

Volební klient v jazyce Flash

A Flash-based Voting Client

Bc. Roman Ryba

Diplomová práce
2012



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
akademický rok: 2011/2012

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Roman RYBA**
Osobní číslo: **A10413**
Studijní program: **N 3902 Inženýrská informatika**
Studijní obor: **Informační technologie**

Téma práce: **Volební klient v jazyce Flash**

Zásady pro vypracování:

1. Analyzujte požadavky na volebního klienta v technologii Flash.
2. Seznamte se s principy tvorby aplikací pomocí technologie Flash.
3. Naprogramujte klienta v programu Flash Professional.
4. Naprogramujte klienta v programu Flash Builder.
5. Oba klienty implementujte.
6. Vytvořte vhodné testovací prostředí.
7. Řešte zabezpečení přenosu citlivých dat mezi klientem a serverem.
8. Provedte vyhodnocení klientů.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. GROVER, Chris. Flash CS5.5: the missing manual. 1st ed. Sebastopol, CA: O'Reilly, 2011, 841 s. Missing manual. ISBN 14-493-9825-1.
2. FLORIO, Chris. Actionscript 3.0 for Adobe Flash Professional CS5 classroom in a book: the official training workbook from Adobe Systems. 1st ed. Berkeley, Calif.: Adobe Press, 2010, 389 s. Classroom in a book. ISBN 03-217-0447-9.
3. FLORIO, Chris. Adobe Flash Professional CS5 classroom in a book: the official training workbook from Adobe Systems. 1st ed. San Jose, Calif.: Adobe, 2010, 375 s. Classroom in a book. ISBN 03-217-0180-1.
4. NOBLE, Joshua J. Flex 4 cookbook: the official training workbook from Adobe Systems. 1st ed. Sebastopol, Calif.: O'Reilly, 2010, 736 s. Classroom in a book. ISBN 05-968-0561-6.
5. TAPLEY, Scott. Adobe Flash Catalyst CS5 classroom in a book: the official training workbook from Adobe Systems. 1st ed. San Jose, CA: Adobe Press, 2010, 273 s. Classroom in a book. ISBN 03-217-0358-8.

Vedoucí diplomové práce:

Ing. Radek Šilhavý, Ph.D.

Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce:

24. února 2012

Termín odevzdání diplomové práce:

21. května 2012

Ve Zlíně dne 24. února 2012

prof. Ing. Vladimír Vašek, CSc.
děkan



doc. Mgr. Roman Jašek, Ph.D.
ředitel ústavu

ABSTRAKT

Cílem diplomové práce je vytvořit softwarového volebního klienta, který bude podporovat distribuované volební schéma pro elektronické volby (vyvíjí se na UPKS). Implementace zabezpečení přenosu mezi klientem a serverem. Vytvoření serverového základu pro klienta pro sběr dat a identifikaci voličů, ověření komunikace a funkčnosti dle schématu elektronických voleb. Vytvoření klienta pomocí Flash Builder a Flash professional, vysvětlení rozdílů principů tvorby a rozdílů technologie tvorby. Rozdíly mezi implementací pro Flash Player a Adobe Air. Součástí bude otestování prototypu a rozhodnutí o vhodnosti či nevhodnosti.

Klíčová slova: Flash, Flex, Flash Player, Adobe Air, Flash Builder, Flash Professional, volby

ABSTRACT

The aim of this thesis is to create election software client that will support distributed election scheme for electronic elections. Implement of security specifications between the client and server. Creating a client-server basis for data collection and identification of voters, verifying the functionality of communication and e-elections according to scheme. Creating a client in Flex Builder and Flash Professional, explain the principles and differences in the technology of creating applications in them. Explain the differences between implementation in Flash Player or Adobe Air. The prototype will be tested to determine the appropriateness.

Keywords: Flash, Flex, Flash Player, Adobe Air, Flash Builder, Flash Professional, voting

Rád bych poděkoval mému vedoucímu diplomové práce Ing. Radku Šilhavému, Ph.D., za odborné vedení, cenné rady a hlavně za čas věnovaný mé práci.

Motto:

*Jestliže jste postavil vzdušný zámek, vaše práce nemusí být zbytečná.
Nyní k němu dodělejte základy.*

Henry David Thoreau

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....

podpis diplomanta

OBSAH

ÚVOD	9
I TEORETICKÁ ČÁST	10
1 FLASH	11
1.1 HISTORIE	11
1.2 ACTIONSCRIPT 3.0	12
1.3 FLASH PROFESSIONAL	12
1.3.1 Pracovní prostředí.....	13
1.3.2 Základní objekty	15
1.3.3 Základní animace	16
1.3.4 Action Script ve Flash Professional	18
1.4 FLASH BUILDER	19
1.5 FLASH CATALYST.....	22
1.6 SPOLUPRÁCE PROGRAMŮ ADOBE.....	25
2 VOLBY	26
2.1 POŽADAVKY ELEKTRONICKÝCH VOLEB	26
2.2 ELEKTRONICKÝ HLASOVACÍ LÍSTEK	26
2.3 KLÍČOVÝ POPIS SCHÉMATU	27
3 BEZPEČNOST	28
3.1 RSA	28
3.2 SHA-2.....	29
3.3 HTTPS	29
II PRAKTICKÁ ČÁST	30
4 POŽADAVKY	31
4.1 NÁVRH	32
4.2 ZABEZPEČENÍ HLASOVACÍHO PROCESU	34
4.3 VYTVOŘENÁ VERZE.....	35
5 FLASH PROFESSIONAL	37
5.1 ZAČÁTEK	37
5.2 GRAFIKA.....	37
5.3 VYTVOŘENÍ INTERAKCÍ	38
6 FLASH BUILDER	40

6.1	PRVOTNÍ NÁVRH	40
6.2	PŘEVOD VE FLASH CATALYST	42
6.3	FINÁLNÍ DODĚLÁNÍ VE FLASH BUILDER.....	45
7	TESTOVACÍ SERVER.....	48
7.1	PŘÍKLAD PŘEDÁVÁNÍ DAT.....	48
7.2	FUNKCE SERVERU	49
7.3	ZABEZPEČENÍ PŘENOSU CITLIVÝCH DAT MEZI KLIENTEM A SERVEREM	49
7.4	PŘÍKLAD ŠIFROVACÍ FUNKCE RSA	50
8	DALŠÍ MOŽNOSTI.....	52
8.1	ADOBE LABS.....	52
8.1.1	Adobe Alchemy	52
9	ZHODNOCENÍ.....	53
9.1	ZHODNOCENÍ POŽADAVKŮ	53
9.2	VÝHODY FLASH TECHNOLOGIE PRO VOLEBNÍHO KLIENTA	54
9.3	ZHODNOCENÍ FLASH.....	54
	ZÁVĚR	56
	ZÁVĚR V ANGLIČTINĚ	57
	SEZNAM POUŽITÉ LITERATURY	58
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	60
	SEZNAM OBRÁZKŮ	61
	SEZNAM TABULEK	62
	SEZNAM PŘÍLOH	63

ÚVOD

Volby jsou důležitým aspektem demokratické společnosti, za svůj život člověk je absolvuje mnohokrát. Proto by bylo dobré vytvořit voliči takové podmínky, aby co nejjednodušeji vyjádřil svůj názor svou volbou a přitom vytvořit i takové podmínky aby zpracování voleb nebylo náročné. Ale nejdůležitější je, aby takovéto hlasování bylo neovlivnitelné z technického hlediska, hlavně při přenosu informací mezi klientem a serverem. Naštěstí v dnešní době už máme dost výkonná zařízení, aby byla možná dostatečně silná šifra, aby dešifrování bez znalosti klíče trvalo roky a nejlépe i staletí. Ale také musí být šifrování a dešifrování možné v reálném čase.

V posledních letech je tato technologie masivněji vylepšována díky úspěchům, které Flash má v aplikacích přehrávání videa, animací a her. Což přineslo spoustu nových možností, které nám Flash nabízí. Hlavní přínos pro tvorbu aplikací je zlepšení výkonu a rychlosti zpracování Action Scriptu. Jedinou nevýhodou oproti například Java, C++ anebo .net C# je že ve Flash nejdou vytvářet uživatelská vlákna pro zpracování dat mimo hlavní smyčku programu. Ale pro účely volebního klienta by to nemělo být překážkou. Nejnáročnější operace, které budou implementovány pro funkčnost klienta, snad nebudou natolik náročné, aby zahltili aplikaci natolik, aby se opožďovalo překreslování okna. Jsou to operace jako, čtení XML jako text a dešifrování na strukturu XML uvnitř Flash, použití digitálního podpisu a nejdůležitější šifrovací algoritmus RSA. Důvodem použití Flash je také že se jedná i o webovou technologii už jen to zaručuje multiplatformnost a to od obyčejného počítače s libovolným operačním systémem přes tablet až k chytřejším mobilním telefonům. S nástupem smartphonů se prostředí Adobe AIR hodně zaměřilo na vývoj aplikací speciálně pro mobilní platformu, takže přibyla spousta funkcí a vylepší výkonu. Takže by volební terminál mohlo být téměř libovolné zařízení podporující Flash v internetovém prohlížeči anebo Adobe AIR.

Cílem práce je vytvořit volebního klienta v jazyce Flash za účelem otestování a rozhodnutí o vhodnosti technologie Flash pro účely volebního klienta.

I. TEORETICKÁ ČÁST

1 FLASH

Flash je komplexní prostředí podporující tvorbu multimedií. Největší využití má v interaktivních počítačových animacích, jejich největší výhodou je že jsou vektorové. Což znamená, že velikost takového souboru s mnoha animacemi a velkou délkou animace je velice malá. Oproti videu, které by se kvalitou obrazu vyrovnalo vektorovému zobrazení, by byla velikost i více než 1000x větší. Další velké využití má ve tvorbě internetových stránek, protože nabízí nejenom tvorbu animací a interaktivních objektů ale také objektový jazyk ActionScript se kterým můžeme dodat spoustu funkcí, anebo také dynamicky generovat obsah a spousta dalších možností.

1.1 Historie

Nejdříve byl kreslicí program SmartSketch, který byl v roce 1995 vylepšen o možnost frame-by-frame animace a přejmenován na FutureSplash Animator. SmartSketch neuspěl sám o sobě u Adobe. V roce 1996 FutureSplash koupila firma Macromedia a spojila jej se svým, podobným, projektem Shockwave. Vzniká tak Shockwave Flash 1.0 (Flash je kombinace slov Future a Splash).



Obr. 1. Původní logo

flash-shockwave

Verze Flash 3.0 byla vylepšena o koncept nazvaný FSCommands, který přinesl jednoduchý způsob implementovat limitovaný počet programovatelných funkcí. Ve verzi 4.0 je po úspěchu FSCommands vytvořena vylepšená verze jazyka nazvaná ActionScript. Snažili se tak prorazit a konkurovat HTML ve tvorbě webových stránek, směřovaly tak vývoj posout více jako programovatelný nástroj pro vývoj webových aplikací. Ve Flash 5.0 je skriptovací jazyk přepracován podle specifikace ECMA, takže měl velmi podobnou strukturu jako JavaScript, a je představena první opravdová verze ActionScript 1.0. Chtěli tak rozšířit svou použitelnost tím že když člověk uměl JavaScript tak se velmi rychle naučil pracovat s ActionScriptem. Ve verzi 6 byly představeny předělané komponenty a nástroje pro video,

kontajner FLV(Flash Video) v dnešní době je nejrozšířenější kontejner streamovaného videa na Internetu. Verze 7.0 je vylepšena o novou objektově orientovanou verzi ActionScript 2.0 a komponenty pro použití zvuku v kódování MP3 a video kontejneru FLV.

V roce 2004 paralelně vznikl aplikační framework Flex (Flash builder), který jako první implementuje ActionScript 3.0, který je pak adaptován i do nové verze původní platformy pro tvorbu Flash. Díky vylepšení ActionScript 3.0 aplikace vytvořené ve Flash běžely jako reálné aplikace spuštěné uživatelem.

V roce 2005 kupuje firma Adobe Systems firmu Macromedia za více než tři miliardy dolarů. Adobe ukončilo a pozměnilo vývoj některých aplikací, které vytvářela Macromedia. Některé Flash implementace, Flash Paper pro publikaci dokumentů do formátu SWF, který ustoupil Acrobatu.

Adobe dobře využilo potenciál platformy. Ta v současnosti je důležitým multimediálním nástrojem ve světě internetu i mimo něj. Denně je používají stovky milionů lidí a většinou ani nemají tušení, že se jedná o Flash.

1.2 ActionScript 3.0

ActionScript 3.0 je objektově orientovaný programovací jazyk uvnitř Flash, je vytvořen podle standardu ECMA, takže se hodně podobá jazykům jako je Java, C#, C++. Pomocí něj dodáváme aplikacím nejen grafickou stránku ale také dynamičnost a interaktivitu. Kdo používal jakýkoliv objektově orientovaný programovací jazyk tak nebude mít problém použít ActionScript.^[13]

1.3 Flash Professional

Flash Professional je základní vývojové prostředí pro tvorbu flash. Je zaměřeno více na práci s grafikou a složitějšími animacemi, ale umí skoro všechno to co Flash Builder neboli Flex. Jednou z výhod pro tvorbu aplikací ve Flash Professional je že se dá nastavit libovolná maximální doba zpracování skriptu. Ve Flex ve starší verzi bylo 15 sekund a nová nabízí 60 sekund, například pro generování RSA klíče o velikosti 2048 bitů je to velice nedostačující. Byla by tak potřeba vyrobit algoritmus sekvenčně což je náročné až nemožné.

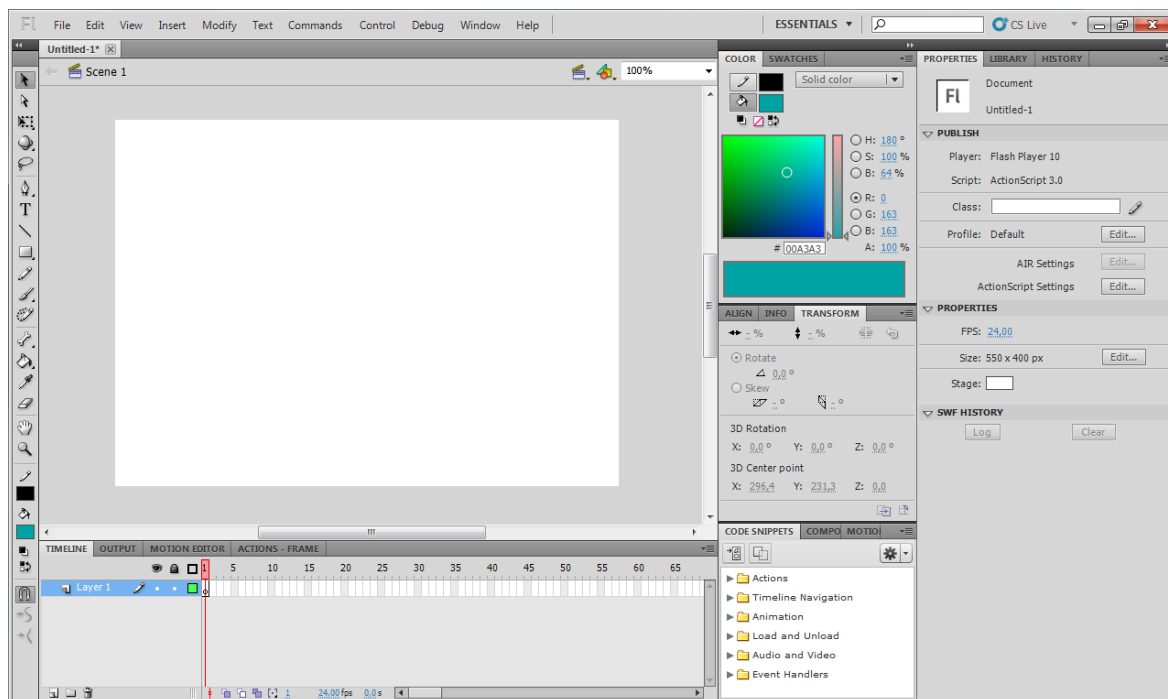
1.3.1 Pracovní prostředí

Úvodní menu nám nabídne základní a nejčastěji používané typy projektů a jiné možnosti. Přednastavené projekty jako jsou projekty typu reklama, animace, bannery, přehrávání medií, prezentace a vzorové projekty. Dále cestu k posledním otevřeným projektům. Také samozřejmě prázdné soubory, se kterými Flash nejčastěji pracuje. A nakonec odkazy na základní výuková videa uložená na stránkách adobe.



Obr. 2. Úvodní menu

Velkou výhodou pracovního plochy je její schopnost přeskupení jednotlivých panelů. Uživatel si tak může poskládat pracovní prostředí tak aby mu co nejvíce vyhovovalo a měl tak optimální workflow.

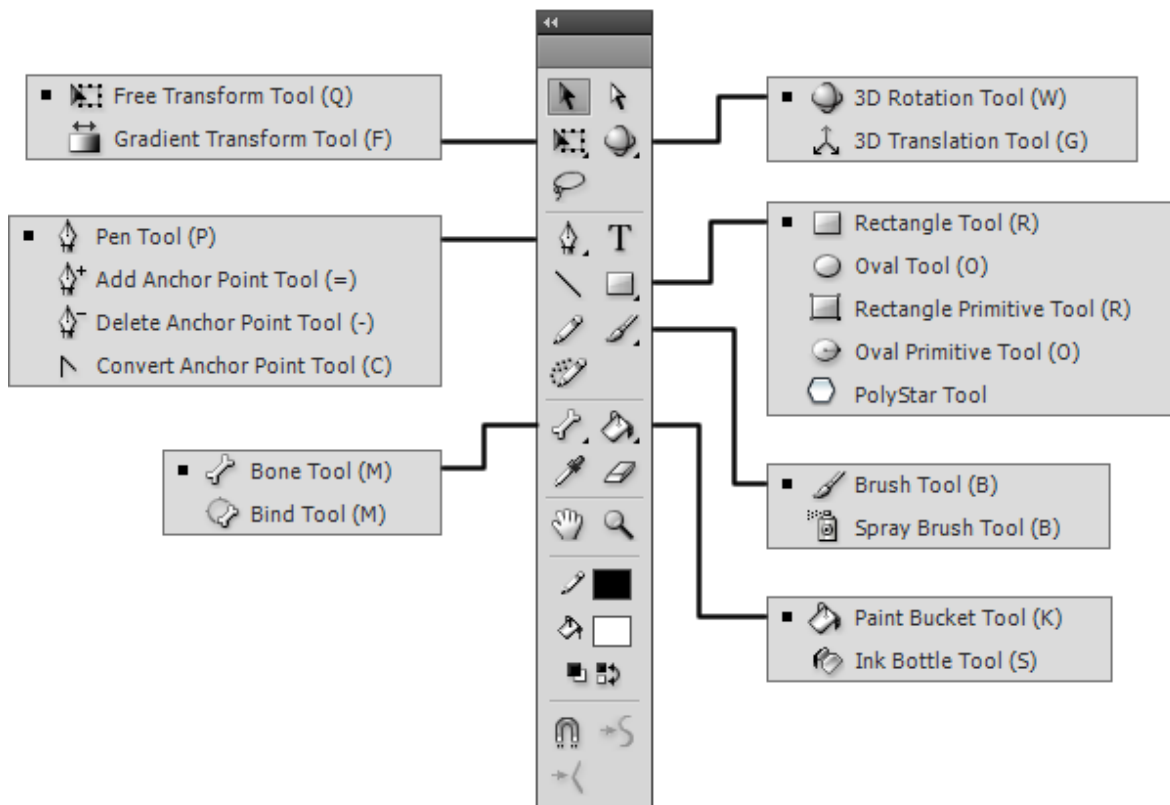


Obr. 3. Pracovní plocha Flash Professional CS5

V horní části okna programu se nacházejí příkazové nabídky a nástroje pro editaci a přidávání prvků do projektu. Prvky se buď vytvářejí přímo ve Flash Professional a nebo importovat z jiných programů Adobe jako je Photoshop, Ilustrátor, After Effects a další.

Flash nám nabízí několik přednastavených pracovních ploch, které mají podle nich ideální rozložení nástrojů. Je 7 přednastavených rozložení pracovní plochy podle účelu, na který zrovna Flash plánujeme použít. Například pro animátory, pro debugování, dizajn a další. Můžeme si také vytvořit vlastní rozložení a uložit si jej abychom tak měli vlastní optimální rozložení pro práci s Flash. Flash Professional obsahuje více než 25 oken, se kterými můžeme libovolně manipulovat a seskupovat je do panelů.^[6]

Velký bílý obdélník v levé části je vymezená pracovní plocha, která bude výstupem projektu, jeho zobrazení se definuje na začátku při vytvoření projektu. Rozlišení pracovní plochy můžeme kdykoliv změnit během práce s projektem. Grafika a objekty mimo tento obdélník nebudou ve výstupu potom vidět, ale může se využít plocha mimo obdélník jako počáteční bod pro objekty, které se animací nebo pomocí Action Scriptu posunou do viditelné oblasti. Pro preciznější vkládání do pracovní plochy se dá využít funkce mřížky a pravítka, které spolupracují s pracovní plochou.



Obr. 4. Panel nástrojů

Panel nástrojů je nejpoužívanější panel ve Flash Professional, obsahuje nástroje, se kterými ovlivňujeme a vytváříme obsah vytvářeného projektu. V horní části jsou nástroje pro výběr a transformace objektů. V druhé části jsou nástroje kreslicí a textové nástroje pro vytváření objektů a grafiky. V třetí části jsou nástroje retušovací a speciální nástroj kosti. Ve čtvrté části jsou navigační nástroje. V páté části je oblast barev. V šesté části je oblast voleb tato část se mění podle toho, který nástroj je aktuálně vybrán.^[3]

1.3.2 Základní objekty

Nezákladnější objekt je Shape je to jednoduchý jednodlitý grafický objekt. Dalším objektem je Graphics. Ten funguje jako kontejner, který zapouzdřuje statické grafické objekty. Nejčastěji se využívá pro animace.

Movie Clip fungují jako kontejner pro grafické objekty a animace. Je to velice výhodné pro znovu využívání těchto objektů jako instancí původního, dá se tak ušetřit velice mnoho místa v celkové velikosti výsledné aplikace.

Button je speciální symbol nemá klasickou timeline ale pouze 4 stavy ve kterých se může tlačítko nacházet. Každý ze stavů automaticky reaguje na eventy myši, které jsou vyvolávané při interakci s objekty. Jsou to stavy Up, Over, Down a Hit. Up je stav tlačítka ve kterém se nachází, když se s ním neprovádí žádná interakce s myši. Over je stav tlačítka ve kterém se nachází, když se nacházíme kurzorem nad aktivní oblastí tlačítka. Down je stav tlačítka ve kterém se nachází, když je tlačítko myši nad aktivní oblastí stlačeno, ale ještě není puštěno. Hit je stav tlačítka, ve kterém se nachází, když na tlačítko klikneme a pustíme.

Classic text je starší verze objektů obsahujících text, jsou 3 základní varianty tohoto objektu. První varianta je Static Text jedná se o text napsaný přímo ve Flash Professional nedá se později nijak upravit. Tomuto objektu nelze ani přiřadit ani jméno instance takže se na něj nedá ani odvolávat pomocí Action Scriptu, to znamená, že se nedá nijak ovlivnit ani jeho pozice což znamená, že text je úplně statický jak napovídá název.

TFL (Text Layout Framework) text je novější vylepšená verze textového pole dostupná až od verze Flash Player 10. Obsahuje mnohem více možností, které můžeme nastavit danému textu uvnitř objektu. Dá se říct, že nám dává naprostou kontrolu nad rozvržením textu. Jednou z nových vlastností je například vytvoření sloupců pro psaní textu přímo v objektu TFLText, není potřeba už používat 2 objekty. A obsahuje také spoustu dalších novinek a vymožeností, které byli převzaty například i z Adobe Illustrator pro úpravu textu.^[1]

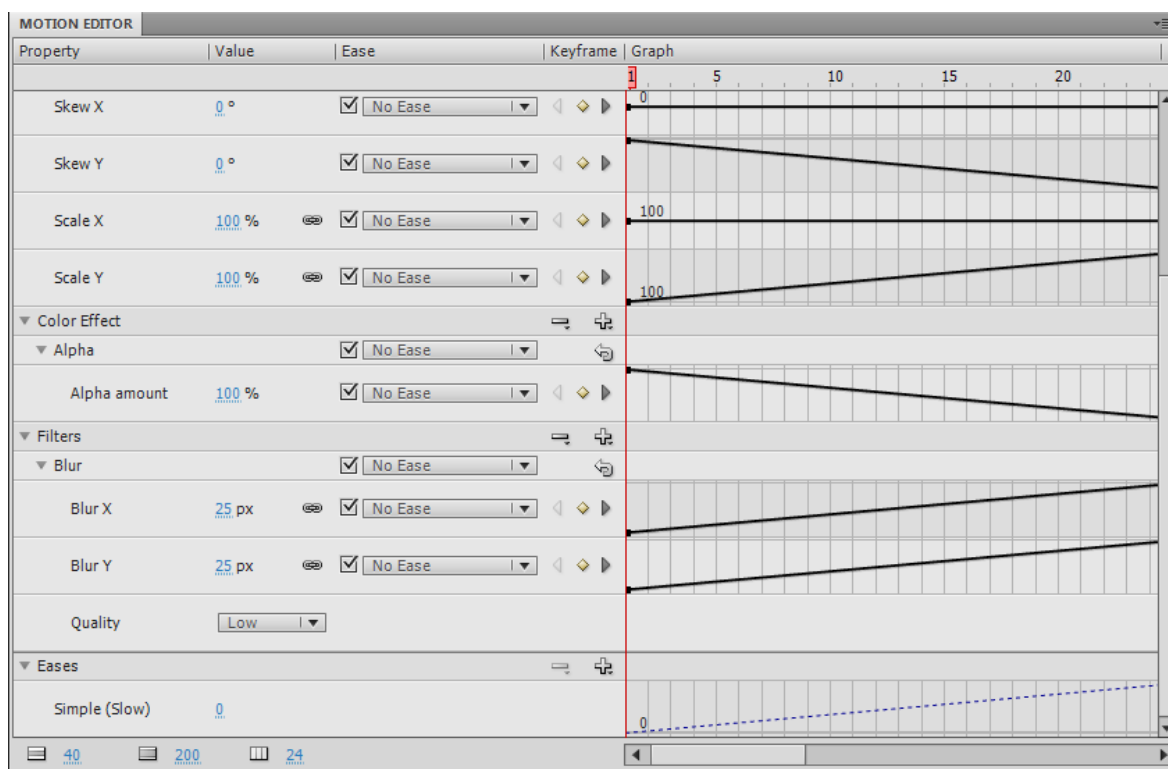
1.3.3 Základní animace

Clasic tween je nejstarší možnost vytváření animací ve Flash Professional. Od něj je odvozen a nově vytvořen novější způsob animování nazvaný Motion tween, který slibuje jednodušší vytváření a plynulejší animace. V podstatě Clasic tween je přechod mezi dvěma stavy stejné instance, což je v podstatě definice animace. Stačí vytvořit dva klíčové snímky. Pravým klikem na první pak vybrat Classic Tween. Aby to fungovalo, musí být v počátečním i koncovém klíčovém snímku instance stejných symbolů, nejde tedy použít čistá grafika jako je shape ale musí to být konvertováno na symbol.

Shape tween funguje podobně jako Classic Tween, jen s tím rozdílem, že v obou klíčových snímcích je jen grafika typu shape, od toho je také odvozen název, Flash pak vykreslí přechod mezi tvary. U složitější grafiky se můžeme setkat s problémem, že při překreslování objektu v čase postupně po snímcích probíhá přeskupování křivek nevhodným způsobem.

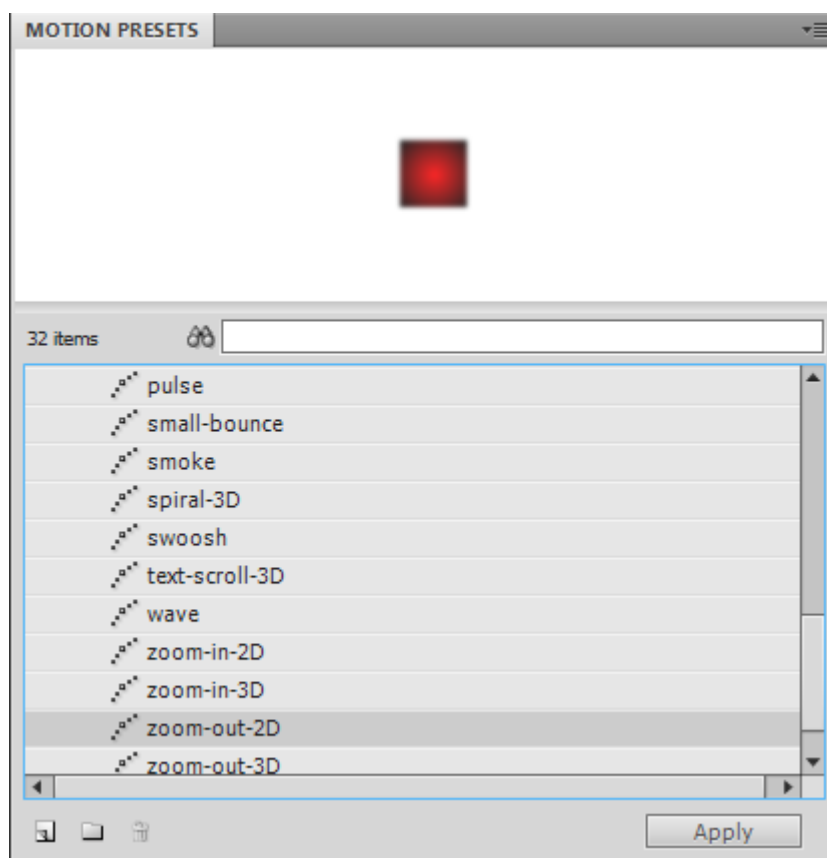
Při použití Shape Tween je možné určit body takzvané Shape Hints. Určují, kde se tento bod má nacházet v průběhu animace, jinak se provádí pouze automatická dedukce a výsledek si bude jen odpovídat v počátečních a koncových tvarech. Body se dají přidat menu v nabídce Modify > Shape > Add Shape Hint. Pak je stačí přesunout na vybrané místo.

Motion tween existuje od Flash Professional CS4. Zpříjemňuje a zjednodušuje práci vytváření animací. Funguje v podstatě stejně jako Classic tween, jen není třeba vytvářet dva klíčové snímky - stačí jen jeden. O vytvoření dalších se Flash postará sám podle toho, jaké se provedou úpravy v jednotlivých snímcích animace. Zároveň se dá i upravit dráha přechodu animovaného objektu stejným způsobem jako se upravuje tvar vektorové křivky. Při vytváření Motion tween můžeme použít pomocný nástroj Motion editor. Ve kterém jsou sjednocené parametry, které můžeme ovlivňovat pomocí Motion tween. Avšak neobsahuje všechny možnosti v základním zobrazení, ale můžeme je jednoduše přidat, jedná se o Color Effects a Filters. Je to proto, že se nedá automaticky očekávat, které z Filters a Color Effects bude uživatel chtít použít, jestli nějaké vůbec použije.^[3]



Obr. 5. Motion Editor

Motion presets jsou přednastavené animace, které se dají aplikovat na objekty na scéně. Motion presets jsou ve své podstatě XML soubory ve kterých je uložen průběh hodnot měnících se v čase. Tím pádem máme k dispozici nástroj, ve kterém můžeme vytvářet předlohy animací, které pak můžeme aplikovat na různé objekty. Také se dají jednoduše převést i do jiného projektu. Parametry animace vložené pomocí Motion presets se můžou libovolně upravovat po aplikování na některý objekt, neovlivní to přednastavený originál, který byl použit.



Obr. 6. Motion Presets

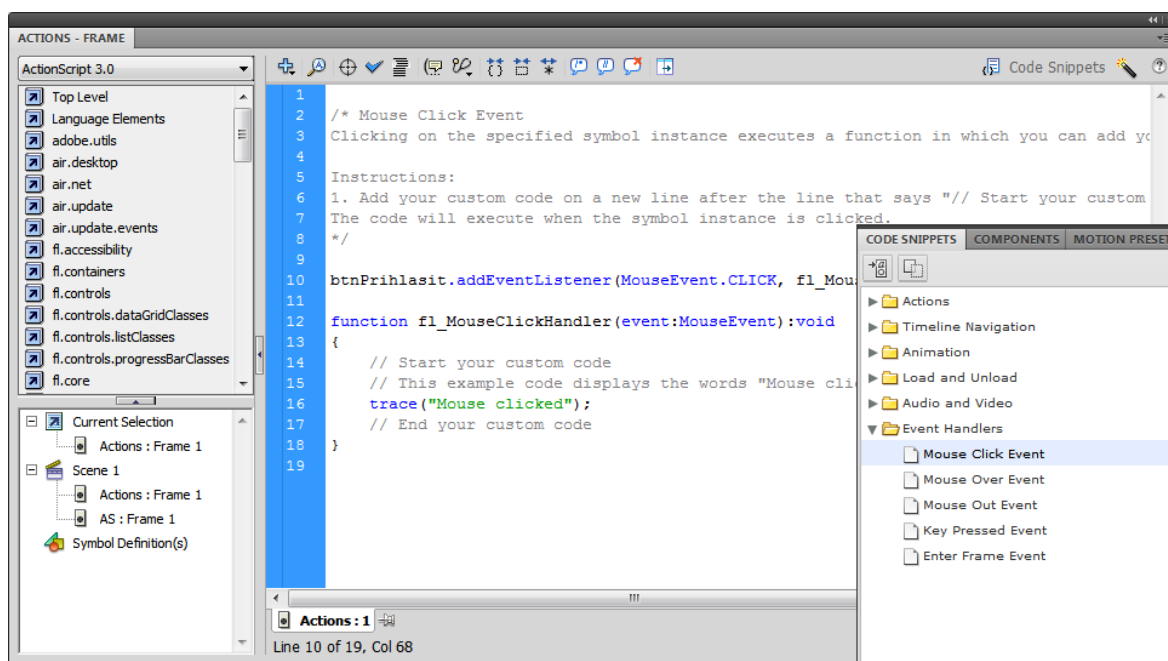
1.3.4 Action Script ve Flash Professional

Action Script se ve Flash Professional píše v Actions Frame která se váže k určitému klíčovému snímku v časové ose. Tento kód se spouští když se aplikace ocitne na daném snímku. Takže se musí dávat pozor na to, kam se daný kód napíše. Mohlo by se stát, že se tak spuštěný kód, bude odkazovat na objekt, který už v daném čase už nebo již neexistuje. To by vedlo k pádu celé aplikace. Stejná pozor je si dávat i při používání eventů, navíc ještě

se musí dát pozor, aby na smazaný nebo neexistující objekt nebyl navázán event. Tohle jsou hlavní věci, na které se musí dát pozor uživatelé začínající s Flash Professional.

Ve Flash Professional CS5 se objevila novinka jménem Code Snippets, která může pomoci s pochopením méně zkušeným uživatelům, jak se využívá ActionScript ve Flash Professional a nejzkušenějším uživatelům ušetřit čas doplněním základu kódu který si pak libovolně upraví. Obsahuje taky textovou nápovědu a návod, který uplatní hlavně méně zkušené uživatelé.

Takzvané Snippets obsahují předvyplněné funkce například pro pohyb na časové ose, pro jednoduché animace, nahrávání externích dokumentů, práce se zvukem a videem, zpracování základních eventů a další.^[3]



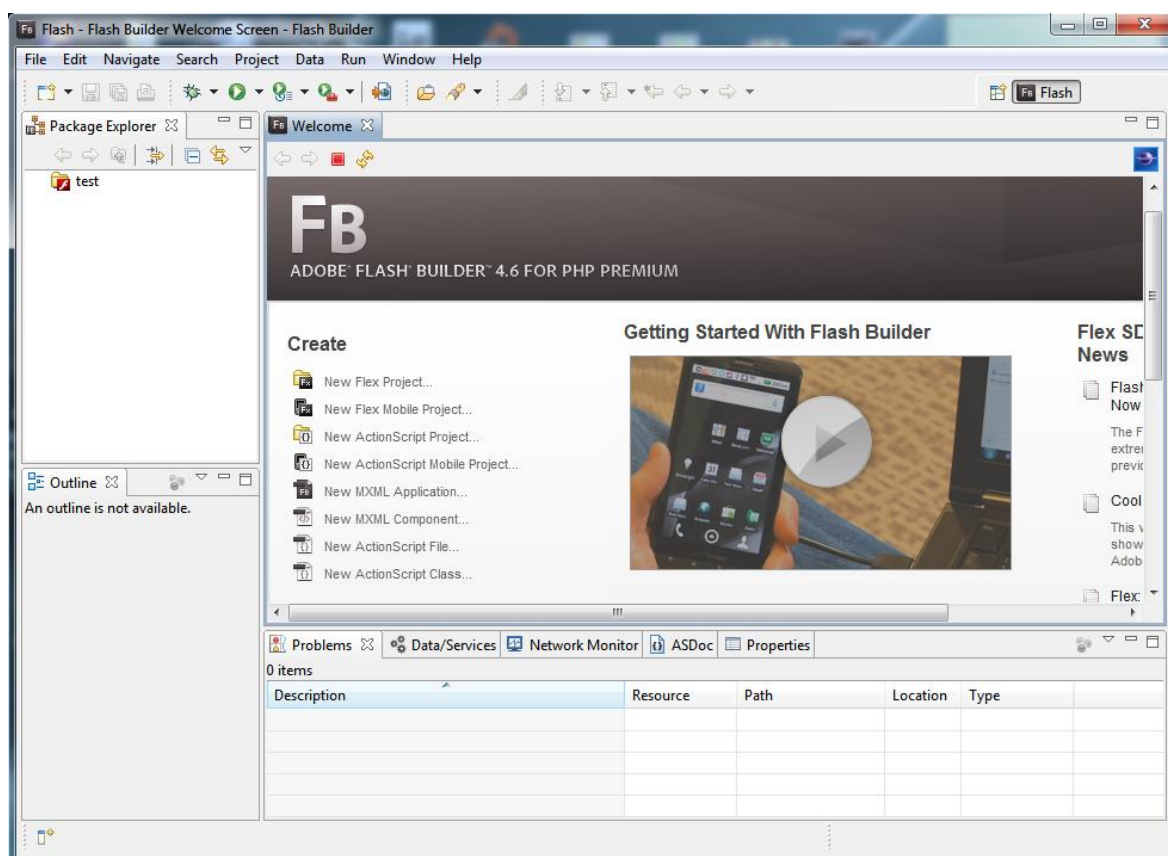
Obr. 7. Code Snippets a Actions Frame

1.4 Flash Builder

Flash Builder je nástroj pro vytváření Flash ale je postaven na jiném principu zvaném Flex je to protože se vytváření obsahu nedá srovnat s prací ve Flash Professional. Ve Flash Builder pracujeme s MXML (Macromedia eXtensible Markup Language), díky tomu že se jedná o nadstavbu XML formátu práce s MXML připomíná novější verze HTML formátu. Dalo by

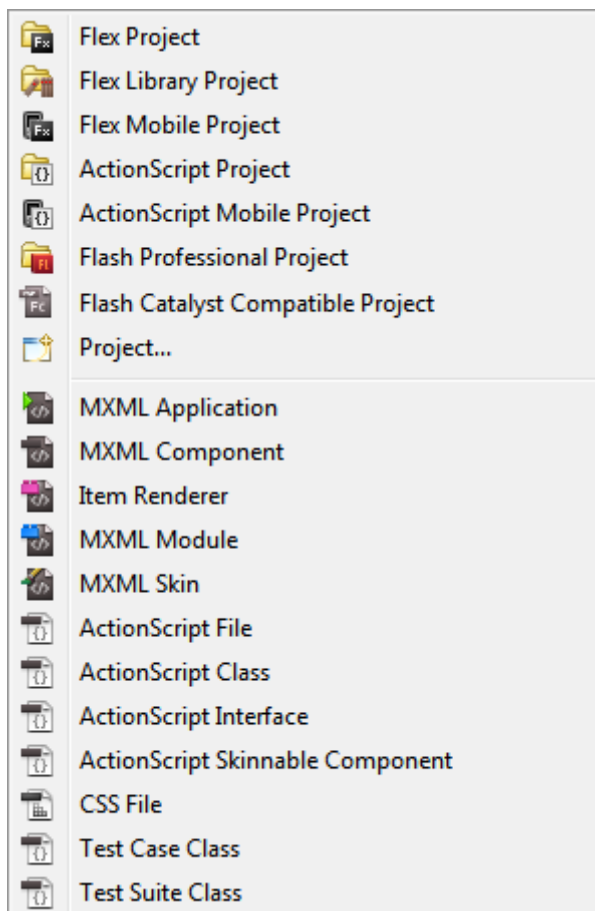
se říci, že je velká podobnost při použití MXML a Action Scriptu s HTML a JavaScriptem. Napovídá tomu i fakt že Action Script svou strukturou je velice podobný JavaScriptu protože chtěli, aby weboví vývojáři měli co nejjednodušší přechod na platformu Flash pokud již umí HTML a JavaScript.^[7]

Pokud pracujeme s Flash Builderem tak pracujeme s Adobe Flex což je v podstatě také Flash ale jde o to na jaké věci je zaměřena. Flex je zaměřen na RIA (Rich Internet Applications) což je odpoutání od původního záměru který měla přinést platforma Flash proto je zde název Flex. Flash a Flex využívají ke svému spuštění Flash Player a nebo Adobe AIR takže se ve své podstatě jedná o Flash pouze je vytvořen jinak. I když vytváříme obsah v MXML tak se při kompilaci všechen kód mění na Action Script.^[10]



Obr. 8. Flash Builder

Flash Builder je postaven jako nástavba prostředí Eclipse takže většina lidí jej může znát už z praxe s jazykem Java, PHP, C++ nebo i jinými. I když je prostředí univerzální nijak to nevede samotné tvorbě. Máme zde možnost pracovat přímo s kódem MXML což je samozřejmě ale i grafické zobrazení pro práci s objekty, které dokáže generovat určité části kódu a usnadnit tak práci uživatelům.



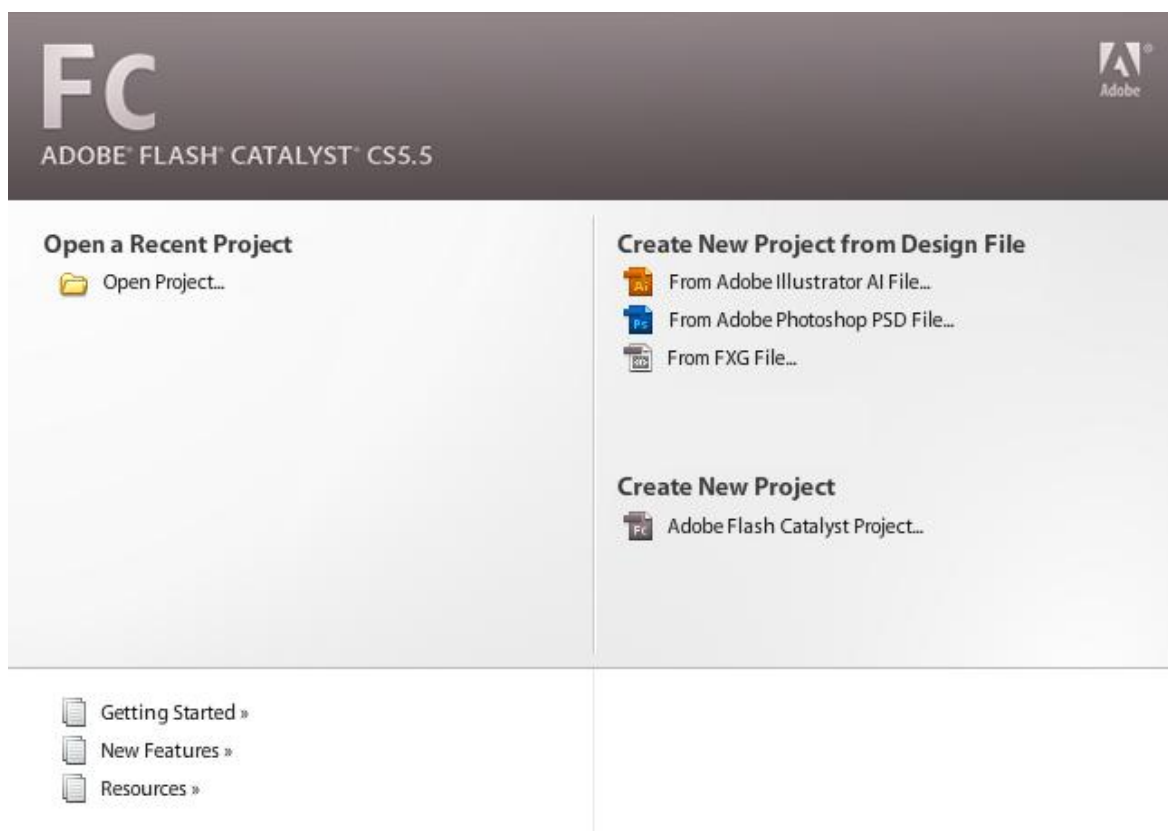
Obr. 9. Flash Builder možnosti

Flash Builder má velkou škálu souborů se kterými může pracovat. Některé možnosti nejsou pouze na projekty vytvářené kompletně ve Flash Builderu ale i pro Flash Professional protože nemá tak dobře implementované prostředí pro psaní Action Scriptu jako je ve Flash Builderu. Další možnost je projekt v kombinaci s Flash Catalyst což je velmi výhodné když chceme vytvořit projekt s náročnějším grafickým prostředím.

V současné době jsou dva druhy MXML komponent, starší jsou nazvané MX a novější Spark. Adobe se snaží postupně o úpravu všech vizuálních komponent ze staršího MX na novější Spark aby byla oddělená data a chování komponenty od její vizuální reprezentace. Dobrá věc je že lze libovolně v projektu používat jak komponenty MX tak Spark anebo jen jednu z nich záleží na tom, jakou aplikaci chceme vytvořit.^[4]

1.5 Flash Catalyst

Flash Catalyst se může brát jako podpůrný program pro jednodušší výrobu grafické části aplikace vytvářené ve Flash Builder a nebo jako samostatný program pro výrobu statictějších aplikací. Je to díky přepracování komponent v MXML tak že se může jednotlivým komponentám přiřadit vlastní skin a tím upravit jeho vzhled. Stejně jako můžeme vytvářet nové vlastní komponenty, se kterými pak budeme pracovat v našem projektu.

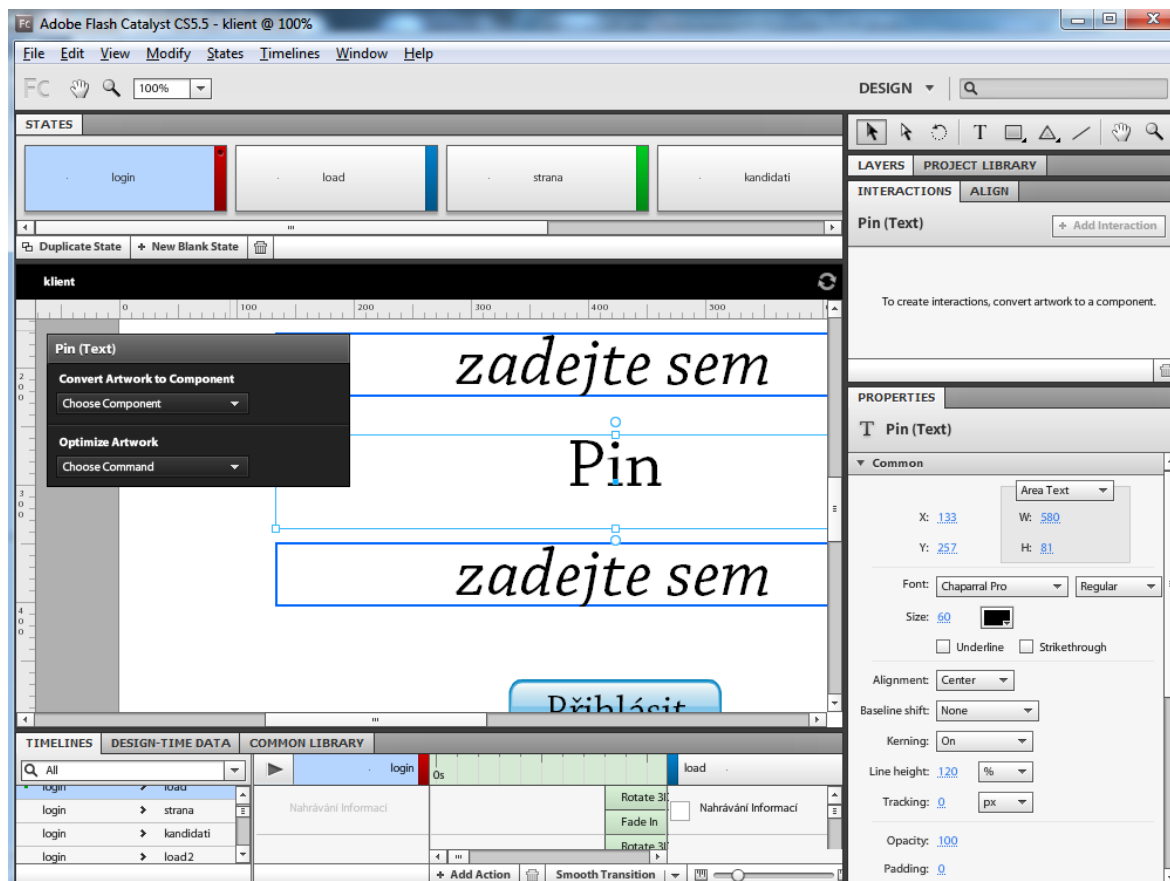


Obr. 10. Flash Catalyst úvodní menu

Velikou výhodou je integrace grafických formátů pro tvorbu aplikace ve Flash Catalyst. Formátu souborů FXG (Flash XML Graphics) je přímo vytvořen pro nejlepší možnou interpretaci a integraci grafických objektů. Anebo lze použít přímo soubor Adobe Illustratoru a nebo Adobe Photoshop a z něj vytvořit Flash Catalyst Projekt.

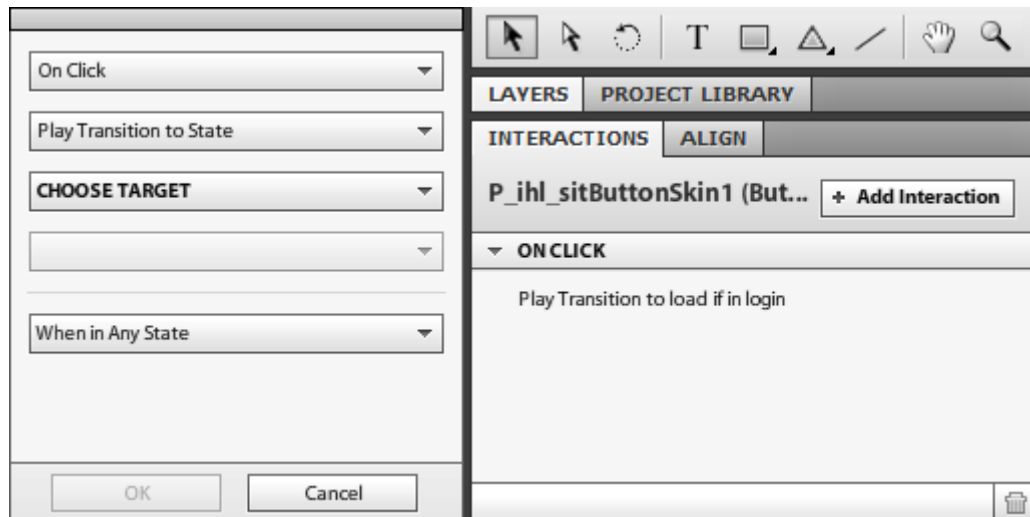
Flash Catalyst je zatím jen ve verzi 1.5, poprvé byl vydán teprve s Adobe CS4 je teda pouze na počátku svého vývoje. Jedinou nevýhodou je pouze omezené možnosti tvorby grafiky uvnitř Adobe Catalyst. Ale věřím, že v budoucnu budou v dalších verzích změny k lepší jednodušší a rychlejší tvorbě komponent a kompletních aplikací.

Velkou výhodou zejména pro práci v týmu kde spolupracuje grafik a programátor. Grafik v aplikaci Flash Catalyst nemusí psát žádný kód a přitom jej vytváří a s ním pak pracuje programátor.



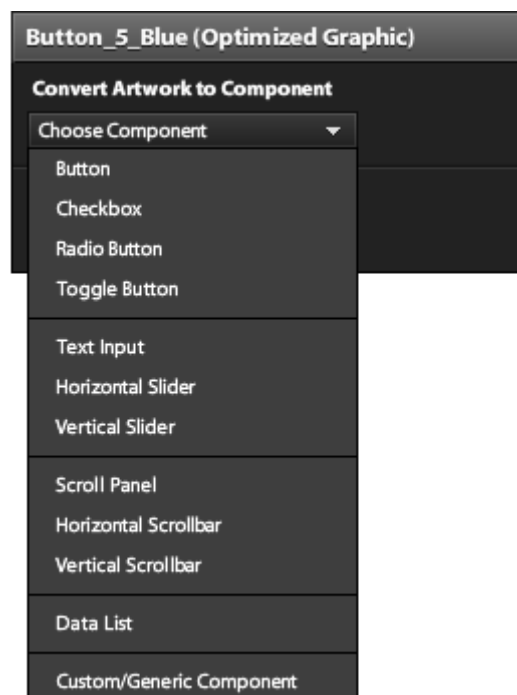
Obr. 11. Pracovní prostředí Flash Catalyst

Pracovní prostředí je velmi jednoduché a těm co už pracovali s Flash Builderem nebude ani potřeba vysvětlovat jednotlivé nástroje. Protože dostal do ruky nástroj, který mu umožní jednoduše vytvářít, to co by ve Flash Builderu musel vytvářet psaním kódu a vytvořením všech stavu jednotlivé komponenty. A také zde jdou jednoduše vytvářet přechody mezi jednotlivými stavy. Dokonce lze nastavit funkčnost tlačítek ale pouze jednoduché úkony jako je přechod mezi stavy aplikace nebo funkce pro přehrávání videa a další. A také můžeme vybrat event, na který se má naslouchat, takže můžeme rozlišit například kliknutí pravého, levého nebo prostředního tlačítka. Anebo pouze stlačení ale ne puštění například pro funkce drag and drop anebo jestli se jedná o dvojklik a další. Vše je jednoduše zobrazeno v nabídce, takže není potřeba nic programovat a všechny interakce jsou zařízeny pouze vybráním položky v nabídce.^[12]



Obr. 12. Panel Interactions

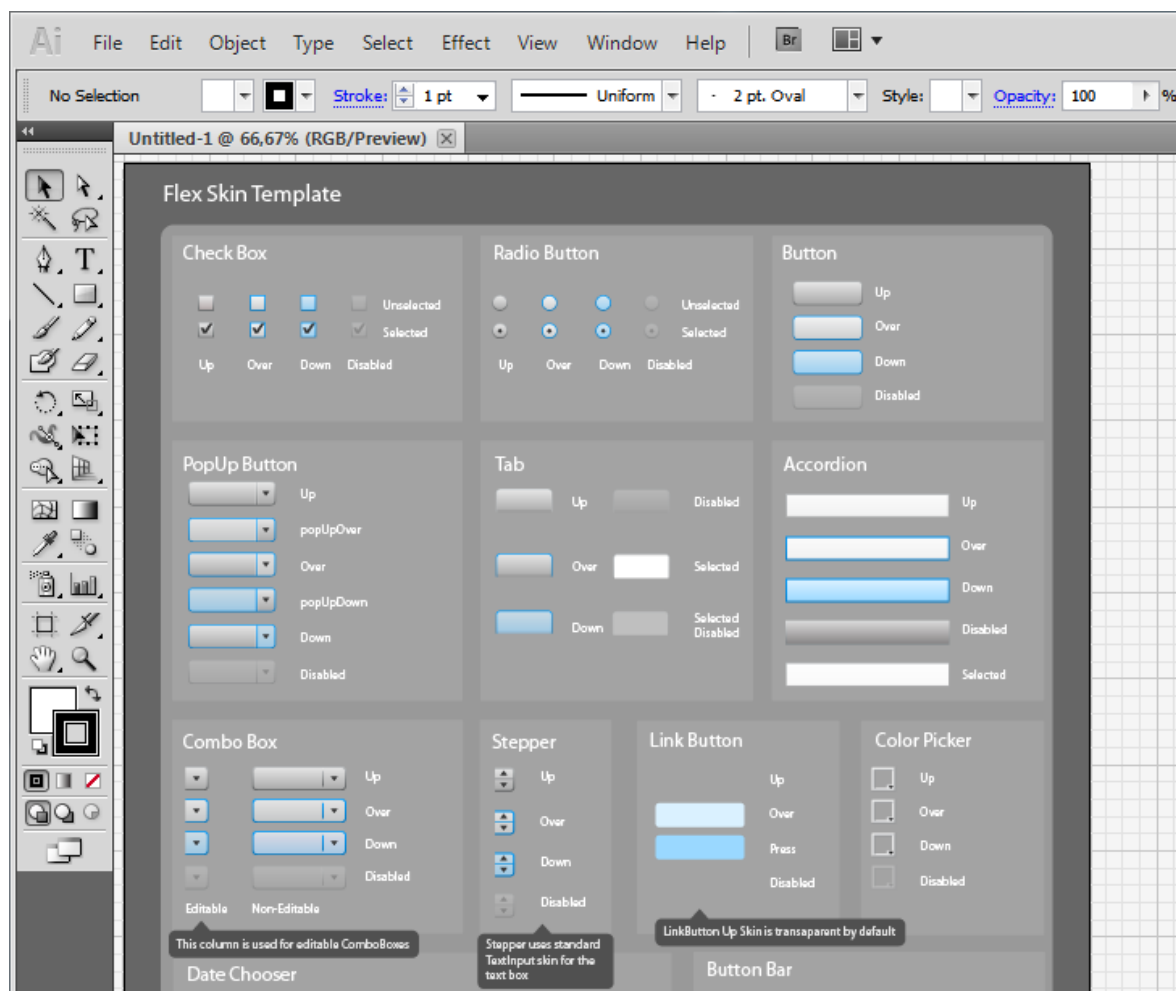
Nabídka Convert je nejdůležitější pro převod grafiky na jednotlivé komponenty. Je na výběr možnost udělat například základní věci jako je například tlačítko. Některé komponenty jako je například Horizontal a Vertical Slider vyžadují, abychom specifikovali určité části dané komponenty. Jednoduše pouze výběrem daného kusu grafiky a potvrzením že se jedná například o táhlo anebo o tlačítka posunutí. Je to výhoda protože se všechno ostatní jako je vygenerování eventů a podobné věci zajišťující funkčnost jsou převzaty z původního vzoru a námi vytvořená grafika je aplikována jako skin dané komponenty.^[5]



Obr. 13. Nabídka Convert

1.6 Spolupráce programů Adobe

Adobe Ilustrátor je díky spojení s Flash Catalyst výborný nástroj pro tvorbu uživatelských komponent a celkového uživatelského prostředí. Dokonce má i přednastavené šablony, které nám umožní rychleji přetvořit komponenty k obrazu svému. Odstraní se tak problémy se zjišťováním velikosti původních komponent a do jakých stavů se daný objekt může dostat, všechno je přehledně zobrazeno na pracovní ploše.



Obr. 14. Adobe Illustrator template

Lze použít i Adobe Photoshop pro vytváření grafického obsahu pro Flex ale dá to více práce než v Adobe Ilustrátoru. Musíme také myslet na to, že v Adobe Photoshop vytváříme rastrovou grafiku, které zabírá více místa a není tak nejvhodnější pro web.

Výborný program pro spolupráci s Flash je Adobe Aftereffects. Dají se tam dělat velmi složité vektorové animace například s textem a spousta další funkcí. Také jeho primárním výstupem je swf soubor.

2 VOLBY

V dálkových elektronických volebních nebo hlasovacích systémech množina autorizovaných voličů vhazuje své hlasy nebo názory z daného hlasovacího zařízení do virtuálního hlasovacího centra skrze komunikační síť. Server nebo množina serverů sestavuje virtuální hlasovací stanici a je zodpovědný za přijímání spojení od voličů a zaznamenání jejich hlasů či názorů vydaných pro jejich další počítání a tabelování. Po tom co jsou tabelovány, výsledky hlasování jsou nakonec zveřejněny, nejčastěji pomocí komunikační sítě samotné.

2.1 Požadavky elektronických voleb

1. Přihlášení voliče – volič se přihlásí za využití subsystému volebního klienta zadáním jednoznačného identifikátoru, který mu je přidělen, do příslušného pole.
2. Zobrazení hlasovacích lístků – autorizovanému voliči se zobrazí elektronický hlasovací lístek. Tento bude čerpán ze subsystému databáze hlasovacích lístků.
3. Zobrazení kandidátů na lístku – tento případ užití navazuje na výběr hlasovacího lístku. Jeho funkcí je naplnit vybraný hlasovací lístek kandidáty a zobrazit je.
4. Úprava hlasovacího lístku – volič provede úpravu hlasovacího lístku za využití standardních periférií – myš, klávesnice nebo za pomoci dotykové obrazovky.
5. Uložení hlasovacího lístku – po úpravě hlasovacího lístku se zobrazí finální náhled hlasu a po odsouhlasení se tento hlasovací lístek uloží.
6. Validace hlasovacího lístku – validace upraveného hlasovacího lístku se provádí před jeho uložením do subsystému databáze hlasovacích lístků.
7. Uložení hlasovacího lístku – lístek je zabezpečen, podepsán a uložen do subsystému databáze hlasovacích lístků.

2.2 Elektronický hlasovací lístek

Elektronický hlasovací lístek je založen na principu tzv. dvoustupňového výběru a je využitelný pro volební systémy poměrného typu s realizací preferenčních hlasů. Lístek se zobrazuje podle příslušnosti voliče do volebního obvodu a to nejprve v podobě nabídky kandidujících subjektů a následně ve formě seznamu kandidátů na příslušné kandidátní listině.

Volič v první fázi volí kandidující stranu či hnutí. Výběr provádí kliknutím polohovacího zařízení. Ve druhé fázi pak označuje 0 až maximálně stanovený počet preferovaných kandidátů. Po kontrole proběhne zabezpečení hlasu a jeho vhození do databáze hlasů.

2.3 Klíčový popis schématu

distribuované volební schéma lze popsat pomocí následujících klíčových bodů:

- 1) Založení elektronického volebního seznamu. Princip národního registru obyvatel doplněný o registr trvalých pobytů, případně zcela nezávislý jednorázová volební seznam, vytvořený v časovém předstihu organizace hlasování.
- 2) Generování transakčního čísla a PINu. Jednorázový volební seznam obsahuje pouze tyto anonymizované transakční čísla a PINy.
- 3) Export a uložení transakčního čísla a PINu do elektronického volebního seznamu.
- 4) Transakční číslo se bude využívat jako identifikace v seznamu hlasujících.
- 5) Transakční číslo a PIN bude hlasující využívat pro přístup k samotnému volebnímu klientu. Voličům bude doručeno na adresu trvalého bydliště v zabezpečené obálce.
- 6) Autorizace oprávněnosti voliče k účasti ve volbách.
- 7) Hlasujícímu bude zobrazen personalizovaný hlasovací lístek – viz elektronický hlasovací lístek.
- 8) Stažení veřejného klíče volební komise a šifrování hlasu – tj. příprava pro přenos.
- 9) Přidání unikátního údaje k řetězci. „Údaj“ zná jen volič a aplikace (např. transakční číslo). Vede k nemožnosti podvržení hlasu.
- 10) Vytvoření digitálního otisku, pomocí hashovacího lagoritmu.
- 11) Případně generování páru klíčů a podpis HASH. Odeslání šifrovaného hlasu a HASH (a případně veřejný klíč k ověření podpisu) na server.
- 12) Po uzavření voleb bude provedena kontrola hlasujících ve volební místnosti a v případě hlasování ve volební místnosti, budou příslušné elektronické hlasy anulovány.
- 13) Pokud volič hlasoval ve volební místnosti, nebude možné již hlasování elektronické. Z databáze se odstraní hlasy těch hlasujících, kteří se dostavili k volbám fyzicky – dle registru oprávněných účastníků.
- 14) Ověření integrity hlasu, respektive jeho započítání do výsledků v elektronické fázi.
- 15) Dešifrování hlasu před samotným sčítáním. ^[14]

3 BEZPEČNOST

Bezpečnost je velmi důležitý faktor voleb, volby musí být neovlivnitelné, jinak by ztratili význam a účel. Chtěl bych se jen v krátkosti zastavit nad technologiemi, které využiji pro komunikaci mezi klientem a serverem. Je velmi důležité šifrovat obsah na straně klienta ve Flash, proto například nelze použít samotné PHP u kterého se odesílají čistá data a šifrování by proběhlo na straně serveru. Mohlo by se využít u PHP pro přenos HTTPS protokol ale to by se dalo považovat jako nedostatečné zabezpečení.

3.1 RSA

Algoritmus RSA publikovali v roce 1978 Ronald Rivest, Adi Shamir a Leonard Adleman, z prvních písmen jejich příjmení vznikl taky název této šifry. Jedná se o asymetrickou šifru, to znamená, že se používá jeden klíč pro šifrování a jiný klíč pro dešifrování. To znamená, že to co zašifrujete pomocí šifrovacího klíče, dá se rozšifrovat pouze pomocí dešifrovacího a naopak.

Šifra je založena na jednoduchém matematickém principu.

- $c = m^e \bmod N$
- $m = c^d \bmod N$

Kdy zašifrujeme m na c a z c potom můžeme zpět vypočítat m .

Postup vytváření veřejného a soukromého klíče

1. nejprve náhodně vygenerujeme dvě velká prvočísla
2. vypočteme číslo N a číslo $\Phi(N)$
 - číslo N je součin dvou náhodně zvolených prvočísel p a $q \dots N = p \cdot q$
 - $\Phi(N)$ je Eulerova funkce určující počet přirozených čísel nesoudělných s N a menších než $N \dots \Phi(N) = (p-1) \cdot (q-1)$; v praxi lze toto číslo nahradit nejmenším společným násobkem čísel $p-1$ a $q-1 \dots L = \text{NSN}(p-1, q-1)$
3. zvolíme náhodné číslo e (šifrovací exponent), kde $1 < e < \Phi(N)$, tak, že největší společný dělitel $\text{NSD}(e, \Phi(N)) = 1$ (tj. e a $\Phi(N)$ jsou nesoudělná)

4. užitím Eukleidova algoritmu vypočteme definované číslo d (dešifrovací exponent) takové, že:

- $1 < d < \Phi(N)$
- $e \cdot d = 1 \pmod{\Phi(N)}$, nebo také $d = e^{-1} \pmod{((p-1)(q-1))}$
Existence čísla d je dána Bautzovou větou.^[9]

Bezpečnost této šifry je založená na problému zvaném faktorizace prvočísel který je neřešitelný pro velká čísla v reálném čase.

3.2 SHA-2

Secure Hash Algorithm – 2 existuje více variant, v tomto případě bude využita verze SHA-256. Algoritmus pracuje pouze jednosměrně, máme-li HASH a dokument, mělo by být velmi obtížné vytvořit jiný dokument se stejnou HASH. Důležitý faktor je, že při změně jednoho bitu se musí změnit celý výsledný HASH. Tomuto aspektu HASH Funkcí se říká bezkoliznost.

Algoritmus u funkcí SHA-224 a SHA-256 pracuje s kontextem 256 bitu. Ten se rozdělí na osm 32 bitových slov A, B, C, D, E, F, G a H. V kompresní funkci pak zpracovává bloky dat o velikosti 512 bitu, pomocí kterých modifikuje obsah jednotlivých slov. Každý blok výpočtu se pak skládá z 64 operací založených na operacích +, and, or, xor, shr a rotr. Velikost vstupních dat může mít délku až 263 bitu.^[8]

3.3 HTTPS

Hypertext Transfer Protocol Secure používá protokol HTTP, ale přenášená data jsou šifrována pomocí SSL/TLS. HTTPS umožňuje zabezpečit šifrováním spojení mezi webovým prohlížečem a webovým serverem před odposloucháváním a podvržením dat. Standardní port pro HTTPS je, 443 kdežto HTTP používá port 80.

Pro zabezpečení pomocí protokolu SSL/TLS jsou potřeba digitální certifikáty. Certifikáty jsou obecně vydávány k různým účelům (např. podpis emailů, podpis domény počítače, atd.). Každý certifikát je vydán tzv. certifikační autoritou, která ručí za pravost certifikátu např. THAWTE, VeriSign, PostSignum. Existuje však možnost vytvoření certifikátu, který si vydavatel sám sobě podepíše tzv. self-signed certificate za použití vlastní certifikační autority.

II. PRAKTICKÁ ČÁST

4 POŽADAVKY

Požadavky na elektronické volby je možné rozdělit do tří základních skupin.

Funkční

- Možnost přihlášení – základní funkce autentizace voliče ve volebním seznamu
- Síťová komunikace – základní funkce pro architekturu server-klient
- Čtení z externího elektronického volebního seznamu – čtení seznamu stran a kandidátů určených pro daného voliče z hlediska volebního okresu
- Možnost vyplnění volebního lístku – základní funkce pro realizaci voleb
- Uložení vyplněného volebního lístku externě – uložení pro další zpracování serverem
- Možnost kryptografie na straně klienta – důležitý aspekt je bezpečnost přenosu dat aby nemohli být volby ovlivněny
- Možnost Hash funkce – digitální podpis dat odesílaných klientu serverem
- Ověření integrity hlasu – potvrzení uložení dat na serveru

Technické požadavky na Flash

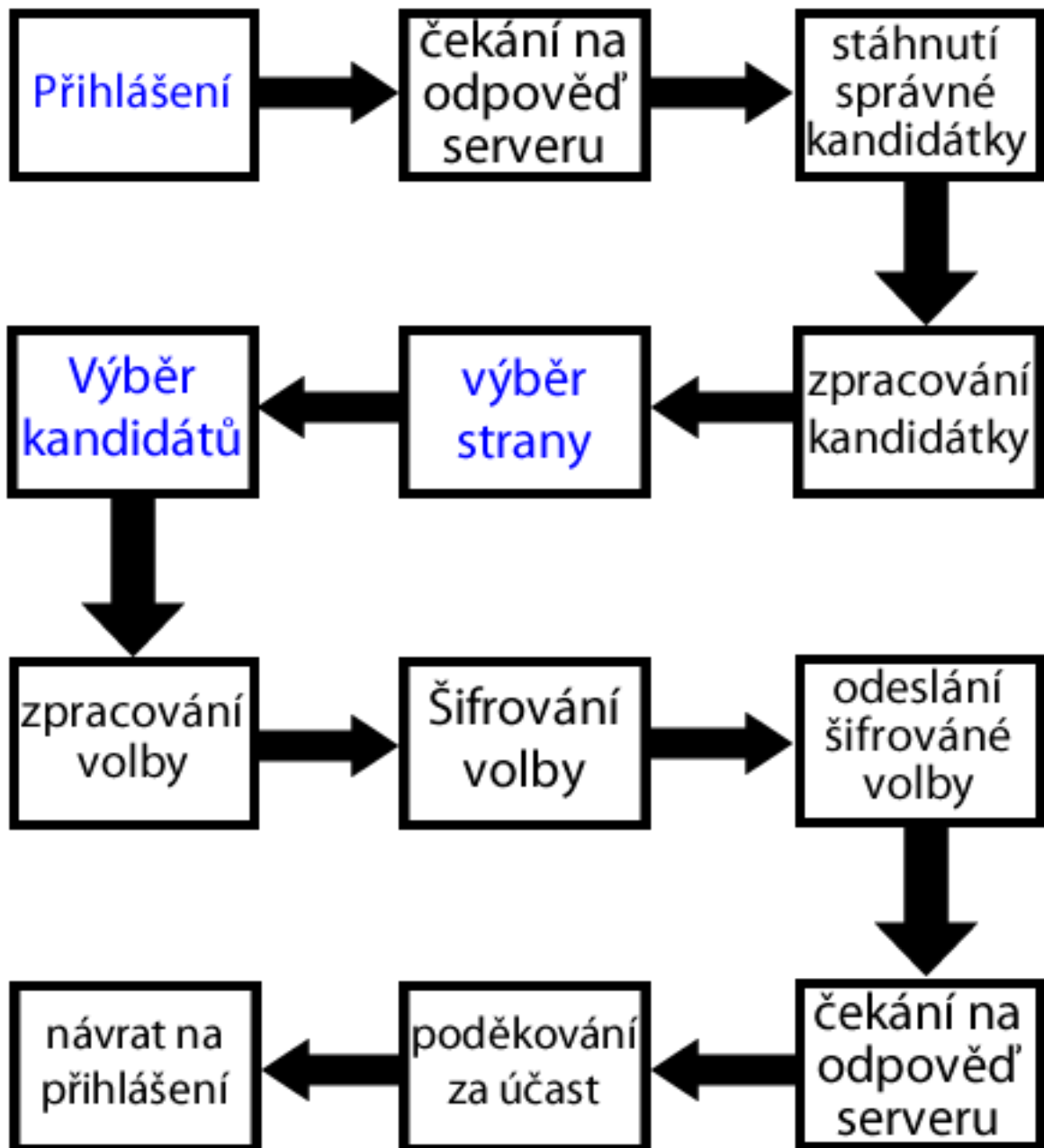
- HTTP/HTTPS – nebo jiný protokol pro komunikaci přes veřejnou síť internet
- RSA šifrování – možnost složitých matematických operací a práce s velkými čísly
- SHA - digitální podpis – jednodušší matematické funkce zajistí rychlejší dešifrování na straně klienta
- XML – pro univerzální formát předávaných dat pro využití jiných technologií pro server
- Uživatelský vstup – základní funkce pro přihlášení a vyplnění volebního lístku
- Výběr z kandidátky – vyplnění volebního lístku elektronicky

Grafické

- Jednoduché – aby nijak nerušilo voliče od voleb
- Intuitivní – nemělo by mít složité ovládání kvůli technicky méně nadaným voličům
- Nepropagující jakoukoliv z volených stran – nestrannost by měl být důležitý faktor

4.1 Návrh

Je důležité si vytvořit návrh toho, co budeme vytvářet. Důležitá je sekvence operací, která se bude muset provést, aby volič byl schopný provést volbu. Další důležitá věc je vytvořit návrh obrazovek, se kterými se uživatel setká při použití aplikace.



Obr. 15. Sekvence událostí

Modrou barvou jsou zobrazeny úkony, které fyzicky provádí uživatel. Černou barvou jsou kroky, které provedou na pozadí bez interakce uživatele. Ale je to jen návrh, takže je možné, že finální produkt bude rozšířen o další funkce. Je viditelné, že klient by měl být co

nejjednodušší pro uživatele na použití. Přeci jen ne všichni kdo jdou k volbám, jsou technicky natolik zdatní, aby intuitivně dokázali ovládat složitější aplikaci.

The image displays six wireframe panels arranged in a 3x2 grid, illustrating a user interface design for a voting application. Each panel is numbered and contains specific UI elements:

- Panel 1:** Labeled '1', it features the text 'Transakční číslo' above a text input field, 'PIN' above another text input field, and a 'login' button at the bottom.
- Panel 2:** Labeled '2', it contains the text 'Loading' centered on the screen.
- Panel 3:** Labeled '3', it is titled 'Výběr strany' and shows a 'radio buttony' option. A 'next' button is located at the bottom right.
- Panel 4:** Labeled '4', it is titled 'Výběr kandidátů' and shows a 'checkboxy' option. 'zpět' and 'odeslat' buttons are positioned at the bottom.
- Panel 5:** Labeled '5', it contains the text 'Zpracovávám' centered on the screen.
- Panel 6:** Labeled '6', it contains the text 'Nashledanou' centered on the screen.

Obr. 16. Předběžný návrh obrazovek

Na obrazovce č. 1 by měly být pole pro zadání PINu a transakčního čísla a tlačítko, které spustí odeslání dat na server. Na obrazovce č. 2 je pouze informativní obrazovka aby volič věděl, že se něco děje na pozadí. Na obrazovce č. 3 jsou zobrazené strany, ze kterých si volič musí vybrat právě jednu proto je tam znázorněný radio button a tlačítko pro přesun na obrazovku výběru kandidátů dané strany. Na obrazovce č. 4 je obrazovka pro výběr kandidátů, kandidátů musí být možno vybrat libovolný počet, proto je tam znázorněn check box. Dále tam také musí být tlačítko pro vrácení na výběr strany, kdyby se náhodou volič rozmyslel, a tlačítko pro odeslání volby na server. Ještě by se mělo po stisknutí tlačítka odeslat objevit upozornění, jestli si je volič svou volbou opravdu jist aby nedošlo náhodou k tomu, že by ukončil a odeslal volbu omylem předčasně. Na obrazovce č. 5 je pouze informativní obrazovka aby volič věděl, že se jeho volba odesílá na server. Na obrazovce č. 6 je poděkování za provedenou volbu a rozloučení s voličem.

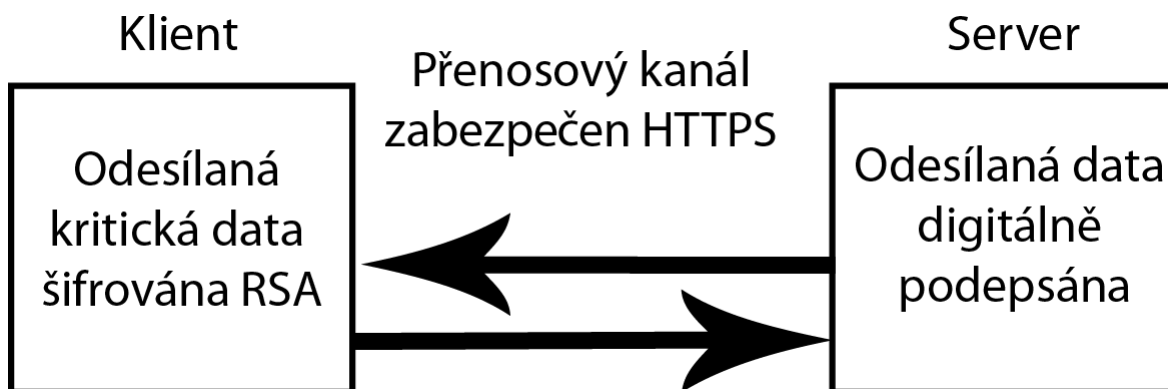
4.2 Zabezpečení hlasovacího procesu

Jako první stupeň zabezpečení je, že celý proces předávání dat mezi voličem a serverem je zapouzdřeno a zabezpečeno pomocí HTTPS, i samotné stažení klienta ze serveru je zabezpečeno pomocí HTTPS. Využívá se například při internetovém bankovníctví, což je proces s vysokým nárokem na bezpečnost, proto je výhodné jej využít i pro volebního klienta. Protokol HTTPS je standardní protokol a Flash jej dokáže využít bez nutnosti složité implementace což je velkou výhodou pro implementaci do volebního klienta.

Druhým stupněm je zabezpečení dat při přenosu uvnitř již zabezpečeného kanálu pomocí HTTPS. Výhodná z hlediska vysoké bezpečnosti je asymetrická kryptografie, důležitý aspekt na použití je časová náročnost a potřeba bezpečného přenosu veřejného klíče ze serveru ke klientu. Z tohoto hlediska jsem zvolil, že pro přenos veřejného klíče využiji digitálního podpisu pro zabezpečení jeho přenosu. Avšak klíč pro vytvoření HASH digitálního podpisu musí být uvnitř klienta, ale díky tomu je to možnost měnit klíč k šifrování RSA a zvýšit tak nemožnost zabezpečení proti přečtení odesílaných dat. Takže data, která odesílá server, jsou zabezpečena pomocí digitálního podpisu, který má za úkol zaručit autenticitu odeslání serverem nikoliv šifrovat obsah. Digitální podpis nešifruje obsah, takže pokud by došlo k prolomení HTTPS protokolu byla by podepsaná data čitelná což je nepřijatelné pro vyplněnou kandidátku, proto se pro vyplněnou kandidátku použije RSA kryptografie. RSA má vysokou míru zabezpečení díky tomu, že zašifrovaná data nejsou

čitelná bez dešifrování a není možné je dešifrovat pomocí údajů, kterými byly zašifrovány. Díky problému nazvanému faktorizace prvočísel je prolomení RSA s vysokým klíčem nemožné v reálném čase, což zajistí vysokou bezpečnost hlasovacího lístku.

Takže ve výsledku jsou data v komunikaci v jednom případě šifrována dvakrát a v druhém šifrována a podepsána. Šlo by použít i další šifrování ale rostl by tak strojový čas potřebný pro zpracování a dešifrování informace.



Obr. 17. Zabezpečení přenosu

4.3 Vytvořená verze

Od původního návrhu se moc neodbočilo, byl pouze rozšířen o některé další funkce. Takže konečný postup je zadání vstupních informací, transakčního čísla a pinu, které je potřeba ověřit na serveru, tato akce je vyvolává stisknutím tlačítka uživatelem pod zadávacími poli. Po stisknutí tlačítka se přesune aplikace na druhou obrazovku, aby vyla indikace toho, že se něco opravdu děje. Po přechodu se spustí kód, který pošle zadané informace na server a čeká na odpověď od serveru.

Od serveru můžou přijít jednoduše řečeno 3 různé odpovědi. První možná odpověď je, že zadané informace nesouhlasí se záznamy v databázi. Zobrazí se dialog, který informuje voliče o tom, že informace které zadal, nejsou platné a pomocí tlačítka ve spodní části dialogu se vrátí zpět na zadání informací. Druhá možnost je, že zadané informace souhlasí a volič ještě nevolil. Tak od serveru ještě dostaneme informaci o tom, jakou kandidátku si má stáhnout, podle toho do kterého okresu volič patří. Tak se odešle další požadavek na stáhnutí správné kandidátky. Po stáhnutí kandidátky se přejde na další stav, kde se zobrazí informace z kandidátky. Anebo třetí možnost že volič už je označen v databázi už, jako že

volil tak dojde k zamítnutí pokračování a dialog oznámí voliči, že už volil a není možné volit dvakrát.

Až se dostane volič do stavu aplikace pro zobrazení stran, tak se dynamicky podle dané kandidátky zobrazí skupina RadioButtonu pro výběr jedné ze stran. Pokud by volič nevybral ani jednu ze stran a kliknul na tlačítko další pro přechod na výběr kandidátů tak bude upozorněn dialogem, že musí vybrat jednu ze stran, aby mohl pokračovat. Pokud bude vybrána jedna ze stran a stiskne se tlačítko pro přesun do dalšího stavu tak se přejde na stav pro výběr kandidátů vybrané strany.

Po přechodu na výběr kandidátů se načtou informace o kandidátech dané strany z kandidátky a vypíší se do skupiny CheckBoxů. Je jen na voliči kolik označí kandidátů. Ve spodní části jsou dvě tlačítka jedno pro vrácení do stavu pro výběr stran a druhé pro odeslání volby. Pokud se klikne na tlačítko odeslání volby tak se objeví dialog, který se voliče zeptá, jestli opravdu chce odeslat aktuální volbu. Je to i pro jistotu kdyby náhodou volič omylem kliknul na odeslání, aby nedošlo k odeslání volby, kterou nechtěl provést.

Po potvrzení odeslání volby se aplikace přesune do stavu s informativní obrazovkou o tom, že se něco děje na pozadí. Jako první se sestaví data k odeslání, což je v podstatě vyplněná kandidátka. Tu je potřeba zašifrovat, v našem případě používám algoritmus RSA s délkou klíče 2048 bitů. Pro zašifrování je potřeba modulo a veřejný exponent, veřejný exponent je uvnitř aplikace a modulo se stáhne ze serveru. RSA se aplikuje na vyplněnou kandidátku a tyto zašifrovaná data se odešlou na server, ten je zpracuje a odešle potvrzení zpět klientovy.

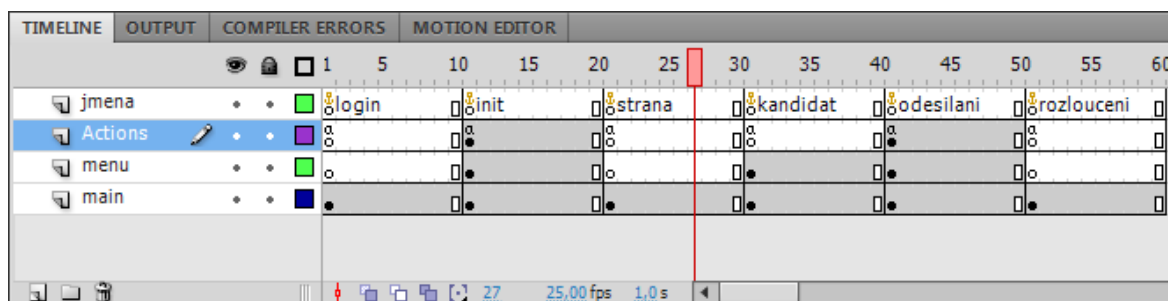
Potom co přijde potvrzení, se aplikace posune do stavu rozloučení s voličem a poděkování za jeho účast ve volbách. Po přesunu do tohoto stavu se aktivuje časovač, který po uplynutí několika sekund přesune aplikaci zpět do stavu přihlášení a může provést volbu další volič na stejném terminálu.

5 FLASH PROFESSIONAL

Flash Professional i když je primárně určen pro animace, není ochuzen o žádné funkce a možnosti které by bránily ve vytváření aplikací anebo webových stránek.

5.1 Začátek

Nejlepší začátek je podle mě rozvrhnout si v časové ose jednotlivé stavy aplikace, ve kterých se bude aplikace nacházet, dokonce by ji šlo vytvořit tak že by se na jednom snímku odehrávalo všechno, ale takový kód by byl o hodně delší a velmi nepřehledný. A ještě vytvořit vrstvy tak abychom měli vše potřebné přehledně rozděleno.



Obr. 18. Timeline

Jednu vrstvu mám pouze na to, aby v ní byly uloženy klíčové názvy snímků, podle kterých se bude orientovat běh aplikace. To že je vedle názvu vidět kotva znamená, že se na tento snímek může přesunout i pomocí ActionScriptu. Druhá vrstva je hlavně pro vytvořený kód a testovací objekty. Ve flash Professional se ActionScript vkládá do určitého snímku, tím se taky určuje, kdy se tento kód aplikuje a provede. Třetí vrstvu mám pro výběrová a oznamovací menu, taky se tím zaručí, že výběrové a oznamovací menu budou vždy nad ostatním obsahem. A poslední vrstva, která obsahuje grafiku samotné aplikace.

5.2 Grafika

Pomocí nástrojů ve Flash Professional jsem vytvořil grafické objekty a těm kterým bylo potřeba i přiřadil instanční jména. Jedinou ne úplně tradiční grafikou jsou ohraničení oblasti pro vkládání RadioButtonu a CheckBoxu. Je to proto, že ve Flash Professional nejsou objekty typu kontejner, který by automaticky řadil data za sebe a nabízel funkce pro práci s těmito daty. Proto je použit obyčejný MovieClip jako kontejner, díky tomu se dá korigovat, ve které vrstvě se mají objevit. Protože jinak pro vložení na samotnou pracovní

plochu by se objevily jako nejvyšší objekty a byly tak nad výběrovými a oznamovacími nabídkami, což by nebylo dobré.

Musí se také počítat s tím, že dané objekty existují pouze v dané sekvenci snímků, ve které jsou vymezené v časové ose. Nedá se na ně odkazovat před ani po daném úseku. Dynamicky vytvořené objekty vložené přímo na hlavní plochu by existovali po celou dobu běhu aplikace, díky tomu že je vložím do MovieClipu, který existuje fyzicky v časové ose. Tím se zařídí, že dané objekty zaniknou spolu s MovieClipem, který je zapouzdřuje.

Jinak na vytváření grafický objektů není nic speciálního, jediné pokud bychom chtěli využít inverzní kinematiku neboli kosti. Ale ty používat nebudu, protože nepoužiji žádné složité animace. Ani drag and drop objekty, které by měli omezení pomocí kostí.

5.3 Vytvoření interakcí

Interakce se vytvářejí hlavně pomocí vyvolávání událostí a jejich zpracování.

Jedním z nejdůležitějších a nejčastěji používaných je využití tlačítek. Díky tomu že používám symboly typu Button tak se události generují automaticky, jsou to události jako kliknutí, přidržení, puštění tlačítka a další. Stačí pouze přidat naslouchání na událost určitého typu k danému objektu. Stačí pouze jednoduše definovat pomocí funkce `objekt.addEventListener(o jakou událost je jedná, funkce která zpracuje událost)`;

Typů událostí a samotných událostí, které se dají zachytit je spousta, stačí jen naslouchat na to, až se daná událost vyvolá a námi vytvořená funkce ji zpracuje.

Takže bylo potřeba vytvořit funkce pro všechny tlačítka a přiřadit pomocí funkce `addEventListener` tyto funkce k danému tlačítku s parametrem `MouseEvent.CLICK` což zaručí, že se funkce provede, když uživatel stiskne a pustí levé tlačítko myši nad daným tlačítkem. Většinou tlačítka budou sloužit pro přesun mezi jednotlivými stavy v časové ose. Ale bude tomu předcházet ve většině případu potvrzovací nabídka. Ta je vytvořena mimo viditelnou oblast pracovní plochy a tím pádem se tváří jako by tam nebyla a po kliknutí na tlačítko se zavolá funkce, která změní pozici tohoto dialogu do viditelné oblasti a aktivuje naslouchání událostí tlačítek uvnitř tohoto dialogu. A tak se buď posune uživatel dál anebo přehodnotí nebo opraví svoji volbu nebo zadání.

Další důležitou událostí je předávání si informací mezi serverem a klientem, to je provedeno díky XMLHttpRequest. XMLHttpRequest je základní funkce, která se hlavně používá pro nahrávání externího obsahu, aby se co nejvíce minimalizovala velikost aplikace a urychlilo se tím nahrání aplikace. Je to jeden z důležitých aspektů webových aplikací, dlouhé načítání je jeden z aspektů, které je dobré co nejvíce minimalizovat. Ale já je budu používat na předávání poměrně malého, ale důležitého obsahu. Pro autentizaci přihlášení, stažení kandidátky a odeslání volby s tím že pomocí jednoho XMLHttpRequestu odešlu a získám informace, je to princip na jakém funguje i AJAX jen s tím rozdílem že nepoužívám JavaScript ale ActionScript.

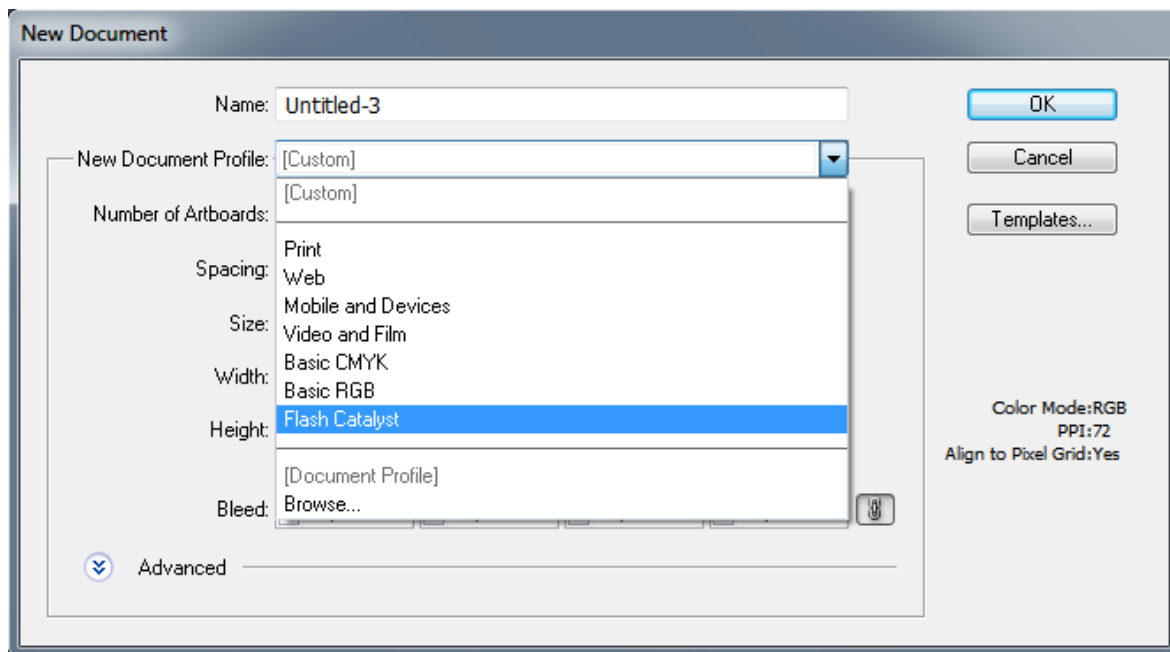
Stačí vytvořit XMLHttpRequest do kterého se vloží potřebné informace a pomocí XMLHttpRequest se tento XMLHttpRequest provede. U XMLHttpRequestu nasloucháme na Event.COMPLETE to nám oznámí, kdy dorazila kompletní odpověď od serveru a může se zpracovat. Data ze serveru jsou uvnitř dané události o dokončení operace, takže jednoduše funkce, která se spustí po dokončení operace, zpracuje přijatá data.

Kód pro šifrování jsou napsány funkce v externí knihovně, která se importuje dovnitř souboru při kompilaci. Takže v podstatě se pracuje s funkcemi, jen se musí používat specifický formát dat. Přesněji Bytová pole protože matematické operace zde pracují s obrovskými čísly, se kterými si standardní proměnné pro čísla neporadí. A taky proto že k šifrovanému textu se chováme jako k obrovskému číslu, ve kterém každý znak má určitou hodnotu. V podstatě se dá říci o šifrování, že se jedná o poměrně jednoduché matematické operace, ale musí se programově aplikovat tak aby daný jazyk dokázal pracovat s nejlépe čísly nekonečné délky. Ještě je důležité, aby tyto operace neměly příliš vysokou dobu zpracování. Protože pracujeme s obrovskými čísly tak taky narůstá doba zpracování jednoduchých matematických operací, máme pouze 32 nebo 64 bitové procesory a například klíč, se kterým chceme pracovat je 2048 bitové číslo, takže je opravdu nutné pracovat sekvenčně a v mnoha krocích.

6 FLASH BUILDER

Pro projekt ve Flash Builder bude předcházet pár kroků, které pomohou vytvořit grafický základ aplikace. A v druhém kroku ve Flash Catalyst dokonce i transformaci grafiky na komponenty, přidání základní interakcí a přechodů mezi jednotlivými stavy.

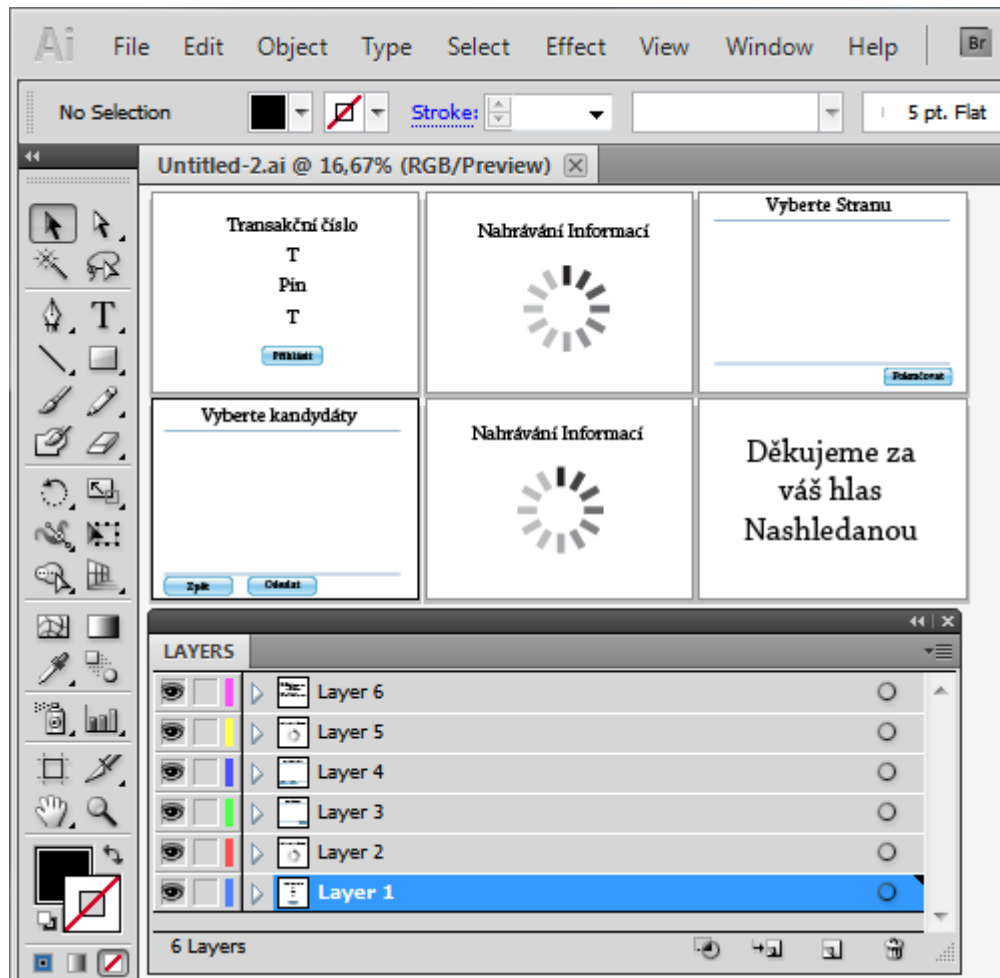
6.1 Prvotní návrh



Obr. 19. Adobe Ilustrátor dokument

V Adobe Ilustrátor existuje možnost vytvoření souboru optimalizovaného pro Flash Catalyst pro další zpracování. Výsledek je pak možno exportovat jako speciální formát FXG která je vytvořen speciálně pro Flash Catalyst a nebo klasický AI soubor který není problém zpracovat po udání několika specifikací jak se má co importovat.

Takže jednoduše při vytváření nového souboru se vybere profil pro Flash Catalyst, dále pak nastavíme požadovaný počet pracovních ploch, které budou reprezentovat jednotlivé stavy aplikace po vložení do Flash Catalyst. Navíc ještě na každé pracovní ploše budu pracovat s objekty v separátních vrstvách.



Obr. 20. Návrh v Adobe Illustrator

Do první pracovní plochy jsem vložil čtyři textová pole. Dvě informativní a dvě, které se později zkonvertují na uživatelská vstupní pole, pro zadání transakčního čísla a pinu určených k autentizaci voliče. A ještě vytvořit grafiku reprezentující tlačítko pro odeslání vložených informací.

Druhá pracovní plocha je pouze informativní, že se zpracovává požadavek. Kdyby náhodou zpracování požadavku trvalo déle, aby si uživatel nemyslel, že aplikace zamrzla.

Třetí pracovní plocha je určená pro výběr strany, v této fázi stačí informativní text, aby uživatel věděl, co má dělat. Dále jednoduché ohraničení umístění kontejneru, do kterého se budou vkládat radiobuttony. A na konec tlačítko, kterým se přesune z výběru stran na výběr daných kandidátů námi vybrané strany.

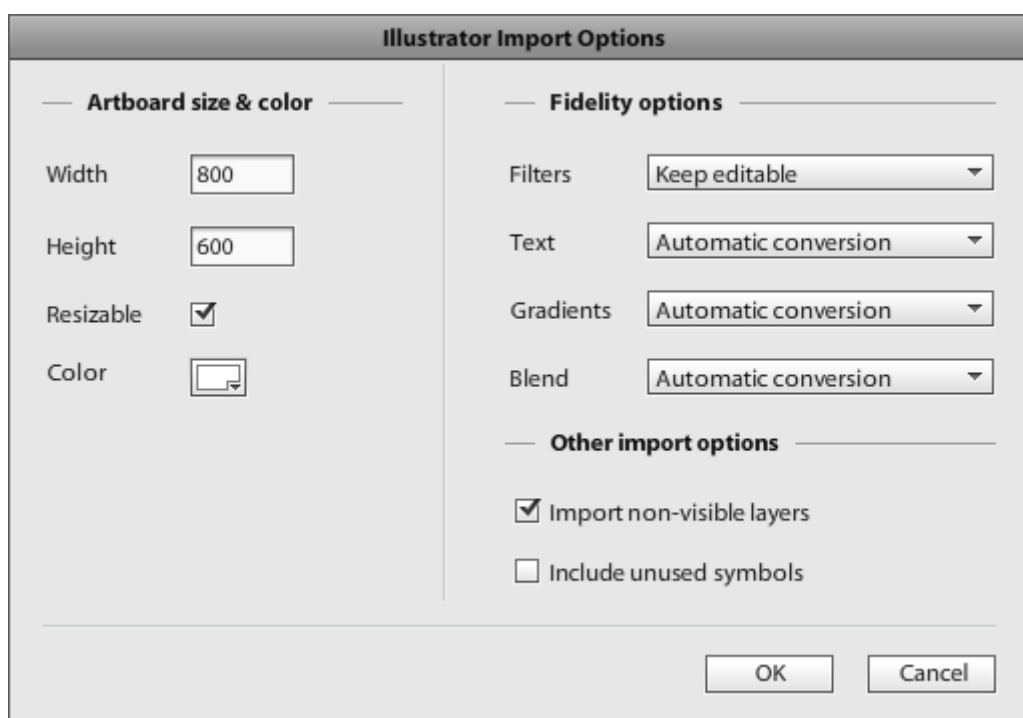
Čtvrtá pracovní plocha je určena pro výběr kandidátů vybrané strany, zase bude potřeba informativní text, aby uživatel věděl, co má dělat. Také bude potřeba ohraničení kontejneru ale tentokrát pro checkboxy, protože musí existovat možnost vybrat i více kandidátů. A dvě grafiky pro tlačítka jedno pro odeslání a druhé pro vrácení zpět na výběr stran.

Pátá pracovní plocha je stejná jako druhá, má pouze informovat uživatele, že se zpracovává požadavek na pozadí aplikace.

Šestá pracovní plocha obsahuje pouze jedno textové pole s poděkováním a rozloučením.

Na rozdíl od Flash Professional si nemusím vytvářet potvrzovací a oznamovací tabulky pokud použiju knihovnu Alert.

6.2 Převod ve Flash Catalyst

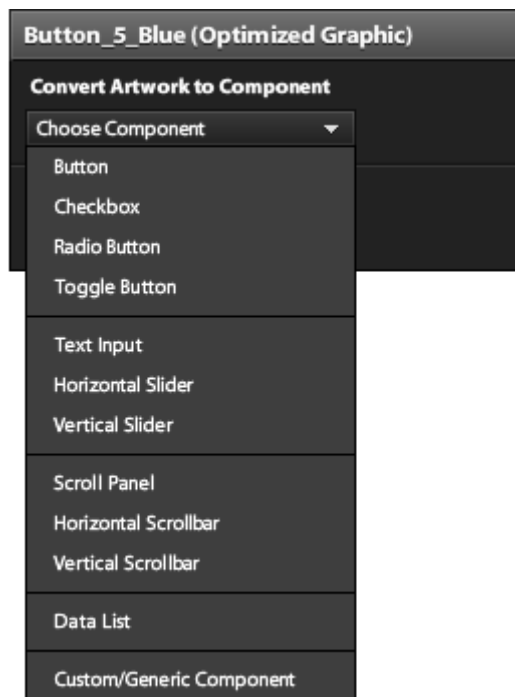


Obr. 21. Import souboru z Illustratoru

Při importu je důležitých několik věcí, zaškrtnutím Resizable můžeme ignorovat rozlišení, protože se výsledná aplikace bude měnit své rozlišení a rozložení komponent podle toho jak velké bude okno aplikace. Což je jedna z dobrých funkcí, které ve Flash Professional nedosáhneme, protože tam není implementována. Důležité je také Import non-visible layers což je importovat i vrstvy které nejsou označeny jako aktuálně viditelné. Je to důležité protože když se dělají například tlačítka tak se dávají jednotlivé stavy přes sebe a nechá se

viditelný pouze jeden ze stavů. Importovat nepoužité symboly z knihovny v ilustratoru nepotřebuji, protože tam žádné nemám.

Provede se analýza souboru, importují se vrstvy a můžeme začít převádět jednotlivé grafické objekty na komponenty. Prostě stačí označit grafický objekt anebo skupinu grafických objektů a vybrat z nabídky o jakou komponentu se bude jednat.



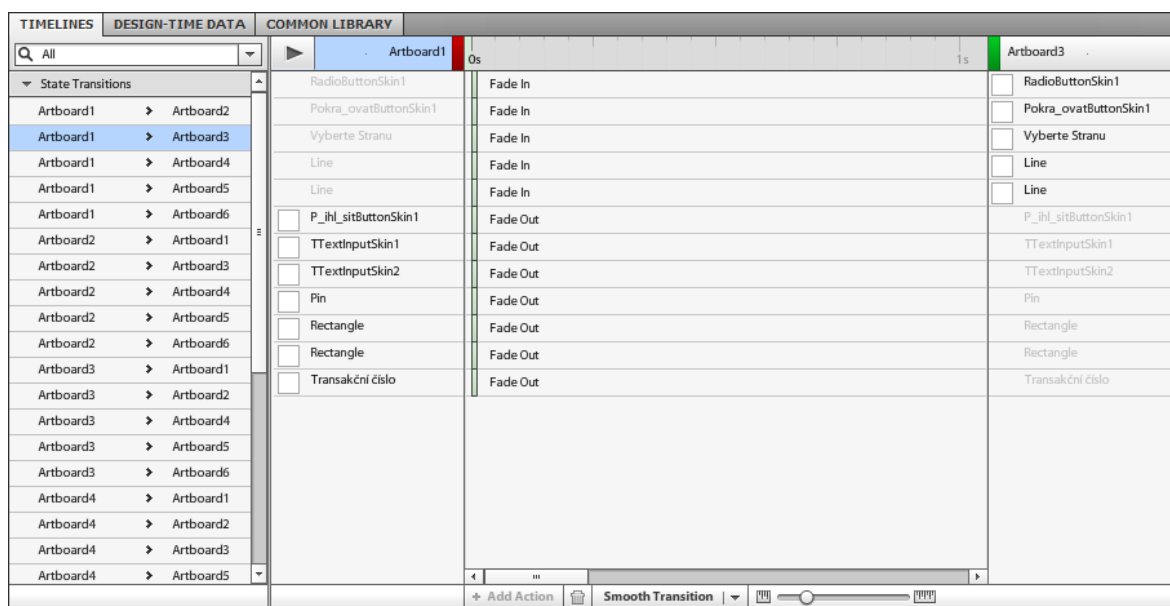
Obr. 22. Nabídka Convert

U tlačítek navíc máme možnost vybrat i textový objekt, který bude reprezentovat objekt label v třídě Button. Tím pádem se pak může pomocí ActionScriptu změnit nápis na tlačítku kdybychom chtěli, sice to nevyužiji, ale je dobré vědět, že i takovou možnost mám, kdybych ji potřeboval. Takhle postupně jsem proměnil všechnu grafiku na potřebné komponenty.

Ve Flash Catalyst je i možnost nastavovat vazby na hrany pro případné změny velikosti okna aby grafika byla na správném místě, ale osobně mi přijde lepší a přehlednější využít panel ve Flash Builder

A ještě jako poslední krok ve Flash Catalyst je nastavení přechodů mezi jednotlivými stavy aplikace. V panelu Timelines je na výběr možnost určení přechodů mezi všemi stavy navzájem ale já budu potřebovat jen přechod z prvního na druhý, z druhého na třetí, z třetího na čtvrtý, ze čtvrtého na pátý, z pátého na šestý a z šestého na

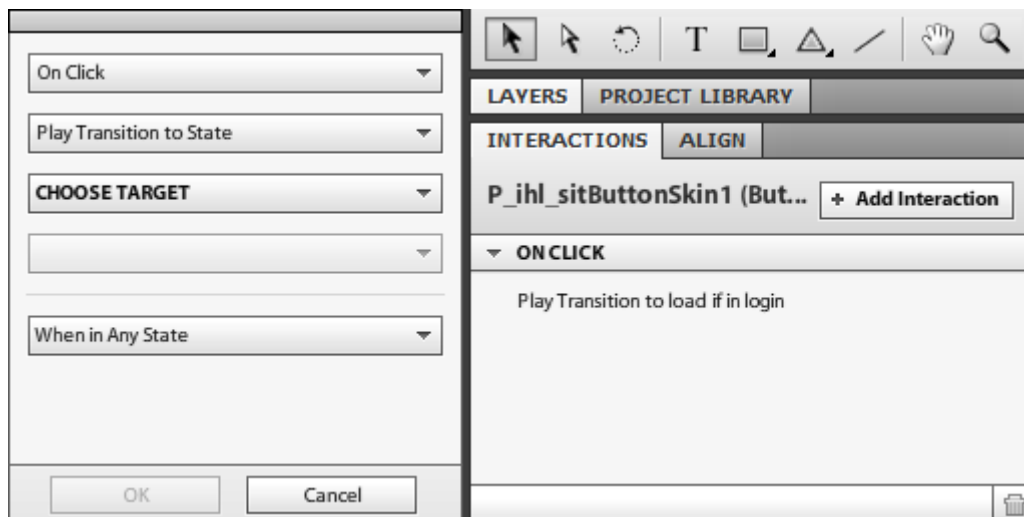
první. Ostatní nebudou potřeba, protože k takovému kroku v aplikaci nemůže dojít, porušil by se tím běh aplikace.



Obr. 23. Flash Catalyst Timelines

Jednoduše se vybere objekt, který se nachází, buď ve stavu ze kterého se přechází, anebo objekt ze stavu na který se přechází. Je možnost i vybrat více objektů najednou a aplikovat na něj některý z efektů přechodu. To se dělá jednoduše pouze kliknutím v dolní liště na Add Action a vybere se, jaká z funkcí se má aplikovat. Funkčně to připomíná Motion tween z Adobe Flash Professional. Tyto přechody by se dali dělat i později ve Flash Builderu pomocí MXML ale Flash Catalyst je to jednodušší, přehlednější a rychlejší. To co Flash Catalyst dělá je, že generuje MXML přechodů podle přednastavených parametrů, takže je možnost i později dělat úpravy ve Flash Builderu ale už bez možnosti náhledu na průběh animace před kompilací aplikace.

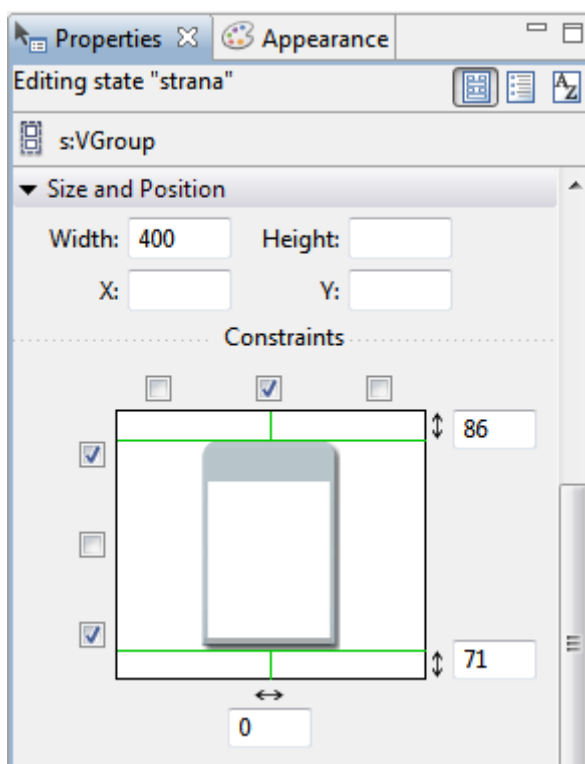
Poslední krok ve Flash Catalyst je nastavení interakcí. Například jako je přehrání animačního přechodu do prvního stavu po nahrání aplikace. A vygenerování základních funkcí pro tlačítka, které budou pak dodělány ve Flash Builderu. Protože ve flash Catalyst není možnost vytvářet vlastní funkce, lze pouze použít přednastavené funkce. Vytvoří se tak základní zpracování eventů jako je nahrání aplikace anebo kliknutí na tlačítko. Jednoduše se vybere objekt a v panelu Interactions se klikne na Add Interaction a v nabídce se vybere jaká akce se má vygenerovat a na jaký event se má naslouchat.



Obr. 24. Panel Interactions

6.3 Finální dodělení ve Flash Builder

Jako první jsem nastavil minimální rozlišení aplikace a pak vazby jednotlivých grafických objektů aby se objekty ideálně rozložili při změně velikosti okna aplikace. Jednoduše se vybere, ke které ze stran se má objekt svázat. Ideální je to pro kontejnery dá se tak zařídit aby se zvětšila jeho velikost podle velikosti stránky.



Obr. 25. Nastavené vazeb

Když už byly konečně všechny vazby nastaveny tak se konečně budu věnovat samotnému funkčnímu kódu. Větší část jsem použil z první verze z Flash Professional ale bylo potřeba několik úprav a některé věci udělat jinak. Například pro RadiobuttonGroup existuje přímo komponenta v MXML, takže nebylo potřeba vytvářet ji pomocí ActionScriptu a samotné RadioButtony se museli nalinkovat z jiné knihovny, aby se použily přímo v architektuře Spark. Dalším rozdílem je že nejde importovat nic z knihovny fl, odtud také pocházely Radiobuttony a CheckBoxy generované ve Flash Professional.

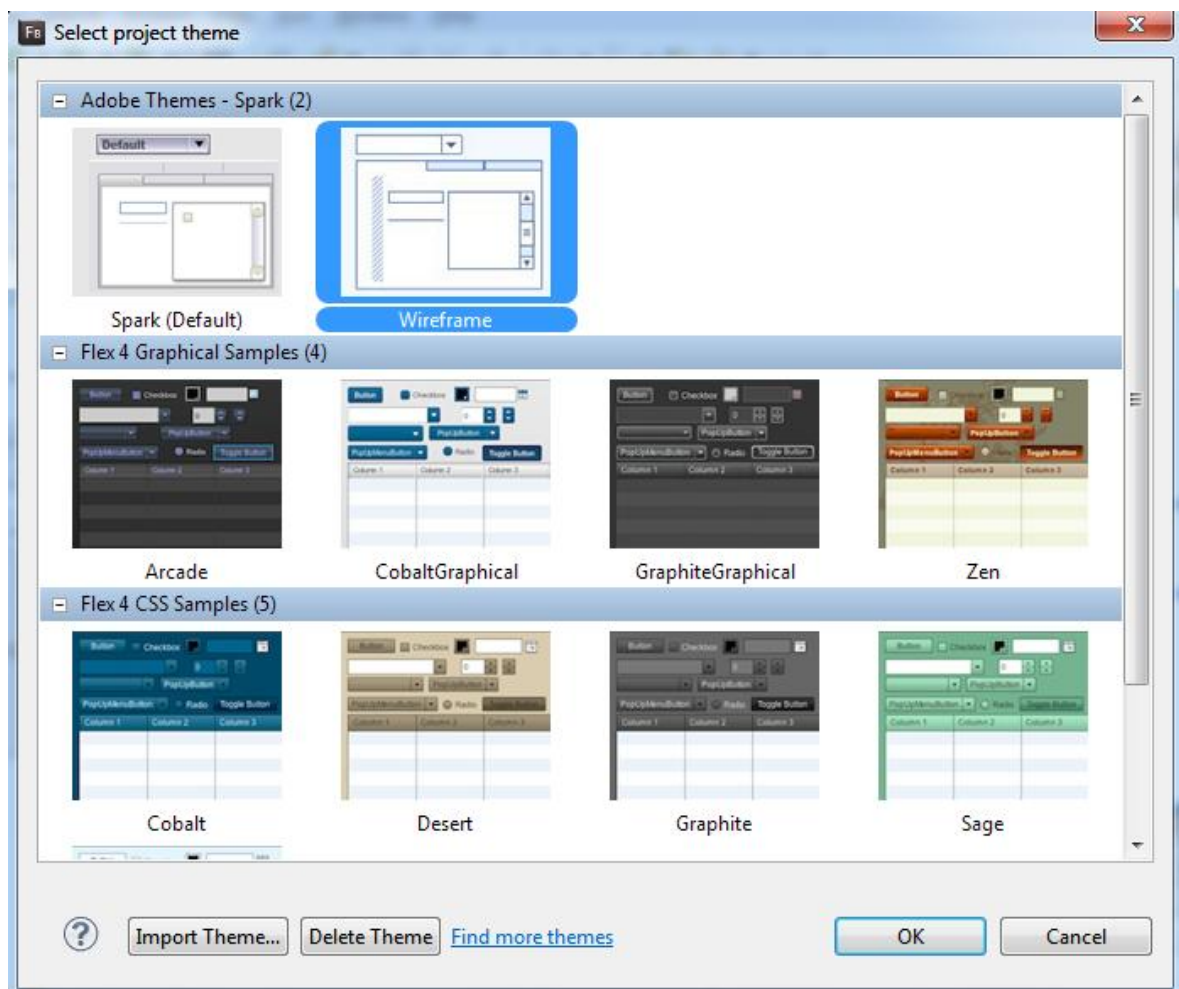
Další rozdíl je že jsem nemusel vytvářet náročně dialogy oznámení a dialogy potvrzení a zpracování jejich eventů. Ve Flash Builder existuje knihovna Alert v architektuře MX, která vytváří takovéto dialogy jednoduše a jejich obsah i vzhled je modifikovatelný takže neztrácíme žádnou funkčnost ani nejsme limitováni grafickou interpretací.

Dalším rozdílem je využití opravdových kontejnerů pro RadioButtony a Checkboxy nejenom obyčejný MovieClip, který ve FlashBuilderu ani použít nejde, hodně to tak zjednoduší samotnou práci s dynamicky vygenerovanými daty. Jednak mám větší a jednodušší přehled o rozmístění vložených objektů tak i jednodušší a lepší přístup k jednotlivým objektům, což rozhodně je velkou výhodou.

Jinak v postupu a funkčnosti aplikace není rozdíl oproti Flash Professional, jen je rozdíl že se nepohybujeme v časové ose ale mezi stavy aplikace. Ale to uživatel nepozná je pouze rozdíl v tom, jak se programově zařídí přechod mezi stavy. Ve flash Builderu stačí pouze změnit hodnotu v currentState na název stavu do kterého se chce přejít a to je vše. Navíc pokud existuje Transition mezi přepínanými stavy tak se automaticky aplikuje při přechodu je to výhodou, že není potřeba mít další starost s tím, aby se přehrál daný přechod.

Jinak kód pro získávání a odesílání informací zůstal úplně stejný, nebylo potřeba jej měnit, stejně tak jako proces šifrování a výpočet Hash, jsou to funkce přímo napsané v ActionScriptu a nevyžívají žádné specifické knihovny, které by vadili univerzálnímu použití v obou programových prostředích.

Jako poslední věc co jsem změnil byl grafický styl. Nelíbilo se mi jak vypadají dialogové okna knihovny Alert, tak jsem změnil jednoduše kompletní barevné schéma aplikace. Protože využívám vlastní grafické skiny pro komponenty tak mi to ovlivnilo pouze dialogy, které pocházely ze standardní knihovny. Dal by se vytvořit i styl přímo jen pro Alert dialog ale to by dalo více práce.



Obr. 26. Select projet theme

Velkou výhodou jsou grafická schémata hlavně při využívání standardních grafických objektů bez aplikování externích stylů objektů. Sjednotí to tak aplikaci barevně do jednotného stylu.

7 TESTOVACÍ SERVER

Zvolil jsem pro testování PHP server s SQL databází. Protože i samotný klient je uzpůsobený pro spolupráci s PHP jelikož pro předávání dat pomocí URLRequestu z Flash se používají stejné postupy jako, když si mezi sebou předávají data jednotlivé stránky PHP, když spolu komunikují.

Metoda ve Flash je URLRequestMethod.POST která je stejná jako když v PHP pracujeme s \$_POST v PHP. Proto v serveru přistupuje k přijatým datům jako by byly odeslány pomocí metody POST, stačí pouze zadat název proměnné ve Flash pomocí třídy URLVariables a pod stejným názvem si v PHP přebereme proměnnou po přijetí pomocí \$_POST['název'].

7.1 Příklad předávání dat

V jednoduchém příkladu demonstřuji co všechno je potřeba pro předání si dat z Flash na server.

Flash

```
1 var loginV:URLRequest = new URLRequest("volba.php");
2 var loadeV:URLLoader = new URLLoader();
3 var phpVarsV:URLVariables = new URLVariables();
4 loginV.method = URLRequestMethod.POST;
5 loadeV.dataFormat = URLLoaderDataFormat.VARIABLES;
6 phpVarsV.tc = txtTC.text;
7 loginV.data = phpVarsV;
8 loadeV.load(loginV);
9 loadeV.addEventListener(Event.COMPLETE, onLoginLoadV);
```

PHP

```
1 $tc = $_POST['tc'];
```

Nejprve je potřeba vytvořit proměnnou typu URLRequest a specifikovat adresu na kterou se bude vztahovat požadavek. Jakou druhou věc je potřeba vytvořit URLLoader který tento požadavek zpravuje. Jako třetí je potřeba vytvořit proměnnou URLVariables pro proměnné, které se budou předávat serveru. Je potřeba nastavit metodu předání jako POST pro

jednoduché zpracování pomocí PHP. Dále je potřeba nastavit formát dat, který budeme předávat. Na sedmém řádku je příklad jak se nastaví proměnná a její hodnota, nemusí se volat žádná funkce, prostě se za tečkou napíše její název a přiřadí se do něj hodnota. Potom co jsou všechny proměnné uvnitř proměnné typu URLVariables, se tato proměnná nastaví, jako data které se budou předávat v URLRequest. Potom už stačí jen zadat URLLoaderu aby daný požadavek uskutečnil. Pokud chceme získat a zpracovat odpověď od serveru musíme nastavit EventListener na naslouchání na dokončení dané operace předávání dat a ve funkci kterou zavolá po dokončení, se zpravují data získané od serveru.

Pro zpracování na PHP serveru si stačí pouze převzít hodnotu pomocí funkce \$_POST a s obsahem pak můžeme libovolně pracovat.

7.2 Funkce serveru

Server byl navržen tak aby odpověděl na všechny možné požadavky, které budou odeslány klientem.

- Autentizace uživatele od klienta přijde transakční číslo a pin zadané voličem, tyto informace se ověří na serveru v SQL databázi a odešle se výsledek zpět klientu, který tuto informaci zpracuje.
- Odeslání správné kandidátky podle okresu, ve kterém je volič zařazen ve volebním seznamu.
- Odeslání veřejného klíče RSA, který klient použije k zašifrování volebního lístku po vyplnění.
- Uložení přijatého vyplněného volebního lístku a odeslání potvrzení o přijetí a uložení volebního lístku

7.3 zabezpečení přenosu citlivých dat mezi klientem a serverem

Zabezpečení přenosu v tomto případě je velmi důležité jelikož volby nesmí být jinak a nikým ovlivněný jinak by ztratily význam.

První krok bezpečnosti je použití protokolu HTTPS který zabezpečuje samotný přenos mezi serverem a klientem. V některých případech voleb by to bylo dostatečné pro zabezpečení

ale u důležitějších voleb by to bylo opravdu nedostatečné. Proto jsem využil jak digitální podpis, tak asymetrickou kryptografii pro vyšší bezpečnost.

Pro digitální podpis je použita SHA-256, podepisují se důležité informace odesílané ze serveru klientu. Využití digitálního podpisu místo šifrování má své výhody i nevýhody. Nevýhodou je, že klíč k podpisu musí být uvnitř klienta, ale tento stejný problém je i u šifrování což je bezpečný přenos klíče ze serveru do klienta, proto bude lepší, když bude uvnitř klienta. Sice to znamená, že pro vyšší bezpečnost při znovuvyužití, se bude muset vytvořit nový spustitelný soubor klienta s jiným klíčem pro další volby.

Jako asymetrickou kryptografii klient používá RSA. Veřejný klíč k šifrování se odesílá digitálně podepsán uvnitř HTTPS protokolu, to by mělo být dostatečně bezpečné pro přenos klíče. Není potřeba zajistit, aby klíč byl nečitelný, musí jen být ochráněn proti změně při přenosu. Kdežto vyplněný volební lístek nesmí být čitelný při přenosu i to je jeden z důvodů použití RSA, po zašifrování veřejným klíčem nejde zpětně dešifrovat veřejným klíčem, zpráva je dešifrována pouze na serveru, který zná privátní klíč pro dešifrování.

7.4 Příklad šifrovací funkce RSA

Tady vidíme příklad šifrovací funkce napsané v ActionScript pro Flash Professional a Flash Builder. Je důležité, že se šifrování provádí na straně klienta a zabezpečí se odesílaná data.

```
1 private function encrypt(src:ByteArray, dst:ByteArray,
2 length:uint, pad:Function):void {
3     if (pad==null) pad = pkcs1pad;
4     var bl:uint = (n.bitLength()+7)/8;
5     var end:int = src.position + length;
6     while (src.position<end) {
7         var block:BigInteger = new BigInteger(pad(src, end, bl,
8         padType), bl);
9         var chunk:BigInteger = doPublic (block);
10        chunk.toArray(dst); }}
11 protected function doPublic(x:BigInteger):BigInteger {
12    return x.modPowInt(e, n);}
```

Funkci se zadá vstupní a výstupní pole, délka vstupu a pokud bychom chtěli použít jinou funkci pro formátování vstupních dat tak ještě o jakou funkci se jedná. Takže, pokud se nezadá formátovací funkce, zvolí se základní. Dále se vytvoří proměnná určující délku bloku, podle délky modula, která se upraví, aby byla v celém násobku osmy zaokrouhlena nahoru. Dále se vytvoří proměnná určující konec šifrované zprávy. Dále se po blocích zpracuje celý vstupní blok, kde se jednotlivé bloky zašifrují pomocí veřejného klíče e a modula n .

Algoritmus pro kódování a dekódování RSA je v principu pravdu jednoduchý. Složitější a časově náročnější je proces vytváření klíče, protože musíme náhodně najít dvě odlišná obrovská prvočísla. Naštěstí klient generovat klíč nemusí, jinak by jedna volba zabrala spoustu času.

8 DALŠÍ MOŽNOSTI

Díky tomu že firma Adobe má tak obrovské úspěchy tak má i vlastní Developer Centrum kde se zabývají novými možnostmi a projekty které buď zaniknou anebo najdou své využití.

8.1 Adobe Labs

Například existuje <http://labs.adobe.com/>

Je tu možnost například si stáhnou beta verze nové verze Adobe Flash Player a nebo Adobe AIR. Nebo Adobe Edge který dokáže převést Flash animace na HTML5, CSS3 + JavaScript. Anebo Adobe AIRLaunchpad který dokáže vytvořit přednastavený projekt pro Adobe AIR ve formě pro Adobe Flash Builder. Avšak nejdůležitější a nejzajímavější pro účely aplikací je Adobe Alchemy.

8.1.1 Adobe Alchemy

Je to technologie, která dokáže přeložit C a C++ kód na ActionScript použitelný tak ve Flash Player a Adobe Air. Samotný Scott Petersen, který se podílí na vývoji, na prezentaci této technologie řekl, že na využití kryptografie napsané v C je to výborné protože by měl dokázat zpracovat přibližně čtyřicet procent instrukcí až desetkrát rychleji po konverzi z C oproti samotnému kódu napsanému přímo v ActionScriptu. Je to i proto že se nevytváří přímo kód, ale dává to k dispozici funkce jako celky. Například máme knihovnu pro C tak se dá jednoduše převést na knihovnu swc pro Flash.

Podmět pro to byl i proto že v C/C++ existuje více, než 6 biliónů řádků kódu které jsou licencované k volnému využití zdarma. A předělávat tisíce řádků kódu pro využití ve Flash by bylo velmi neefektivní, proto by se hodila jednoduchá konverze místo pracného přepisování.

Největší demonstrace této technologie byla ve spolupráci s Epic games kdy převedli kompletní Unreal Engine do Flash. Engine sám o sobě měl přibližně 1,1 milionu řádek kódu. Po konverzi sice vzniklo přibližně 250 milionů řádek ActionScriptu ale všechno bylo funkční a běželo plynule. Navíc takto vznikl podmět pro nový compiler pro ActionScript který dokáže zpracovat obrovské množství ActionScriptu rychleji.

9 ZHODNOCENÍ

9.1 Zhodnocení požadavků

Ani o jednom požadavku vytčeném v kapitole 4 se nedá říci, že by nebyl splněn, pouze grafické prostředí by zasloužilo, aby bylo navrženo profesionálním grafikem, ale i přesto je výsledná aplikace intuitivní a jednoduchá.

Funkční

- Možnost přihlášení – splněno – server a klient si předají informace a umožní tak autentizaci voliče
- Síťová komunikace – splněno – je potřeba připojení k Internetu pro vytvoření spojení mezi serverem a klientem
- Čtení z externího elektronického volebního seznamu – splněno – klient odešle požadavek a výsledek zpracuje ale důležitější část je zpracování na serveru odeslání správné kandidátky z hlediska volebního okrsku
- Možnost vyplnění volebního lístku – splněno – dynamicky generované výběrová pole vytvořená podle přijaté kandidátky slouží k vytvoření vyplněného volebního lístku
- Uložení vyplněného volebního lístku externě – splněno – po vyplnění volebního lístku se lístek zašifruje a odešle na server, který jej uloží.
- Možnost kryptografie na straně klienta – splněno – Flash běží na lokální stanici, to zajišťuje, aby se vyplněný volební lístek mohl zašifrovat před odesláním
- Možnost Hash funkce – splněno – implementována funkce pro vytvoření Hash
- Ověření integrity hlasu – splněno – server potvrdí uložení dat na serveru

Technické požadavky na Flash

- HTTP/HTTPS – splněno – Flash dokáže pracovat základně v HTTPS
- RSA šifrování – splněno – aplikováno RSA šifrování
- SHA – splněno – aplikován digitální podpis SHA-256
- XML – splněno – Flash obsahuje knihovny pro veškerou možnou práci s XML

- Uživatelský vstup – splněno – Flash obsahuje objekty statické i dynamické pro sběr zadaných uživatelských dat
- Výběr z kandidátky – splněno – vyplnění volebního lístku elektronicky pomocí

9.2 Výhody Flash technologie pro volebního klienta

Technologie Flash je obsahem primárně určena pro web, tak zvané RIA což jsou komplexní webové aplikace anebo samostatně spustitelné aplikace na klientské stanici. Díky tomu že Flash je primárně určena pro web jsou zde komunikace mezi serverem a klientem jednoduše implementovatelné a jejich zpracování jako události je také jednoduše zpracovatelné, stejně jako informace o průběhu těchto událostí.

Další výhodou je spolupráce s nástroji pro výrobu složitých grafických prostředí a jejich jednoduchá implementace do samotného programu. Je to i výborná vlastnost kdy vytváříme program ve spolupráci s grafikem, aby aplikace měla co nejvyšší úroveň a požadovaný vzhled. Neznám žádný jiný program, který by měl k dispozici možnost přímého využití takto vyspělého grafického prostředí.

Jako další výhodu bych viděl, že se dá vytvořit klient jako serverová aplikace stažená a běžící pak lokálně na stanici. Jednoduše se tak také dají dělat případné změny a bez následku nutnosti náročné redistribuce. Distribuce klienta je tím pádem velmi jednoduchá. I samotný přenos klienta k voliči probíhá pomocí HTTPS tím pádem je to i bezpečné.

Důležitou výhodou je možnost šifrování dat na straně klienta před odesláním na server. S tím je i spojen vysoký výpočetní výkon, takže šifrování RSA ve Flash netrvá dlouho, je to přibližně 50 milisekund. Díky tomu že Flash je neustále vylepšován k uspokojení neustále se zvyšujících nároků je zde i možnost ještě dalšího vylepšení výkonu v budoucnu.

9.3 Zhodnocení Flash

Pokud se bude hodnotit výkonnostně rozdíl mezi aplikací napsanou ve Flash Professional a nebo Flash Builder, tak rozdíl v podstatě neexistuje. Při testování rychlosti šifrování RSA rozdíl byl minimální. Takže v závislosti na výkonu nezáleží, v které z aplikací bude klient vytvořen.

Pokud se bude hodnotit samotné vytváření aplikace ve Flash Professional a Builder.

Flash Professional má výhodu v možnosti vytvářet mnohem složitější a komplexnější animace díky tomu, že Flash Professional je primárně vytvořen pro animace. I rozdíl toho že používá časovou osu, jako primární smyčku programu není překážkou, nabízí to větší přehled nad staticky vygenerovanými objekty. Ale nevýhodou je ne úplně tak propracovaný intellisense, tento aspekt zpomaluje výrobu aplikace, ale věřím, že v dalších verzích přijde krok více kupředu a vylepšení. A největší nevýhodou, která přidává práci je absence strukturovaných kontejnerů pro grafické objekty.

Flash Builder při využití i Flash Catalyst je podle mého názoru výhodnější z hlediska, že Flash Builder je primárně určen pro RIA a vytváření složitějších struktur a kódu v ActionScriptu. Další výhodou je taky možnost flexibilního rozlišení aplikace a možnost využití kontejnerů pro jednoduché zpracování dynamicky zobrazovaných dat. V podstatě Flash Builder si poradí i s animacemi ale musí se dělat pomocí knihoven a ActionScriptu, což je náročnější a není k tomu grafický editor, takže se musí aplikace kompilovat, aby byly viditelné vytvořené animace.

Pokud se zaměřím na Flash technologii jako takovou oproti jiným webovým technologiím tak jedinou nevýhodou je potřeba stáhnout si Adobe Flash Player. Výhodou Flash je že dokáže spolupracovat s ostatními technologiemi jako je například PHP, ColdFusion, WSDL a další. Výhodou je také že nemusíme řešit, jaký prohlížeč uživatel používá, protože se všechno zpracovává uvnitř Flash Playeru.

Nevýhodou oproti standardním jazykům jako je Java nebo C/C++ je že nejdou přímo ovládat USB a podobné periferie počítače, kromě webkamery. Musely by se vytvořit další aplikace v jiném jazyku jako aplikace typu klient-server nebo P2P.

Z vytvořeného klienta, testování výkonnosti při šifrování a zhodnocení výhod a nevýhod jsem vyhodnotil, že technologie Flash je velmi vhodná pro účely volebního klienta.

ZÁVĚR

Ve své diplomové práci jsem se zaměřil na tvorbu aplikací, speciálně pro potřebu volebního klienta, ve Flash Professional a Flash Catalyst + Flash Builder. Ale bylo potřeba také připravit server pro účely testování funkčnosti požadavků odesílaných a přijímaných klientem.

V praktické části jsem se zaměřil na postup ve vytváření klienta jak ve Flash Professional tak Flash Builder, aby bylo vidět jak moc je postup vytváření Flash aplikace rozdílný z hlediska použité aplikace pro vytvoření. Také postup a proces návrhu aplikace klienta a její vylepšení do konečné fáze.

U volební aplikace je velmi důležité zabezpečení aplikace, aby nedošlo ke změně zasílaného obsahu mezi serverem a klientem a došlo tak k ovlivnění voleb. Byla potřeba vytvořit návrh zabezpečení přenosu a přenášených dat mezi severem a klientem. Bylo velmi výhodné použít protokol HTTPS díky jeho standardizaci a rozsahu použití. Z hlediska vyšší bezpečnosti se využívá digitální podpis pro přijímaná data od serveru a RSA šifrování pro odesílaná data serveru. RSA má délku klíče 2048 bitů, což by mělo být dostatečné pro zabránění v dešifrování obsahu třetí stranou v reálném čase, z časového hlediska šifrování by nebyl problém použít i větší délku klíče ale dešifrování zabere více času.

Bude výhodné, až bude hotová finální verze Adobe Alchemy aplikovat šifrování pomocí tohoto nástroje a C/C++ knihoven mělo by to zvýšit rychlost šifrování a vést tak k možnosti použití ještě vyššího klíče pro kódování a zvýšit tak bezpečnost přenášených dat.

Na konec bych chtěl jen dodat, že jsem velice rád, že jsem si mohl rozšířit znalosti vytváření Flash o vývoji aplikace pomocí Flash Builder a Flash Catalyst.

ZÁVĚR V ANGLIČTINĚ

In my thesis I focused on needs of creating applications, specifically on the needs of the voting client, in Flash Professional and Flash Catalyst + Flash Builder. But it was also required to prepare a server for testing of the functionality of the requests sent and received to client.

In the practical part is focused on the creation process as the client is build in Flash Professional and Flex Builder to see the differences in creating Flash applications in the two different programs. Also the process of designing a client application and enchanting it to its final stage.

For the elections is very important to use security applications to prevent changes of content sent between the server and client and was no way to influence elections. It was necessary to design transfer security of data transmitted between the server and the client. It was very convenient to use the HTTPS protocol because of its standardization and mass use. In terms of increased security is used digital signature for data received from the server and the RSA encryption for transmitted data to server. RSA key length is 2048 bits, which should be sufficient to prevent decryption of the contents by third party in real time, over time, for the encryption would not be a problem to use a larger key length but it take more time to decipher.

It will be beneficial, for the use for client, when Adobe finishes Adobe Alchemy to apply encryption using this tool and C / C ++ libraries, it should increase the speed of encryption and lead to the possibility of using even higher key for encoding and increase the security of transmitted data.

In the end I just wanted to add that I am very glad I was able to extend the knowledge on the development and creating Flash applications using Flex Builder and Flash Catalyst.

SEZNAM POUŽITÉ LITERATURY

- [1] GROVER, Chris. Flash CS5.5: the missing manual. 1st ed. Sebastopol, CA: O'Reilly, 2011, 841 s. Missing manual. ISBN 14-493-9825-1.
- [2] FLORIO, Chris. Actionscript 3.0 for Adobe Flash Professional CS5 classroom in a book: the official training workbook from Adobe Systems. 1st ed. Berkeley, Calif.?: Adobe Press, c2010, 389 s. Classroom in a book. ISBN 03-217-0447-9.
- [3] FLORIO, Chris. Adobe Flash Professional CS5 classroom in a book: the official training workbook from Adobe Systems. 1st ed. San Jose, Calif.: Adobe, c2010, 375 s. Classroom in a book. ISBN 03-217-0180-1.
- [4] NOBLE, Joshua J. Flex 4 cookbook: the official training workbook from Adobe Systems. 1st ed. Sebastopol, Calif.: O'Reilly, 2010, 736 s. Classroom in a book. ISBN 05-968-0561-6.
- [5] TAPLEY, Scott. Adobe Flash Catalyst CS5 classroom in a book: the official training workbook from Adobe Systems. 1st ed. San Jose, CA: Adobe Press, c2010, 273 s. Classroom in a book. ISBN 03-217-0358-8.
- [6] SHUMAN, Jim. Adobe flash cs5 revealed: the official training workbook from Adobe Systems. 1st ed. Clifton Park, NY: Delmar Cengage Learning, 2010, 273 s. Classroom in a book. ISBN 11-111-3057-4.
- [7] BOREK, Bernard. Adobe Flex: co je a co není. [online]. 2008 [cit. 2012-05-03]. Dostupné z: <http://interval.cz/clanky/adobe-flex-co-je-a-co-neni/>
- [8] SANADHYA, K. S., SARKAR, P.. New Collision attacks Against Up To 24-step SHA-2. [online]. [cit. 2012-05-03]. Dostupné z WWW: <http://eprint.iacr.org/2008/270.pdf>
- [9] MARTIN MAREŠ. Algoritmy okolo teorie čísel [online]. 7. 11. 2011 [cit. 2012-05-04]. Dostupné z: <http://mj.ucw.cz/papers/numth.pdf>
- [10] BERNARD, Borek. Adobe Flex: kompletní průvodce tvorbou interaktivních aplikací. Vyd. 1. Brno: Computer Press, 2011, 397 s. ISBN 978-80-251-2765-0.
- [11] BERNARD, Borek. Adobe Flash CS5 Professional: oficiální výukový kurz. Vyd. 1. Brno: Computer Press, 2010, 392 s. ISBN 978-80-251-3224-1.

- [12] BERNARD, Borek. Adobe Flash Catalyst CS5: oficiální výukový kurz. Vyd. 1. Brno: Computer Press, 2011, 280 s. ISBN 978-80-251-3254-8.
- [13] BERNARD, Borek. ActionScript 3.0: oficiální výukový kurz. Vyd. 1. Překlad Kristýna Konopková. Brno: Computer Press, 2011, 381 s. ISBN 978-80-251-3335-4.
- [14] Šilhavý, R., Šilhavý, P., Prokopová, Z.: Architecture of COOPTO Remote Voting Solution, Springer Science+Business Media B.V., Advanced Techniques in Computing Science and Software Engineering, Dordrecht, 2010, 477-479, ISBN-ISMN 978-90-481-3959-9

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

XML	eXtensible Markup Language .
HTML	HyperText Markup Language
RIA	Rich Internet Applications.
MXML	Macromedia eXtensible Markup Language.
AJAX	Asynchronous JavaScript and XML
SQL	Structured Query Language
PHP	Hypertext Preprocessor (Personal Home Page)
RSA	Rivest, Shamir a Adleman
SHA	Secure Hash Algorithm
SSL	Secure Sockets Layer
WSDL	Web Services Description Language
P2P	Peer-to-peer

SEZNAM OBRÁZKŮ

Obr. 1.	Původní logo flash-shockwave	11
Obr. 2.	Úvodní menu	13
Obr. 3.	Pracovní plocha Flash Professional CS5	14
Obr. 4.	Panel nástrojů	15
Obr. 5.	Motion Editor	17
Obr. 6.	Motion Presets.....	18
Obr. 7.	Code Snippets a Actions Frame	19
Obr. 8.	Flash Builder.....	20
Obr. 9.	Flash Builder možnosti.....	21
Obr. 10.	Flash Catalyst úvodní menu.....	22
Obr. 11.	Pracovní prostředí Flash Catalyst	23
Obr. 12.	Panel Interactions	24
Obr. 13.	Nabídka Convert.....	24
Obr. 14.	Abobe Illustrator template	25
Obr. 15.	Sekvence událostí	32
Obr. 16.	Předběžný návrh obrazovek	33
Obr. 17.	Zabezpečení přenosu.....	35
Obr. 18.	Timeline.....	37
Obr. 19.	Adobe Ilustrátor dokument	40
Obr. 20.	Návrh v Adobe Illustrator	41
Obr. 21.	Import souboru z Illustratoru	42
Obr. 22.	Nabídka Convert.....	43
Obr. 23.	Flash Catalyst Timelines.....	44
Obr. 24.	Panel Interactions	45
Obr. 25.	Nastavené vazeb	45
Obr. 26.	Select projet theme	47

SEZNAM TABULEK

SEZNAM PŘÍLOH

CD s prací v pdf formátu a všemi zdrojovými soubory