Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky

TECHNOLOGIE SMIL A TVORBA WEBOVÝCH MULTIMEDIÁLNÍCH PREZENTACÍ

Bakalářská práce

Autor práce: Tomáš Vlasák
Vedoucí práce: PaedDr. Petr Pexa
Datum odevzdání: 25. 4. 2008

Prohlašuji, že svoji bakalářskou práci jsem vypracoval/-a samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské/diplomové práce, a to v nezkrácené podobě pedagogickou fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách.

V Českých Budějovicích dne

Na tomto místě bych rád poděkoval PaedDr. Petru Pexovi za odborné rady a cenné připomínky k této práci.

Anotace

Cílem této bakalářské práce je vytvořit uživatelskou příručku technologie SMIL jako moderní metody pro tvorbu webových multimediálních prezentací. Dále jsem se pokusil o porovnání těchto tří technologií: SMIL, Flash a JavaScript. Součástí práce je také konkrétní webová multimediální prezentace, vytvořená pomocí SMIL technologie.

Abstract

The aim of this Bachelor work is to create a user guide of SMIL technology as a modern method for creation of web multimedia presentations. Furthermore, I tried to compare these three technologies: SMIL, Flash and JavaScript. The work also contains the specific web multimedia presentation which was created by means of SMIL technology.

OBSAH

1 ÚVOD

Mnoho z nás si nedokáže představit svůj život bez internetu. Pro některé – např. pro webmastery - znamená internet způsob obživy. Abychom mohli být úspěšnými webmastery, musíme jít s dobou a umět používat i ty nejmodernější technologie. Jednou z takovýchto moderních technologií je technologie SMIL. Tato méně známá technologie slouží k vytváření webových multimediálních prezentací.

Již delší dobu se zajímám o tvorbu webových stránek, a proto mě zaujalo toto téma. Cílem této bakalářské práce je vypracovat uživatelskou příručku technologie SMIL. Pomocí této příručky by i méně zkušený webmaster měl být schopen vytvořit vlastní multimediální prezentaci.

Práce je rozdělena na část teoretickou a praktickou. V teoretické části představuji jazyk SMIL, zmiňuji jeho historii a popisuji jednotlivé verze tohoto jazyka. Dále se zabývám rozšiřujícími profily SMILu pro webové stránky. Jedná se o profily HTML+TIME a XHML+SMIL. Praktická část práce obsahuje příklady k jednotlivým verzím a profilům jazyka SMIL. V závěru práce jsem se pokusil o srovnání s tradičními technologiemi jako je Adobe Flash a JavaScript. Také jsem otestoval podporu této technologie v dostupných nejnovějších verzích webových prohlížečů.

Informace potřebné k napsání této práce jsem čerpal výhradně z internetových zdrojů.

Součástí bakalářské práce je CD, které obsahuje sbírku příkladů.

2 SPECIFIKACE JAZYKA SMIL

Jazyk SMIL, neboli Synchronized Multimedia Integration Language vychází z jazyka XML. Jedná se o snadno naučitelný značkovací jazyk, podobně jako HTML, sloužící k vytváření multimediálních prezentací obsahujících audio, video, obrázky a text.

Je třeba zmínit, že SMIL neslouží k vytvoření videa nebo obrázku, ale pracuje s již hotovými multimediálními a textovými objekty, určuje, který objekt a kdy se má zobrazit nebo posunout. Výsledná prezentace obsahuje pouze zdrojový kód jednotlivých animací, nikoli samotné multimediální objekty.

„SMIL přináší univerzální, výkonné a přitom nenáročné řešení, které je pro mobilní platformu mnohem důležitější než pro oblast PC.“[1]

2.1 Co tedy můžeme pomocí jazyka SMIL vytvořit?

Pomocí jazyka SMIL můžeme vytvořit internetové prezentace, podobné prezentacím v PowerPointu. Ty mohou současně zobrazit různé typy souborů (text, video, audio), z různých webových serverů. Prezentace může také obsahovat ovládací tlačítka (start, stop, next, atd.) a hypertextové odkazy na jiné prezentace.

[1] GRIMMICH, Šimon. SMIL - Timed Interactive Multimedia Extensions for HTML. *Interval.cz* [online]. 2004 [cit. 2008-03-04]. Dostupný z WWW: <http://interval.cz/clanky/smil-timed-interactive-multimedia-extensions-for-html/>.

2.2 Vlastnosti a funkce jazyka SMIL

Jazyk SMIL
- umožňuje nastavit vzhled prezentace.
- umí nastavovat pozici a velikost vkládaného objektu.
- dokáže měnit pozici či viditelnost objektu v reálném čase.
- nám umožňuje definovat sekvenci, délku trvání, pozici a viditelnost elementu.
- stejně jako JavaScript umožňuje testování různých podmínek (verze prohlížeče, rychlost připojení k internetu, rozlišení obrazovky, atd.)

3 HISTORIE JAZYKA SMIL

Jazyk SMIL byl vyvinut teprve v nedávné době, a proto jeho působení ve světě internetu a multimédií není moc dlouhé.

Tato kapitola se pokouší nastínit nejdůležitější okamžiky v historii jazyka SMIL. Podrobnou historii můžeme nalézt na webových stránkách W3C.org.[2]

W3C vyvíjelo SMIL od roku 1997 jako jazyk pro choreografické multimediální prezentace, ve kterých můžeme zpracovávat audio, video, text v reálném čase.

V listopadu 1997 byla vydána první verze SMILu.

V únoru 1998 byla vydána druhá verze SMILu (Second Public Edition).

V listopadu 1999 byl vydán SMIL-Boston, který byl později přejmenován na SMIL 2.0.

V dubnu 2001 byl zpřístupněn pro veřejnost fungující koncept SMILu 2.0.

V srpnu 2001 byl publikován nový profil SMILu (XHTML + SMIL Profile) a SMIL 2.0 se stal standardem W3C.

V prosinci 2001 vyšla první tištěná publikace o jazyce SMIL (SMIL: Adding Multimedia to the Web).

V červenci 2002 byl definován nový profil SMILu pro MMS (Multimedia Messaging Service) nazvaný 3GPP SMIL Profile.

V červnu 2003 byl profil 3GPP SMIL Profile využit v mobilním telefonu Nokia 6600.

V lednu 2005 vyšla nová verze SMIlu 2.0 obsahující Service Pack.

V únoru 2005 vydala organizace SYMM Working Group SMIL 2.1, který se ještě téhož roku v prosinci stal standardem.

V prosinci 2006 byl poprvé zveřejněn SMIL 3.0.

[2] http://www.w3.org/AudioVideo/ 4.3.2008

4 SPECIFIKACE JEDNOTLIVÝCH VERZÍ SMILu

Tato kapitola pojednává o jednotlivých verzích jazyka SMIL a poukazuje na nejdůležitější rozdíly mezi nimi. Dále zde uvádím základní kostry jednotlivých verzí.

4.1 SMIL 1.0

Jedná se o první a tudíž i nejjednodušší verzi SMILu. V porovnání s novějšími verzemi tato obsahuje jen několik značek, které si můžeme snadno zapamatovat.

Základní kostra dokumentu je podobná základní kostře dokumentu HTML s tím rozdílem, že základním elementem dokumentu není element <html>, ale element <smil> . Samozřejmě také musíme vložit správné DTD.[3]

```
<!DOCTYPE    smil    PUBLIC    "-//W3C//DTD    SMIL    1.0//EN"
            "http://www.w3.org/TR/REC-smil/SMIL10.dtd">
<smil>
      <head>

      </head>

      <body>

      </body>
</smil>
```

4.1.1 Jaké elementy můžeme dát do hlavičky dokumentu?

Mezi párový element <head> můžeme napsat meta-tagy podobně jako v HTML nebo XHTML, avšak nejdůležitějším elementem je **párový** element <layout>. Do „layoutu“ můžeme zapsat **nepárový** element <root-layout>, který slouží k nastavení vzhledu a rozměrů stránky a také element **nepárový** <region>, který slouží jako kontejner pro multimediální objekty.

[3] *W3.org* [online]. 2008 [2008-1-6]. Dostupný z WWW: < http://www.w3.org/TR/REC-smil/#dtd/>.

```
...
      <head>
      <layout>
            <root-layout />
            <region id=‘reg1‘ ... />
            <region id=‘reg2‘ ... />
      </layout>
      </head>
...
```

Výše uvedené elementy root-layout a region používají tyto atributy:

id – slouží jako identifikátor (používá se k označení regionu)
width – používá se k nastavení šířky regionu
height – používá se k nastavení výšky regionu
background-color – používá se k nastavení barvy pozadí regionu nebo stránky
top – odsazení z horní strany
left – odsazení z levé strany
z-index – používá se pro možnost překrývání dvou regionů (vrstvy)

Pomocí zde uvedených atributů můžeme vytvořit následující layout.

```
...
      <layout type=‘text/smil-basic-layout‘>
            <root-layout width=‘260‘ height=‘100‘ background-
color=‘silver‘ />
            <region id=‘reg1‘ width=‘100‘ height=‘100‘ top=‘0‘
left=‘0‘ />
            <region id=‘reg2‘ width=‘150‘ height=‘80‘ top=‘0‘
left=‘110‘ />
      </layout>
...
```

Takto nastavený layout nastaví šířku, výšku a pozadí stránky. Dále vytvoří dva regiony, které budou umístěny vedle sebe s desetipixelovou mezerou. Pokud použijeme tento layout a do elementu <body> přidáme např. nějaký obrázek, musíme mu nastavit atribut region.

```
...
      <body>
            <img src=‘obrazek.gif‘ region=‘reg1‘ ... />
      </body>
...
```

Tímto řádkem jsme určili umístění obrázku do regionu reg1. Je pochopitelné, že velikost obrázku by měla korespondovat s velikostí regionu.

4.1.2 Jaké elementy můžeme dát do těla dokumentu?

Do těla dokumentu vkládáme především multimediální objekty jako jsou audio, video a obrázky, ale můžeme také vkládat odkazy na jiné prezentace či textové soubory.

Ve SMILu existují dva základní typy animací:

- seq - sekvenční animace (jednotlivé animace se spouštějí postupně)
- par – paralelní animace (animace se budou spouštět současně)

Tyto dva druhy animací můžeme spojovat a vytvářet tak opravdu rozmanité prezentace. Obě varianty mohou mít tyto atributy:

- begin – čas začátku animace v sekundách
- end – čas konce animace v sekundách

Elementy umístěné v elementu body mohou mít následující atributy:

region - id regionu
alt - popis obsahu objektu
longdesc – dlouhý popis
src - cesta k zdroji objektu
type – typ objektu
dur - délka trvání v sekundách
ref – reference
repeat - počet opakování

Multimediální prvky audio, video, animation, ref a textstream mohou mít navíc další atributy (uvedené níže). Tyto dva atributy nám mohou posloužit pro ořezávání videa nebo audia.

clip-begin	– čas začátku přehrávání v objektu
clip-end	– čas konce přehrávání v objektu

4.2 SMIL 2.0

SMIL 2.0 má dva hlavní záměry:

Formuluje jazyk založený na XML, který uživateli umožní vytvořit interaktivní multimediální prezentaci. Pomocí SMILu 2.0 může uživatel charakterizovat dočasné fungování multimediální prezentace, spojovat odkazy s mediálními objekty a popsat rozvržení prezentace na obrazovce.

Umožňuje opětovné použití syntaxe a sémantiky SMILu 2.0 v jiných jazycích založených na XML, zvláště v těch jazycích, které znázorňují načasování a synchronizaci. Například komponenty SMILu 2.0 se používají při začleňování načasování do XHTML a SVG. SMIL 2.0 je definován jako sada značkovacích modulů, které formulují sémantiku a XML syntaxi pro určité oblasti funkčnosti SMILu.[4]

Základní kostra dokumentu psaného ve SMILu 2.0 vypadá následovně:

```
<?xml version="1.0"?>
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 2.0//EN"
"http://www.w3.org/2001/SMIL20/SMIL20.dtd">
<smil xmlns="http://www.w3.org/2001/SMIL20/Language">
      <head>
      </head>
      <body>
      </body>
</smil>
```

Tato kostra se od předchozí verze moc neliší, rozdíl mezi verzí 1.0 a verzí 2.0 spočívá v jiném DTD a XML deklaraci. Dalším rozdílem je změna názvů atributů odebráním pomlčky mezi dvouslovnými atributy. Příkladem takovéto změny je atribut clip-begin, který má nyní název clipBegin.

[4] http://www.w3.org/TR/2005/REC-SMIL2-20050107/introduction.html 4.3.2008

4.2.1 Modularizace a Interaktivita

„SMIL 2.0 přidává především moduly pro interaktivitu (uživatel může reagovat na prezentaci - měnit parametry objektů) a animace (změna barvy, parametru, pohyb po křivce). Závažnou změnou je modularizace, tedy rozdělení prvků SMILu do skupin - modulů, což umožňuje použít vybrané vlastnosti SMILu k doplnění jiného formátu. Kupříkladu animační moduly jsou použity v SVG. SMIL 2.0 obsahuje celkem 45 modulů rozdělených do deseti skupin“[5]

1. Časování (Timing)
2. Práce s časem (Time Manipulation)
3. Animace (Animation)
4. Řízení obsahu (Content Control)
5. Rozvržení (Layout)
6. Odkazy (Linking)
7. Mediální objekty (Media Objects)
8. Metainformace (Metainformation)
9. Struktura (Structure)
10. Změny a přechody (Transitions)

Timing - Jedná se o kategorii modulů, které slouží pro plánování a časování. Tato kategorie obsahuje celkem 19 modulů, které nebudu přesněji popisovat, pouze zde uvedu jejich výčet: AccessKeyTiming, BasicInlineTiming, BasicTimeContainers, EventTiming, ExclTimeContainers, FillDefault, MediaMarkerTiming, MinMaxTiming, MultiArcTiming, RepeatTiming, RepeatValueTiming, RestartDefault, RestartTiming,

[5] GRIMMICH, Šimon. SMIL - základní elementy a konstrukce. *Interval.cz* [online]. 13.7.2004 [cit. 2008-03-01]. Dostupný z WWW: <http://interval.cz/clanky/smil-zakladni-elementy-a-konstrukce/>.

SyncbaseTiming, SyncBehavior, SyncBehaviorDefault, SyncMaster, TimeContainerAttributes, WallclockTiming.

Time Manipulation – V této kategorii se nachází pouze jediný modul, který je nazván stejně jako kategorie. Mluvíme tedy o Time Manipultion Modulu, který obsahuje následující atributy: accelerate, decelerate, autoReverse a speed. První dva slouží, jak je již patrné z anglického názvu, ke zrychlování a zpomalování. Atribut autoReverse použijeme u animací, kde požadujeme po skončení animace opětovné přehrání, ovšem v opačném sledu. Poslední atribut speed udává poměr rychlosti přehrávání vůči rodičovskému elementu.

Animation – Jedná se o kategorii modulů sloužící k animacím. Obsahuje pouze dva moduly, jedním z nich je BasicAnimation modul a druhým modulem je SplineAnimation modul. Nejdůležitějšími atributy, které obsahují tyto moduly, jsou attributeName a targetElement. Atribut targetElement určuje element, se kterým budeme něco dělat, přičemž atribut attributeName určuje, jaký atribut vybraného elementu budeme měnit.

Content Control – Tato kategorie obsahuje čtyři moduly, které slouží pro kontrolu a řízení obsahu (rozhodovací konstrukce). Jsou jimi: BasicContentControl, CustomTestAttributes, PrefetchControl a SkipContentControl. „Jde o moduly, které zjišťují informace o systému návštěvníka - rychlost připojení, rozlišení, jazyk a podobně. Závisle na zjištěných datech je možné poskytnout návštěvníkovi odpovídající obsah, například audio v jím preferovaném jazyce. Část tohoto modulu byla obsažena již ve SMIL 1.0, a to v podobě elementu switch pro rozhodovací konstrukci.“

Layout – Kategorie obsahuje tyto čtyři moduly: AudioLayout, BasicLayout, HierarchicalLayout a MultiWindowLayout. Tyto moduly slouží

především k nastavení celkového vzhledu prezentace - s výjimkou prvního modulu, který slouží k ovládání hlasitosti audio či video objektů.

Linking – Další kategorií modulů je kategorie Linking, která obsahuje tři moduly pro tvorbu odkazů. Díky těmto třem modulům můžeme rozlišit tři různé způsoby vytváření odkazů. Na jednotlivé typy odkazů se podíváme později, prozatím jen vypíši názvy modulů, které patří do této kategorii: BasicLinking, LinkingAttributes a ObjectLinking.

Media Objects - Tato kategorie obsahuje sedm modulů sloužících k vkládání objektů (audio, video, obrázky, text, atd.) do prezentace. Tyto moduly nebudu přesněji popisovat, pouze zde uvedu jejich výčet: BasicMedia, BrushMedia, MediaAccessibility, MediaClipping, MediaClipMarkers, MediaDescription a MediaParam.

Metainformation - V této kategorii se nachází pouze jediný modul, který je nazván stejně jako kategorie. Mluvíme tedy o Metainformation Modulu, který obsahuje popis meta elementů, které známe z (X)HTML. K dispozici je navíc element metadata, který pracuje s RDF (Resource Description Framework).

Structure – Tato kategorie obsahuje také jen jediný modul, který opět nese totožný název s názvem kategorie. Structure modul je tvořen elementy smil, head a body.

Transitions - Tyto moduly slouží ke změnám objektů nebo k přechodům mezi jednotlivými scénami. Tato kategorie obsahuje tyto tři moduly: BasicTransitions, InlineTransitions a TransitionModifiers.

4.3 SMIL 2.1

Můžeme říct, že verze 2.1 je zcela modulární. Obsahuje 50 modulů, z nichž je sestaven základní profil – SMIL 2.1 Basic Profile a dva rozšířené profily pro mobilní zařízení – SMIL 2.1 Mobile Profile a SMIL 2.1 Extended Mobile Profile. Těchto 50 modulů je opět rozděleno do 10 kategorií. Změny byly provedeny v kategoriích Timing, Layout, Media Objects a Transitions. Přibylo celkem 7 nových modulů, dva moduly byly smazány a u čtyř modulů byl změněn jejich obsah. Tentokrát zde nebudu uvádět jednotlivé kategorie, pouze uvedu ty, ve kterých byly provedeny nějaké změny:

Timing – V této kategorii přibyly dva nové moduly a jeden modul byl odstraněn. Přesněji řečeno modul ExclTimeContainers byl nahrazen modulem BasicExclTimeContainers. Dalším novým modulem je BasicPriorityClassContainers obsahující krom jiných elementů element priorityClass, který slouží k ovládání přerušení.

Layout – Do této kategorie přibyly celkem čtyři nové moduly. Prvním z nich je AlignmentLayout, který slouží pro přesné zarovnávání elementu k určitým stranám či na přesnou pozici. Druhý modul s názvem BackgroundTilingLayout popisuje atributy backgroundImage a backgroundRepeat. Zvláště zajímavý je backgroundRepeat, který umožňuje opakovaní obrázku v pozadí tak, jak to známe z (X)HTML, nebo můžeme nechat obrázek opakovat pouze v horizontální či vertikální ose. Třetím novým modulem je modul `OverrideLayout. Tento modul nedefinuje žádné nové elementy, pouze` rozšiřuje element ref. Čtvrtým a posledním modulem je SubRegionLayout nahrazující původní HierarchicalLayout modul.

Transitions – Do této kategorie přibyl jeden modul s názvem FullScreenTransitionEffects. Jak již napovídá samotný název, tento modul slouží k fullscreenovým přechodům a změnám.

Základní kostra dokumentu psaného ve SMILu 2.1 vypadá následovně:

```
<?xml version="1.0" ?>
 <!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 2.1//EN"
"http://www.w3.org/2005/SMIL21/SMIL21.dtd">
<smil xmlns="http://www.w3.org/2005/SMIL21/Language">
      <head>
      </head>

      <body>
      </body>
</smil>
```

Základní kostra obsahuje opět jiné DTD, jinak je oproti verzi 2.0 beze změn.

4.3.1 SMIL Mobile Profile

SMIL 2.1 Mobile Profile je kolekce modulů SMILu 2.1, které jsou v kontextu s mobilními zařízeními. Jedná se o verzi nadřazenou SMILu 2.1 Basic Profile, která byla ještě vylepšena na SMIL Extended Mobile Profile, o kterém si povíme později. SMIL 2.1 Mobile Profile se velkou mírou podobá profilu SMILu, který 3GPP definoval pro MMS. V porovnání s vylepšeným profilem 3GPP, tedy s 3GPP2 má SMIL Mobile Profile navíc tyto tři moduly: BackgroundTilingLayout, AlignmentLayout a FullScreenTransitions. Ale oproti 3GPP2 mu chybí tyto dva moduly: MultiArcTiming a BasicAnimation.

SMIL Mobile Profile podporuje základní vlastnosti SMILu 2.1, k čemuž používá pouze nezbytně nutné moduly ze SMILu 2.1. Obsahuje pouze 24 modulů, které jsou opět rozděleny do již známých kategorií. Kategorií je pouze osm, protože zde chybí tyto dvě kategorie: Animation a Time Manipulation. Přehled jednotlivých kategorií je uveden níže.

Přehled jednotlivých kategorií:[6]

SMIL 2.1 Mobile Profile	
Collection Name	**Elements in Collection**
ContentControl	switch, prefetch
Layout	region, root-layout, layout, regPoint
LinkAnchor	a, area
MediaContent	text, img, audio, video, ref, textstream, param, paramGroup
Metainformation	meta, metadata
Structure	smil, head, body
Schedule	par, seq
Transition	transition

[6] *W3.org* [online]. 2008 [cit. 2008-1-6]. Dostupný z WWW: < http://www.w3.org/TR/2005/REC-SMIL2-20051213/smil21-mobile-profile.html/>.

Základní kostra dokumentu psaného ve SMILu 2.1 Mobile Profile:

```
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 2.1 Mobile//EN"
"http://www.w3.org/2005/SMIL21/SMIL21Mobile.dtd">
<smil xmlns="http://www.w3.org/2005/SMIL21/Mobile">
        <head>
        </head>

        <body>
        </body>
</smil>
```

Základní kostra se opět liší pouze v jiném DTD.

4.3.2 SMIL Extended Mobile Profile

Proč SMIL Extended Mobile Profile? Původní SMIL Mobile Profile byl vytvořen pro mobilní zařízení, které v tu dobu nebyly příliš výkonné a měly nízké rozlišení displeje. SMIL Extended Mobile Profile byl vytvořen pro mobilní zařízení roku 2005, které již v tuto dobu obsahovaly procesory s rychlostí 206MHz a poměrně kvalitní displej.

Extended Mobile Profile je kolekce modulů, které poskytují téměř kompletní podporu SMILu 2.1 v kontextu s pokročilými mobilními zařízeními. SMIL Extended Mobile Profile celkem obsahuje 30 modulů, které jsou rozděleny do devíti kategorií. Přehled jednotlivých kategorií je uveden níže.

Přehled jednotlivých kategorií:[7]

SMIL 2.1 Extended Mobile Profile	
Collection Name	**Elements in Collection**
Animation	animate, set, animateMotion, animateColor
ContentControl	switch, prefetch
Layout	region, root-layout, layout, regPoint
LinkAnchor	a, area [anchor]
MediaContent	text, img, audio, video, ref, animation, textstream, brush, param, paramGroup
Metainformation	meta, metadata
Structure	smil, head, body
Schedule	par, seq, excl
Transition	transition

[7] *W3.org* [online]. 2008 [cit. 2008-1-6]. Dostupný z WWW: < http://www.w3.org/TR/2005/REC-SMIL2-20051213/smil21-extended-mobile-profile.html/>.

Základní kostra dokumentu psaného ve SMILu 2.1 Extended Mobile Profile:

```
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 2.1 Extended
Mobile//EN"
"http://www.w3.org/2005/SMIL21/SMIL21ExtendedMobile.dtd">
<smil xmlns="http://www.w3.org/2005/SMIL21/ExtendedMobile">
        <head>
        </head>

        <body>
        </body>
</smil>
```

Základní kostra se opět liší pouze v jiném DTD.

5 HTML+TIME

HTML+TIME (Timed Interactive Multimedia Extensions) představuje časované interaktivní rozšíření HTML. Jedná se o přidání časování a synchronizace do HTML dokumentu, které umožní prohlížení prezentací napsaných v jazyce SMIL v internetových prohlížečích.

5.1 Specifikace

HTLM + TIME je „ořezaná“ verze jazyka SMIL, která je určena přímo pro webové prohlížeče. Tento profil neobsahuje některé elementy a atributy původní verze, protože v HTML dokumentu již nejsou potřeba, neboť jsou nahrazeny pomocí CSS. Příkladem je element layout či region. Dále kořenový element smil je nahrazen elementem html.

5.2 Atributy

Díky rozšíření o TIME můžeme v HTML dokumentu používat nové atributy, které nám umožní synchronizaci a načasování jednotlivých animací. Následující atributy můžeme přiřadit k libovolnému elementu v HTML.

begin – určuje čas začátku, u tohoto atributu máme několik možností zápisu:

- begin=10 – začne 10 sekund po rodičovském elementu
- begin=10;15;20 – začne v 10.,15. a 20. sekundě
- begin=elementName.click – začne po kliknutí na vybraný element
- begin=elementName.begin – začne současně s vybraným elementem
- begin=elementName.end – začne po ukončení vybraného elementu
- begin=5;elementName.click – začne v 5. sekundě, nebo vždy po kliknutí na vybraný element

Atribut begin je univerzálním atributem pro nastavování začátku animace. Existují i specializované atributy jako jsou např. beginWith či beginEvent, které také slouží k určování začátků animací. Tyto atributy jsou navázané na konkrétní událost.

beginWith – začne současně s elementem
beginAfter – začne po skončení elementu
beginEvent – začne při události např. onClick
dur – délka trvání v sekundách, pro nekonečno indefinite
end – určí čas konce
endWith – určuje současné ukončení
endEvent – ukončovací událost
repeat – určuje počet opakování, pro nekonečno: indefinite
repeatDur – určuje čas, po který se bude opakovat
timeAction – tento atribut slouží k nastavování viditelnosti elementu, k dispozici máme tyto hodnoty: display, visibility, style a onOff, defaultně je nastavena hodnota visibility. Jedná se o nejpoužitelnější hodnotu, díky níž můžeme v určitém čase zobrazit element. Za zmínku také stojí hodnota style, která nám umožní načtení inline stylu až ve chvíli, kdy to požadujeme.

5.3 Základní kostra dokumentu

Základní kostra dokumentu v HTML + TIME[8]

```
<html xmlns:t ="urn:schemas-microsoft-com:time">

<head>
<?import namespace="t" implementation="#default#time2">
<style>
  .time {behavior: url(#default#time2);}
</style>
</head>

<body>
  <div class="time">
  ...HTML+TIME
  </div>
</body>
</html>
```

[8] http://msdn2.microsoft.com/en-us/library/ms533099(VS.85).aspx 3.4.2008

6 XHTML+SMIL

Jedná se o umístění prezentací vytvořených pomocí jazyka SMIL 2.0 do internetových prohlížečů, především do internetového prohlížeče IE 6.0.

Z důvodu zbytečnosti opět nebyly implementovány všechny moduly SMILu 2.0. Jedná se o tyto moduly: layout, linking, structure a metainformation. Jsou nahrazeny XHTML elementy nebo pomocí CSS.

Tento profil se skládá z 19 modulů, které jsou rozděleny do 6 kategorií. Jednotlivé moduly a kategorie zde nebudu vypisovat, protože se shodují s moduly, které již byly uvedeny.

6.1 Základní kostra dokumentu

Základní kostra dokumentu psaného v XHTM+SMIL

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML+SMIL //EN"
"http://www.w3.org/2001/SMIL20/WD/xhtmlplussmil.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"
xmlns:smil="http://www.w3.org/2001/SMIL20">
```

Na webové stránce můžeme prezentaci zobrazit pomocí elementu embed:

```
<embed width=100% height=100% fullscreen=yes src="SMIL.smi" />
```

Toto řešení má velkou výhodu a tou je oddělení zdrojového kódu HTML od zdrojového kódu SMILu. Na druhé straně velká nevýhoda tohoto řešení spočívá v tom, že musíte mít nainstalovaný nějaký přehrávač (např. RealPlayer nebo QuickTime player), který je asociovaný pro soubory *.smi či *.smil.

Proto je lepší použít styl, tak jako v případě HTML+TIME

```
<head>
  <style>.time {behavior: url(#default#time2)}</style>
</head>
<body

</body>
</html>
```

7 Praktické ukázky

Tato kapitola obsahuje sbírku příkladů, které jsou v kontextu s jednotlivými profily jazyka SMIL.

7.1 SMIL 1.0

Zde je důležité připomenout, že k přehrávání prezentací vytvořených pomocí této technologie potřebujeme nějaký přehrávač (např. RealPlayer či QuickTime player).

Abychom mohli začít tvořit, musíme nejprve použít nějakou základní kostru. Základní kostra této specifikace SMILu je uvedena v kapitole 4.1. Tato základní kostra nám poslouží jako šablona pro naše příklady.

Abychom byli schopni zobrazit nějaký objekt v určitém čase, musí být uveden v hlavičce dokumentu párový element layout, který obsahuje nastavení vzhledu, nebo-li root-layout. Dále musí obsahovat alespoň jeden region, do kterého budeme umisťovat multimediální objekty nebo text.

7.1.1 Příklad 1

V našem příkladě jsme nastavili pomocí elementu root-layout šířku na 500 pixelů,výšku na 300 pixelů a pozadí na bílé. Dále jsme vytvořili jeden region s názvem logo, který je umístěn 30 pixelů od levého i horního okraje, je 40 pixelů vysoký, 436 pixelů široký a má nastavené žluté pozadí.

```
...
<head>
  <layout type="text/smil-basic-layout">
      <root-layout width="500" height="300" background-
color="#ffffff" />
      <region id="r1" left="30" top="30" height="40"
width="436"  background-color="#ffff00" ></region>
  </layout>
</head>
...
```

Do těla dokumentu umístíme nějaký objekt, v našem případě obrázek. Všimněme si atributu region, který určuje id regionu, do kterého se má obrázek umístit. Element img je vložen mezi párové elementy par, které nám říkají, že se bude jednat o paralelní průběh prezentace.

```
...
<body>
      <par>
       <img id="smil-logo" begin="3s" dur="5s" region="r1"
src="smil-logo.gif" />
      </par>
</body>
...
```

Obrázek má nastaveny čtyři velmi důležité atributy. Prvním z nich je region, který určuje místo, kde se obrázek zobrazí. Druhým je atribut dur, který určuje délku trvání, tedy jak dlouho má být obrázek zobrazen. Atribut begin určuje čas, ve kterém se má obrázek zobrazit. A atribut src určuje obrázek, tedy pomocí tohoto atributu nastavujeme cestu k požadovanému obrázku.

Po vložení této hlavičky a těla do základní kostry, vznikne funkční příklad prezentace o rozlišení 500x300 pixelů. Tento příklad (P1) je k dispozici ve sbírce příkladů, která je součástí této práce.

7.1.2 Příklad 2

Druhý příklad vznikne pouhou modifikací příkladu 1. Nejprve změníme rozměry prezentace a to z původní výšky 300 pixelů na 500 pixelů. Následně přidáme do hlavičky dokumentu druhý region, kterému pomocí atributu id, dáme název r2. Výsledná podoba hlavičky dokumentu je následující:

```
<head>
      <layout type="text/smil-basic-layout">
      <root-layout width="500" height="500" background-
color="#ffffff" />
      <region id="r1" left="30" top="30" height="40"
width="436"  background-color="#ffff00" ></region>
      <region id="r2" left="65" top="130" height="300"
width="370"  background-color="#ffff00" ></region>
       </layout>
</head>
```

Do těla dokumentu přidáme ještě jeden obrázek a pomocí atributu region jej umístíme do regionu2, tedy do r2. Dále mu nastavíme jinou délku trvání dur a pomocí atributu begin jej necháme zobrazit o něco později. Výsledná podoba těla dokumentu je následující:

```
<body>
  <par>
      <img id="smil-logo" begin="3s" dur="6s" region="r1"
src="smil-logo.gif" />
      <img id="obraz" begin="5s" dur="4s" region="r2"
src="smil-logo2.png" />
  </par>
</body>
```

Všimněme si délky trvání dur a atributu begin u obou obrázků. První má nastavený čas začátku na 3s a druhý na 5s, což způsobí, že druhý obrázek se zobrazí o 2 sekundy později. Délka trvání prvního obrázku je 6 sekund, zatímco u druhého pouze 4 sekundy. Tím je zajištěno současné zmizení obou obrázků. Tento příklad (P2) je také k dispozici ve sbírce příkladů, která je součástí této práce.

7.2 SMIL 2.0

K přehrávání prezentací, které jsou vytvořené pomocí technologie SMIL 2.0, potřebujeme nějaký přehrávač (např. RealPlayer či QuickTime player).

Základní kostra této specifikace SMILu je uvedena v kapitole 4.2. Tato základní kostra nám poslouží jako šablona pro naše příklady.

Nejprve opět musíme nastavit layout a nějaké regiony pro umisťování multimediálních objektů.

7.2.1 Příklad 1

Nejprve opět vytvoříme nějaký layout. Tento layout nastaví rozměry prezentace a vytvoří jeden region s názvem r1, který bude roztažený téměř přes celý dokument a bude mít nastavenou žlutou barvu pozadí. Zdrojový kód může vypadat například takto:

```
<layout type="text/smil-basic-layout">
   <root-layout width="500" height="500" background-
color="#ffffff" />
   <region id="r1" left="25" top="25" height="450" width="450"
background-color="#ffff00">
   </region>
</layout>
```

Do těla dokumentu opět vložíme nějaký obrázek, kterému nastavíme atribut region na r1, tedy jej umístíme do regionu s id r1.

```
<img id="obrazek1" region="r1" dur="10s" src="smil.gif" />
```

Dále připíšeme element animateMotion, který zajistí pohyb obrázku v regionu. Pomocí atributu targetElement určíme element, se kterým budeme hýbat. V našem příkladě jej nastavíme na obrázek1. Pak už jen stačí nastavit odkud kam se element s id obrazek1 má pohnout. To uděláme pomocí atributu from a atributu to. Nakonec už jen stačí zmínit, že atribut fill s hodnotou hold zajistí setrvání elementu ve výsledné pozici i po skončení animace.

```
<animateMotion begin="0s" targetElement="obrazek1" from="0,0"
to="0,30" dur="1s" fill="hold" />
```

Pomocí atributu targetElement můžeme hýbat s jakýmkoli objektem, který má nastavenou pozici. V našem příkladě můžeme hýbat i s celým regionem, stačí jen nastavit hodnotu atributu targetElement na název regionu, se kterým budeme pohybovat.

```
<animateMotion begin="4s" targetElement="r1" from="25,25"
to="500,25" dur="3s" fill="hold" />
```

Nesmíme zapomenout vložit tyto řádky do paralelní animace, tedy mezi párové elementy par. Výsledný kód těla dokumentu vypadá následovně:

```
<body>
   <par>
     <img id="obrazek1" region="r1" dur="10s" src="smil.gif" />
     <animateMotion begin="0s" targetElement="obrazek1"
from="0,0" to="0,30" fill="hold" dur="1s"  />
     <animateMotion begin="4s" targetElement="r1" from="25,25"
to="500,25" fill="hold" dur="3s"  />
   </par>
</body>
```

Funkční ukázka tohoto příkladu je uvedena ve sbírce příkladů ve složce SMIL 2.0 pod názvem P1.

7.2.2 Příklad 2

Nejprve musíme opět vytvořit nějaký layout prezentace. Pro naše účely můžeme použít layout z předchozího příkladu. Do hlavičky dokumentu ještě přidáme několik řádků.

```
<transition id="fade_1" dur="2s" type="fade" />
<transition id="star" dur="2s" type="starWipe"
subtype="fivePoint "/>
```

Tyto řádky slouží k nastavení přechodů, tedy k nastavení změn mezi jednotlivými prvky prezentace. Nejdůležitější atributy elementu region jsou: id, dur, type, popř. subtype. Atribut id slouží k pojmenování přechodu. Atribut dur

je délka trvání přechodu. Atributem type se nastavuje typ přechodu (např. fade [prolínání], snakeWipe či starWipe). U členitých typů přechodu se používá také atribut subtype, který určuje podkategorii přechodu. Například u typu přechodu starWipe můžeme použít atribut subtype, jehož možné hodnoty jsou: fourPoint, fivePoint a sixPoint.

Výsledná hlavička dokumentu se skládá z předchozího layoutu a ze dvou výše uvedených přechodů. Zdrojový kód hlavičky je následující:

```
<head>
<layout type="text/smil-basic-layout">
   <root-layout width="500" height="500" background-
color="#ffffff" />
   <region id="r1" left="25" top="25" height="450" width="450"
background-color="#ffff00"></region>
</layout>
<transition id="fade_1" dur="2s" type="fade" />
<transition id="star" dur="2s" type="starWipe"
subtype="fivePoint "/>
</head>
```

Do těla dokumentu vložíme nějaké obrázky. Pro lepší viditelnost přechodů použijeme větší obrázky než v předchozím příkladě. Velikosti obrázků jsou shodné s velikostí regionu, tedy jsou roztaženy přes celý region.

```
<img id="obrazek1" region="r1"
  transIn="fade_1" src="450x450.jpg" begin="2.00s" dur="5.00s"
/>
<img id="obrazek2" region="r1"
  transIn="fade_1" src="450x4502.jpg" begin="6.00s" dur="5.00s"
/>
<img id="obrazek3" region="r1"
  transIn="star" src="450x4503.jpg" begin="10.00s" dur="5.00s"
/>
```

Všechny tři obrázky jsou umístěny do stejného regionu r1. Avšak každý z nich má nastavenou jinou dobu zobrazení pomocí atributu begin. První dva obrázky jsou si velmi podobné, mají nastavený i stejný typ přechodu fade a liší se jen v barevném odstínu. Třetí obrázek má nastavený typ přechodu na hvězdu, kde jsme pomocí atributu subtype s hodnotou fivePoint určili, že hvězda bude pěticípá.

Podobně jako u předchozího příkladu nesmíme zapomenout umístit obrázky do sekvenční nebo do paralelní animace. Tento příklad je optimalizován pro paralelní animaci.

Pokud byste chtěli použít sekvenční animaci můžete dosáhnout stejných výsledků, ovšem doporučuji změnit hodnotu atributu begin na 0 sekund či jej smazat úplně.

Funkční ukázka tohoto příkladu je uvedena ve sbírce příkladů ve složce SMIL 2.0 pod názvem P2.

7.2.3 Příklad 3

Opět do stejného layoutu, který byl použit v příkladě 1 i 2, vložíme nějaké multimediální objekty. Pro celkové zjednodušení můžeme použít předchozí příklad a připsat následující řádek nad obrázky.

```
<audio id="muzika" begin="1s" dur="16" src="neco.mp3" />
```

Přidáním tohoto řádku docílíme toho, že nám v prezentaci bude hrát na pozadí mp3. Pomocí atributů clipBegin a clipEnd můžeme nastavit čas, od kterého má mp3 začít hrát. Chceme-li ukázku z nějaké mp3 nebo z nějakého video souboru, můžeme ji vystřihnout pomocí těchto atributů.

```
.. clipBegin="10s" clipEnd="22s" ..
```

V tomto případě se přehraje dvanáctisekundová ukázka, která začne v desáté sekundě a skončí ve dvaadvacáté sekundě.

Multimediální prvky nemusíme načítat pouze pomocí relativních odkazů z lokálního disku, ale také je můžeme načítat prostřednictvím sítě internet.

```
<audio id="muzika" begin="1s" dur="16"
src="http://www.play.cz/radio/beat128.mp3.m3u" />
```

Funkční ukázky těchto příkladů jsou uvedeny ve sbírce příkladů ve složce SMIL 2.0 pod názvem P3.

7.3 HTML+TIME

K přehrávání prezentací, které jsou vytvořené pomocí této technologie, nepotřebujeme žádný přehrávač. Vystačíme si s internetovým prohlížečem Internet Explorer. Základní kostra této specifikace SMILu je uvedena v kapitole 5.3. Tato základní kostra nám poslouží jako šablona pro naše příklady.

Zde je důležité si uvědomit, že u HTML+TIME nemusíme nastavovat layout a regiony, neboť nastavení této části prezentace uděláme prostřednictvím kaskádových stylů CSS.

7.3.1 Příklad 1

Nejprve vytvoříme nějaký objekt, který bude sloužit jako kontejner pro další elementy či multimediální objekty. V našem příkladě jím bude element div. Tomuto elementu přiřadíme inline styl, který bude určovat barvu pozadí, rozměry prezentace atd.

```
<div id="slide1" style="border: 2px solid black; padding: 10px;
position: relative; width: 800px; margin: 10px; height:600px;
background-color: #CCFF99;">
      ... sem budeme vkládat objekty ...
</div>
```

Pomocí tohoto divu jsme nastavili základní vzhled prezentace. Do tohoto elementu vložíme nějaký nadpis. Pro tyto účely použijeme element h1, kterému pomocí atributu id přiřadíme název nad1 a pomocí inline stylu mu nastavíme barvu pozadí na světle zelenou.

```
<h1 id="nadpis1" style="color: white">Specifikace jazyka</h1>
```

Element animateColor umožňuje plynulou změnu z jedné barvy v barvu druhou prostřednictvím atributů from a to. Stačí jen nastavit hodnoty těchto atributů např. na žlutou a červenou a výsledek bude zobrazen jako plynulý přechod ze žluté, oranžové a jejich všech odstínů až po červenou barvu.

Pomocí atributu targetElement a atributu atributeName, který je nastaven na hodnotu color, jednoznačně určíme element a atribut, na který se má efekt aplikovat. Atribut dur nám v tomto případě určuje počet sekund, kterých bude potřeba ke změně z jedné barvy na barvu druhou. Použití atributu autoReverse způsobí přehrání efektu i v opačném pořadí. A atribut repeatCount, který je v našem příkladě nastaven na hodnotu indefinite, způsobí nekonečné opakování efektu.

```
<t:animateColor autoReverse="true" targetElement="nadpis1"
dur="3s" attributeName="color" from="gold" to="red" fill="hold"
repeatCount="indefinite" />
```

Tím bychom měli vytvořený nadpis, který plynule mění svoji barvu. Dále vytvoříme tabulku, která bude mít jeden sloupec a dva řádky. Pomocí atributu width nastavíme šířku tabulky. Tato tabulka s nastavenou šířkou nám zjednoduší formátování textu, neboť text se bude lámat dle nastavené šířky. Do obou řádků vložíme nějaký text pomocí elementu span. Pomocí již známých atributů jim nastavíme různé názvy, délku trvání a dobu, ve kterou se mají zobrazit. Aby vše fungovalo, jak má, musíme elementům span nastavit třídu time.

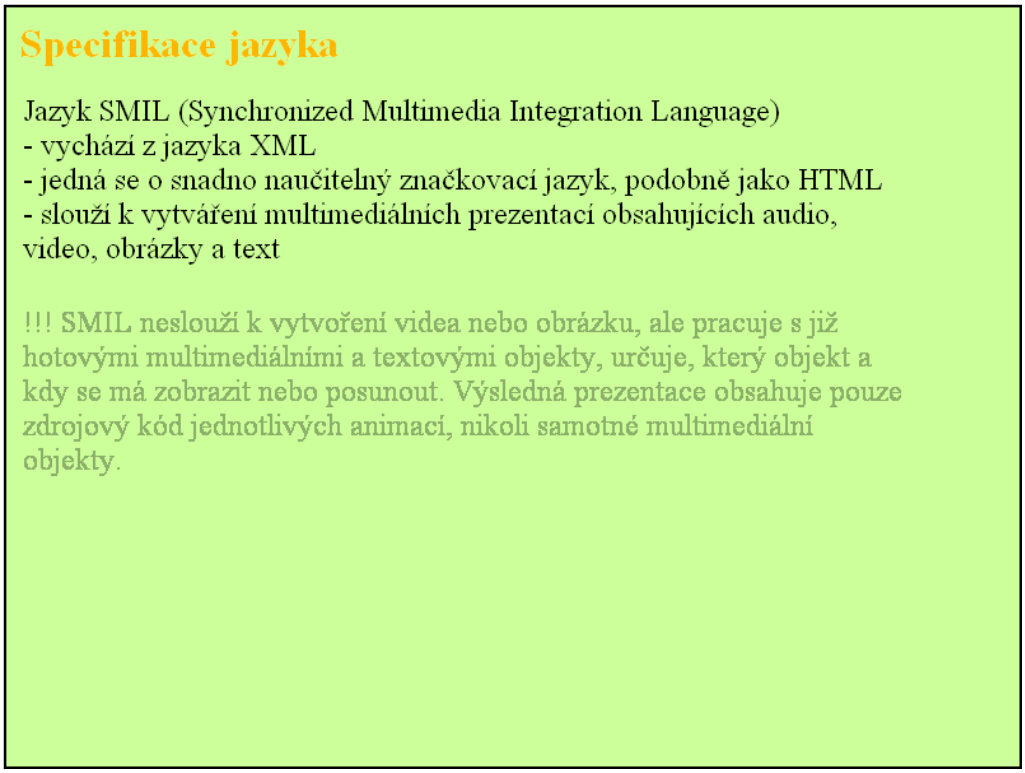
```
<table width="700"><tr><td>
<span id="span1" begin="nadpis1.onClick" style="width:700px;
font-size:18pt;" class="time">
Jazyk SMIL (Synchronized Multimedia Integration Language)<br />
- vychází z jazyka XML<br />
- jedná se o snadno naučitelný značkovací jazyk, podobně jako
HTML<br />
- slouží k vytváření multimediálních prezentací obsahujících
audio, video, obrázky a text <br /><br />
</span>
</td></tr><tr><td>

      <span id="span2" begin="span1.begin+5"
style="width:700px; font-size:18pt;" class="time" >
      !!! SMIL neslouží k vytvoření videa nebo obrázku, ale
pracuje s již hotovými multimediálními a textovými objekty,
určuje, který objekt a kdy se má zobrazit nebo posunout.
Výsledná prezentace obsahuje pouze zdrojový kód jednotlivých
animací, nikoli samotné multimediální objekty.
</span>
</td></tr></table>
```

Důležitým atributem, který je třeba připomenout je atribut begin. V našem příkladě je jeho hodnota nastavena na událost onClick, která je navázána na element nadpis. Což znamená, že element s názvem span1 se objeví přesně ve chvíli, kdy klikneme na barevně měnící se nadpis.

Nakonec přidáme ještě dva řádky, kterými nastavíme přechody nebo-li transitions. Tyto řádky můžeme umístit buď nad, nebo pod tabulku.

```
<t:transitionfilter targetelement="span1" type="fade" dur="2"
begin="nadpis1.onclick" />
<t:transitionfilter targetelement="span2" type="fade" dur="2"
begin="span1.begin+5" />
```

Protože je tento příklad trochu komplikovanější než předchozí příklady, uvádím zde kompletní zdrojový kód tohoto příkladu. Budete-li postupovat přesně tak, jak popisuji výše, měli byste dosáhnout totožného výsledku.



Výše uvedený obrázek ukazuje, jak vypadá tento příklad v internetovém prohlížeči Internet Explorer.

Kompletní zdrojový kód tohoto příkladu:

```
<html xmlns:t ="urn:schemas-microsoft-com:time">

<head>
<?import namespace="t" implementation="#default#time2">
<style>
  .time {behavior: url(#default#time2);}
</style>
</head>

<body>

<div id="slide1" style="border: 2px solid black; padding: 10px;
position: relative; width: 800px; margin: 10px;
height:600px;background-color: #CCFF99;">
<h1 id="nadpis1" style="color: white">Specifikace jazyka</h1>
<t:animateColor autoReverse="true" targetElement="nadpis1"
dur="3s" attributeName="color" from="gold" to="red" fill="hold"
repeatCount="indefinite" />

<table width="700"><tr><td>
<span id="span1" begin="nadpis1.onclick"
style="width:700px;font-size:18pt;" class="time">
Jazyk SMIL (Synchronized Multimedia Integration Language)<br />
- vychází z jazyka XML<br />
- jedná se o snadno naučitelný značkovací jazyk, podobně jako
HTML<br />
- slouží k vytváření multimediálních prezentací obsahujících
audio, video, obrázky a text <br /><br />
</span>
</td></tr>


<tr><td>
<span id="span2" class="time" style="width:700px;font-
size:18pt;" begin="span1.begin+5">
!!! SMIL neslouží k vytvoření videa nebo obrázku, ale pracuje s
již hotovými multimediálními a textovými objekty, určuje, který
objekt a kdy se má zobrazit nebo posunout. Výsledná prezentace
obsahuje pouze zdrojový kód jednotlivých animací, nikoli
samotné multimediální objekty.
</span>
</td></tr></table>
<t:transitionfilter targetelement="span1" type="fade" dur="2"
begin="nadpis1.onclick" />
<t:transitionfilter targetelement="span2" type="fade" dur="2"
begin="span1.begin+5" />

</div>

</body>
</html>
```

7.3.2 Příklad 2

V tomto příkladě si ukážeme, jak aplikovat dva přechody na jeden element. Nejprve vytvoříme nějaký objekt, na který budeme moct přechody aplikovat.

```
<div class="time" id="div1" style="background-color:silver;
width:520px; font: italic bold large arial; color:red;
padding:10px;">Na tento div jsou aplikovány dva přechody.</div>
```

Elementu div jsme přiřadili třídu time, nastavili atribut id na div1 a pomocí inline stylu jsme nastavili barvu pozadí, použité písmo atd.

Nyní vytvoříme dva různé přechody, kterým nastavíme atribut targetElement a atribut begin na stejnou hodnotu. Tedy určíme, že oba přechody se mají současně aplikovat na element s názvem div1.

První z přechodů nám umožní postupné zobrazení zleva doprava.

```
<t:transitionFilter begin="div1.begin" type="barWipe" dur="5"
targetElement="div1"/>
```

Druhý přechod tzv. fade způsobí prolnutí celého divu.

```
<t:transitionFilter begin="div1.begin" type="fade" dur="5"
targetElement="div1"/>
```

Výsledek je pak zobrazen jako postupně načítající se div zleva doprava, přičemž navíc postupně přechází ze světlé do tmavší barvy (fade).

7.3.3 Příklad 3

Abychom si usnadnili práci, použijeme k vytvoření další ukázky zdrojový kód z předchozího příkladu. V tomto příkladě budeme pohybovat elementem div pomocí elementu animateMotion a také si ukážeme rozdíl mezi sekvenční a paralelní animací. Abychom mohli pohybovat elementem po obvodu čtverce, musíme Element animateMotion použít vícenásobně, tedy pro

každou stranu čtverce jeden. Do těla dokumentu, hned pod řádek s druhým přechodem (fade), přidáme následující konstrukci, která představuje sekvenční průběh animace. Konkrétní nastavení atributů v této sekvenční animaci nám zajistí spuštění animace ve chvíli, kdy klikneme na element s nazvem div1. Dále atribut repeatCount zajistí nekonečné opakování animace. Jelikož zajistíme, aby se element div vrátil po skončení animace na své původní místo, nemusíme přidávat atribut fill.

```
<t:seq begin="div1.onClick" repeatCount="indefinite">

</t:seq>
```

Do této konstrukce vepíšeme čtyřikrát element animateMotion a každý bude mít nastaven atribut targetElement a dur na stejné hodnoty. Důležitou roli hraje atribut fill s nastavenou hodnotou na hold, který zajistí setrvání elementu div na výsledné pozici.

První z elementů animateMotion má za následek posun elementu div dolu o 300 pixelů, pomocí atributu dur bude tento přesun trvat čtyři sekundy.

```
<t:animateMotion dur="4s"  targetElement="div1" by="0,300"
fill="hold"/>
```

Podobně je na tom i druhý z elementů, který posune element div doprava o 300 pixelů. Tento přesun bude opět trvat čtyři sekundy.

```
<t:animateMotion dur="4s"  targetElement="div1" by="300,0"
fill="hold"/>
```

Nyní nám zbývají dva elementy. První z nich slouží pro přesun směrem nahoru a druhý pro přesun směrem doleva, čímž vrátíme div do předchozí pozice.

```
<t:animateMotion dur="4s"  targetElement="div1" by="0,-300"
fill="hold"/>
<t:animateMotion dur="4s"  targetElement="div1" by="-300,0"
fill="hold"/>
```

Tento příklad můžeme ještě modifikovat a to přepsáním sekvenční animace na paralelní. Nahrazením elementu seq za element par, však dosáhneme jen toho, že po kliknutí na element div se element pouze rozklepe. To se sice může někdy hodit, například pro zvýraznění reklamního textu či obrázku, ale v našem příkladě je tento efekt nežádoucí. Abychom dosáhli stejného výsledku jako v případě použití sekvenční animace, museli bychom přidat každému elementu animateMotion atribut begin. U prvního elementu animateMotion nemusí být nastaven atribut begin, neboť hodnota nula sekund nesmí být uvedena. Pokud bychom atribut chtěli i přesto použít, musíme mu nastavit hodnotu blízkou nule (např. 0.001). Atributy dalších elementů jsou nastaveny na čtyři, osm a dvanáct sekund.

```
<t:animateMotion           dur="4s"  targetElement="div1"
by="0,300"  fill="hold"/>

<t:animateMotion begin="4s" dur="4s"  targetElement="div1"
by="300,0"  fill="hold"/>

<t:animateMotion begin="8s" dur="4s"  targetElement="div1"
by="0,-300"  fill="hold"/>

<t:animateMotion begin="12s" dur="4s"  targetElement="div1"
by="-300,0"  fill="hold"/>
```

Funkční ukázky a kompletní zdrojové kódy těchto příkladů jsou uvedeny ve sbírce příkladů ve složce HTML+TIME pod názvem příklad 3.

8 Srovnání technologií SMIL, JavaScript a Flash

V této kapitole bych se rád věnoval i jiným technologiím, které se také používají k vytváření animací a které pak dále můžeme umístit na webové stránky. Velká výhoda technologií Flash a JavaScript spočívá v tom, že jsou podporovány internetovými prohlížeči. Animace, které jsou napsané v JavaScriptu, se zobrazí ve všech nejznámějších prohlížečích. Podobné je to i u technologie Flash. Pomocí programu Adobe Flash (dříve Macromedia Flash) se dá poměrně snadno a rychle vytvořit téměř jakákoli animace, která je následně exportována do swf souboru. Tento soubor pak můžeme pomocí elementu embed či elementu object umístit do webové stránky. Musíme si ale uvědomit, že podpora technologie Flash není součástí webových prohlížečů, ale teprve pomocí pluginu či externích přehrávačů (jako je např. Adobe Flash Player) jsou animace umístěny do webových stránek.

8.1 Flash

„Jedná se o program schopný vytvářet vektorové animace opatřené zvukovými efekty. Vektorová grafika má oproti bitmapové dvě hlavní výhody. Tou první je možnost zvětšování či zmenšování bez ztráty ostrosti a kvality a tou druhou výhodou - a pro web docela podstatnou - je velikost výsledného souboru a ta je řádově nižší než např. u animovaného gifu.“[9]

8.2 JavaScript

„JavaScript je programovací jazyk, který se používá v internetových stránkách. Zapisuje se přímo do HTML kódu, což je velká výhoda, protože je to jednoduché. JavaScript je klientský skript. To znamená, že se program odesílá se stránkou na klienta (do prohlížeče) a teprve tam je vykonáván.“[10]

[9] KALDA, Martin. Macromedia FLASH 4 - úvod. *Interval.cz* [online]. 1999 [cit. 2008-04-12]. Dostupný z WWW: <http://interval.cz/clanky/macromedia-flash-4-uvod/>.
[10] JANOVSKÝ, Dušan. *Jakpsatweb.cz : Úvod do JavaScriptu* [online]. 1998 [cit. 2008-04-12]. Dostupný z WWW: <http://www.jakpsatweb.cz/javascript/javascript-uvod.html>.

8.3 Ukázka

Na této ukázce si ukážeme srovnání technologie SMIL s technologiemi JavaScript a Flash. Ukážeme si, jak pomocí těchto tří různých technologií vytvořit stejný příklad.

SMIL	Java Script	Flash
		
SMIL	Java Script	Flash
		

Nejprve začneme technologií SMIL. Pomocí této technologie můžeme snadno popsat ve dvou až třech řádcích změnu velikosti objektu. V JavaScriptu je to dost podobné, stačí správně použít metodu onClick a výsledek je také jen na dva řádky. Ovšem pokud budeme chtít plynulou změnu velikosti obrázku u JavaScriptu, zdrojový kód bude značně rozsáhlý. Budeme nuceni napsat několik funkcí či metod a pokud si v průběhu programování vzpomeneme, že by bylo lepší udělat animaci i v opačném směru, délka kódu se zdvojnásobí. Technologie Flash a SMIL jsou na tom v tomto ohledu mnohem lépe. Pomocí programu Adobe Flash můžeme dynamicky vytvářet i měnit průběh animace.

U technologie SMIL máme k dispozici atribut autoReverse, který zajistí zpětný chod animace.

8.3.1 Ukázka – část 1

V první části programu je ukázka jednorázové změny velikosti obrázku. Po kliknutí na obrázek dojde k okamžité změně velikosti obrázku.

Technologie SMIL

Pomocí technologie SMIL musíme nejprve do HTML stránky vložit obrázek, na který budeme efekty aplikovat. To uděláme pomocí elementu img. Aby technologie SMIL mohla správně fungovat, musíme k obrázku přiřadit třídu time a nastavit některé potřebné atributy.

```
<img id="kostka" src="cube.jpg" class="time"
style="height:72px; width: 72px;" />
```

Do paralelní animace přidáme dva řádky. První z nich slouží pro změnu obrázku ve vertikálním směru a druhý pro změnu obrázku ve směru horizontálním. Díky tomu, že jsou tyto řádky umístěny mezi párové elementy par, provedou se tyto dvě změny současně.

```
<t:par begin="kostka.onClick" fill="hold">
 <t:animate targetElement="kostka" dur="0.01s"
attributeName="height" to="152" fill="hold" />
 <t:animate targetElement="kostka" dur="0.01s"
attributeName="width" to="152" fill="hold" />
</t:par>
```

Pomocí atributu dur, v tomto případě nastaveném na 10ms, jsme obrázkům zajistili změnu, která proběhne během 10ms, což je pro lidské oko tak krátký okamžik, že se nám změna jeví jako jednorázová.

Technologie JavaScript

U technologie JavaScript je jednorázová změna velikosti obrázku velmi jednoduchá. Pomocí integrovaných funkcí můžeme snadno na událost onClick nastavit požadovanou změnu. Nejprve musíme do HTML stránky vložit nějaký obrázek, což uděláme pomocí elementu img.

```
<img src="cube.jpg" width="76" height="76" name="kostka2" />
```

Nyní události onClick přiřadíme integrované funkce JavaSccriptu pro změny velikosti obrázku.

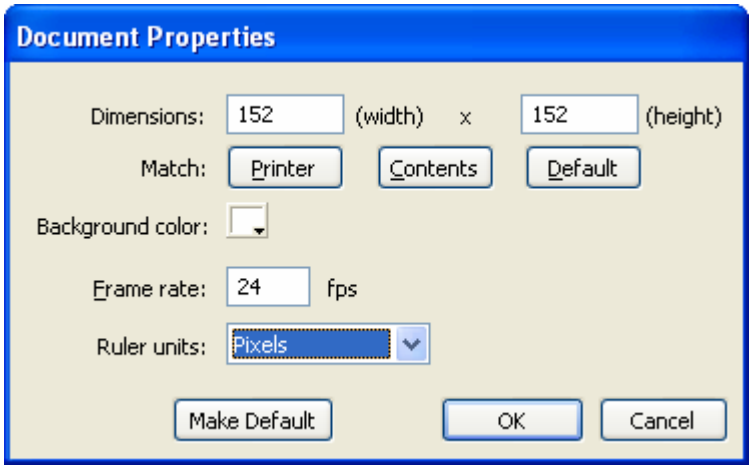
```
onclick="javascript:kostka2.height=152;kostka2.width=152"
```

Tak jak je uvedeno výše, je JavaScript nejvýhodnějším řešením této části příkladu, neboť výsledek se skládá pouze z těchto řádků.

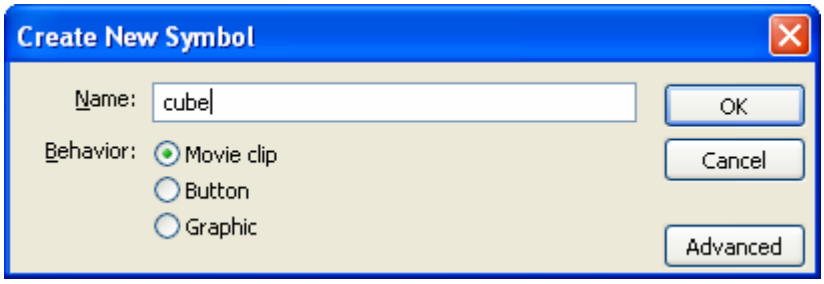
```
<img src="cube.jpg" width="76" height="76" name="kostka2"
onclick="javascript:kostka2.height=152;kostka2.width=152"/>
```

Technologie Flash

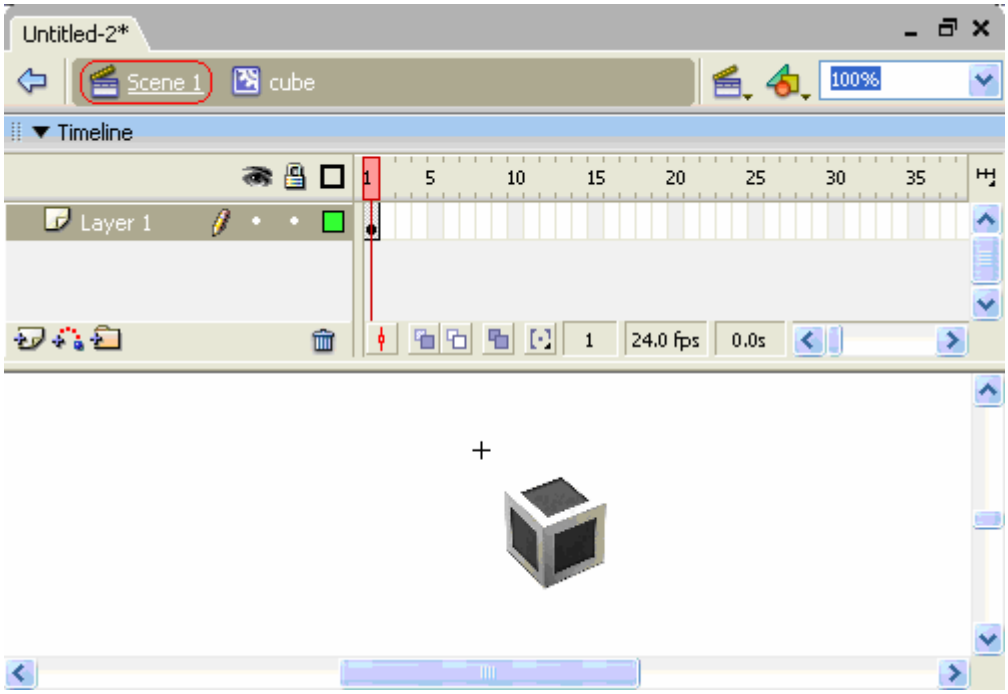
Technologie Flash umožňuje snadnou změnu velikosti obrázku. Tuto práci nám ve velké míře usnadňuje vývojové prostředí Adobe Flash. Pomocí tohoto programu nejprve nastavíme v sekci Dokument Properties počet snímků za sekundu, barvu pozadí a rozměry animace.



Poté co jsme nastavili rozměry animace, vytvoříme symbol Movie clip (CTRL+F8) s názvem cube.

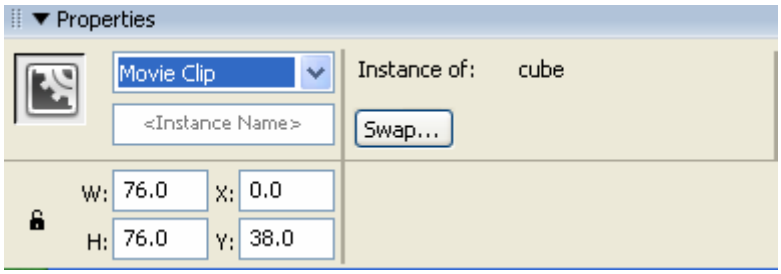


Nyní načteme do knihovny požadovaný obrázek (File -> Import -> Import to Library), odkud ho následně vložíme do tohoto Movie clipu s názvem cube. Z editace Movie clipu se zpět přepneme do scény. To uděláme pomocí červeně zvýrazněného odkazu s názvem scene1.



Do prvního keyframu časové osy scény přetáhneme z knihovny souborů náš Movie clip cube. Nejprve mu budeme muset nastavit rozměry a pozici, což můžeme udělat prostým roztažením a přesunutím myší. Ovšem mnohem lepší

je nastavit číselné hodnoty těchto parametrů a to tak, jak je vidět na následujícím obrázku.



Nyní Movie clipu cube přidáme actionScript, který bude zajišťovat spuštění animace pouze ve chvíli, kdy na něj klikneme. To uděláme pomocí následujících řádků.

```
on (press)
{
      _root.play();
}
```

Do časové osy scény scene1 přidáme druhý keyframe. Oběma těmto keyframům přidáme pomocí actionScriptu funkci stop();. Umístěním této funkce do obou framů potlačíme automatické spuštění a nekonečné opakování animace.

```
stop();
```

8.3.2 Ukázka – část 2

V druhé části programu je ukázka plynulé změny velikosti obrázku. Po kliknutí na obrázek dojde k pozvolné změně velikosti obrázku.

Technologie SMIL

Pomocí technologie SMIL musíme nejprve do HTML stránky vložit obrázek. To uděláme pomocí následujícího řádku.

```
<img id="kostka3" src="cube.jpg" class="time"
style="height:72px; width: 72px;" />
```

Do paralelní animace opět vložíme dva řádky, které slouží ke změně rozměrů objektu.

```
<t:par begin="kostka3.onClick" fill="hold">
 <t:animate targetElement="kostka3" dur="1s"
attributeName="height" to="152" fill="hold" />
 <t:animate targetElement="kostka3" dur="1s"
attributeName="width" to="152" fill="hold" />
</t:par>
```

Jediná změna oproti předchozímu příkladu spočívá v jiném nastavení hodnoty atributu dur. Z původních 10ms jsme jej nyní nastavili na 1s, což znamená, že změna velikosti obrázku proběhne během 1s.

Technologie JavaScript

U technologie JavaScript je plynulá změna velikosti obrázku poměrně komplikovaná. Nevystačíme si s integrovanými funkcemi a bude potřeba trochu více programovat. K tomuto účelu použijeme zdrojový kód Jiřího Zachara[11]

Pokud již máme napsané funkce pro měnění velikosti obrázku, můžeme do HTML stránky vložit obrázek, kterému nastavíme událost onClick následovně:

```
onclick="resize_plus('imageResize', 152);"
```

Výsledné vložení obrázku do HTML stránky vypadá takto:

```
<img src="cube.jpg" border="0" width="76px" id="imageResize"
style="border: 0px" align="center"
onclick="resize_plus('imageResize', 152);">
```

[11] http://www.zaachi.com/cs/items/javascript-plynula-zmena-velikosti-obrazku.html 12.4.2008

Upravený zdrojový kód Jiřího Zachara[12]:

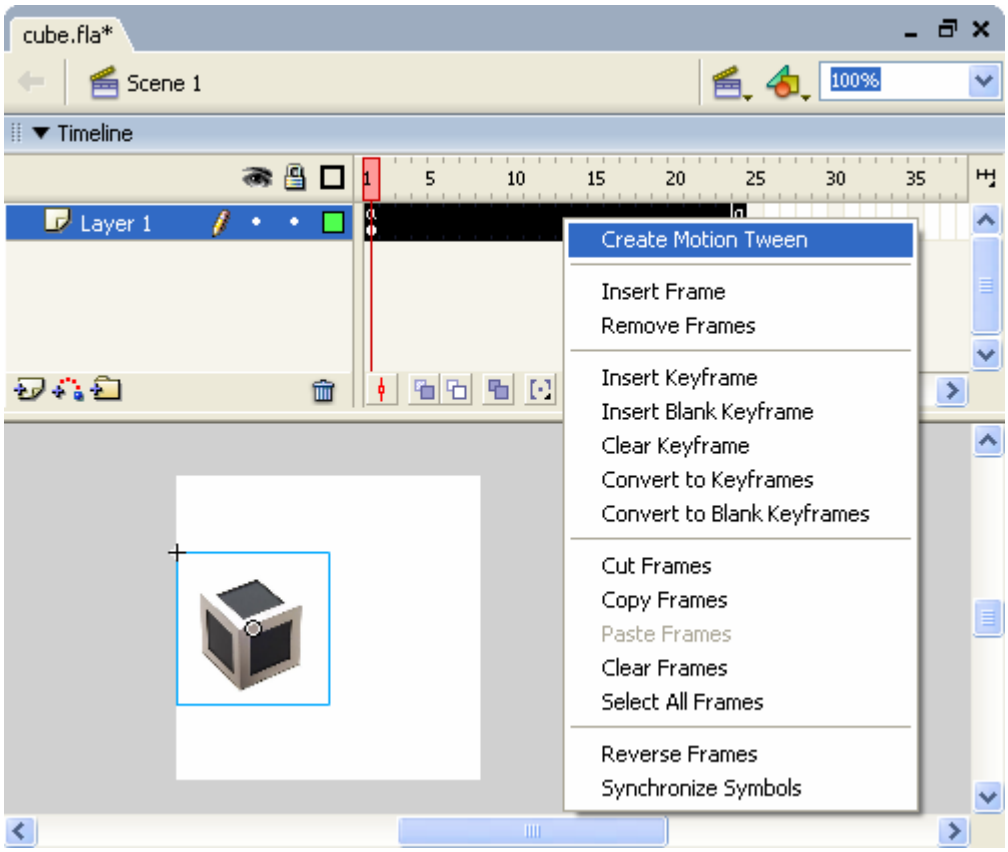
```
<script language="JavaScript">

function image_width_plus( img_width, id, max ) {
    timer = null;
    //objekt z daneho identifikatoru
    idImg = document.getElementById(id);
    //podminka pro porovnani velikosti
    if(img_width <= max ) {
        //v pripade pravdy vytvorime novou velikost
        img_width += 3;
        //a nastavime nvoou velikost
        idImg.style.width = ( img_width ) + "px";
        //rekurzivne volame funkci pro zvetsovani obrazku
        timer  =  setTimeout("image_width_plus("  +  img_width  +
","'" + id + "', " + max + ");", 30);
    } else {
        //clearTimeout('timer');
        return true;
    }
}

function resize_plus( id, max ){
    //objekt z daneho identifikatoru
    idImg = document.getElementById(id);
    //volame funkci pro zvetseni obrazku
    //jako prvni parametr uvedeme soucasnou velikost obrazku
    image_width_plus( idImg.width, id, max );
}

function paint_images(){
    //vytvorime pole pro nove velikosti obrazku
    var values = new Array(123, 145, 100, 145, 88, 40, 110, 120
);
    //v cyklu zavolame funkci pro zvetseni obrazku
    for( i = 1; i < 9; i++ ){
        resize_plus("a" + i, values[i-1]);
    }
}

function set_default(){
    //v  cyklu  volame  metodu  pro  zmenseni  obrazku  na  hodnotu
width 30px
    for( i = 1; i < 9; i++ ){
        resize_minus("a" + i, 30);
    }
}

</script>
```

[12] http://www.zaachi.com/cs/items/javascript-plynula-zmena-velikosti-obrazku.html 12.4.2008

Technologie Flash

Technologie Flash umožňuje snadno vytvořit i plynulou změnu velikosti obrázku. Tu vytvoříme editací předchozího příkladu. Pouhým přidáním jednadvaceti framů mezi první a druhý keyframe získáme animaci, jejíž délka bude 1s. V našem případě jsme docílili toho, že po kliknutí na Movie clip dojde k jednorázové změně velikosti obrázku, ale až po jednosekundové prodlevě.

Nyní už jen stačí označit všechny framy animace a vytvořit Motion Tween, což uděláme pomocí pravého tlačítka myši.



Výsledek už pak stačí jen exportovat do swf souboru, který následně pomocí párového elementu <object> vložíme do webové stránky.

Tento příklad je k dispozici ve sbírce příkladů ve složce HTML+TIME pod názvem p4.

9 ZÁVĚR

V porovnání s technologiemi Adobe Flash a JavaScript patří jazyk SMIL k jednoduchým značkovacím jazykům. Díky jednoduché syntaxi tohoto jazyka můžeme snadno popsat i složitější animaci.

Jazyk SMIL je novým značkovacím jazykem, vývoj tohoto jazyka pořád pokračuje. V lednu tohoto roku vyšla další revize jazyka SMIL. Tato nová verze 3.0 rozšiřuje verzi SMILu 2.1 o několik dalších modulů.

Největší přínos této technologie spočívá v tom, že pomocí jazyka SMIL můžeme vytvořit totožné prezentace jako např. v technologii Flash, ale s tím rozdílem, že prezentace vytvořená ve SMILu bude mít menší nároky na hardware počítače, zejména na procesor. Z tohoto důvodu se SMIL uplatňuje zejména v mobilních technologiích, kde je výkon mnohem důležitější než v oblasti počítačů.

Hlavní nevýhoda jazyka SMIL spočívá v tom, že se jedná o novou málo známou technologii, díky čemuž je podpora ve webových prohlížečích velmi slabá. Jediný Internet Explorer podporuje rozšíření webových stránek o třídu time. Nezbývá nám nic jiného než doufat, že absence zásuvných modulů do prohlížečů Opera a FireFox, nebude trvat věčně. Existuje však Java Applet, který umožňuje spouštět prezentace jazyka SMIL na webových stránkách, ovšem rychlost tohoto Java Appletu není nejideálnější a navíc tento Applet podporuje jen základní prvky jazyka SMIL verze 1.0.

Cílem této práce bylo seznámit uživatele internetu a webmastery s touto novou technologií a také jim umožnit vytvářet vlastní multimediální prezentace pomocí této uživatelské příručky. Čím větší ohlas bude tato technologie mít, tím dříve se můžeme těšit na podporu ostatních webových prohlížečů.

10 LITERATURA

http://www.w3.org/TR/NOTE-HTMLplusTIME 3.3.2008

http://www.w3.org/TR/XHTMLplusSMIL/ 4.3.2008

http://www.w3.org/TR/2005/REC-SMIL2-20050107/smil-modules.html 8. 3.2008

http://www.w3.org/TR/2005/REC-SMIL2-20050107/ 8.3.2008

http://www.w3.org/2005/SMIL21/SMIL21.dtd 23.3.2008

http://www.w3.org/TR/2005/REC-SMIL2-20051213/smil-transitions.html 23.3.2008

http://www.w3.org/TR/2005/REC-SMIL2-20051213/smil-modules.html 23.3.2008

http://interval.cz/clanky/smil-timed-interactive-multimedia-extensions-for-html/ 4. 3. 2008

http://www.w3.org/AudioVideo/ 4.3.2008

http://www.w3.org/TR/REC-smil/ 6.1.2008

http://www.w3.org/TR/REC-smil/#dtd/ 6. 1. 2008

http://www.w3.org/TR/2005/REC-SMIL2-20050107/introduction.html/ 6. 1. 2008

http://interval.cz/clanky/smil-zakladni-elementy-a-konstrukce/ 1. 3. 2008

http://interval.cz/clanky/smil-jazyk-pro-multimedialni-prezentace/ 2.3.2008

http://interval.cz/clanky/smil-tvorba-prezentace-v-praxi/ 3.3.2008

http://interval.cz/clanky/smil-animace-a-prezentacni-efekty/ 3.3.2008

http://www.w3.org/TR/2005/REC-SMIL2-20051213/smil21-mobile-profile.html/ 6. 1. 2008

http://www.w3.org/TR/2005/REC-SMIL2-20051213/smil21-extended-mobile-profile.html/ 6. 1. 2008

http://www.microsoft.com/cze/windows/ie/features.mspx 8.4.2008

http://www.zaachi.com/cs/items/javascript-plynula-zmena-velikosti-obrazku.html 12.4.2008

http://interval.cz/clanky/macromedia-flash-4-uvod/ 12.4.2008

http://www.jakpsatweb.cz/javascript/javascript-uvod.html 12.4.2008

http://msdn2.microsoft.com/en-us/library/ms533099(VS.85).aspx 3.4.2008