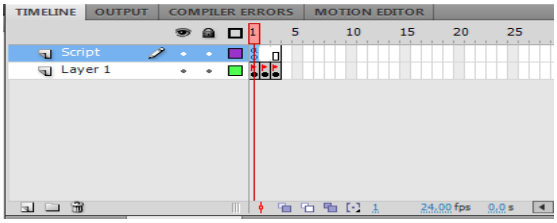
Tento zdrojový kód mi zastaví objekty tváre, ktoré sa striedali na prvej snímke Úsmev. Teraz napíšem funkciu pre dychčanie keď na postavičku kliknem myšou. Pre túto funkciu musím zavolať poslucháča udalostí addEventListener (vždy musíme dodržovať malé a veľké písmená.

```
stop();
        //zadychčí keď na ústa kliknem myšou
addEventListener(MouseEvent.CLICK, ustaDychcanie);
        // vysvetlenie ustaDychcanie
function ustaDychcanie(m:MouseEvent){
        // choď do rámu so symbolom Dychčanie
  gotoAndStop("Dychcanie");
       // po dvoch sekundách sa vráti Dychčanie na Úsmev
   setTimeout(ustaUsmev, 2000);
}
```

```
function ustaUsmev(){
  gotoAndStop("Usmev");
}
```

Navigačný príkaz gotoAndStop("Dychčanie"); by sme mohli napísať aj tak, že do zátvorky by sme miesto Dychčanie napísali (3). To znamená že kód sa zastaví na tretej snímke. Tak isto musím aj túto snímku premenovať na Dychčanie aby program vedel kde sa má funkcia zastaviť. Po napísaní mena snímky, vidím, že na časovej osy v snímke ktorá má meno sa objavila červená vlajka. Udalosť MouseEvent.CLICK znemená, že keď kliknem na ústa v animácii sa prehrávacia hlava zastaví na tretej snímke. Pri každej udalosti musím vysvetliť, čo jednotlivé udalosti znamenajú. Zdrojový kód setTimeout(ústaÚsmev, 2000) znamená, že po dvoch sekundách sa dychčanie vráti naspäť na prvú snímku Úsmev. Aby jazyk ActionScript vedel rozoznať všetky snímky musím si ich všetky pomenovať. Na všetkých teda budem mať po jednej červenej vlajke. Funkcia sa vždy musí písať do zložených zátvoriek.

Obrázok 19 Označenie Časovej osy



Zdroj: Vlastné spracovanie

Nasleduje zapísanie kódu pre symbol Telo, ktorý si budem musieť vyhľadať v Knižnici a s dvojitým kliknutím sa dostať k jeho Časovej osy. Na časovej osy si vytvorím novú vrstvu a pomenujem ju Script. Znova musím zavolať poslucháča udalostí addEventListener.

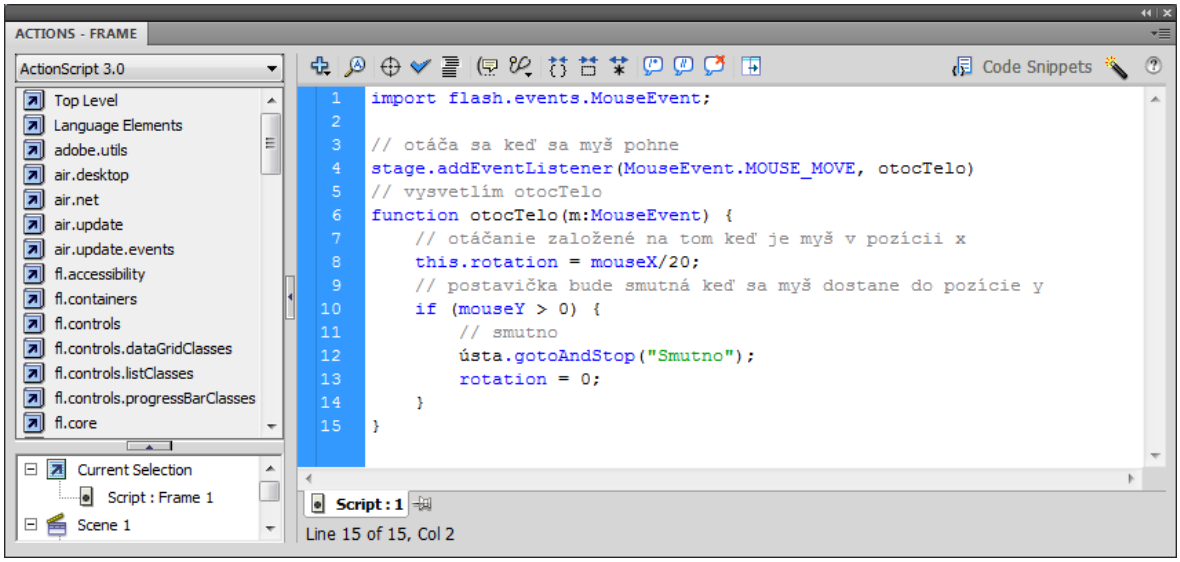
```
import flash.events.MouseEvent;


        // otáča sa keď sa myš pohne
stage.addEventListener(MouseEvent.MOUSE_MOVE, otocTelo)
        // vysvetlím otocTelo
function otocTelo(m:MouseEvent) {
        // otáčanie založené na tom keď je myš v pozícii x
  this.rotation = mouseX/20;
        // postavička bude smutná keď sa myš dostane do pozície y
  if (mouseY > 0) {
```

```
        // smutno
usta.gotoAndStop("Smutno");
rotation = 0;
  }
}
```

Po napísaní kódu stage.addEventListener(MouseEvent.MOUSE_MOVE, otocTelo) sa mi automaticky vloží príkaz import. Funguje s nastavenými radami pri písaní kódu v predvoľbách prostredia Flash. Kód v štvrtom riadku Akcie znamená, že pri hýbaní myšou na stage, takže iba na pozadí, sa postavička bude otáčať okolo svojej osy pri pohybe myši. MouseEvent je udalosť myši. Potom musím vysvetliť vo funkcii, čo znamená otocTelo a do kódu dopíšem v akej pozícii X a Y sa myš bude nachádzať a akú rotáciu postavička urobí. V dvanástom a trinástom riadku je napísané, že pri nulovej rotácii sa potavička bude tváriť smutno.

Obrázok 20 Zdrojový kód pre vrstvu Script v Telo



Zdroj: Vlastné spracovanie

Cez hlavnú Scénu animácie postupne klikám na šošovku, až kým sa nedostanem k jej časovej osy aby som mohla napísať kód pre pohyb šošoviek za myšou.

```
import flash.events.Event;

        //V každom ráme sa šošovka bude pozerať za myšou
addEventListener(Event.ENTER_FRAME, pozriNaMys);
        //vysvetlenie pozriNaMys
function pozriNaMys(e:Event){
  //uhol na myš
```

```
  var uhol = Math.atan2(mouseY,mouseX);
            //pohyb šošovky, v tomto smere, začiatok v 0, 0
  var desiredX = Math.cos(uhol) * 8;
  var desiredY = Math.sin(uhol) * 8;
           //zmiernim polohu, miesto prichytenia
  sosovka.x += (desiredX - sosovka.x) * 0.2;
  sosovka.y += (desiredY - sosovka.y) * 0.2;
}
```

Predchádzajúci zdrojový kód znamená: pridávam poslucháča udalosti, ktorý má v zátvorke udalosť šošovky pozriNaMys s vysvetleným vo funkcii.

Na pozadí som vytvorila farebnú hviezdu. Preto som musela vytvoriť ďalšiu vrstvu Script aby som do jej zdrojového kódu napísala aby sa pohybovala po pozadí a po kliknutí na hviezdu sa otáčala o desať stupňov. Jej zdrojový kód vyzerá takto:

```
import flash.events.MouseEvent;
import flash.events.Event;
hviezda.addEventListener(MouseEvent.CLICK, otocHviezdu);
function otocHviezdu(e:MouseEvent) : void {
  hviezda.rotation +=10;
}
addEventListener(Event.ENTER_FRAME, posunHviezdu);
function posunHviezdu(e:Event): void {
  if (hviezda.x < stage.stageWidth){
            hviezda.x += 2;
}   else {
  hviezda.x=0;
}
}
```

V tomto zdrojom kóde sa mi automaticky priradili tri importy. Prvý je privolaný pre kliknutie na hviezdu, druhý pre pohyb hviezdy. Pri prvom volaní poslucháča udalosti som vytvorila udalosť myši, ktorý popisuje klik myši a následné otočenie o desať stupňov. Pri druhom volaní udalosti animujem vodorovnú pozíciu hviezdy s vlastnosťou x.

Obrázok 21 Konečná animácia

Zdroj: Vlastné spracovanie

ZÁVER

V mojej práci som sa venovala opisu bohatých klientskych aplikácií v skratke RIA (Rich client applications). V názve bakalárskej práce bolo určené písať o pojme CRA čo znamená Client Rich Applications ale týmto pojmom sú vo všeobecnosti označované dynamicky generované webové aplikácie, ktoré svojim kvalitne spracovaným, vysoko interaktívnym ovládaním dosahujú takmer úroveň bežných desktopových aplikácií. V tejto súvislosti sa častejšie stretávame so synonymickým označením RIA. Prvá kapitola je zameraná na teoretický rozbor a charakteristiku hlavných pojmov v predmetovej oblasti. V jej ďalších podkapitolách je splnený prvý čiastkový cieľ a to vymedzenie pojmov v predmetovej oblasti. Druhá kapitola sa zaoberá opisom tvorby aplikácií typu RIA v prostredí Adobe Flash s využitím jazyka ActionScript, tým ja splnený druhý čiastkový cieľ práce. Kapitola sa zaoberá hlavnými metódami, tvorbou premenných, podmienenými príkazmi, modifikátormi prístupu, parametrami a udalosťami skriptovacieho jazyka ActionScript. V tejto kapitole je poukázané na programy od spoločnosti Adobe, v ktorých je umožnené tvoriť bohaté klientske aplikácie s jazykom ActionScript. Tretia kapitola je zameraná na vytvorenie animácie v prostredí Adobe Flash Professional CS5 s využitím jazyka ActionScript 3.0. Na základe uvedeného sa domnievam, že hlavné a čiastkové ciele práce boli splnené.

ZOZNAM POUŽITEJ LITERATÚRY

1. ASLESON, R., SCHUTTA, N. *Vytvárime vysoce interaktivní webové aplikace*. První vydání. Brno : Computer Press, 2007. ISBN 80-251-1285-3.
2. BASL, J, BLAŽÍČEK, R, *Podnikové informační systémy,* Grada, 328 s. ISBN 978-80 247-4307.
3. BURIAN, P. 2012. *Webové a agentové technológie*. Praha : Grada, 2012. 376 s. ISBN 978-80-247-4376-9.
4. CROFT, J., LIOYD, I., RUBIN, D. *Mistrovství v CSS* : *Pokočilé techniky pro webové designéry a vývojáře*. První vydání. Brno : Computer Press, 2007. ISBN 978-80-251-1705-7.
5. Flash Development for Android Cookbook. 2011. Packt Publishing. ISBN 184-969-142-8.
6. *Getting started with Adobe Flex.IBM Corporation*. 2010. 239 s.
7. GORALSKI, G. 2013. *Foundation Flex for Designers*. Published by Springer-Verlag. New York, Inc. ISBN 978-1-59059-877-1.
8. HALÁK, J. Rýchly vývoj aplikácií v jazyku Visual Basic 2010 pre systém Windows 7. 2013.71 s. ebook
9. LABRECQUE, J. 2011. *What´s is new in Adobe AIR 3*. O´Reilly Media, 2011. 104 s. ISBN 978-1-4493-1107-0.
10. McCUNE, D. , DEEPA S. *Adobe® Flex® 3.0 For Dummies®.* Published by Wiley Publishing. ISBN: 978-0-470-27792-8.
11. POWELL, Thomas, A.,*Web design,* Computer Press , 2004, ISBN 80-722-694-96.
12. SLENDER, G. *Developing with Ext GWT – Enterprise RIA Development,* Apress, 140 s. ISBN 978-143-021-940-8.
13. SUSAN, S. *101 Ways to Promote Your Web Site* [e-book]. 7th edition. GulfBreeze : Maximum Press, 2009. ISBN 978-1-931-644-65-5.
14. FERONATO, E. *Programujeme hry ve Flashi* Brno:Computer Press, 2012. ISBN 978-80-251-3697-3.
15. ADOBE CREATIVE TEAM. *Adobe Flash Professional CS6 – Oficiální výukový kurz.* Brno: Computer Press, 2013. ISBN 978-80-251-3802-1
16. ADOBE CREATIVE TEAM. *ActionScript 3.0 – Oficiální výukový kurz.* Brno: Computer Press, 2011. ISBN 978-80-251-3335-4

Zoznam obrázkov

Zoznam tabuliek