UNIVERZITA PALACKÉHO V OLOMOUCI
PEDAGOGICKÁ FAKULTA

Katedra technické a informační výchovy

Karolína Šmatlová

III. ročník - prezenční studium

Obor: Český jazyk a základy technických věd

Tvorba a prvky multimediálního webu prostřednictvím specifikace HTML 5

Bakalářská práce

Vedoucí práce: Mgr. Jan Kubrický, Ph.D.

OLOMOUC 2014

Čestné prohlášení

Prohlašuji, že jsem práci vypracovala samostatně a použila jen uvedených pramenů.

V Olomouci dne 8. 4. 2014

…….

Karolína Šmatlová

Poděkování

Poděkovat bych chtěla hlavně Mgr. Janu Kubrickému, Ph. D. za odborné rady a trpělivý přístup, hlavně při dokončování bakalářské práce.

Obsah

Úvod

Bakalářská práce pojednává o multimediálních objektech vytvářených specifikací HTML5 a o možnostech využití vytvořených multimediálních objektů.

Hlavním cílem teoretické části práce je analyzovat možnosti tvorby a využití multimediálních prvků s využitím standardu HTML5.

Cílem praktické části je představit možnosti práce s rozhraním Canvas, práci s multimediálním prvkem video pomocí specifikace HTML5 a XHTML a základní rozdíly ve struktuře zdrojového kódu specifikace HTML5 a XHTML.

Multimediální prvky jsou již stálou součástí internetových stránek. Pro práci s multimediálními prvky vznikají nové technologie. Předmětem této práce se staly multimediální prvky vytvořené pomocí specifikace HTML5.

Specifikace HTML5 přináší nové prvky pro vytvoření grafiky. Jeden z multimediálních prvků pro tvorbu grafiky na internetových stránkách je rozhraní Canvas, které pracuje s pomocí JavaScriptu. Dalším zajímavým multimediálním prvkem je video a audio. Zaměřením na rozhraní Canvas a element <video> chci poukázat na zajímavé možnosti tvorby internetových stránek v jazyce HTML5. U specifikace HTML5 nejsou elementy <canvas> a <video> jedinými možnostmi grafiky. Dalo by se říct, že specifikace HTML5 se hodně zaměřuje na práci s multimediálními prvky.

Teoretická část

1 Specifikace HTML5

Hypertext Markup Language 5 je nejnovější a stále se vyvíjející specifikace pro tvorbu internetových stránek, která se označuje zkratkou HTML5. V kapitole specifikace HTML5 se zaměřím na vývoj jazyka HTML5 a na budoucnost jazyka HTML5.

Pro orientaci v bakalářské práci vysvětluji pojmy, které používám pro osoby, které určitým způsobem pracují s HTML5:

- **Vývojáři HTML5** – lidé nebo týmy, kteří se podílí přímo na vývoji specifikace HTML5.
- **Tvůrci internetových stránek** – skupiny lidí, kteří vytvářejí internetové stránky.
- **Uživatelé internetových stránek** – lidé, kteří navštěvují vytvořené internetové stránky.

1.1 Vývoj HTML5

Jazyk HTML se začal vyvíjet v 90. letech 20. století. V této době se představilo hned několik verzí jazyka HTML a to verze 2.0, 3.2, 4.0, 4.01. Vývoj HTML se následně zastavil. Vývoj webových standardů se přesunul k vývoji XML a XHTML. [3, 6]

HTML dlouho nezůstává v pozadí. V roce 2004 malá skupina lidí založila WHATWG (Web Hypertext Application Technology Working Group) a vytvořila specifikaci HTML5. Vývojáři se zaměřili na oblasti internetových aplikací, které podle nich strádají nejvíce. V roce 2006 se k vývoji HTML5 přidává konsorcium W3C. Nový pracovní návrh HTML5 zveřejňují v roce 2008. O rok později přichází na trh XHTML2. [4, 6]

Specifikace HTML5 řeší velmi praktické problémy, a proto se začínají k HTML5 přiklánět vývojáři internetových prohlížečů. Experimenty na straně internetových prohlížečů přináší značné zpětné vazby pro vývojáře HTML5. Práce na specifikaci HTML5 pořád ještě neskončila. Vývojáři vydávají prohlášení, že by chtěli přesně definovat jazyk HTML5 do roku 2022. [6]

1.2 Vývojáři HTML5

Na vývoji specifikace HTML5 se podílejí tři světové organizace.

Organizace WHATWG (Web Hypertext Application Technology Working Group), která stála u vzniku HTML5. Organizace se nezaměřuje jen na vývoj HTML, ale také na vývoj rozhraní API. [6]

Další organizací je konsorcium W3C (WordlWide Web Consortium). Konsorcium W3C se podepsalo pod vývojem HTML. Na chvíli od vývoje HTML odstoupila, ale jak už jsem se zmínila ve vývoji HTML5 v roce 2006 se vrací k vývojářským týmům. Nyní je konsorcium W3C zodpovědné za dodávání nových specifikací HTML5. [6]

Poslední organizací podepsanou pod vývojem HTML5 je IETF (Internet EngineeringTaskForce). Tato organizace definuje nové rozhraní API WebSocket, které staví na protokolu WebSocket. [6]

1.3 HTML5

Klíčovým úkolem vývojářů HTML5 se stala kompatibilita. Příkladem je nový element <*header*>. Společnost Google analyzovala miliony stránek a odhalila, že typickým elementem je DIV s identifikátorem header. A tak vývojáři přebrali identifikátor header a vytvořili z něj element. [3, 4, 6]

Vývojáři HTML5 se snaží upřednostnit tvůrce internetových stránek. Ve specifikaci HTML5 můžeme využít několik způsobů psaní zdrojových kódů (viz. Syntax 1). Všechny tři způsoby, které jsou uvedeny v ukázce Syntax 1, platí pro zdrojový kód HTML5. Tvůrcům internetových stránek usnadňují tvorbu internetových stránek, jelikož nemusí řádek po řádku vyhledávat chybu. Podívejme se na tento problém ze strany uživatele internetových stránek. Může se stát, že uživateli se internetová stránka nezobrazí nebo na internetové stránce nastane nečekaná chyba. [3, 4, 6]

Zdrojový kód 1: Ukázka platného zdrojového kódu:

```
id="prohtml5"
id=prohtml5
ID="prohtml5"
```

Důležitou věcí, kterou vývojáři kladou jako velkou přednost HTML5 je zjednodušení programovacího kódu. Tento přístup se zaměřuje na nativní podporu

internetových prohlížečů na místo komplexního kódu v JavaScriptu, zjednodušený DOCTYPE, zjednodušená deklarace znaků a rozhraní API v HTML5. Deklaraci typu dokumentu a znakové sady více představím v praktické části bakalářské práce. [3, 4, 6]

HTML5 dále nabízí podporu funkcí, které byly dříve možné pouze s použitím zásuvných modulů. Na téma funkcí, které mají nahradit zásuvné moduly, se více zaměřím v kapitole Flash vs. HTML5. [3, 4, 6]

1.4 HTML5 – tablety, mobilní zařízení

Rozvoji technologie tabletu a mobilních zařízení se museli přizpůsobit i vývojáři HTML5. Hlavní prioritou vývoje je ovládání dotykových zařízení. Za důležitou funkci u mobilních zařízení a tablet považuji funkci orientace. [6]

Událost, která napomáhá vyvolat funkci orientace internetových stránek v mobilním zařízení nebo tabletu, tvůrci internetových stránek uvádí ve zdrojovém kódu v elementu body (viz. Zdrojový kód 2). Dokument, který obsahuje zdrojový kód uvedený ve zdrojovém kódu 2, se orientuje do čtyř stran (viz. Tabulka č. 1). [6]

Zdrojový kód 2: Ukázka změny orientace zařízení:

```
<body onorientationchange="rotateDisplay();">
```

Hodnota	Popis
0	Stejná orientace jako při načtení internetové stránky.
-90	Internetová stránka se otočí po směru hodinových ručiček.
180	Internetová stránka se otočí vzhůru nohama.
90	Internetová stránka se otočí proti směru hodinových ručiček.

Tabulka 1. Hodnoty orientace[1]

Poněkud složitější operací je naprogramování dotyků. Dotykové události totiž musí být schopny reprezentovat více dotyků současně. Základní dotykové události uvádím v tabulce č. 2. [6]

[1] Uvedená tabulka převzatá z knihy HTML5 Programujeme moderní webové aplikace, napsal kolektiv autorů Peter Lubbers, Brian Albers, Frank Salim

Událost	Popis
touchstart	Uživatel umístí prst na dotykovou obrazovku.
touchmove	Jeden či více prstů na obrazovce změnilo svou polohu, byl přetažen jinam.
touchend	Uživatel odstranil z obrazovky jeden či více prstů.
touchcancel	Došlo k přerušení dotekové operace.

Tabulka 2. Dotykové události.[2]

1.5 Budoucnost HTML5

Vývojáři se zaměřují na rozhraní WebGL. Rozhraní aplikace WebGL tvoří 3D grafiku internetových stránek. Princip práce je založený na elementu canvas, který je doplněn 3D kontextem. Dříve pracovali někteří vývojáři internetových prohlížečů na experimentálním rozhraní pro 3D grafiku v JavaScriptu. V roce 1992 vytvořila skupina TheKhronos Group OpenGL pro 3D kreslení. Specifikace OpenGL se používá v herním průmyslu a grafických aplikacích. Kontext pro WebGL umožňuje místo kreslení čar a vyplňování cest, práci s texturami a vyrovnávací pamětí. [3, 6]

Zajímavostí také je, že HTML5 usiluje o vytvoření XHTML5, aby nástroje pro práci s XML mohli generovat platný kód HTML5. [6]

[2] Uvedená tabulka převzatá z knihy HTML5 Programujeme moderní webové aplikace, napsal kolektiv autorů Peter Lubbers, Brian Albers, Frank Salim

2 JavaScript a kaskádové styly

Při tvorbě internetových stránek nepracuji jen s HTML5. Proto v této kapitole okrajově představím JavaScript a kaskádové styly. S těmito technologiemi budu dále pracovat v praktické části bakalářské práce.

2.1 JavaScript

JavaScript je skriptovací programovací jazyk, který umožňuje vytvářet dynamické internetové stránky. Činnost JavaScriptu zabezpečuje internetový prohlížeč. Internetové prohlížeče posuzují JavaScript pomocí modulu DOM. Možností specifikace HTML5 je potřeba využívat při práci JavaScrip pro Rozhraní Canvas. [9]

Využití JavaScriptu:

- záměna obrázků na stránce v závislosti na poloze kurzoru myši (efekt rollover)
- vytvoření prvků ulehčujících navigaci v podobě hierarchických (folder) a roletových (pull-down) nabídek
- kontrola správnosti údajů zadaných návštěvníkem stránky do formuláře (před jejich odesláním na server) a jejich korekce
- práce s okny prohlížeče stránek
- využití údajů o aktuálním datu a času a jejich zobrazení (hodiny, kalendáře).[3]

2.2 Kaskádové styly

Cascading Style Seets (zkratka CSS) vzniká v roce 1997. V českém jazyce se pro CSS ujalo označení kaskádové styly. CSS umožňuje grafickou úpravu internetových stránek. Hlavní smysl kaskádových stylů je oddělení vzhledu od struktury a obsahu navrhované stránky. [10, 11]

Specifikace HTML5 pracuje s verzí kaskádových stylů CSS3. CSS3 poskytuje pokročilé sektory, grafická vylepšení a lepší podporu fontů. V případě propojení HTML5 a CSS3 je možné stylovat téměř veškeré elementy (audio, video, canvas,…). CSS4 dokáže vytvářet i základní animace. [10, 11]

[3] Údaje o využití JavaScriptu přepsané z knihy JavaScript: efektní nástroj oživení www stránek, autor Slavoj Písek

3 HTML5 - audio a video

Novými a převratnými elementy specifikace HTML5 jsou elementy <*audio*> a <*video*>. V této kapitole se zaměřím na základní práci s elementy <*audio*> a <*video*>. O těchto elementech by se dala napsat samostatná bakalářská práce, a proto jsem se zaměřila jen na základní práci s těmito elementy.

3.1 Elementy <audio> a <video>

O elementech <*video*> a <*audio*> v HTML5 se začalo mluvit v roce 2005. První zkušební verze podporována internetovým prohlížečem Safari prorazila v roce 2007. S nástupem roku 2010 elementy <*video*> a <*audio*> podporuje i Internet Explorer. [4, 6, 8]

Před příchodem elementu <*video*> a <*audio*> se mohlo do internetových stránek přidávat video a audio jen pomocí elementu <*objekt*> a <*embed*>. Problémem bylo, že tyto elementy vyžadovaly zásuvné moduly v prohlížeči a také se spouštěli pomocí nainstalovaného programu v systému (např. Windows Media Player). [4, 6, 8]

Tvůrci musejí brát v potaz tři různé formáty pro zpracování videa a audia na internetových stránkách. Důvodem jsou internetové prohlížeče. Prohlížeče podporují kodeky WebM, OggTheora, MPEG-4 H. 264 pro video a WAV, Ogg Vorbis, MP3 pro audio. [4, 6, 8]

Pokud je element <*video*> a <*audio*> v základním tvaru obsahuje jen atribut src. Atribut src určuje cestu ke zdroji videa. Používání samotného základního atributu src lze jen v tom případě, že video je ovládáno prostřednictvím JavaSriptu. Atribut src není jediným atributem pro element <*video*> a <*audio*>. V následujících podkapitolách si představíme další atributy elementu <*video*> a <*audio*>. [4, 6, 8]

Zdrojový kód 3: Element <video> a <audio>

```
Ogg videa          <video src="video1.ogv"></video>
WebM videa         <video src="video1.webm"></video>
MPEG-4 videa       <video src="video1.mp4"></video>
WAV audia          <audio src="audio1.wav"></audio>
Ogg Vorbis audia   <audio src="audio1.wav"></audio>
MP3 audia          <audio src="audio1.wav"></audio>
```

3.2 Atributy elementu <*video*>

Pro element <*video*> existují atributy, kterými můžeme upravovat vlastnosti vloženého videa. Samozřejmě výběr atributů při tvorbě internetových stránek záleží na nás. Pouze atribut src nesmí chybět v elementu <*video*>. V následující podkapitole představuji několik atributů a jejich funkce vzhledem k elementu <*video*>. [6, 8]

Atribut autoplay

Atributem autoplay spustíme video automaticky. Jestliže atribut autoplay použijeme, video se přehraje od začátku do konce. Pokud nedojde k žádným zásahům ze strany uživatele nebo s problémem kompatibility s internetovým prohlížečem.

Atribut nepřebírá žádné hodnoty. Logicky pokud je atribut autoplay přítomen nastavení je true. Pokud přítomen není nastavení je false. [6, 8]

Zdrojový kód 4: Atribut autoplay:

```
<video src="video1.mp4" autoplay> </video>
```

Atribut loop

Atribut loop slouží k dalšímu automatickému přehrávání poté, co video dokončilo první přehrávání. Vlastně atributem loop způsobíme nekonečnou smyčku. U použití atributu loop nemůžeme počet smyček specifikovat. Počet smyček můžeme specifikovat jen v případě použití API nebo JavaScriptu. Atribut loop je také logický, takže pro něj platí stejné nastavení jako pro atribut autoplay. [6, 8]

Zdrojový kód 5: Zdrojový kód atributu loop:

```
<video src="video1.mp4" autoplay loop> </video>
```

Atribut poster

Atribut poster specifikuje obrázek, který bude reprezentovat video na internetové stránce. Atribut poster používáme u videí, které se automaticky nepřehrávají. Důležitou součástí atributu poster je zadání hodnoty tzn. cesty k obrázku. [6, 8]

Zdrojový kód 6: Zdrojový kód atributu poster:

```
<video src="video1.mp4" poster="obrazek1.jpg"> </video>
```

Atribut width a height

Tímto atributem nastavujeme rozměry videa. Nejlépe uděláme, když rozměry zobrazení videa na internetových stránkách nastavíme podle rozměrů přidávaného videa. Pokud se rozměry videa nebudou shodovat. Video bude vycentrováno a po stranách se zobrazí pruhy. [6, 8]

Pokud velikost videa neupravíme sami, internetový prohlížeč si ji upraví podle velikosti prvního snímku vloženého videa. [6, 8]

Zdrojový kód 7: Zdrojový kód atributu width a height:

```
<video src="video1.mp4" width="300" height="150"> </video>
<video src="video1.mp4" width="20%" height="20%"> </video>
```

Atribut controls

Atribut controls zobrazuje ve videu na internetové stránce ovládací prvky. Uživatel internetové stránky může videm manipulovat (spustit video, vypnout video,…). Specifikujeme-li atribut controls bez atributu autoplay ovládací prvky se v prohlížeči zobrazí buď hned, nebo po potáhnutí kurzorem myši na video. [6, 8]

Zdrojový kód 8: Zdrojový kód atributu controls:

```
<video src="video1.mp4" controls></video>
```

Atribut preload

Atributem preloud tvůrci webových stránek nastavují pokyny pro internetový prohlížeč. Co má internetový prohlížeč dělat u elementu <*video*> ohledně stahování. Atribut preload přebírá jednu ze tří hodnot (none, metadata nebo auto). [6, 8]

Hodnotu none použijeme, pokud nepředpokládáme, že uživatel video spustí (např. galerie videí,…). [6, 8]

Hodnotu metadata nastavíme, pokud chceme zobrazit první snímek, ale nechceme zahájit progresivní stahování. Metadata jsou výchozí hodnotou atributu preload. [6, 8]

Poslední hodnotou auto zahájíme progresivní stahování i v tom případě kdy není nastavený atribut autoplay. [6, 8]

Atribut preloud většinou používají jen pokročilý tvůrci internetových stránek. [6, 8]

Zdrojový kód 9: Atributu preload (none,metadata,auto):

```
<video src="video1.mp4" preload="none" controls></video>
<video src="video1.mp4" preload="metadata" controls></video>
<video src="video1.mp4" preload="auto" controls></video>
```

3.3 Atributy elementu <*audio*>

Atributy, které spadají pod element <audio>, mají stejné funkce jako atributy u elementu <video>. Mezi atributy patřící pod element <audio>, kromě atributu src, se řadí atribut autoplay, atribut loop, atribut controls a atribut preload. [6, 8]

3.4 Náhradní obsah

V rámci videa a audia můžeme vytvořit tzv. náhradní obsah. Náhradní obsah se objeví jen v případě, kdy internetový prohlížeč nebude kompatibilní s elementem <*video*> nebo <*audio*> v HTML5. [4, 6, 8]

Zdrojový kód 10: Příklad Náhradního obsahu:

```
<video src="video1.mp4">
Video nelze zobrazit, jelikož Váš webový prohlížeč nepodporuje prvek video v HTML5.
</video>
```

4 Rozhraní Canvas

HTML5 přináší nové rozhraní s názvem Canvas. V této kapitole si představíme rozhraní Canvas a jeho využití. Rozhraní Canvas se budu také věnovat v praktické části bakalářské práce, kde uvedu praktickou ukázku vytváření obrázku (loga) a grafu.

4.1 Canvas

Rozhraní, které umožňuje generovat a vykreslovat grafiku, grafy, obrázky i animace. Canvas vytváří obrázky a animace v internetovém prohlížeči programováním v JavaScriptu. Rozhraní Canvas pracuje na principu plátna. Tzv. princip plátna nejprve představil Apple ve WebKitu pro MacOS X. Než vůbec přišla možnost pracovat pomocí plátna, bylo možné malovat pomocí zásuvných modulů Flash, SVG (ScalableVectorGraphics), VML (VectorMarkupLanguage) nebo pomocí JavaScriptu. Apple nejprve bojoval za svá práva, když organizace WHATWG přišla ze specifikací plátna, ale nakonec souhlasil s použitím volné licence. [3, 4, 6]

Za pomocí JavaScriptu můžete s elementem canvas jakkoliv pracovat. Můžete přidávat grafiku, čáry, texty. Toto rozhraní skýtá i možnost malování a vkládání pokročilých animací. Rozhraní se skládá ze stejných dvourozměrných grafických operací, které nalezneme ve většině moderních operačních systémů a framewroku. [3, 4, 6]

Canvas podporuje tyto kategorie funkcí:

- práce s Canvas – vytvoření kreslící oblasti, 2D kontext, ukládání a obnovování stavu
- kreslení základních tvarů – obdélníky, cesty, úsečky, oblouky, Bezierovy akvadratické křivky
- práce s obrázky – vytváření, kreslení, změna měřítka a výřezy
- stylování – barvy, styly výplní, styly tahů, průhlednost, styly úseček, gradienty, stíny a vzorky
- transformace – přesuny, otáčení, změna měřítka, transformační matice
- kompozice – ořezávání a překrývání kreslené kompozice
- animace – vykonávání kreslících funkcí v průběhu času (sdružením časových intervalů a timeoutů).[4]

[4] Údaje o podpoře funkcí rozhraní Canvas přepsané z knihy HTML5 audio a video, autor Silvia Pfeiffer

4.2 Plátno

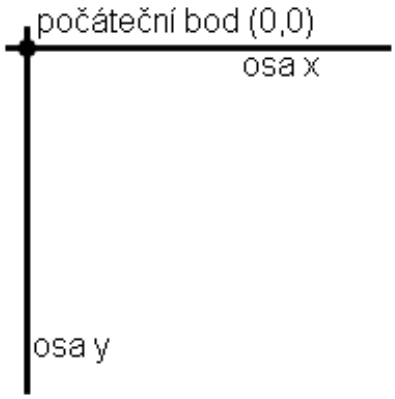
Pro rozhraní Canvas je velmi důležité plátno. Použitím elementu canvas se na stránce objeví obdélníková oblast široká 300 pixelů a vysoká 150 pixelů. Samozřejmě velikost plátna si můžete měnit dle svého za pomocí atributu width a height v elementu <*canvas*>. Plátno má taky své souřadnice jeho úplný počátek tedy hodnoty x=0 a y=0 nalezneme v levém horním rohu. Horizontálně se nachází osa x a vertikálně osa y. [4, 6]

Zdrojový kód 11: Příklad nastavení vlastní velikosti plátna:

```
<canvas width="700" height="500">
</canvas>
```

4.3 Počáteční bod plátna

Důležité pro práci s rozhraním Canvas je uvědomit si, kde se nachází počáteční bod na ose x, y. Počáteční body x, y (0,0) se nachází v levém horním rohu plátna. Horizontálně se nachází osa x a vertikálně osa y. [4, 6]



Obrázek č. 1: Počáteční bod plátna

5 HTML 5 podpora v režimu off-line

Mnoho aplikací vyžaduje neustálé připojení k internetu. Bohužel ani v dnešní době není možné neustálé připojení k internetu. V případě, že nutně potřebujeme otevřít např. svůj e-mail na místě bez připojení k internetu, by nám mohl pomoci režim off-line. V kapitole podpora v režimu off-line se zaměřuji na práci s režimem off-line a jeho součástmi.

5.1 Režim off-line

Režim off-line závisí na tzv. ukládání do mezi-paměti. Ukládání do mezi-paměti právě umožňuje specifikace HTML5. Nevýhodou je ovšem, že tuto funkci jde použít explicitně. Jelikož žádný internetový server vám neumožní ukládání dat do mezi-paměti pro off-line. Mezi-paměť umožňuje off-line aplikacím přizpůsobit se nedostupné síti. Režim off-line by se mohl uplatnit při čtený a psaní e-mailů, editaci dokumentů, editaci a zobrazení prezentací a k vytváření poznámek. [4, 6]

Tvůrce internetových stránek si sám může specifikovat prostředky HTML5 (např. soubory HTML, CSS, JavaScriptu, obrázky), tak aby byly dostupné v režimu off-line. Záleží na tvůrci internetových stránek, jak si řídí obsah mezi-paměti. [4, 6]

5.2 Off-line aplikace

Před samotným vytvořením off-line aplikace nezapomínejme ověřit podporu prohlížeče. [4, 6]

Zdrojový kód 12: Podpora prohlížeče pro off-line aplikaci:

```
if (window.applicationCache) {///prohlížeč podporuje off-line webové aplikace}
```

Manifest

Pokud chceme vytvořit soubory (např. HTML, CSS, JavaScript), které budou spadat do aplikace off-line musíme nejprve do elementu <*html*> vložit atribut manifest.

Zdrojový kód 13: Atribut manifest:

```
<html manifest="aplikace.manifest"> </html>
```

Manifest obsahuje seznam souborů, které budou uloženy do mezi-paměti pro případ stavu off-line. V manifestu musí být každý soubor, který chceme zobrazit v režimu off-line. [4, 6]

Zdrojový kód 14: Ukázka obsahu souboru manifest:

```
CACHE MANIFEST
ukazka.html
ukazka.js
ukazka.css
ukazka.gif
```

Soubor manifestu má typ MIME text/cachemanifest. Aby soubor manifest pracoval pod typem MIME musíme do souboru *{Instalační složka Pythonu}/Lib/mimetypes.py přidat '.manifest': 'text/cache-manifest manifest"*. [4, 6]

ApplicationCache

Rozhraní ApplicationCache pracuje s aplikační mezi-pamětí. ApplicationCache generuje události spojené se stavem mezi-paměti. [4, 6]

Jednou z těchto událostí je událost *status*. Uvádí aktuální stav mezi-paměti. Mezi-paměť se může nacházet v jednom ze šesti stavů:

Hodnota vlastnosti (state)	Stav mezi-paměti
0	UNCACHED
1	IDLE
2	CHECKING
3	DOWNLOADING

Tabulka 3. Přehled stavů mezi-paměti[5]

Typickým stavem je stav IDLE. Všechny prostředky aplikace jsou uloženy a neprobíhá na nich žádná aktualizace. Pokud dříve existoval obsah mezi-paměti, ale

[5] Uvedená tabulka převzatá z knihy HTML5 Programujeme moderní webové aplikace, napsal kolektiv autorů Peter Lubbers, Brian Albers, Frank Salim

soubor manifestu z nějakých důvodů chybí, stav paměti se změní na OBSOLETE. Každý stav mezi-paměti, kromě hodnoty stavu 0, má svou vlastní událost. Pokud přejdeme do některého ze stavů, automaticky se vygeneruje událost, která k tomuto stavu připadá. [4, 6]

Událost	Stav mezi-paměti
CHECKING	Necking
DOWNLOADING	Downloading
UPDATEREADY	Updateready
OBSOLETE	Obplete
IDLE	Cached

Tabulka 4. Běžné události a s nimi korespondující stav mezi-paměti[6]

[6] Uvedená tabulka převzatá z knihy HTML5 Programujeme moderní webové aplikace, napsal kolektiv autorů Peter Lubbers, Brian Albers, Frank Salim

6 Srovnání HTML5 s technologiemi Flash a SilverLight

Dlouhou dobu byly aplikace (např. animace, videa, atd.) vytvářeny pomocí technologie Flash. V této kapitole se podívám jak technologie Flash začala ustupovat HTML5.

SilverLight je technologie, která využívá zásuvné moduly. V této kapitole se zabývám jeho aktuálností porovnání se specifikací HTML5.

6.1 Flash

Pod vývojem Flashe se podepisuje společnost Adobe. Flash je založen na vektorové grafice. Je určen k vytváření animací, prezentací, ale taky her. Flash pracuje za pomoci programovacího jazyka ActionScript. [16]

HTML5 vs Flash

Společnost Adobe začala vyvíjet aplikace pro mobilní zařízení, které podporovaly Flash. Aplikace s podporou technologie flash neměli dlouhého trvání. Začali je nahrazovat aplikace vytvořené pomocí HTML5. Aplikace vytvořené v jazyce HTML5 nepotřebují instalaci zásuvných modulů. Postupně se Flash začal přizpůsobovat specifikaci HTML5. [13]

Flash Professional CS6 obsahuje sadu nástrojů pro rozhraní CreateJS. Rozhraní CreateJS umožňuje přechod z aplikace Flash do formátu HTML5. Flash Professional CS6 umožňuje vytváření animací a multimediálního obsahu. Vytváří interaktivní obsah pro počítače, tablety, smartphony, ale také i televizory. [16]

6.2 SilverLight

SilverLight vyvíjí společnost Microsoft. Je určen k vytváření interaktivních internetových aplikací. SilverLight usnadňuje práci s animacemi, textem a videm. Většina současných internetových prohlížečů podporuje SilverLight. [5]

Na internetových serverech jsou mnohem rozšířenější aplikace Flash než SilverLight.

HTML5 vs SilverLight

S příchodem páté verze Microsoft SilverLight5. Microsoft vydává prohlášení, že SilverLight5 je poslední verzí na trhu. SilverLight začne ustupovat specifikaci HTML5.

Ale samozřejmě to neznamená, že ze dne na den se z technologie SilverLight přejde na specifikaci HTML5 a tak se SilverLight vytratí ze světa. Nástup technologie HTML5 není tak rychlí a ještě dlouho bude trvat, než tato specifikace dosáhne takřka své dokonalosti. Proto Microsoft plánuje stálou podporu technologie SilverLight5 až do roku 2021. [13]

7 Kompatibilita HTML5 s internetovými prohlížeči

Kapitola kompatibilita s internetovými prohlížeči je zaměřená na čtyři nejpoužívanější internetové prohlížeče a jejich zobrazování specifikace HTML5. Nejprve ve stručnosti představím internetové pohlížeče (Opera, Internet Explorer, Mozilla Firefox, Google Chrome). Poté se zaměřím na jejich kompatibilitu s elementy a rozhraními, které byly představené v dřívějších kapitolách.

7.1 Internetové prohlížeče

V následující podkapitole se ve zkratce zaměřím na vývoj čtyř nejpoužívanějších internetových prohlížečů na světě.

Opera

V roce 1994 vzniká výzkumný projekt Opera určená pro norskou telekomunikační společnost Telenor. Roku 1996 vychází první komerční prohlížeč Opera 2.1 pro Windows. Postupem času Opera začala podporovat nejen operační systémy (Windouws, Linux, Solaris), ale také mobilní systémy. Při vydání Opery 8.5 v září roku 2005 přešla Opera z komerčního prohlížeče na freeware. Obsahuje webový prohlížeč, e-mailového klienta, IRC klienta, RSS a Atom čtečku, čtečku Usenetových skupin a další funkce. [1]

Internet Explorer

Předchůdcem Internet Exploreru je prohlížeč NCIS Mosaic. Od roku 1995 se tento webový prohlížeč vyvíjí bez licence zdrojového kódu. Internet Explorer je součástí operačních systémů Microsoft Windows. Delší dobu vedl v používanosti webových prohlížečů. [1]

Mozilla Firefox

První verzi Moziilly Firefox vydali v roce 2002 Dave Hyatt a Blake Ross ovšem pod názvem Phoenix. Následně je projekt přejmenován na Mozilla Firebird a poté na Firefox. Mozilla Firefox je jeden z nejstahovanějších webových prohlížečů dokonce se díky tomuto zapsal do Guinessovy knihy rekordů. [1]

Google Chrome

První verze pro Microsoft Windows vyšla v roce 2008. V tomto roce nejprve vychází v září beta verze a v prosinci stabilní verze. V roce 2010 vychází verze, která rychleji zpracovává JavaScript. Google Chrome velmi rychle začíná podporovat funkce HTML5, ale také podporuje obdobné doplňky jako Mozilla Firefox. [1]

7.2 Přehled kompatibility

Kompatibilita je důležitou součástí používání internetových stránek. Kolikrát se stává, že si otevřeme internetovou stránku a nezobrazuje se nám daná stránka správně. Protože HTML5 je poměrně novou specifikací potřebuje mít přehled, které funkce specifikace HTML5 pracují správně v internetových prohlížečích.

Video a audio

Pro spuštění videa nebo audia existují tři základní formáty. Pro video kodek WebM (.webm), kodek OggTheora (.ogv), Kodek MPEG-4 H.264 (.mp4) a pro audio kodek WAV (.wav), kodek MP3 (.mp3) a kodek OggVorbis (.ogg). Následující tabulka ukazuje kompatibilitu videa a audia s internetovými prohlížeči. Z tabulky jasně vyplývá, že internetovým prohlížečem, který podporuje všechny formáty jak u videa, tak u audia je Google Chrome. [6, 17]

	Opera	Internet Explorer	Mozzilla Firefox	Google Chrome
Video	webm; ogv	mp4	webm; ogv	webm; ogv; mp4
Audio	wav; ogg	mp3	wav; ogg	wav; ogg; mp3

Tabulka 5. Přehled podpory kodeků v internetových prohlížečích

Rozhraní Canvas

Rozhraní Canvas podporují všechny uvedené internetové prohlížeče s výjimkou starších verzí Interet Eploreru. Canvas podporuje internetový prohlížeč Internet Explorer u verze 9. [6, 17]

Malým problémem může být podpora rozhraní Canvas Text. Důvodem je pozdější přidání do specifikace HTML5. [6, 17]

Off-line

Off-line webové aplikace podporuje Opera, Mozilla Firefox, Google Chrom. Internet Explorer nepodporuje vůbec off-line internetové aplikace. Jelikož podpora jednotlivých prohlížečů se liší. Je vhodné nejprve ověřit podporu off-line internetové aplikace. [6, 17]

Formuláře

Podpora formulářů je ze strany prohlížečů omezená. Jediný prohlížeč, který podporuje všechny typy vstupných polí, je Opera. [6, 17]

Vstupní pole pro zadání e-mailu, čísla, telefonu, adresy URL, čísla vyhledávání a částečně rozsahu podporuje Google Chrome od verze 10. [6, 17]

Základní podporu nabízí Mozilla Firefox od verze 4.0. Verze 4 podporuje vstupní pole pro e-mail, telefon, URL adresu a vyhledávání. [6, 17]

Internet Explorer vůbec nepodporuje nové vstupní pole formulářů. Nové vstupní pole se v Internet Exploreru zobrazí jako běžná textová pole. [6, 17]

7.3 Modernizr

Pro otestování podpory internetového prohlížeče je možné využít knihovnu Modernizr. Knihovna Modernizr otestuje schopnost prohlížeče pracovat s použitými prvky. Poté přizpůsobí použité prvky schopnostem daného internetového prohlížeče. [18]

Praktická část

8 HTML5 srovnání s XHTML

V kapitole HTML5 srovnání s XHTML se podívám na základní změny v tvorbě internetových stránek. Zaměřím se na nové sémantické znaky HTML5 v porovnání se sémantickými znaky XHTML. Dále poukážu na jednoduchost nového elementu <video> proti elementu <objekt>.

8.1 XHTML

Specifikace XHTML vychází z jazyka XML. Vývojáři převzali syntaxi z XML a vytvořili novou specifikaci XHTML, jenž se stala základem pro kaskádové styly. Zkratka XHTML znamená eXtensible Hypertext Markup Language. Extensible v českém jazyce znamená rozšiřitelný. Ovšem ve skutečnosti jde o zúžení a osekání specifikace XHTML. [2, 10, 11]

XHTML existuje v několika verzích (např. XHTML 1.0 transitional, XHTML 1.0 strict, XHTML 1.1). Pro tvorbu internetových stránek používám verzi XHTML 1.0 transitional. [2, 10, 11]

Pro XHTML je podstatné, aby každý element byl zřetelně ukončen, buď ukončovací značkou, nebo lomítkem na konci značky. Vlastnost elementu nebo atributu musí být uzavřeny v uvozovkách. [2, 10, 11]

8.2 Deklarace typu dokumentu

Když začínám psát zdrojový kód internetové stránky, musím nejprve nadefinovat deklaraci typu dokumentu. Deklarace pomáhá validátorům určit, jaký typ validace je třeba při ověřování kódu zvolit. Pro deklaraci typu dokumentu používám nepárový element <DOCTYPE>.

Při tvorbě internetových stránek v XHTML byl zápis deklarace typu dokumentu složitý a těžko zapamatoval. Specifikace HTML5 přichází s novým zápisem deklarace typu dokumentu. Jak je patrné z příkladu zdrojového kódu 10. – 11. <DOCTYPE> byl výrazně zkrácen. Už od začátku psaní zdrojového kódu v jazyce HTML5 je patrné, že vyhrává jednoduchost.

Zdrojový kód 15: DOCTYPE HTML5:

```
<!DOCTYPE html>
```

Zdrojový kód 16: DOCTYPE XHTML:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

8.3 Znaková sada

Stejně jako u deklarace typu dokumentu došlo k zjednodušení deklarace znakové sady. Pro deklaraci znakové sady používám nepárový element <meta>.

V elementu <meta> uvádím atributy. V případě XHTML používám tři atributy (http-equiv, content a charset). Deklarace znakové sady v jazyce HTML5 element <meta> obsahuje jen atribut charset. V atributu charset nastavujeme jazyk internetových stránek. V mém případě definuji atributem charset český jazyk.

Zdrojový kód 17: Znaková sada HTML5:

```
<meta charset="utf-8">
```

Zdrojový kód 18: Znaková sada XHTML:

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
```

8.4 Struktura internetové stránky

Specifikace HTML5 přináší nové sémantické elementy, které nahrazují element . Neznamená to, že element úplně zmizel ze zdrojového kódu jazyka. Element použiji i při tvorbě internetové stránky v jazyce HTML5. Použitím nových sémantických elementů se zdrojový kód internetové stránky stává mnohem přehlednější.

K elementu se připojoval atribut id, který sloužil jako identifikátor. Používání identifikátorů u XHTML bylo pravidlem, abychom odlišily sekce internetové stránky pro práci s kaskádovými styly. V HTML5 nemusím používat identifikátory. Pokud ve zdrojovém kódu uvedu vícekrát některý sémantický element a chci, aby jeho vzhled pomocí kaskádových stylů byl jiný, přiřadím k elementu identifikátor.

V textu jsem se zmínila o kaskádových stylech. Pomocí kaskádových stylů upravuji grafiku internetových stránek. V případě XHTML kaskádový styl CSS a pro HTML5 kaskádový styl CSS3.

Nové sémantické elementy

Sémantické elementy rozdělují internetové stránky na sekce. Nové sémantické elementy pro tvorbu internetových stránek jsou element <header>, <nav>, <section>, <article>, <aside> a <footer>. V tabulce 4. uvádím použití sémantických elementů. Všechny sémantické elementy jsou párovými značkami, musí bý ukončeny.

Element	Popis
header	Obsah záhlaví (stránky anebo sekce stránky)
nav	Navigace
section	Sekce internetové stránky
article	Nezávislý obsah článku
aside	Související obsah anebo citace
footer	Obsah zápatí (stránky anebo sekce stránky)

Tabulka 6. Nové sémantické elementy HTML5[7]

Struktura internetové stránky v HTML5

Pro vytvoření struktury internetových stránek ve specifikaci HTML5 využiji nové sémantické značky. Strukturu internetových stránek zapisuji v elementu <body>. Začínám záhlavím, které specifikuje element <header>. V tomto případě použiji identifikátor <header id="zahlavi">, protože element <header> se mi objeví i v dalších sekcích.

Pro navigaci používám element <nav>. V němž uvádím odkazy na další internetové stránky pomocí elementu <a href>, který je párovou značkou.

Hlavní stránku pro psaní článku si rozdělíme na několik sekcí. Celou sekci ohraničíme elementem <sekcion>. Pro vytvoření oblasti pro článek použijeme element <article>, který bude obsahovat ještě jedno záhlaví pro název článku <header id="clanek">. Na konec stránky umístím patičku, pro níž máme nový element <footer>.

Identifikátory, které jsem přidávala do elementu <header> slouží pro práci s CSS3.

[7] Uvedená tabulka převzatá z knihy HTML5 Programujeme moderní webové aplikace, napsal kolektiv autorů Peter Lubbers, Brian Albers, Frank Salim

Zdrojový kód 19: Struktura internetové stránky stránky v HTML5:

```
<body>
        <header id="zahlavi">
        </header>
        <div>
            <nav>
            </nav>
            <section>
            <article>
            <header id="clanek">
            </header>
            </article>
            </section>
            <footer>
            </footer>
         </div>
</body>
```

Struktura internetové stránky v XHTML

Se strukturou internetové stránky napsané ve specifikaci XHTML pracuji podobně. Rozdíl v psaní struktury internetové stránky v XHTML a HTML5 je takový, že v XHTML používám jen element . K elementu přidáváme identifikátory pro úpravu v CSS.

Zdrojový kód pro strukturu internetové stránky v XHTML začíná vycentrováním stránky na střed <div id="center>. Teprve potom definuji záhlaví internetové stránky . Pro určení místa článku a navigace nejprve nadefinuji celek . Teď mohu určit umístění navigace. V příkladě se navigace nachází v levé části stránky . V pravé části layoutu budeme psát článek. Nakonec umístím patičku dokumentu .

Zdrojový kód 20: Struktura internetové stránky v XHTML

```
<body>
        <div id="center">
        <div id="zahlavi">
        </div>
        <div id="celek">
```

```
        <div id="levy_sloupec">
         </div>
         <div id="pravy_sloupec">
         </div>
         <div id="paticka">
         </div>
          </div>
          </div>
</body>
```

Shodné elementy HTML5 a XHTML

Specifikace HTML5 a XHTML se neliší ve všech elementech Pro tvorbu nadpisu u HTML5 i XHTML používám párový element <h.>. Dalším stejným elementem pro obě specifikace je element odstavec <p>. Stejným způsobem jako u XHTML definuji element <a href>, který slouží jako odkaz. Pokud chci na stránky vložit obrázek, mohu použít element <img>. Pro vložení obrázku na internetové stránky v HTML 5 mám možnost použít element <canvas>.

9 Video HTML5 vs. video XHTML

Než se zaměřím na srovnání videa vloženého v jazyce HTML5 a XHTML, ukážu na praktických příkladech, jak vkládání videa na internetové stránky probíhá. Aby video vypadlo na stránkách vizuálně hezky, využila jsem úpravy v kaskádových stylech.

9.1 HTML5 video

Obrázek č. 2: Ukázka videa v HTML5

Jak už jsem se zmínila v teoretické části v kapitole HTML5 audio-video pro vložení videa na internetové stránky vytvořili vývojáři HTML5 element <video>. Do elementu <video> vložím atribut src, v kterém definuji cestu k videu. Aby vložení videa proběhlo, musím mít video ve formátu webm, ovg nebo mp4.

Pro snadnou manipulaci s videem na internetových stránkách vložím do elementu <video> atribut controls, který jsem popsala v podkapitole Atributy elementu <video>. Samozřejmě můžeme definovat i další atributy zmíněné v kapitole Atributy elementu <video>.

Zdrojový kód 21: HTML5 – video:

```
<video src="video/video1.ogv" controls>
Váš internetový prohlížeč nepodporuje prvek video.
</video>
```

HTML5 video – css

V teoretické části jsem zmínila možnost úpravy videa pomocí atributu width a height elementu <video>. Chtěla bych poukázat, že změnu velikosti videa nemusím provádět jen zmíněnými atributy, ale také v kaskádových stylech příkazem width. Další možností u změny velikosti je uvedení v procentech.

Rámeček kolem videa způsobí příkaz border, kde nastavuji velikost rámečku v pixelech, barvu rámečku a nepřerušovanou linku rámečku (solid). Zaoblení rohů rámečku dosáhnu příkazem border-radius. U příkazu border-radius uvedu úhel zaoblení v pixelech. Odsazení rámování od videa provádím příkazem padding. Hodnota příkazu padding opět uvádím v pixelech.

Kolem rámování utvářím stínování příkazem box-shadow. Do příkazu box-shadow uvádím velikost stínování a barvu stínu.

Odsazení videa od okrajů internetové stránky provádím příkazem margin.

Zdrojový kód 22: HTML5 video - css:

```
video {
    width:90%;
    padding: 10px;
    margin: 30px;
    border: 5px solid #7B3F00;
    border-radius: 15px;
    box-shadow: 10px 10px 5px gray;
    box-sizing: border-box;
    }
```

9.2 XHTML video

Obrázek č. 3: Ukázka videa v XHML

Pro vložení videa na internetové stránky vytvořené v jazyce XHTML existuje více způsobů. Představím způsob připojení videa na naše internetové stránky z pomocí serveru www.youtube.com a elementu <object>.

Na internetové stránky vkládám vlastní video. Proto nejprve video nahraji na internetovou stránku www.youtube.com, kde potřebuji mít založený účet. Na své internetové stránky vložím odkaz na video, který se bude moct přehrát přímo na mých stránkách.

Do zdrojového kódu XHTML uvedu element <object>, který bude obsahovat čtyři atributy (type, data, width, height). Všechny údaje uvedené v atributech musí být umístěny v uvozovkách.

K atributu type připíši instrukce pro připojení k Flashové aplikaci (application–x-shockwave-flash).

Abych mohla doplnit atribut data, najdu si video, které jsme vložila na internetový server www.youtube.com. Z adresního řádku zkopíruji kód nahrávky. Kód nahrávky je uveden za písmenem v=. Zkopírovaný kód vložím k atributu data, ale nejprve před kód nahrávky uvedu adresu internetového serveru http:www.youtube.com/v/. Za v lomítko přidám, zkopírovaný kód nahrávky.

V elementu <objekt> dále nadefinuji rozměry videa atributy width a hight. Rozměry videa uvádím v pixelech.

Dále musím uvést parametry objektu. Pro parametry objektu používáme element <param>. Element <param obsahuje dva atributy (name, value).

Atribut name obsahuje jméno proměnné. V tomto případě v atributu name uvádím jméno proměnné movie.

Atributem value vyjadřujeme hodnotu proměnné. Hodnota v atributu value je shodná s atributem data v elementu <object>.

Element <video> a <object> jsou párovými značkami a proto musí být ukončeny </video> a </object>.

Zdrojový kód 23: XHTML – video:

```
<object
type="application/x-shockwave-flash"
data="http://www.youtube.com/v/ov5x0wcd5Z0"
width="400" height="300">
<param name="movie" value="http://www.youtube.com/v/ov5x0wcd5Z0"/>
</object>
```

XHTML video – css

Videu ve specifikaci XHTML přidám rámování příkazem border-style, šířku rámování příkazem border-width a barvu příkazem border-color. Dvojité rámování vyvolám příkazem double.

Pro odsazení opět používám příkaz margin. Video odsadím z levé strany, z horního okraje a z dolního okraje internetové stránky.

Zdrojový kód 24: XHTML video - css:

```
object {
    border-style: double;
    border-width: 10px;
    border-color: #7B3F00;
    margin-top: 40px;
    margin-left: 150px;
    margin-bottom: 40px;
    }
```

9.3 Srovnání video HTML5 vs. XHTML

Jako kritéria jsem si stanovila přehlednost a množství zdrojového kódu, funkčnost, propojení s kaskádovými styly, kompatibilita s internetovými prohlížeči.

Co se týče přehlednosti a množství zdrojového kódu HTML5 je daleko přehlednější a jednodušší. Za zmínku stojí, že video vložené pomocí zdrojového kódu XHTML, bylo nejprve vloženo na jiný server. Teprve potom jsem mohla napsat zdrojový kód. U specifikace HTML5 video kdykoliv upravím a znovu jen vložím na své internetové stránky.

U videa vloženého pomocí elementu <video> nemusím spoléhat na přídavné moduly (např. flash). Ale u videa vloženého ze serveru www.youtube.com činí špatná instalace nebo zastaralý modul obtíže.

Obě specifikace lze jednoduchým způsobem propojit s kaskádovými styly. Pomocí kaskádových stylů je možné upravit i vložené video. HTML5 může pracovat jak s kaskádovými styly CSS tak CSS3, kdežto XHTML pracuje jen s CSS.

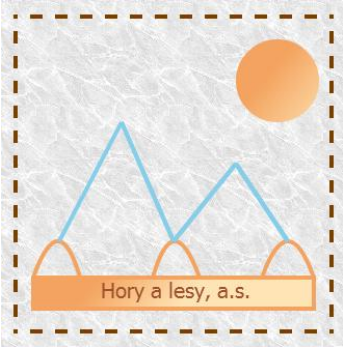
Kompatibilitu s internetovými prohlížeči u specifikace HTML5 jsem již uvedla v teoretické části. Kompatibilita v XHTML v mém případě závisí na kompatibilitě internetového prohlížeče s internetovým serverem www.youtube.com.

10 Práce s rozhraním Canvas

Jak už jsem se zmínila v teoretické části v rozhraní Canvas vytváří grafické prvky internetových stránek. V praktické části bakalářské práce se zaměřím na základní tvary, které se dají vytvořit v rozhraní Canvas. Vytvořím pomocí základních tvarů jednoduché logo. V druhé části kapitoly Práce s rozhraním se zaměřím na sloupcový graf. Pro práci s rozhraním Canvas použiji JavaScript.

10.1 Canvas – obrázek (logo)

Obrázek (logo) vytvořím v rozhraní Canvas pomocí základních tvarů. Obrázek (logo) se skládá z obdélníku, kvadratických křivek, cest a kružnice. Než začnu tvořit samotný obrázek (logo), musím nejprve nadefinovat plátno. Poté v JavaScriptu zadám příkazy pro vyvolání tvarů.



Obrázek č. 4: Canvas - obrázek (logo)

Canvas – definice plátna

Rozhraní Canvas vykresluje obrázky na tzv. plátno. Nejprve tedy nadefinuji plátno, abych mohla pracovat s rozhraním Canvas. Definici plátna provedu v elementu <canvas>. V elementu <canvas> nastavím identifikátor, který potřebujeme pro práci s JavaScriptem a velikost plátna atributy width a height. Element <canvas> je párovým elementem </canvas>.

Zdrojový kód 25: Canvas – definice plátna:

```
<canvas id="logo" width="300" height="300">
</canvas>
```

Pro vykreslení obrázku (loga) na plátno použiji JavaScript. Abych mohla vytvářet obrázek (logo) v JavaScriptu, musím nejprve zkontrolovat element <canvas>.

Do zdrojového kódu JavaScriptu vložím příkaz document.getElementByld(“logo“), který obsahuje identifikátor elementu <canvas>. Poté vyvolám context příkazem patno.getContext(“2d“). Context obsahuje vlastnosti a metody 2d grafiky.

Zdrojový kód 26: JavaScript – nalezení plátna a context

```
<script>
  var platno = document.getElementByld(“obrazek“);
  var context = platno.getContext(“2d“);
</script>
```

Canvas – základní tvary a cesty

Obdélník (čtverec)

Pro vytvoření obdélníku (čtverce) použiji metodu rect, která je určená k vyvolání obdélníku (čtverce). Do metody rect zadám parametry počátečního bodu obdélníku (čtverce) na ose x, y. Ke stejně metodě připíši parametry šířky (width) a výšky (height). Mám zadané počáteční body a šířku, výšku obdélníku (čtverce), ale pro zobrazení základního tvaru je nutné nadefinovat ohraničení metodou stroke.

Zdrojový kód 27: Canvas – obdélník

```
ctx.rect(15,250,270,30);
ctx.stroke();
var gradient = ctx.createLinearGradient(120, 200, 180, 250);
gradient.addColorStop(0, "#F4A460");
gradient.addColorStop(1, "#FFE4B5");
ctx.fillStyle = gradient;
ctx.fill();
```

Kvadratická křivka

Počáteční bod kvadratické křivky si nadefinuji metodou moveTo. Do zdrojového kódu připíši metodu quadraticCurveTo, která mi určí, že se jedná o kvadratickou křivku. Teď zadám do metody quadraticCurveTo tzv. kontrolní a koncový bod. Kontrolní bod slouží v kvadratické Bézierové křivce k výpočtu. Koncový bod umístím na osu x, y. Nesmím zapomenout na metodu stroke, která nám zadanou křivku vykreslí.

Jak je patrné s obrázku skládá se ze tří kvadratických křivek. Zdrojový kód kvadratických křivek je téměř shodný. Rozdíl spočívá v umístění počátečního, kontrolního a koncového bodu.

Zdrojový kód 28: Canvas – kvadratická křivka:

```
//křivka 1
ctx.moveTo(15, 250);
ctx.quadraticCurveTo(40, 180, 60, 250);
ctx.lineWidth = 4;
ctx.strokeStyle = "#F4A460";
ctx.stroke();
ctx.beginPath();
 //křivka 2
ctx.moveTo(130, 250);
ctx.quadraticCurveTo(150, 180, 175, 250);
ctx.lineWidth = 4;
ctx.strokeStyle = "#F4A460";
ctx.stroke();
ctx.beginPath();
//křivka 3
ctx.moveTo(235, 250);
ctx.quadraticCurveTo(260, 180, 285, 250);
ctx.lineWidth = 4;
ctx.strokeStyle = "#F4A460";
ctx.stroke();
ctx.beginPath();
```

Cesta

Cestu vytvořím tak, že nejprve nadefinuji počáteční bod na ose x, y do metody moveTo a koncový bod na ose x, y do metody lineTo. Samozřejmě opět přidám metodu stroke. U cesty se dají upravit konce metodou linecap. Linecap používá tři hodnoty round, butt a square. Použitím hodnoty round zaoblím konce linky. Hodnotou butt definujeme ukončení linky kolmým řezem v místě koncových bodů. U hodnoty square dochází k prodloužení linky.

V obrázku vytvářím čtyři cesty. Liší se v umístění počátečního a koncového bodu. Vytvořené cesty směřují kolmo, protože počáteční a koncový bod je nadefinován v kolmém směru.

Zdrojový kód 29: Canvas - cesta:

```
 //cesta 1
ctx.moveTo(260, 215);
ctx.lineTo(210, 140);
```

```
ctx.stroke();
ctx.lineWidth = 4;
ctx.strokeStyle = "#87CEEB";
//cesta 2
ctx.moveTo(210, 140);
ctx.lineTo(150, 215);
ctx.stroke();
ctx.lineWidth = 4;
ctx.strokeStyle = "#87CEEB";
//cesta 3
ctx.moveTo(150, 215);
ctx.lineTo(100, 100);
ctx.stroke();
ctx.lineWidth = 4;
ctx.strokeStyle = "#87CEEB";
//cesta 4
ctx.moveTo(100, 100);
ctx.lineTo(40, 215);
ctx.stroke();
ctx.lineWidth = 4;
ctx.strokeStyle = "#87CEEB";
```

Kružnice

Kružnici definuji metodou arc. Nejprve si musím nadefinovat střed kružnice na ose x, y. Dále uvádím do metody arc radius (průměr) kružnice. Další parametry metody arc se týkají úhlu počátku a úhlu konce kružnice. V mém případě je úhel počátku 0 a úhel konce 2*Math.Pi. Ovšem úhel počátku nemusí být vždy nulový a úhel konce 2*Math.Pi. Dají se použít další úhly, které mohou zapříčinit, že kružnice bude otevřená. Pro vytvoření kružnice musím připsat metodu stroke. V mém příkladu byla metoda stroke odstraněna a nahrazena výplní kružnice.

Zdrojový kód 30: Canvas - kružnice:

```
ctx.beginPath();
ctx.arc(250, 60, 40, 0, 2 * Math.PI);
var grd = ctx.createLinearGradient(80, 120, 180, 200);
grd.addColorStop(0, "#FFFF00");
grd.addColorStop(1, "#FF6347");
ctx.fillStyle = gradient;
ctx.fill();
```

Canvas – text

V příkladu obrázku se nachází ještě text, který vykreslím metodou fillText. Text, který chci zobrazit na plátnu, vložím do uvozovek metody fillText. Umístění textu na plátně vytvořím v metodě fillText počátečním bodem na ose x, y. Nastavení velikosti písma v pixelech a fontu písma vyvolávám metodou font().

Zdrojový kód 31: Canvas - text:

```
ctx.font = "20px Tahoma";
ctx.lineWidth = 4;
ctx.fillStyle = "#A0522D";
ctx.fillText("Hory a lesy, a.s.", 80, 270);
```

Canavas – úprava cest a tvarů

Barevné kompozice obrázku (loga) docílím úpravou šířky, barvy cest a změnou výplně tvarů. Grafickou úpravu cest a tvarů uvádím v příkladech zdrojových kódů 27 – 31.

Šířku cest upravím metodou width. Kde uvádím šířku v axelech. Metodou strokeStyle změním barvu cest. Barvu cest uvádím v uvozovkách.

Výplň tvarů lze udělat jednobarevnou nebo přechodem barev. Jednobarevnou výplň vytvořím metodou fill a metodou fillStyle. Kde do metody fillStyle uvádím barvu výplně. Vytvoření barevného přechodu začínám metodou fill a fillStyle, kde zadám parametr gradient. Dále vyvolám funkci lineárního přechodu nebo radiálního přechodu. Nejprve vyvolám metodu lineárního přechodu createLinearGradient. Parametrem pro metodu createLinearGradient jsou souřadnice počátečního a koncového bodu přechodu na ose x,y. Metodou addColorStop nastavím barvu a specifikuji pozici gradient, která může být od 0 do 1.

Canvas – css

Plátnu jsem přidala rámování pomocí kaskádových stylů. V kaskádovém stylu nastavím tloušťku rámečku (border-width), styl rámování (border-style), barvu rámečku (border-color) a odsazení obrázku (margin).

V mém příkladu jsem použila rámeček tloušťky 4px, styl rámování přerušovaná čára vyvolaná příkazem dashed a barvu rámečku čokoládově hnědou (#7B3F00).

Abych obrázek odsadila od okrajů internetové stránky, použiji element margin. Nastavím odsazení z levé strany (margin-left), od horního okraje (margin-top) od spodního okraje (margin-bottom) internetové stránky.

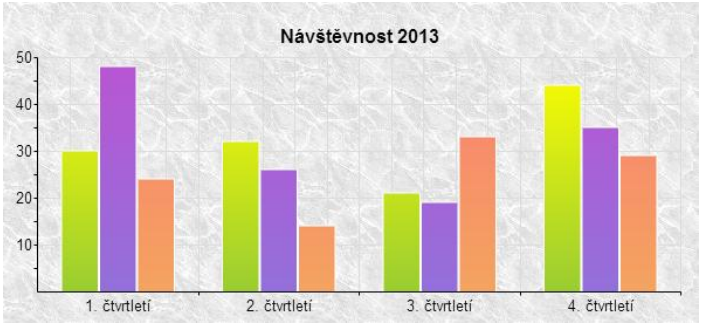
Zdrojový kód 32: CSS3 – rámeček plátna:

```
canvas#obrazek { border-width: 4px;
                 border-style: dashed;
                 border-color: #7B3F00;
                margin-top: 40px;
               margin-left: 200px;
              margin-bottom: 40px;}
```

Canvas – závěr

Tvoření obrázku v rozhraní Canvas je jednoduchý a užitečný způsob vkládání grafiky na internetové stránky. Výhodou vytvoření obrázku v rozhraní Canvas je možnost jeho úpravy přímo ve zdrojovém kódu internetové stránky. Nemusím používat žádný program pro tvorbu grafiky.

10.2 Canvas – graf



Obrázek č. 5: Canvas - graf

V rozhraní Canvas nemusím pracovat jen s tvary, mohu vytvořit i graf. Dříve se k vytvoření grafu používal Flash, ale tvoření grafu v rozhraní Canvas je podstatně jednodušší. Pro vytvoření sloupcového grafu využiji knihovnu RGraph, která je volně ke stažení na internetových stránkách www.rgraph.net.

Do hlavičky zdrojového kódu vložím element <script>. Do elementu <script> přidám atribut scr s cestou k dokumentu napsaném v JavaScriptu. Název dokumentu v JavaScriptu bude mít koncovku .js. Pro vytvoření sloupcového grafu využiji tři knihovny RGraph. common. core. js, RGraph.bar.js. a jquery.min.js.

Zdrojový kód 33: JavaScript – Rgraph:

```
<script src="libraries/RGraph.common.core.js" ></script>
<script src="libraries/RGraph.bar.js" ></script>
<script src="libraries/jquery.min.js" ></script>
```

V příkladu si uvádím sloupcový graf s hodnotami návštěvnosti za rok 2013. Stejně jako u vkládání obrázku do rozhraní Canvas nejprve nadefinuji plátno. Postupuji úplně stejně jako u obrázku. Nastavím identifikátor a velikost plátna. Do zdrojového kódu grafu nejprve připíši metodu newRGraph.bar. Do které zadám identifikátor elementu <canvas>. Za identifikátor elementu <canvas> vyplním do hranatých závorek hodnoty sloupců grafu. Další řádky budou začínat metodou sloupec.Set. V prvním řádku metody sloupec.Set jsem uvedla značku labels pro název sloupců. Název sloupců se nachází na ose x. Na ose y v metodě sloupec.Set vyvolám značku ymax, která vykresluje maximální hodnotu sloupců.

Pro lepší přehlednost grafu vložím nadpis. Nadpis grafu vyvolám značku title. Kde jako parametr uvádím text nadpisu.

Sloupcový graf, který jsem vytvořila, se v každém čtvrtletí skládá z tří sloupců, které označují měsíce v jednom čtvrtletí. Pro lepší přehlednost sloupců grafu vyvolám značku colors. Rozhodla jsem se udělat u sloupců barevný přechod. Proto pro každý sloupec zvlášť nadefinuji parametr gradient. Parametr gradient bude obsahovat vždy dvě barvy, které budou odděleny dvojtečkou. Nemusím používat jen dvě barvy, ale klidně barevný přechod s více barvami.

Sloupcům v grafu vytvářím ohraničení, které definuji značkami strokestyle a linewidth. V případě strokestyle definuji barvu rámování a linewidth definuji šířku rámování.

Na závěr zdrojového kódu přidám metodu sloupec.Draw, která mi požadovaný graf vykreslí.

Zdrojový kód 34: JavaScript - sloupcový graf:

```
<canvas  width="600" height="250" id="graf">
</canvas>
<script>
var sloupec = new RGraph.Bar('graf', [[30,48,24],[32,26,14],[21,19,33],[44,35,29]])
sloupec.Set('labels', ['1. čtvrtletí', '2. čtvrtletí', '3. čtvrtletí', '4. čtvrtletí'])
sloupec.Set('ymax', 50)
 sloupec.Set('title', 'Návštěvnost 2013')
sloupec.Set('strokestyle', 'white')
sloupec.Set('linewidth', 2)
sloupec.Set('colors',['Gradient(#9ACD32:#FFFF00)','Gradient(#9370DB:#BA55D3)',
'Gradient(#F4A460:#FA8072)'])
sloupec.Draw();
</script>
```

Graf – závěr

Vykreslování grafu pomocí knihovny RGraph v rozhraní Canvas je snadné. Dříve se grafy vytvářeli pomocí flashových aplikací. Vytvořením grafu v rozhraní Canvas si nemusím dělat starosti se zásuvnými moduly. Graf může mít potíže při vykreslování s kompatibilitou některých internetových prohlížečů.

11 Závěr

Internetové stránky navštěvujeme dnes a denně. Málo, kdy si však uvědomujeme kolik, práce a úsilí vynaloží tvůrci k vytvoření jednoduché internetové stránce. Ve své bakalářské práci jsem se zaměřila na specifikaci HTML5 a její multimediální elementy. Cílem v teoretické části bakalářské práce bylo pojednat o nových vlastnostech specifikace HTML5. Vyhledala jsem odbornou literaturu, která se zabývala tématikou specifikace HTML5. Zkoumala jsem multimediální elementy <video> a <audio>, rozhraní Canvas a také rozhraní off-line aplikace. Praktickou část bakalářské práce jsem věnovala bližšímu seznámení s rozhraním Canvas, elementem <video> a základním rozdílům struktury zdrojového kódu mezi HTML5 a XHTML. Svoje tvrzení jsem dokládala příklady zdrojových kódů.

Z teoretické části bakalářské práce vyplývá, že se specifikací HTML5 se dá pracovat v mnoha oblastech. Specifikace HTML5 bude mít velký dopad pro tvorbu internetových stránek. Specifikace HTML5 se stále vyvíjí a přinese mnoho dalších zajímavých způsobů pro tvorbu internetových stránek. Bez pochyby nejzajímavější je 3D grafika.

Z praktické části bakalářské práce vyplývá, že rozhraní Canvas je velmi zajímavé pro tvorbu obrázků a grafů. Největším přínosem vidím to, že rozhraní Canvas je přímo ve zdrojovém kódu a já nemusím řešit cestu dalšího dokumentu (např. obrazek vložený pomoci img). Srovnáním elementu <video> a <object> jsem si potvrdila jednoduchost specifikace HTML5. Jednoduchostí oplývá HTML5 i v deklaraci typu dokumentu a znakové sady. Velkým přínosem je struktura internetové stránky, která je mnohem přehlednější než u XHTML.

12 Slovník pojmů

Rozhraní API

Zkratka pro Application Programmin Interface. Api je rozhraní pro programování aplikací. S rozhraním API pracují například Mapy Google.

SVG

Zkratka pro Scalable Vecctor Graphics. Specifikace pro vytváření vektorové grafiky pomocí XML.

VML

Zkratka pro Vector Markup Language. Jazyk vyvinutý Microsoftem pro tvorbu vektorové grafiky.

WebGL

Zkratka pro Web Graphics Library. WebGL je založeno na principu JavaScriptového API rpo nativní zobrazování interaktivní 3D grafiky.

WebKit

Název pro renderovací jádra prohlížeče postavené na frameworku, který využívá aplikace operačního systému Mac OS X.

DOM

Zkratka pro Document Object Model (objektový model dokumentu). Objektově orientovaná reprezentace XML a HTML dokumentu.

13 Literatura

1. BEDNÁŘ, Vojtěch. *Alternativní webové prohlížeče: Firefox, Opera, Mozilla, Maxthon a další.* Vyd. 1. Brno: Computer Press, 2006, 168 s. ISBN 80-251-0566-0.

2. CASTRO, Elizabeth. HTML*, XHTML a CSS: názorný průvodce tvorbou WWW stránek.* Vyd. 1. Brno: Computer Press, 2007, 438 s. ISBN 978-80-251-1531-2.

3. GOLDSTEIN, Alexis. *HTML5 a CSS3 pro webové designéry.* 1. vyd. Brno: Zoner Press, 2011, 286 s. ISBN 978-80-7413-166-0.

4. HOGAN, Brian P. *HTML5 a CSS3: výukový kurz webového vývojáře.* Vyd. 1. Brno: Computer Press, 2011, 272 s. ISBN 978-80-251-3576-1.

5. LACKO, Luboslav. *Silverlight: výukový průvodce tvorbou interaktivních aplikací*. 1. vyd. Brno: Computer Press, 2010, 464 s. ISBN 978-80-251-2716-2.

6. LUBBERS, Peter, Brian ALBERS a Frank SALIM. *HTML5: programujeme moderní webové aplikace.* Vyd. 1. Brno: Computer Press, 2011, 304 s. ISBN 978-80-251-3539-6.

7. NAUMANN, Friedrich. *Dějiny informatiky: od abaku k internetu*. Vyd. 1. Překlad Michaela Voltrová. Praha: Academia, 2009, 422 s. Galileo, sv. 40. ISBN 978-802-0017-307.

8. PFEIFFER, Silvia. *HTML5 - audio a video: kompletní průvodce.* Vyd. 1. Brno: Zoner Press, 2011, 350 s. Encyklopedie webdesignera. ISBN 978-80-7413-147-9.

9. PÍSEK, Slavoj. *JavaScript: efektní nástroj oživení www stránek*. 1. vyd. Praha: Grada, 2001, 231 s. ISBN 80-247-0014-X.

10. PROCHÁZKA, David. CSS *a XHTML: tvorba dokonalých WWW stránek krok za krokem*. 2., aktualiz. vyd. Praha: Grada, 2011, 175 s. Průvodce (Grada). ISBN 978-80-247-3897-0

11. SCHAFER, Steven M. *HTML, XHTML a CSS: bible [pro tvorbu WWW stránek]* 1. vyd. Praha: Grada, 2009, 647 s. ISBN 978-80-247-2850-6.

12. SKLENÁK, Vilém. *Data, informace, znalosti a Internet*. Vyd. 1. V Praze: C.H. Beck, 2001, xvii, 507 s. C.H. Beck pro praxi. ISBN 80-717-9409-0.

14 Internetové zdroje

13. Článek Silverlight 5 je prý poslední. Konec soubeje s Flashem. Autor Jakub Čížek. Dostupne z <http://www.zive.cz/bleskovky/silverlight-5-je-pry-posledni-konec-souboje-s-flashem/sc-4-a-161472/default.aspx>.

14. Další informace o tvorbě internetových stránek použité ze serveru Jak psát web. Dostupné z < http://www.jakpsatweb.cz/ >.

15. Dostupné JavaScripty pro tvorbu grafů v rozhraní Canvas. Dostupné z <http://www.rgraph.net/download>.

16. Informace o aplikaci Flash Professional CS6. Dostupné z <http://www.adobe.com/cz/products/flash/flash-to-html5.html>.

17. Kompatibilita internetových prohlížečů s obsahem internetových stránek v HTML5. Dostupné z <http://www.w3schools.com/html/html5_intro.asp>.

18. Modernizr 2 - testování schopností na míru. Autor Martin Malý. Dostupné z <http://www.zdrojak.cz/zpravicky/modernizr-2-testovani-schopnosti-na-miru/>.

19. Základní zdrojové kódy pro práci s rozhraní Canvas. Dostupné z <http://www.w3schools.com/html/html5_canvas.asp>.

Jméno a příjmení:	Karolína Šmatlová
Katedra:	Technické a informační výchovy
Vedoucí práce:	Mgr. Jan Kubrický, Ph. D.
Rok obhajoby:	2014

Název práce:	Tvorba a prvky multimediálního webu prostřednictvím specifikace HTML 5
Název v angličtině:	Creation and elements multimedia web by specifications HTML5
Anotace práce:	Bakalářská práce pojednává o multimediálních objektech vytvářených specifikací HTML5 a o možnostech využití vytvořených multimediálních objektů. Hlavním cílem teoretické části práce je analyzovat možnosti tvorby a využití multimediálních prvků s využitím specifikace HTML5. Cílem praktické části je představit možnosti práce s rozhraním Canvas. Práci s multimediálním prvkem video pomocí specifikace HTML5 a XHTML a základní rozdíly ve struktuře zdrojového kódu HTML5 a XHTML.
Klíčová slova:	HTML5, Rozhraní Canvas, Elementy video a audio, XHTML
Anotace v angličtině:	The bachelor dealsof multimedia objects created specifications HTML5 and the possibilities use created multimedia objects. The main intention theoretical part analyze options creation and use multimedia elements using specifikacions HTML5. Intention practical part is to present opportunities to work with interface Canvas. Work with multimedia elements video using the specification HTML5 and XHTML and fundamental differences in the structure of source code HTML5 and XHTML.

Klíčová slova v angličtině:	HTML5, Interface Canvas, Elements of video and audio, XHTML
Přílohy vázané v práci:	
Rozsah práce:	51
Jazyk práce:	Čeština