TECHNICKÁ UNIVERZITA V LIBERCI
Ekonomická fakulta

HTML 5 A MULTIMÉDIA

Diplomová práce

Studijní program: N6209 – Systémové inženýrství a informatika
Studijní obor: 6209T021 – Manažerská informatika

Autor práce: **Bc. Radim Danihelka**
Vedoucí práce: Ing. Petr Weinlich, Ph.D.

Liberec 2015

Prohlášení

Byl jsem seznámen s tím, že na mou diplomovou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé diplomové práce pro vnitřní potřebu TUL.

Užiji-li diplomovou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Diplomovou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé diplomové práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum: 6. 5. 2015

Podpis: Daniela

Poděkování

Rád bych touto cestou poděkoval panu Ing. Petru Weinlichovi, Ph.D. za jeho cenné rady a připomínky při zpracování této diplomové práce. Děkuji také svému nejbližšímu okolí a rodině za podporu během celé doby studia.

Anotace

Diplomová práce se zabývá novou verzí značkovacího jazyka HTML5 a zejména pak elementy pro vkládání audia a videa a s nimi související problematikou multimediálních formátů a jejich distribucí. Dále se práce zabývá novou verzí kaskádových stylů CSS3. V práci jsou rozebrány novinky a přínosy, které tyto jazyky přináší. Práce popisuje jednotlivé nové elementy, ukazuje jejich syntaxi doplněnou o ukázky zdrojových kódů a také je zmíněna podpora v internetových prohlížečích a mobilních zařízeních. Následuje zkoumání trhu podle zařízení a jejich parametrů, ze kterých uživatelé nejčastěji navštěvují webové stránky. V poslední části je ukázáno praktické využití HTML5 a CSS3 pro distribuci multimédií.

Klíčová slova

HTML5, multimédia, video, audio, CSS3

Annotation

HTML5 and multimedia

This thesis describes new version of markup language HTML5, particularly elements for inserting audio and video. There is also described related issues of multimedia formats and their distribution. The thesis is also looking into new version of cascading style sheet language CSS3. Thesis describes new elements, shows their syntax along with source code examples. Also support among web browsers and mobile devices is noticed. The following is market research that objective is internet access by type of device and its parameters. In the last part is demonstrated practical usage of HTML5 and CSS3 for distribution multimedia.

Key Words

HTML5, multimedia, video, audio, CSS3

Obsah

Seznam zkratek

API	Application Programming Interface
CSS	Cascading Style Sheets
DASH	Dynamic Adaptive Streaming over HTTP
DTD	Document Type Declaration
DRM	Digital rights managment
GIF	Graphics Interchange Format
HLS	HTTP Live Streaming
HTML	HyperText Markup Language
IE	Internet Explorer
JS	JavaScript
MP3	MPEG-1 nebo MPEG-2 Audio Layer III
MP4	MPEG-4 Part 14
MPEG	Moving Picture Experts Group
OGV	Ogg Video
SVG	Scalable Vector Graphics
TS	Transport Stream
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
WAV	Waveform audio file format
WHATWG	Web Hypertext Application Technology Working Group
WOFF	Web Open Font Format
XHTML	Extensible Hypertext Markup Language
XML	Extensible Markup Language

Seznam tabulek

Seznam obrázků

Úvod

HTML a CSS jsou technologie používané k tvorbě webových prezentací a které tvoří základní součást prakticky každé webové stránky. Obě technologie prošli dlouhým vývojovým cyklem a jejich nejnovější revize přinášejí řadu technologických inovací. HTML5 již plní funkci standardu avšak na CSS3 se stále pracuje, přičemž většina zásadních modulů již dosáhla finálního stádia vývoje.

Obsah práce se zabývá novými vlastnostmi HTML5 a CSS3, které tyto nové standarty přinášejí do vývoje webových stránek zejména pak v oblasti multimediálního obsahu.

V první části je nastíněna historie vývoje multimediálního obsahu na webu a s ním souvisejících technologií. V další kapitole je rozebráno samotné HTML počínaje vývojem a historií, pak již následuje popis samotného HTML5 a jeho nových prvků. Následuje kapitola věnující se multimediálnímu obsahu na webu. Tato kapitola je zaměřena zejména na nové elementy video a audio a související problematiku audio a video formátů. Jsou zde zmíněny jednotlivé možnosti distribuce těchto multimédií a také jejich alternativy. V další kapitole je zacíleno na CSS3 a jeho vlastnosti usnadňující tvorbu multimediálních stránek. Následující kapitola je zaměřena na webové uživatele a trh. V první části je uvedena statistika, z jakých zařízení konzumují webový obsah. Následuje statistika zabývající se zobrazovacími možnostmi uživatelských zařízení. V poslední části této kapitoly jsou rozebrána typická mobilní zařízení, která jsou aktuálně dostupná na trhu. Cílem této kapitoly je odhalit na jaká zařízení by měl vývojář optimalizovat své webové stránky. V poslední části je popsán postup tvorby webových stránky s responzivním designem a postup tvorby HTML5 video přehrávače sloužícímu pro distribuci multimediálního obsahu.

Cílem práce je shrnout situaci v oblasti vývoje HTML a multimediálního obsahu na webu. Ukázat jaké výhody a také, jaká omezení přináší použití HTML5 pro distribuci multimediálního obsahu.

Literární rešerše

V diplomové práci je čerpáno z několika typů literatury. Mezi hlavní patří odborná literatura, odborné články, online knihovny a weby. Jelikož HTML5 je čerstvým standardem byla mimo tištěné literatury hlavním zdrojem také stránka W3C, která poskytuje aktuální specifikace tohoto standardu: HTML5 Specifications. 2014. *W3C* [online]. [cit. 2015-02-01]. Dostupné z: http://www.w3.org/TR/html5/.

Díky možnosti přístupu studentů Technické univerzity v Liberci do placené databáze ProQuest Central bylo čerpáno i z této databáze. Příkladem může být článek z časopisu „Video Edge“ pojednávající o adaptivním stremování, zejména pak o standardu DASH: SIMPSON, Wes. 2013. Adaptive Streaming Adoption. *Video edge* [online]. (3) [cit. 2015-03-04]. ISSN 2372-9244. Dostupné z: http://search.proquest.com/docview/1637940701/fulltext/17D080027EBB4355PQ/1?accountid=17116 .

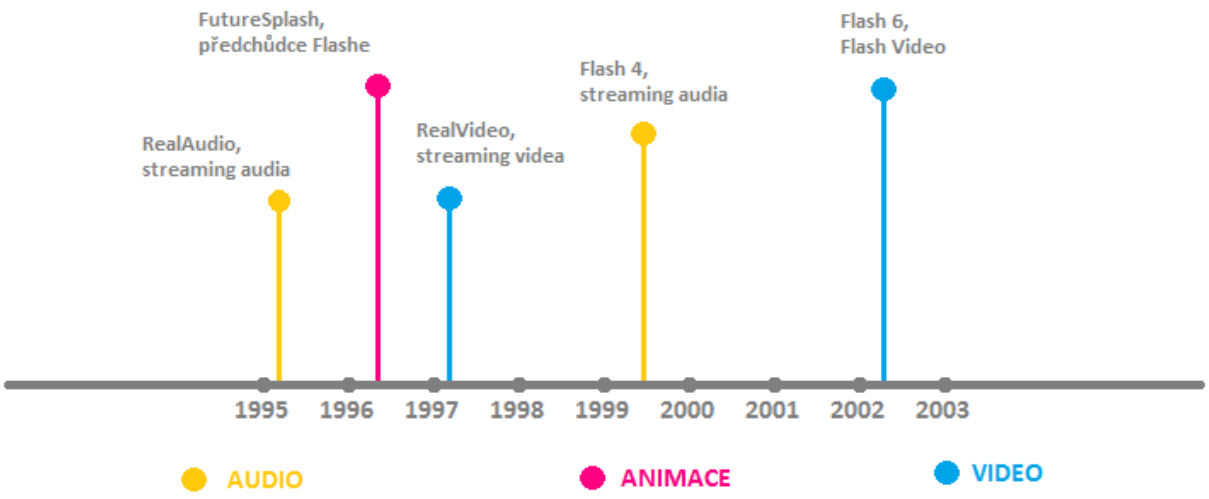
Technologie HTML5 a CSS3 jsou velice dobře vysvětleny v knižní publikaci: CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. 1. vyd. Brno: Computer Press, 2012, 439 s. ISBN 978-80-251-3733-8. Publikace se nezabývá pouze novými prvky těchto standardů, ale je kompletním průvodcem i pro začínající vývojáře. Obdobně zaměřená je kniha od Petera Lubbers, která však cílí již na pokročilejší vývojáře: LUBBERS, Peter, Brian ALBERS a Frank SALIM. 2011. *HTML5: programujeme moderní webové aplikace*. Vyd. 1. Brno: Computer Press, 304 s. ISBN 978-80-251-3539-6. V publikaci autora Makzan najdeme názorné použití kreslícího plátna Canvas pro vytváření her a dalších interaktivních aplikací: MAKZAN,. 2012. *Programujeme hry v HTML5*. 1. vyd. Brno: Computer Press, 320 s. ISBN 978-80-251-3731-4. V knize od Marka Pilgrima můžeme nalézt praktické rady k přípravě audio a video obsahu vhodného pro webové stránky: PILGRIM, Mark, Brian ALBERS a Frank SALIM. 2010. *HTML5: up and running*. 1st ed. Sebastopol, CA: O'Reilly, 304 s. ISBN 05-968-0602-7. Speciálně na multimediální prvky audio a video a jejich praktické použití včetně názorných příkladů je zaměřena kniha od Silvie Pfeiffer: PFEIFFER, Silvia. 2010. *The definitive guide to HTML5 video*. New York: Distributed to the book trade worldwide by Springer Science Business Media, xiv, 321 p. Expert's voice in Web development. ISBN 14-302-3090-8.

1 Webové stránky a multimediální obsah

Multimediálním obsahem se rozumí obrázky, animace, zvuk a video. Největší objem obrázků tvoří fotografie a loga, s animacemi se můžeme setkat například u reklamních bannerů, her a interaktivních grafů. Zvuk přehrávají některé stránky na pozadí, ale především je použit u internetových rádií a nakonec video používají internetové televize a služby pro sdílení videa.

1.1 Historie vývoje multimediálního obsahu na webu

Web prošel za celou svou dobu existence překotným vývojem. Zprvu sloužil pouze k zobrazování textu, ale časem vznikla potřeba zobrazovat také obrázky a další média. Hlavní milníky vývoje multimediálních technologií na webu v časovém horizontu jsou nastíněny na obrázku níže.



Obr. 1: Multimédia na webu z pohledu času
Zdroj: vlastní

Obrázky se staly prvním multimediálním prvkem webových stránek. K vložení obrázku na stránku slouží nepárový tag <img>. Obrázky se používaly také k tvorbě webové grafiky, pro tyto účely je nejvhodnějším formátem GIF, který je schopen ukládat 256 barev. GIF byl také prvním formátem umožňujícím tvorbu jednoduchých animací. Jeho nástupcem se

stal formát PNG, který přinesl zejména rozšíření barevné hloubky na 24 bitů. Pro zobrazování fotografií se používá formát JPEG, který díky svému kompresnímu algoritmu ukládá obrázky do souborů poměrně malé velikosti. Není však vhodný pro čárovou grafiku, kde vytváří artefakty a také dochází k rozmazání hran. [1]

Dalším milníkem bylo zakomponování audia. Průkopníkem v této oblasti se stala společnost RealNetworks, která v dubnu roku 1995 audio formát RealAudio. Tento formát umožňoval streaming audia což znamená, že zvuková stopa byla přehrávána v reálném čase bez potřeby stáhnutí na lokální uložiště v PC. Toho využívaly především internetová rádia ke svému vysílání. [2]

V květnu roku 1996 byl uveden FutureSplash Animator, předchůdce populárního Adobe Flash. Jeho úkolem byla tvorba vektorové grafiky a animací. Tato technologie umožňovala tvorbu interaktivních aplikací (zejména her) a webových prezentací.

Posledním uvedeným multimediálním obsahem bylo video. Jako první přišla v roce 1997 společnost RealNetworks se svým formátem RealVideo. Použitý kompresní formát byl H.263. RealVideo bylo párováno s RealAudiem a distribuováno v kontejneru RealMedia.[2]

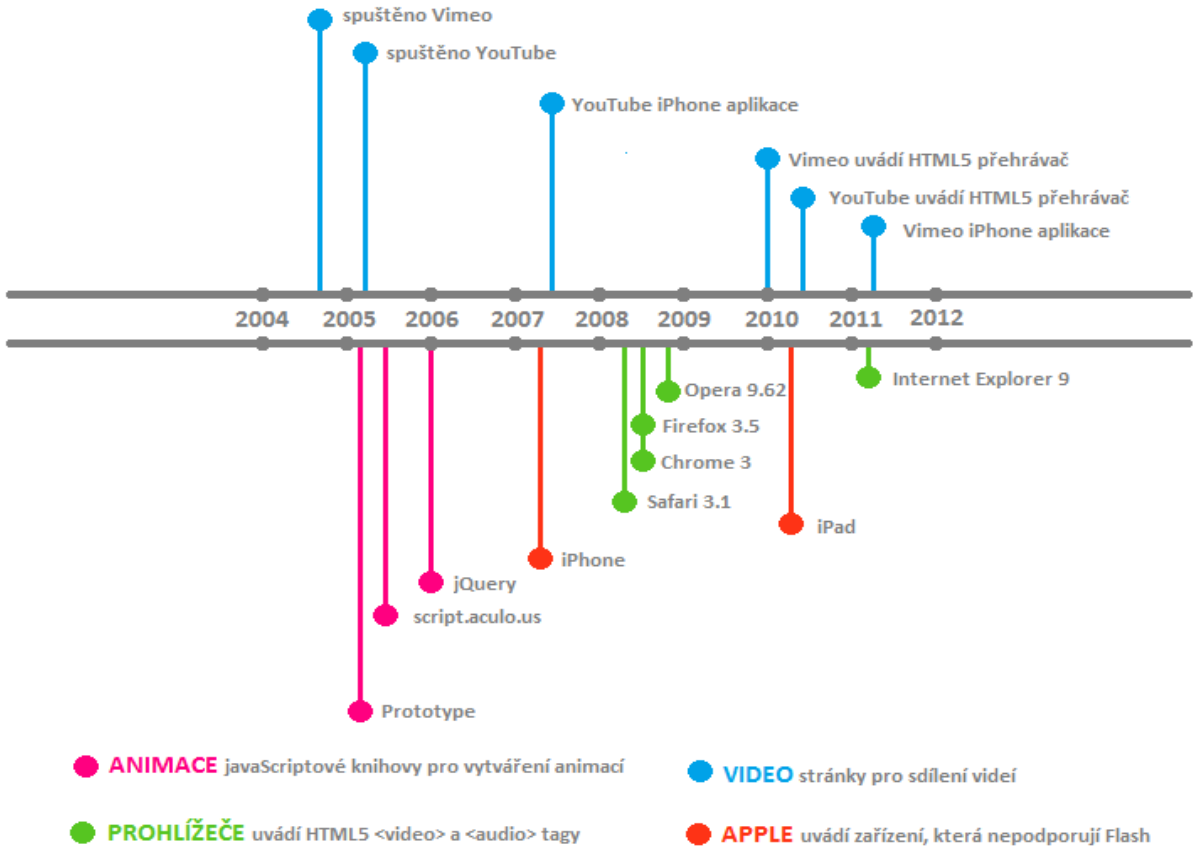
S implementací svých proprietárních formátů přišly také další společnosti. Apple přišel se svým multimediálním frameworkem QuickTime a zejména video formátem QuickTime Movie. Microsoft nabízel svůj multimediální framework Windows Media se svým Windows Media Video (WMV) formátem. Nevýhodou výše zmíněným metod distribuce videa na webové stránce je nutnost, aby měl uživatel nainstalovaný přehrávač těchto formátů ve formě pluginu v prohlížeči.

V roce 2002 přidala firma Adobe podporu pro video do Flash Playeru verze 6. První verze Flash Videa stavěla na proprietární implementaci komprese H.263, nazvanou Sorenson Spark. Novější verze pak již založeny na podpoře kompresního formátu H.264. Flash Video se stalo de-facto standardem pro streaming videa na webu. Postupně se začaly objevovat webové stránky určené pro sdílení video obsahu. Jako první se na scéně objevilo

[1] DUCKETT, Jon. *HTML: design and build websites*. Indianapolis, IN: Wiley, c2011, 490 p. ISBN 11-180-0818-9.

[2] Our Story. 2014. *RealNetworks* [online]. [cit. 2015-05-02]. Dostupné z: http://www.realnetworks.com/our-story/

v listopadu roku 2004 Vimeo následováno YouTube a dalšími. Díky tomu je Flash Player naistalován na 99 % všech počítačů a odsunuje další technologie do pozadí, tyto události reflektuje obrázek Obr. 2.[3] Další významný krok bylo uvedení JavaScriptových knihoven. Například v roce 2005 byla uvedena JavaSriptová knihovna Prototype, která měla za úkol usnadnit tvorbu animací a grafických efektů. V roce 2006 ji následovala knihovna jQuery, která se stala velice oblíbenou. Důkazem toho je, že našla využití u více než 60% z 10 000 nejnavštěvovanějších stránek[4].



Obr. 2 – Milníky na poli multimédií
Zdroj: vlastní

[3] FLV and F4V File Format Specification (Version 10.1). 2010. *Adobe.com* [online]. [cit. 2015-05-02]. Dostupné z: http://download.macromedia.com/f4v/video_file_format_spec_v10_1.pdf

[4] Top JavaScript Technologies. In: *SimilarTech* [online]. 2014 [cit. 2015-02-05]. Dostupné z: https://www.similartech.com/categories/javascript

Dostáváme se do roku 2007, kdy Apple vydává svůj iPhone, který nepodporuje Flash. To pro uživatele znamená nemožnost přehrávání videí na YouTube a dalších serverech, proto YouTube vydává intuitivní aplikaci pro přehrávání svého obsahu.

V roce 2008 nastává revoluce v oblasti prohlížečů kdy hlavní hráči jako Safari, Chrome, Firefox a Opera nasazují podporu pro nové HTML5 tagy <audio> a <video>. Internet Explorer v této podpoře zaostává a s podporou přichází až v roce 2011 u Internetu Explorer 9. Mezitím už servery jako YouTube a Vimeo uvádí HTML5 verze svých přehrávačů. [5]

[5] DUCKETT, Jon. *HTML: design and build websites*. Indianapolis, IN: Wiley, c2011, 490 p. ISBN 11-180-0818-9

2 HTML5

HTML je zkratka pro HyperText Markup Language, což je značkovací jazyk pro hypertext. Je hlavním z jazyků pro vytváření stránek v systému World Wide Web. HTML5 je jeho poslední pátou revizí.

2.1 Historie HTML

Jazyk HTML prošel za celou svou dobu existence rozsáhlým vývojem. Novější verze neznamená úplné přeformulování jazyka, ale obsahuje prvky z předchozích verzí a nové prvky, kupříkladu: *article*, *header*, *footer*, *video* a *audio*. Některé prvky byly naopak z novější verze odstraněny u HTML5 jsou to například: *font*, *frame*, *strike* a *center*.

Vývoj značkovacích jazyků prbíhal od prvních verzí HTML přes HTML4 a XHTML (Extensible Hypertext Markup Language) až k HTML5, které se začíná postupně prosazovat. Nejdůležitější milníky ve vývoji HTML jsou:

- 1991 – první vývoj HTML 1.0 – textový režim – Tim Berners-Lee
- 1992 – první webový prohlížeč Mosaic
- 1993 – HTML 2.0 – neoficiální verze, první grafický prohlížeč
- 1995 – HTML 2.0 – oficiální verze, HTML 3.0 – neoficiální rozšíření
- 1996 – HTML3.2 – přidání tabulek
- 1997 – HTML 4.0 – vylepšení a rozšíření tabulek, přidání formulářů, rámů
- 1999 – HTML 4.01
- 2004 – započaty práce na specifikaci HTML5 skupinou WHATWG
- 2012 – HTML5 vstupuje do fáze „ Candidate Recommendation“
- 28 listopadu 2014 – uvádí W3C HTML5 do statusu „Recommendation“, což znamená, že se stává oficiálním standardem.[6][7]

Za specifikací jazyka HTML5 stojí skupina WHATWG (The Web Hypertext Application Technology Working Group), která byla nespokojena s dalším postupem skupiny W3C,

[6] Historie HTML. 2012. *Garth.cz* [online]. [cit. 2015-05-02]. Dostupné z: http://www.garth.cz/uvod-do-html/historie-html/

[7] HTML5: A vocabulary and associated APIs for HTML and XHTML Editor's Draft 13 November 2014. *W3C* [online]. 2014 [cit. 2015-05-02]. Dostupné z: http://www.w3.org/html/wg/drafts/html/CR/

kdy další verze jazyka XHTML 2.0 měla sice přinést nové zajímavé vlastnosti, ale avšak za cenu porušení kompatibility s předchozími verzemi jazyka HTML a XHTML. Pro vývojáře prohlížečů to byl příliš revoluční krok, ti spíše preferovali konzervativnější přístup v postupném vylepšování současných technologií HTML 4.0.1 a XHTML 1.0.[8]

Nespokojenost těchto vývojářů vyústila k založení již výše zmíněné skupiny WHATWG. Na její půdě pak společně připravovali specifikaci platformy pro webové aplikace běžící v prohlížeči. Kromě rozšíření jazyka HTML specifikace zahrnovala i definice důležitých rozhraní pro využití ve skriptovacím jazyku JavaScript.

Toto rozštěpení dalšího vývoje HTML nebylo žádoucí. Nakonec i uvnitř skupiny W3C převážil názor, že XHTML 2 je slepá cesta ve vývoji, protože jej výrobci prohlížečů ignorují. V roce 2007 se proto síly W3C a WHATWG spojily a pracovní skupina HTML vzala jako základ specifikace vytvořené týmem WHATWG a na jejich základě začalo vznikat HTML5.[9]

HTML5 navazuje na HTML 4.0.1 a přidává pro vývojáře webových aplikací spousty nových vlastností, kvůli kterým je výhodné tuto nejnovější verzi jazyka používat. Zároveň zachovává možnost používat pro zápis stránek syntaxi XML a používat tak de facto XHTML. Co přináší HTML5 vývojářům se dozvíme v další kapitole.

2.2 Detekce podpory HTML5 v prohlížečích

Zdali prohlížeč podporuje HTML5 nelze jednoznačně určit, protože se skládá z jednotlivých částí, jež jsou v prohlížeči implementovány. Nemůžeme tedy určit, zda prohlížeč podporuje HTML5, ale můžeme detekovat podporu jednotlivých vlastností, např. zda podporuje video, canvas či API pro geolokaci.

[8] Historie a vývoj HTML. In: *Http://htmlguru.cz* [online]. 2013 [cit. 2015-05-02]. Dostupné z: http://htmlguru.cz/uvod-historie.html

[9] tamtéž

Techniky detekce

Prohlížeč při vykreslování stránky vytváří Document Object Model (DOM), což je kolekce všech objektů, které představují jednotlivé HTML prvky stránky. To znamená, že každý <p>, a atd. je v DOMu samostatným objektem.

Všechny objekty DOMu sdílí sadu společných vlastností, některé objekty mají určité vlastnosti navíc. V prohlížečích s podporou vlastností HTML5 budou mít některé objekty unikátní vlastnosti. Díky této skutečnosti můžeme rychlým pohledem na DOM zjistit, jaké vlastnosti prohlížeč podporuje.

Na této skutečnosti staví čtyři základní principy detekce podporovaných vlastností:

1. Ověření zda existuje určitá vlastnost globálního objektu (např. *window* nebo *navigator*) tímto způsobem můžeme detekovat například podporu geolokace
2. Vytvoření HTML prvku a následně ověření zda u něj existuje určitá vlastnost. Takto by se dala detekovat podpora canvasu
3. Vytvoření HTML prvku a zjištění zdali podporuje určitou metodu a následné zavolání této metody a zkontrolování její návratové hodnoty.
4. Vytvoření HTML prvku, nastavení vlastnosti na určitou hodnotu a následné testování zda si vlastnost tuto hodnotu uchovala[10]

Na výše uvedených principech je postavena JavaScriptová open source knihovna Modernizr, která slouží k detekci řady vlastností HTML a CSS3. Modernizr se spouští automaticky a vytváří globální objekt s názvem Modernizr, který obsahuje sadu vlastností nabývajících hodnotu boolean. Například pokud prohlížeč podporuje canvas bude Modernizr.canvas nabývat hodnotu true a naopak.[11]

2.3 Novinky a přínosy HTML5

Tato kapitola bude zaměřena na to, co nového nám nová verze HTML přináší. V první části se budeme věnovat nové sémantice, syntaxi a novým elementům. Následně se

[10] PILGRIM, Mark, Brian ALBERS a Frank SALIM. 2010. *HTML5: up and running*. 1st ed. Sebastopol, CA: O'Reilly, 304 s. ISBN 05-968-0602-7

[11] *Modernizr* [online]. 2014 [cit. 2014-04-08]. Dostupné z: http://modernizr.com/

přesuneme k novým audio a video elementům, u kterých podrobněji rozebereme problematiku audio a video formátů a celou podkapitolu zakončíme prvky pro zobrazování grafiky tedy canvas a SVG.

Nové značky v HTML5

Dnes je většina prvků spojována do logických celků elementem , ten však nenese žádný sémantický význam, a tak je vyloučeno strojové zpracování. Tento přístup slouží hlavně k rozdělení dokumentu pro stylování. Často jsou těmto divům přiřazovány rozumné atributy tříd a id, aby bylo zřejmé kam si prvek zařadit (např. hlavička, header, patička, footer). Pro stroje toto řešení však není vhodné, protože se nemůže spolehnout na vývojáře, zda takto danou třídu pojmenoval.[12]

HTML5 tak vstupuje do hry s mnoha elementy, které mají za úkol sémanticky popsat různé části dokumentu. Ve specifikaci tak najdeme elementy <article>, <nav> a další. Než se podíváme přímo na jednotlivé nové elementy, začneme úplně od začátku, tedy novou sémantikou u stávajících elementů. Tedy novým zápisem pro DOCTYPE, zápisem pro vkládání skriptů a stylů.

DOCTYPE

Doctype nám deklaruje typ dokumentu, tedy nese informaci pro prohlížeč, na jaký typ dokumentu nahlíží. *Doctype* musí být vždy umístěn ještě před tagem <html>. Hlavní roli *doctype* sehrál, pokud měly být stránky validní, musela se totiž dodržovat přesná syntaxe podle verze *doctype* jakou jsme si zvolili. Ta mohla vypadat pro HTML 4.01 například takto:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

Nový zápis je velice zjednodušen na pouhé: `<!DOCTYPE html>`

[12] GOLDSTEINOVÁ, Alexis, Louis LAZARIS a Estelle WEYLOVÁ. *HTML5 a CSS3 pro webové designéry*. 1. vydání. Jan Pokorný. Brno: Zoner Press, 2011, 286 s. ISBN 978-80-7413-166-0

V zápisu již není uvedena verze značkovacího jazyka, jelikož to pro něj není podstatné. Nyní to ovšem není důležité ani pro validaci stránky v případě, že se použije validátor pro HTML5.

Kódování dokumentu

Zjednodušení se projevilo i u zápisu kódování dokumentu. Dříve mohl zápis vypadat například takto:

```
<meta http-equiv="Content-Type" content="text/html; charset="windows-1250">
```

Nový zápis pak vypadá např. následovně: `<meta charset="windows-1250">`

Načítání kaskádových stylů

Načítání kaskádových stylů bylo zjednodušeno z původního `<link rel="stylesheet" type="text/css" href="styl.css">` na `<link rel="stylesheet" href="styl.css">`, ze zápisu byla odstraněna definice *type*, ta má již nastavenou defaultní hodnotu `type="text/css"`.

Prvek <script>

Nyní se podíváme, jak se zjednodušil zápis pro vkládání skriptů. Dnes se již na webu bez JavaScriptu téměř neobejdeme, vkládáme jak svoje vlastní skripty, tak linkujeme již hotové knihovny např. jQuery.

Původní zápis mohl vypadat např. takto: `<script src="script.js" type="text/javascript"></script>` dnes bychom použily nový zápis, který vznikne vynecháním atributu type: `<script src="script.js"></script>`. Další změnou u elementu <script> je nový atribut *async*, který má zajišťovat asynchronní zpracování skriptu a atribut *defer*, který naopak pozdrží zpracování skriptu až na okamžik, kdy byl dokument zparsován.

Jak je vidět HTML5 přináší vítané změny v zjednodušení kódu. Nyní si konečně ukážeme, co přinášejí již výše zmíněné nové sémantické elementy. Jednotlivé nové elementy jsou uvedeny v přehledné tabulce níže. Co každý element přináší podrobněji, se dozvíme dále v textu.

Tab. 1 – Nové sémantické prvky v HTML5

Tag	Popis
<article>	Definuje odstavec
<aside>	Definuje doplňující obsah k obsahu hlavnímu
<details>	Definuje dodatečné podrobnosti, které uživatel může zobrazit nebo skrýt
<figcaption>	Definuje titulek pro element <figure>
<figure>	Určuje soběstačný obsah jako ilustrace, diagramy, fotografie, zdrojové kódy atd.
<footer>	Definuje patičku stránky nebo sekce
<header>	Definuje hlavičku stránky nebo sekce
<main>	Specifikuje hlavní obsah dokumentu
<mark>	Definuje označený/zvýrazněný text
<nav>	Definuje navigační odkazy
<section>	Definuje sekci v dokumentu
<summary>	Definuje viditelný nadpis pro element <details>
<time>	Definuje datum a čas

Zdroj: vlastní zpracování podle HTML5 Specifications. 2014. W3C *[online]. [cit. 2015-02-01]. Dostupné z: http://www.w3.org/TR/html5/*

Prvek <section>

Prvek *section* představuje obecný úsek dokumentu nebo aplikace. Úsek v tomto kontextu znamená tematické seskupení obsahu, typicky s nadpisem. Příkladem úseků mohou být kapitoly, panely v dialogovém okně uspořádaném do panelů. Úvodní stránka webové stránky může být rozdělena do sekcí pro úvod, aktuality a kontaktní informace.

Prvek <article>

Prvek *article* představuje část stránky, která tvoří samostatný celek v dokumentu, stránce, aplikaci nebo webu a která je určena k nezávislé distribuci či opakovanému použití. Může

se jednat o příspěvek ve fóru, časopisecký nebo novinový článek, uživatelský komentář, či jakýkoli jiný prvek nebo obsah.

Prvek <aside>

Prvek *aside* představuje část stránky tvořenou obsahem, který přímo nesouvisí s hlavním obsahem stránky. Prvek lze použít pro typografické efekty, jako jsou citace vytažené z textu, komentáře na okraji, reklamy, skupiny prvků nav a pro další obsah, který je vnímán jako oddělený od vlastního obsahu stránky.

Prvek <header>

Prvek *header* představuje typicky skupinu úvodních nebo navigačních prostředků pro nejbližší sekční obsah. V praxi to znamená, že obaluje stejný obsah, jaký tradičně obalovali divy představující hlavičku, tedy například název webu, navigační menu, logo atd. Prvek se muže na stránce vyskytovat vícekrát v podobě hlaviček jednotlivých sekcí a také jako hlavička celého webu.

Prvek <footer >

Prvek *footer* představuje patičku pro část stránky, nebo pro celou stránku. Patička typicky obsahuje informace o své sekci, jako autora, odkazy na související dokumenty, autorská práva a podobně. Patička se nemusí nutně vyskytovat na konci části, ale obvykle to tak bývá.

Prvek <nav>

Prvek *nav* reprezentuje část stránky, která odkazuje na další stránky, nebo na části aktuální stránky. Nejčastěji se bude tedy jednat o primární navigaci stránky. Ne všechny skupiny odkazů na stránce musí být uvnitř prvku nav – pouze části tvořící hlavní navigační bloky jsou vhodné pro prvek nav.

Prvek <hgroup>

Prvek *hgroup* reprezentuje nadpis části. Prvek se používá k seskupení několika prvků *h1–h6*, když má nadpis několik úrovní, jako jsou podnadpisy, alternativní titulky nebo slogan.

Prvek <time>

Prvek *time* slouží k vkládání času. Element vznikl, protože lidé zapisují čas jinak než stroje. Právě pro stroje je určen formát času, který je zapsán v atributu *datetime*. Tím umožňuje webovým vyhledávačům lépe filtrovat výsledky podle času. Zápis za použití prvku *time* a jeho atributu *datetime* by mohl vypadat např. takto:

```
<p>Tento článek byl publikován <time datetime="2015-13-02">13. ledna 2014</time>.</p>
```

Prvky <figure> a <figcaption>

Element *figure* slouží k obalení doplňkového obsahu k obsahu hlavnímu, který je teoreticky možno vyjmout, aniž by došlo k ovlivnění významu hlavního obsahu. Tímto doplňkovým obsahem může být např. odstavec textu, obrázek či video.[13] K doplňkovému obsahu často přidáme popisek pomocí elementu *figcaption*. Na obrázku níže můžeme vidět příklad kódu a jeho výsledné zobrazení:

```
<figure>
  <img src="logo_html5.gif" alt="HTML5 logo">
  <figcaption>Obr.1 - HTML5 logo</figcaption>
</figure>
```

Obr.1 - HTML5 logo

Obr. 3 – Ukázka použití tagů <figure> a <figcaption>
Zdroj: vlastní, použitý obrázek ze stránky HTML(5) Tutorial. 2012. W3Schools [online]. [cit. 2015-05-03]. Dostupné z: http://www.w3schools.com/html/default.asp

[13] Webdesignérův průvodce po HTML5. 2010. *Zdrojak.cz* [online]. [cit. 2015-05-02]. Dostupné z: http://www.zdrojak.cz/clanky/webdesigneruv-pruvodce-po-html5-nova-semantika/

Prvek <details> a <summary>

Slouží k zobrazení a následnému skrytí dodatečných informací. S tímto elementem se pojí element <*summary*>, který slouží jako jeho titulek. Pokud uživatel klikne na tento titulek, zobrazí se obsah elementu *details*. Dříve bychom museli k tomuto efektu využít služeb JavaScriptu za použití HTML5 tato nutnost odpadá. Je ovšem nutné podotknout, že tento element ještě není podporován prohlížečem Firefox a Internet Explorer. Příklad použití:

```
<details>
    <summary>Poznámka: Element details není podporován ještě všemi
prohlížeči</summary>
     <ul>
       <li><strong>Opera</strong> - ano od verze 15.0</li>
       <li><strong>Chrome</strong> - ano od verze 12.0</li>
       <li><strong>Safari</strong> - ano od verze 6 </li>
       <li><strong>Firefox</strong> - nepodporuje</li>
       <li><strong>Exlorer</strong> - nepodporuje</li>
     </ul>
</details>
```

Prvek <main>

Tento element specifikuje hlavní obsah dokumentu. Obsah tohoto elementu by měl být unikátní, neměl by obsahovat žádný obsah, který se opakuje např. postranní panely, navigační odkazy, copyright atd. Podle specifikace W3C nesmí být následovníkem žádného z těchto prvků: <article>, <aside>, <footer>, <header> a elementu <nav>.

Formuláře

Vylepšení se dočkaly také formulářové prvky. Tyto vylepšení zejména usnadňují tvorbu formulářů tím, že se obejdeme pro jejich základní validaci bez použití JavaScriptu.

Nové typy vstupních prvků

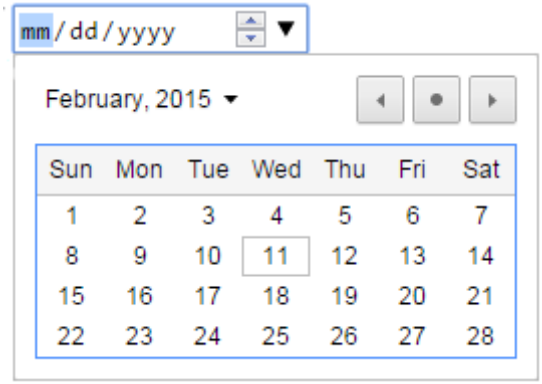
Dříve většinu vstupů obhospodařoval obyčejný <input type="text">, kterému sekundoval JavaScript a v něm napsaným kódem ošetřoval, aby do inputu mohli být zadána např. pouze čísla. HTML5 tyto potřeby řeší novými hodnotami atributu *type*. Dnes mají tyto nové atributy význam především pro uživatele dotykových mobilních zařízení. Vzhledem

k tomu, že tato zařízení mají virtuální klávesnici, mohou jí upravovat dle toho, co má uživatel zrovna zadávat. Tyto změny rozložení kláves v závislosti na typu vstupu ilustruje obrázek níže.

Obr. 4 – Různé rozložení kláves v iOS pří odlišných typech vstupních prvků
Zdroj: Webdesignérův průvodce po HTML5. 2010. Zdrojak.cz *[online]. [cit. 2015-03-02]. Dostupné z: http://www.zdrojak.cz/clanky/webdesigneruv-pruvodce-po-html5-tahame-data-od-navstevnika/*

Typ *number* nám tedy povoluje zadávat pouze čísla a klávesnice na dotykových zařízeních se přizpůsobí k zadávání čísel. Obdobně je to i pro typ *email*, kdy si prvek sám ověří, zdali se jedná o emailovou adresu a opět se tomu i přizpůsobí klávesnice dotykových zařízení, která nyní usnadňuje zadávání emailové adresy, stejně tomu bude i u typu *url*, který slouží pro zadávání *url* adresy.

Nakonec se dostáváme k typům sloužícím pro výběr data, které zahrnují tyto typy: *date*, *month*, *week*, *time*, *datetime* a *datetime-local*. Obrázek níže ilustruje výběr data, máme-li zvolen typ *date*.



Obr. 5 – Novinka v HTML5 v podobě výběru data
Zdroj: vlastní

Nové atributy formulářových prvků

Mezi nové atributy patří atribut *autocomplete*, který slouží k automatickému vyplňovaní formulářových polí, což nám může velice usnadnit práci. Ne vždy je tato funkce žádoucí a to zejména v případě pokud se jedná o citlivá data a tak ji může vývojář zakázat.

Dalším užitečným atributem je *placeholder*, který slouží k tomu, aby v poli inputu byla zobrazena nápověda co má uživatel vyplnit, tedy jaká je očekávaná hodnota pole.

Atribut *autofocus* se stará o to, zdali má být danému formulářovému prvku přiřazen „focus“ při načtení stránky. *Autofocus* může být nastaven pouze pro jeden prvek na stránce. Jedná se o pouhé přemístění kurzoru do daného formulářového prvku. Příkladem může být úvodní stránka Google nebo Seznamu, kde je kurzor při načítání stránky umístěn do vyhledávacího pole.

Atribut *required* zajišťuje, aby formulářové pole nezůstalo nevyplněné. Užitečné jsou také atributy *min* a *max*, které zajištují aby hodnota vstupů byla v požadovaném rozpětí. Jsou podporovány tyto typy vstupů: *number, range, date, datetime, datetime-local, month, time* a *week.*

Posledním atributem je atribut *pattern*, který nám umožňuje požadovat určitý formát neboli vzor vstupu. Tento vzor se zadává pomocí regulárních výrazů. Tímto například můžeme zajistit požadavky na sílu hesla.

2.4 SVG a Canvas

K HTML5 kromě nové sémantiky a tagů patří nové související technologie. Jedněmi z nich jsou SVG a Canvas, které slouží k tvorbě a zobrazování grafiky. V prohlížečích je jejich podpora již na velice dobré úrovni.

Tab. 2 – Podpora SVG a Canvas v prohlížečích

Vlastnost	Firefox	Chrome	IE	IE Mobile	Opera	Safari	iOS Safari	Android
SVG	3+	3+	9+	10+	10.0+	2.0+	1.0+	1.0+
Canvas	4+	6+	9+	10+	10.5+	3.2+	3.2+	4.0+

Zdroj: vlastní zpracování podle dokumentace jednotlivých prohlížečů

SVG

SVG je vektorový grafický formát založený na XML. Jeho obsah může být statický, dynamický, interaktivní či animovaný. Navíc může být stylován pomocí CSS a dynamicky se s ním může pracovat skrze SVG DOM. SVG obrázky lze vytvářet v pouhém textovém editoru, ale mnohem příjemnější je použití nějakého kreslícího programu, jako je například freewarový Inskape.[14]

SVG najde využití při tvorbě ikon, firemních log, fontů, vzorů a dokonce i celých fontů. Jeho výhodou je nízká velikost výsledných obrázků. Díky tomu, že se jedná o vektorový formát, tak obrázek můžeme jakkoliv zvětšovat či zmenšovat bez ztráty kvality.

Existují čtyři způsoby jak vložit SVG grafiku na webovou stránku:

První způsob znamená přímé vložení do HTML kódu:

```
<svg ..>
      <path ../>
</svg>
```

Druhým způsobem je vložení pomocí tagů <object>:

```
<object type="image/svg+xml" data="img.svg"></object>
```

[14] *Inkscape* [online]. [cit. 2015-07-03]. Dostupné z: https://inkscape.org/en/

Třetí možnost je využití tagu <img>:

```
<img src="obrazek.svg">
```

Posledním způsobem je vložení pomocí CSS:

```
.class{
background: url(obrazek.svg);
}
```[15]

Canvas

Canvas je jakési kreslící plátno, do kterého je naopak možné kreslit bitmapovou grafiku s využitím Javascriptu. Umožňuje nakreslit různé tvary, rovné čáry, oblouky, gradienty a další. Jeho prostřednictvím je dokonce možné manipulovat s jednotlivými pixely v obrázcích nebo videu. *Canvas* můžeme využít pro vytváření interaktivní grafiky, jejichž změny reagují na uživatelské akce. Příkladem mohou být aplikace pro kreslení, interaktivní grafy a v neposlední řadě například i hry vložené přímo v HTML stránce bez nutnosti použití dalších pluginů.[16]

[15] The ultimate SVG guide. 2014. *PSDtoWORDPRESS* [online]. [cit. 2015-05-03]. Dostupné z: https://psdtowp.net/svg.html

[16] FULTON, Steve a Jeff FULTON. 2011. *HTML5 Canvas: native interactivity and animation for the Web*. 1st ed. Sebastopol, CA: O'Reilly, xix, 628 p. ISBN 978-1-449-39390-8.

3 Audio a video

Audio a video jsou dlouho očekávané nové elementy, které standardizují vkládání multimediálních souborů do webových stránek. Jak už bylo zmíněno v úvodní kapitole, tak se dříve na tyto účely používali technologie, které vyžadovaly instalaci pluginů a samozřejmě také pravidelnou aktualizaci. Nejvíce rozšířený byl potom Adobe Flash. Ovšem na trhu se začala objevovat mobilní zařízení, která přestávala Flash podporovat, nejdříve to byl Apple, později se přidal také Android. Apple upustil od podpory zejména, protože chtěl podporovat otevřené standardy, jako jsou HTML, CSS a JavaScript a ne proprietární Adobe Flash, který se pomalu adaptuje na nové technologie. Dalším důvodem byla výdrž na baterii. K dosažení dlouhé výdrže na baterii je potřeba, aby zařízení dekódovalo video hardwarovou cestou, protože softwarové dekódování spotřebovává příliš mnoho energie. Spoustu mobilních zařízení již obsahuje HW dekodér komprese H.264. I když Flash nakonec přidal podporu pro H.264, drtivá většina stránek s Flash videem stejně vyžaduje podporu staršího dekodéru takže musí být video dekódováno softwarovou cestou. Když by měly všechny tyto stránky překódovat videa s použitím H.264, tak ho rovnou mohou nabídnout bez potřeby Flashe díky HTML5. Dalším argumentem je dotykové ovládání s podporou multi-dotyků a různých gest, na které Flash nebyl navrhnut. U těchto zařízení funkce jako *hover* a *mouse over* přestaly mít význam. [17]

Nyní se podíváme, jak jsou na tom uživatelé a jejich prohlížeče s podporou pro nové video a audio elementy. Vzhledem k tomu, že do hlavních prohlížečů byla podpora implementována již v průběhu roku 2008, tak se dnes již globálně dostáváme k číslu 90,75% prohlížečů vybavených podporou pro přehrávání multimédií. Mezi nepodporované prohlížeče patří: Internet Explorer 8 a verze nižší s 3 % uživatelskou základnou, dalších 3,14 % tvoří uživatelé prohlížeče Opera Mini. Zbylá procenta tvoří uživatelé navštěvující web pomocí prohlížečů minoritních platforem, jako jsou např.: Windows Mobile, Xbox, Kindle, Symbian atp. Na základě této statistiky můžeme postavit závěr, že vývojáři můžou nové elementy plně využívat.[18]

[17] JOBS, Steve. Steve Jobs explains why Apple's not a fan of Flash. In: *NETWORKWORLD* [online]. 2010 [cit. 2015-02-11]. Dostupné z: http://www.networkworld.com/article/2230600/steve-jobs-explains-why-apple-s-not-a-fan-of-flash.html

[18]The State of HTML5 Video. In: *JWPlayer* [online]. 2015 [cit. 2015-02-11]. Dostupné z: http://www.jwplayer.com/html5/

S příchodem video a audio elementů bylo nutné rozhodnout, které formáty budou považovány za standartní a jejich podpora bude implementována napříč prohlížeči. Bohužel výrobci prohlížečů se nedohodli na jednotném formátu pro video ani audio. Jedna skupina prosazovala otevřené formáty, další skupina oponovala, že patentem chráněné H.264 pro video disponuje lepší kvalitou.

Pro přehlednost si uděláme jasno v pojmech jako kompresní formát, kodek a kontejner. Kompresní formáty mohou být ztrátové nebo bezztrátové a slouží zejména za účelem snížení datového toku, tedy aby výsledný soubor byl co nejmenší při zachování požadované kvality. Implementace specifického kompresního formátu se nazývá kodek, který může mít podobu jak softwarové knihovny, tak se může jednat o specializovaný hardware. Kodek se skládá ze dvou částí kodéru a dekodéru. Kodér vykonává funkci komprese (kódování) a dekodér vykonává funkci dekomprese (dekódování). Kontejner je potom obálka souboru nebo datového toku, obsahující jeden nebo více proudů multimediálních dat. U video kontejnerů se můžeme setkat s jednou video stopou a několika zvukovými stopami v různých jazycích a se stopami titulků.

3.1 Video kompresní formáty

Nyní se konečně můžeme plně věnovat jednotlivým kompresním formáty pro video, které můžeme použít na webu. U jednotlivých formátů budou také uvedeny nástroje vhodné pro jejich kódování. V přehledu jsou uvedeny nejen standardizované formáty skupinou W3C , ale i formáty nové, které jsou již také využívány nebo v blízké budoucnosti budou.

H.264

Formát H.264 byl vyvinut a standardizován v roce 2003 skupinou MPEG jako nástupce formátu MPEG-2. Našel uplatnění u digitálního pozemního vysílání, stal se standardem u Blu-Ray disců a dočkal se i implementace do Flash Videa. Dělí se na několik profilů, od základních, které jsou určeny pro web a mobilní zařízení až po nejvyšší používaných právě u Blu-Ray disců. [19]

[19] WIEGAND, T., G.J. SULLIVAN, G. BJONTEGAARD a A. LUTHRA. 2003. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video*

Jeho hlavní výhodu je, že mnoho přenosných zařízení obsahuje i jeho hardwarovou podporou. Hlavním problémem H.264 je jeho patentová chráněnost. A také nutnost platit licenční poplatky za jeho implementaci.

Kódování H.264 videa

V podstatě všechny open-source nástroje pro kódování H.264 videa jsou založeny na knihovně x264, která je uvolněna pod licencí GNU GPL. Jedním z nejlepších je nástroj FFmpeg, který postrádá uživatelské rozhraní a ovládá se za pomocí příkazové řádky. Alternativou mu může být např. software Handbrake a VLC, které již disponují uživatelským rozhraním. My se zaměříme na nástroj FFmpeg, protože je to univerzální nástroj, který můžeme ze zdrojového kódu zkompilovat na jakoukoliv platformu. Navíc není rozhodující, který nástroj použijeme, ale půjde nám o porozumění jednotlivým profilům a parametrům pro kódování.

MPEG-4 H.264 zahrnuje několik přednastavených profilů s různou charakteristikou. Již dostupné množství profilů může činit problémy, kdy jaký profil zvolit. Profily slouží k definování vlastností, jaké může kodér použít. Vyšší profily zaznamenají vyšší nároky na výkon zařízení pro kódování i dekódování, ale zároveň vyšší kvalitu výstupu. Nejpoužívanější profily jsou:

- Baseline Profile: nižší míra komprese, méně náročný na HW, vhodný pro starší mobilní zařízení
- Main Profile: podporují novější mobilní zařízení, používaný pro televizní vysílaní a na webu
- High Profile: nejvyšší kvalita, použití u Blu-ray disců

Dále k jednotlivým profilům můžeme přiřadit H.264 level, který přidává další omezení jako maximální datový tok, maximální počet snímků za sekundu a také rozlišení. U každého zařízení by v manuálu mělo být uvedeno jaký maximální profil a level podporuje. Pro zajištění nejvyšší míry kompatibility na cílových zařízeních bychom video měli

kódovat v Baseline profilu level 3, tím ale poměrně naroste datový tok potřebný k dosažení stejné kvality jako u vyšších profilů.[20]

Kód níže ukazuje příklad příkazů pro kódování zdrojového videa ve formátu dv do kontejneru mp4, použit je profil baseline, a datový tok je nastaven na hodnotu 3000 Kbit/s. Audio je kódováno do formátu ACC s datovým tokem 192 kbit/s.

```
ffmpeg -i zdrojovevideo.dv \
-profile:v baseline -level 3.0
-vcodec libx264 -vpre baseline -vb 3000k \
-acodec libfaac -ab 192k \
-threads 0 vystupnivideo.mp4
```

Theora

Theora je otevřený formát pro kompresi videa využívající ztrátovou kompresi. Formát není zatížený patenty ani poplatky za jeho implementaci. Byl vyvinut skupinou Xiph.org Foundation v roce 2004. Stejnojmenná skupina také nabízí video kodek libtheora, který je referenční implementací Theory napsán v jazyce C a uvolněn pod licencí BSD. [21]

Kódování Theora videa

Open source nástroje pro kódování do tohoto formátu jsou založeny na výše zmíněne knihovně libtheora. Mezi nejpoužívanější nástroje patří ffmpegtheora a FFmpeg, které jsou bez grafického rozhraní. Nástroje s grafickým rozhraním jsou např. VLC a Miro Video Converter. Další alternativou může být plug-in FireFogg pro prohlížeč Firefox. My použijeme nástroj ffmpegtheora.

Následující příkaz převede zdrojové video do formátu Theora a jeho zvukovou stopu do formátu Vorbis:

```
ffmpeg2theora vstupnivideo -o vystupnivideo.ogv
```

Rozlišení a snímková frekvence zůstanou na stejných hodnotách jako u zdrojového videa.

Pokud chceme upravit kvalitu a rozlišení výstupu stačí změnit příslušné příkazy. Na příkladu níže je kvalita videa nastavena na 9, kvalita audia na hodnotu 6, šířka videa na

[20] OZER, By Jan. *Video compression for Flash, Apple devices and HTML5*. Galax, VA: Doceo Publ, 2011. ISBN 09-762-5950-8.
[21] Theora benefits. 2011. *Theora.org* [online]. [cit. 2015-03]. Dostupné z: http://www.theora.org/benefits/

720 px a výška na 576 px. Kvalita videa a audia lze nastavovat v rozmezí 0 – 10, vyšší hodnota znamená lepší kvalitu.

```
ffmpeg2theora zdrojovevideo.mp4 -o vystupnivideo.ogv \  --videoquality 9\
--audioquality 6   --width 720 --height 576 \[22]
```

VP8

Kompresní formát VP8 byl původně vyvinut a uvolněn v roce 2008 společností On2 Technologies. V roce 2010 On2 koupila společnost Google a uvolnila specifikace formátu pod licencí Creative Commons Attribution 3.0 license. Téhož roku Google uvolnil jeho referenční implementaci - knihovnu libvpx pod licencí BSD.

Kódování VP8 videa

Existuje řada kodérů využívajících knihovnu libvpx: Direct show filtry, VP8 SDK, Gstreamer a FFmpeg. VLC a Miro Video Converter jsou nástroje disponující grafickým uživatelským rozhraním.

Příkaz níže využívá softwaru FFmpeg a slouží pouze k převedení zdrojového videa do formátu VP8 v kontejneru WebM a audiem ve formátu Vorbis.

```
ffmpeg -i vstupnivideo vystupnivideo.webm
```

Následující sada příkazů vytvoří video s námi definovanými parametry výšky videa a také datovým tokem videa i audia:

```
ffmpeg -o vystupnivideo.webm -p 720p \
--videobitrate 3000 \
--audiobitrate 192 \
vstupnivideo
```

VP9

Kompresní video formát VP9 je nástupcem formátu VP8, který již byl vyhotoven v režii společnosti Google. Jeho cílem bylo zejména znatelné snížení datového toku při zachování kvality videa. To se mu také podařilo, protože při polovičním datovém toku dosahuje

[22] GERBER, Jan. 2009. *OGG THEORA COOK BOOK* [online]. [cit. 2015-05-03]. Dostupné z: http://en.flossmanuals.net/ogg-theora/encoding/_booki/ogg-theora/ogg-theora.pdf

srovnatelné kvality s formátem H.264. Efektivní komprese nachází využití hlavně u vysokých rozlišení např. 4K videa, kdy objem dat oproti FullHD videu narůstá na čtyřnásobnou hodnotu.[23]

Ačkoliv není formát VP9 oficiálním standardem pro HTML5 video, tak již je podporován řadou prohlížečů. Server YouTube dokonce distribuuje video v tomto formátu a videa o vysokém rozlišení 4K již ukládá pouze v něm.

H.265

Kompresní formát H.265 někdy označovaný také jako MPEG-4/HEVC je konkurentem pro formát VP9. Přínos formátu spočívá zejména ve snížení datového toku na polovinu při stejné kvalitě videa jako nabízí jeho předchůdce H.264. To také vedlo ke znatelné zvýšení kvality výstupu při shodném datovém toku. Oproti h.264 nabízí řadu vylepšení které umožňují citelné snížení datového toku. Ke kompresi i dekompresi je potřeba mnohem výkonnějšího hardwaru než u předchozího standardu. Řad zařízení již disponuje hardwarovým dekodérem.

Stejně jako předchůdce nabízí řadu profilů a levelů. Novinkou je profil Main 10, který umožňuje 10 bitové kódování barevné hloubky. Opensource implementací pro kódování je knihovna x265, na které staví například nástroje FFmpeg, VLC a HandBrake. [24]

Obr. 6 – Porovnání kvality výstupu komprese H.265 a H.264 při stejném datovém toku
Zdroj: X265 HEVC Upgrade [online]. 2013. [cit. 2015-09-03]. Dostupné z: https://x265.com/

[23]VP9 Video Codec. 2013. *WebMProject* [online]. [cit. 2015-05-04]. Dostupné z: http://www.webmproject.org/vp9/
[24] OZER, By Jan. *Video compression for Flash, Apple devices and HTML5*. Galax, VA: Doceo Publ, 2011. ISBN 09-762-5950-8.

Audio kompresní formáty

MP3

MP3 nebo-li MPEG-2 Audio Layer III je formát ztrátové komprese pro zvukové soubory. Za jeho vznikem stojí skupina MPEG. Při zachování poměrně dobré kvality umožňuje zmenšit velikost souborů v CD kvalitě přibližně na desetinu. MP3 může podporovat až dva zvukové kanály s konstantním nebo variabilním datovým tokem. MP3 si patentovala společnost Thomson/RCA. Nejznámějším kodérem pro tento formát je kodek LAME[25].

AAC

AAC nebo-li MPEG-4 Advanced Audio Coding je velmi kvalitní kompresní audio formát, který umožňuje téměř neomezený počet kanálů a rozsah datového toku. Formát je patentován skupinou MPEG. Můžeme se s ním setkat jako RAW streamem s koncovkou .aac nebo v kontejneru MP4 s koncovkami .mp4 a .m4a.[26]

Vorbis

Jedná se o formát, která vyvinula nadace Xiph.org. Nabízí velmi dobrou kvalitu již u nízkých datových toků. Používá formu ztrátové komprese. Není zatěžkán patenty a jeho použití pro ne komerční účely je zdarma.

Ještě bychom mohli zmínit formáty Speech a Opus[27], které byli vyvinuty zejména pro komprimaci mluveného slova. Jejich podpora, ale není standardizována.

3.2 Audio a video kontejnery

Nyní se podíváme jaké jednotlivé video a audio kompresní formáty můžeme kombinovat v jednotlivých kontejnerech. V tabulce Tab. 4 jsou uvedeny kombinace video a audio formátů v jednotlivých kontejnerech standardizovaných skupinou W3C. V tabulce můžeme

[25] *The LAME Project* [online]. 2011. [cit. 2015-07-03]. Dostupné z: http://lame.sourceforge.net
[26] PILGRIM, Mark, Brian ALBERS a Frank SALIM. 2010. *HTML5: up and running*. 1st ed. Sebastopol, CA: O'Reilly, 304 s. ISBN 05-968-0602-7
[27] Opus Interactive Audio Codec. 2014. *Opus* [online]. [cit. 2015-05-04]. Dostupné z: http://www.opus-codec.org

najít přípony kontejnerů a jejich příslušný MIME typ, který jak uvidíme dále, najde využití u elementu source.

Tab. 3 – Přehled video kontejnerů a jejich příslušných audio a video formátů podle standardu skupiny W3C

Kontejner	Video	Audio	Přípona	MIME typ
MP4	H.264	ACC	.mp4	video/mp4
OGG	Theora	Vorbis	.ogg	video/ogg
WebM	VP8	Vorbis	.webm	video/webm

Zdroj: vlastní zpracování podle HTML <video> Tag. W3Schools [online]. [cit. 2015-02-03]. Dostupné z: http://www.w3schools.com/tags/tag_video.asp

Následující tabulka ukazuje podporu výše zmíněných standardizovaných video kontejnerů v jednotlivých populárních prohlížečích. V tabulce je uvedeno od jaké verze prohlížeč formát podporuje.

Tab. 4 – Přehled video kontejnerů a jejich podpory v prohlížečích

Kontejner	Firefox	Chrome	IE	IE Mobile	Opera	Safari	iOS Safari	Android
MP4	22+*	3+	9+	10+	24	3.1+	3.2+	4.4+
Ogg	3.5+	3+	–	–	10.5+	–	–	2.3+
WebM	4+	6+	9+* *	10+* *	10.6+	–	–	2.3+

Zdroj: vlastní zpracování podle dokumentace jednotlivých prohlížečů
**odvíjí se od použité platformy*
*** nutný dodatečný software*

Z tabulky patrné, že jediný video kontejner, který je podporován všemi prohlížeči je MP4. Nemůže však spoléhat na uživatele, že má nainstalován aktuální prohlížeč. Proto bychom měli video poskytnout alespoň ještě v jednom formátu a to nejlépe v kontejneru WebM. Podpora Firefoxu u MP4 formátu není integrována přímo v prohlížeči a to z důvodů patentového zatížení a tak jeho podpora je závislá od podpory operačního systému či hardwaru. Na platformě Windows Vista a novější je podpora zajištěna od Firefox verze 22

a výše, pro Linux to je Firefox 26+ a pro android Firefox 20+. Pro podporu WebM u Internet Explorer musíme naistalovat dodatečný software[28].

Obdobný závěr můžeme vyvodit i pro podporu audio formátů (viz tabulka níže), kdy formát MP3 podporují všechny aktuální verze hlavních prohlížečů. I zde bychom měli pro jistotu zvuk poskytnout alespoň v jednom dalším formátu.

Tab. 5 – Přehled audio formátů a jejich podpora v prohlížečích

Kontejner	Firefox	Chrome	IE	IE Mobile	Opera	Safari	iOS Safari	Android
MP3	21+	3+	9+	10+	14	3.1+	3.2+	2.3+
Vorbis OGG	3.5+	9+	–	–	10.5+	3.1+	3.2+	2.3+
Wav	3.5+	3+	–	–	11+	3.1+	3.2+	2.3+

Zdroj: vlastní zpracování podle dokumentace jednotlivých prohlížečů

Celou situaci zkomplikovali nové formáty. Zatímco H.265 zatím nenachází žádnou podporu v prohlížečích. Formát VP9 již nachází podporu v prohlížečích v kombinaci s audio formátem Opus v kontejneru WebM. Podporu shrnuje tabulka Tab. 6. Samotný Opus se může vyskytovat také v OGG kontejneru. Brzké nasazení podpory H.265 lze očekávat v prohlížečích Safari. Zároveň se od těchto prohlížečů neočekává podpora pro VP9, jelikož to je proprietární formát společnosti Google a Apple nehodlá na svých zařízeních podporovat proprietární formáty. H.265 bude umístěn v MP4 kontejneru spolu s ACC audiem.

Tab. 6 – Podpora prohlížečů pro video ve formátu VP9 s Opus audiem v kontejneru WebM

Kontejner	Firefox	Chrome	IE	IE Mobile	Opera	Safari	iOS Safari	Android
WebM (VP9 + Opus)	28+	29+	–	–	16+	–	–	–

Zdroj: vlastní zpracování podle dokumentace jednotlivých prohlížečů

[28] WebM Video for Microsoft Internet Explorer 9+. 2014. *Tools Google* [online]. [cit. 2015-03-04]. Dostupné z: https://tools.google.com/dlpage/webmmf/

3.3 Element <video>

Element video slouží pro vkládání videí na webovou stránku. Jak již bylo řečeno výše, tak bohužel neexistuje jediný formát podporovaný všemi prohlížeči a tak musíme video vložit v různých formátech a v tom nám vypomáhá element source. Element source obsahuje atribut src, který ukazuje cestu k souboru a atribut type, který specifikuje formát kontejneru zdrojového videa jako MIME typ. Uvedením atributu type, usnadňujeme prohlížeči určení, zda může daný formát přehrát bez zbytečného stahování dat. Vložení videa z různých zdrojů potom vypadá následovně:

```
<video controls width="640" height="360">
      <source src="video.mp4" type="video/mp4">
      <source src="video.webm" type="video/webm">
      <source src="video.ogg" type="video/ogg">
</video>
```

V příkladu jsou uvedeny tři zdroje v různých formátech, z nichž bude přehrán jako první ten, který je podporován prohlížečem.

Tab. 7 – Přehled atributů tagu <video>

Atribut	Hodnota	Popis
autoplay	autoplay	Specifikuje, zda se má přehráváni spustit automaticky ihned jak je k dispozici
controls	controls	Určuje, zda mají být zobrazeny ovládací prvky
loop	loop	Zapne přehrávání ve smyčce
muted	muted	Ztlumí zvuk
preload	auto, none metadata	Specifikuje formu načítání do mezipaměti
src	URL	Cesta k souboru
poster	URL	Náhledový obrázek videa

Zdroj: vlastní

U elementu *video* může být uveden atribut *controls*, který slouží k zobrazení ovládacích prvků. Pokud bychom neuvedli tento atribut, stále můžeme video ovládat skrze kontextové menu. Bohužel vzhled ovládacích prvků se liší napříč prohlížeči i jejich jednotlivými

verzemi. To může činit problém při zakomponování přehrávače do celkového vzhledu stránek. Vzhled ovládacím prvků můžeme nastylovat pomocí CSS a JavaScriptu.

Povinnými atributy jsou atributy *width* a *height*, které definují výšku a šířku videa. Hodnota těchto atributů je nejčastěji v pixelech.

Firefox 28 Internet Explorer 11

Opera 25 a Chrome 40 Safari 7

Obr. 7 – Vzhled ovládacích prvků v jednotlivých prohlížečích
Zdroj: vlastní

Atribut poster slouží k vložení náhledového obrázku. Tento obrázek je na pozadí přehrávače dokud uživatel video nespustí. Měl by hlavně sloužit k identifikaci obsahu videa.

Dalším atributem je *autoplay*, který slouží k automatickému spuštění videa ihned po načtení stránky. Spolu s atributem *loop*, který zajišťuje přehrávání videa stále dokola, najde využití např., když video tvoří pozadí stránky, jinak jej není příliš vhodné používat.

Atribut *preload* umožňuje nastavit načítání videa do mezipaměti. Může nabývat hodnot: *auto, none* a *metadata*.[29]

Při hodnotě auto se po načtení stránky začne načítat do paměti také celý multimediální soubor. Nastavení na hodnotu auto není příliš vhodné, pokud máme na stránce více multimediálních souborů nebo pokud si nejsme jisti, jestli uživatel bude chtít obsah spouštět.

Při nastavení na hodnotu *metadata* se načtou pouze *metadata* což znamená například načtení rozměrů videa, délku přehrávání, seznam skladeb a prvních pár snímku videa které se použijí jako náhled. Hodnota *metadata* je výchozí nastavenou hodnotou.

Při hodnotě *none* se nebudou přednačítat do paměti žádná data. Z této situace je vhodné použít atribut poster, aby se uživateli zobrazil náhled videa, jinak by se zobrazila pouze černá plocha. Použití nastavení na *none* je vhodné zejména u mobilních zařízení, kdy nechceme zatěžovat datový tok.

Dalším elementem úzce souvisejícím s videem je element <track>, který se vkládá před uzavírací tag videa. Element může být použit například pro překlad u videa. Element obsahuje hned několik atributů. Prvním atributem je atribut *kind* specifikující druh popisků, který může nabývat hodnot: *subtitles, captions, descriptions, chapters* a *metadat*.

Typ *subtitles* slouží k vložení stopy titulků, *captions* najde uplatnění jako popisky k dialogům pro neslyšící, *descriptions* slouží jako popisek videa, který je zvukově syntetizován, pokud je video skryto nebo pokud je uživatel slepý. *Chapters* slouží k rozdělení videa do kapitol za účelem jednodušší navigace. Mezi jednotlivými kapitolami je poté umožněna navigace pomocí interaktivního menu. *Metadata* slouží jako popis videa pro skripty, pro uživatele je skryt.

Zdrojový kód níže znázorňuje použití elementu track s dvěma stopami titulků:

```
<video width="320" height="240" controls>
  <source src="film.mp4" type="video/mp4">
  <source src="film.ogg" type="video/ogg">
  <track src="subtitles_en.vtt" kind="subtitles" srclang="en"
label="English">
  <track src="subtitles_cs.vtt" kind="subtitles" srclang="cz"
```

[29] PFEIFFER, Silvia. *The definitive guide to HTML5 video*. Distributed to the book trade worldwide by Springer Science Business Media, xiv, 321 p. ISBN 14-302-3090-8.

```
label="Česky">
</video>
```

V příkladu vidíme atribut *src*, který specifikuje umístění textové stopy, *srclang* určuje jazyk textové stopy a je vyžadován právě pokud kind="subtitles". Poslední atribut label slouží jako popisek textové stopy.

3.4 Element <audio>

S elementem audio se pracuje obdobně jako s elementem video, avšak již nejsou podporovány některé atributy. Přehled atributů pro prvek audio je uveden v tabulce níže.

Tab. 8 – Přehled atributů tagu <audio>

Atribut	Hodnota	Popis
autoplay	autoplay	Specifikuje, zda se má přehrávání spustit automaticky ihned jak je k dispozici
controls	controls	Určuje, zda mají být zobrazeny ovládací prvky
loop	loop	Zapne přehrávání ve smyčce
muted	muted	Ztlumí zvuk
preload	auto, none, metadata	Specifikuje formu načítání do mezipaměti
src	URL	Cesta k souboru

Zdroj: vlastní

Stejně jako u videa je vhodné uvést soubor ve více formátech a to z výše zmíněných důvodů ohledně kompatibility. Pokud neuvedeme atribut *controls*, tak se nezobrazí ovládací prvky, ale není možno ani ovládání skrze kontextové menu. Zvuk je spuštěn na pozadí.

3.5 Distribuce

Progresivní stahování

Progresivní stahování je nejjednodušším a zároveň nejlevnějším způsobem pro distribuci digitálních multimédií. Pro distribuci je použit běžný HTTP server, nejčastěji Apache. Nezbytná je podpora protokolu HTTP ve verzi 1.1.

K plynulému přehrávání je nutné, aby rychlost internetového připojení uživatele byla větší než je datový tok videa, jinak uživatel musí čekat na načtení videa.

Jelikož je při progresivním stahování video či audio souboru stahován na fyzický nosič zařízení je velmi jednoduché ho následně kopírovat. Kvůli tomuto nedostatku není tento způsob distribuce vhodný pro distribuování autorsky chráněného obsahu.

Progresivní stahování v rámci HTML5 umožňuje dokonce i přejít na určité místo ve videu aniž by tato část byla již stažena. To je umožněno díky implementaci HTTP 1.1 Byte Range požadavků, které umožňují klientům si vyžádat pouze část souboru.[30]

Streaming za použití RTP/RTSP

Streamování je proces, kdy datový tok není ukládán paměťový nosič klientského zařízení, ale je rovnou přehráván. Streamované soubory jsou buď umístěny na serveru jako tzv. streamy, při přehrávání se pouze stahují k uživateli a přehrávají (On demand), nebo jsou generovány a v reálném čase přenášeny (online, např. živé televizní a rádiové vysílání).

Ke streamování pomocí RTP/RTSP (Real Time Streaming Protocol) musí být použit speciální server. Tímto serverem může být např. VideoLAN, QuickTime Streaming Server či Helix od společnosti RealMedia.[31]

Tento protokol byl vyvinut zejména pro použití u video konferencí a telefonních hovorů (voice-over-IP). Pro streaming obsahu na vyžádání se příliš nehodí, protože stejného výsledku můžeme dosáhnout pomocí výše zmíněného progresivního stahování a za použití běžného webového serveru. Pro živé vysílání je naopak vhodnější níže zmíněná

[30] PFEIFFER, Silvia. *The definitive guide to HTML5 video*. Distributed to the book trade worldwide by Springer Science Business Media, xiv, 321 p. ISBN 14-302-3090-8.

[31] OZER, By Jan. *Producing streaming video for multiple screen delivery*. Galax, Va: Doceo Publishing, 2013. ISBN 09-762-5954-0

technologie HTTP Streaming. Z těchto důvodů nenachází podporu u výrobců prohlížečů. Jeho implementaci v prohlížečích je tedy nepravděpodobná.

Proprietární protokol RTMP od společnosti Adobe je velmi podobný protokolu RTSP, s nímž sdílí i jeho nedostatky. Streaming médií zajišťuje Flash Media Server. Výhodou tohoto řešení je šifrování vysílaných streamů, které znesnadňuje kopírování přenášených audiovizuálních dat.

HTTP Streaming

Protože streaming pomocí RTMP vyžaduje speciální server, což přináší nemalé další náklady. Bylo cílem v setrvání použití běžných webových serverů bez dalších softwarových rozšíření a přenesení problematiky streamování na stranu klientského softwaru.

Hlavním přínosem je vlastnost adaptive HTTP streaming, která umožňuje změnu kvality videa během přehrávání, tzn. jeho datového toku. Toto řešení umožňuje nepřetržité přehrávání při měnících se podmínkách v datové propustnosti připojení a také v dostupnosti aktuálních systémových prostředků zařízení. V praxi to znamená, že například s poklesem datové šířky nebo při vytíženém hardware je uživateli distribuován stream s datovým tokem, který je schopen plynule přehrát. Naopak s větší datovou šířkou a s dostatečným výkonem hardwaru je uživateli distribuován stream s vyšším datovým tokem a tedy i výstupní kvalitou. Tato změna probíhá zcela plynule, uživatel pouze zaznamená změnu kvality výstupu.[32]

Dostupné technologie na trhu pro adaptivní streamování jsou:

- Apple HTTP Live Streaming
- Microsoft Smooth Streaming
- Adobe HTTP Dynamic Streaming
- MPEG-DASH

Výše zmíněné technologie pracují na společném principu: několik verzí multimediálního zdroje je kódováno s různými datovými toky a v různých rozlišeních a v segmentech jsou

[32] OZER, By Jan. *Producing streaming video for multiple screen delivery*. Galax, Va: Doceo Publishing, 2013. ISBN 09-762-5954-0.

poté umístěny na webovém serveru. Klientský software je informován o těchto segmentech skrz soubor, ve kterém jsou uvedeny informace o jejich umístění, rozlišení a datovém toku. Přehrávač médií klienta neustále analyzuje stav přehrávání a na základě dostupné šířky pásma připojení a výkonu CPU, pak vybírá nejvhodnější segmenty k přehrání.

Apple zabudovalo podporu pro HTTP Live Streaming do svých zařízení iPad, iPhone a iPod. Dále jeho podporu nalezneme v desktopovém přehrávači QuickTime a ve formě HTML5 videa v prohlížeči Safari. Bohužel Safari je jediným prohlížečem s podporou HLS u ostatních musí být jeho podpora implementována využitím Flash přehrávače. Podpora protokolu se objevila také v operačním systému Android od verze 3. Microsoft ve svém Silverightu podporuje, jak Microsoft Smooth Streaming, tak Apple HLS. Adobe HTTP Dynamic Streaming se na trhu objevil spolu s uvedením Flash Playeru 10.1.

Posledním řešením adaptivního streamování prostřednictvím HTTP protokolu je mezinárodní standard Dynamic Adaptive Streaming over HTTP nebo-li DASH, za kterým stojí skupina MPEG. DASH je podporován v HTML5 prostřednictvím Media Source Extensions (MSE) a podporu pro DRM zajištují Encrypted Media Extensions. MSE se v současné době nachází ve stádiu Candidate Recomendation. V současné době už využívají tohoto API služby YouTube a Netflix. Také byla vytvořena opensource JavaScriptová knihovna DASH.js, která usnadňuje tvorbu HTML5 přehrávačů s podporu DASH.[33]

Tabulka níže znázorňuje podporu MSE a formátů, u kterých je možné jej využít. Dále je zde uvedena podpora pro EME. Z tabulky je patrná nevelká podpora v prohlížečích a to znamená nutnost použití Flashe pro zpětnou kompatibilitu. V praxi to znamená, že pro živé vysílání či pro vysílání chráněného obsahu se nemůžeme spolehnout pouze na HTML5, ale je nezbytné využití dalších technologií pro zpětnou kompatibilitu se všemi možnými zařízeními a prohlížeči.

[33] SIMPSON, Wes. 2013. Adaptive Streaming Adoption. *Video edge* [online]. (3) [cit. 2015-03-04]. ISSN 2372-9244. Dostupné z: http://search.proquest.com/docview/1637940701/fulltext/17D080027EBB4355PQ/1?accountid=17116

Tab. 9 – Přehled podpory Media Source Extensions a Encrypted Media Extensions v prohlížečích

Prohlížeč	MSE	Podporované kontejnery	EME
IE	11+*	MP4	11+*
Firefox	–	–	–
Chrome	34+	MP4, WebM	–
Opera	15+	WebM	–
Safari	8+	MP4, TS	8+
iOS	–	–	–
Android	–	–	–
IE Mobile	8.1+	MP4	8.1+

Zdroj: vlastní zpracování podle dokumentace jednotlivých prohlížečů
**pouze Windows 8 a verze novější*

Media Presentation Description (MPD) je XML soubor, který tvoří přehled jednotlivých segmentů a jejich parametrů potřebných k přehrání. Tyto soubory je možné vytvořit např. pomocí nástroje MP4Box, který se ovládá pomocí příkazové řádky.

Nyní se podíváme jak takový MPD soubor vytvoříme. Nejprve si musíme upravit zdrojové video, tak aby každý segment začínal klíčovým snímkem. To znamená, pokud chceme mít např. segmenty dlouhé 4 s a snímková frekvence videa bude 24 s, musíme parametr keyint nastavit na hodnotu 96, tzn. každý 96. snímek bude snímek klíčový. Na stejnou hodnotu nastavíme také parametr *min-keyint*, abychom docílili konstantního intervalu mezi klíčovými snímky. Nakonec parametrem *no-scenecut* vypneme automatické rozhodování o umisťování klíčových snímků. Kompletní sadu příkazů ukazuje příklad níže:

```
x264 --output vystupnivideo.mp4 --fps 24 --preset slow --bitrate 2400 --
vbv-maxrate 4800 --vbv-bufsize 9600 --min-keyint 96 --keyint 96 --
scenecut 0 --no-scenecut --pass 1 --video-filter
"resize:width=1280,height=720" vstupnivideo.mkv
```

Dalším krokem je již samotné segmentování pomocí MP4Boxu.

```
MP4Box -dash 4000 -frag 4000 -rap -segment-name segment_zdrojovevideo.mp4
```

Příkaz *dash* určuje délku segmentů, v našem konkrétním případě jsou to 4 s tedy 4000 ms. Příkaz *frag* slouží k vytvoření sub segmentů v rámci segmentů, protože nechceme vytvářet sub segmenty, tak hodnota zůstává stejná jako u příkazu *dash*. Příkaz rap vynutí vytváření segmentů v bodech, kde se nacházejí klíčové snímky. Příkaz *segment-name* slouží k pojmenování jednotlivých segmentů. V příkladu je uvedena hodnota segment_, jednotlivé segmenty budou potom pojmenovány takto: segment_1.m4s, segment2.m4s atd. Výstupem toho příkazu bude jedno segmentované video, inicializační segment a MPD soubor.

Stejný postup budeme opakovat pro video s jiným datovým tokem a rozlišením. Výstup z MPD souborů překopírujeme do původního MPD souboru. Obdobný postup platí i pro audio stopu. Pokud nemáme audio jako samostatný soubor, můžeme si pomoci audio selektorem:

```
MP4Box -dash 4000 -frag 4000 -rap -segment-name segment_ video.mp4#audio
```

Posledním krokem je vložení všech součástí na webový server a předaní odkazu na MPD soubor přehrávači s podporou DASH.[34]

[34] MPEG-DASH CONTENT GENERATION USING MP4BOX AND X264. 2014. *Bitdash* [online]. [cit. 2015-04]. Dostupné z: https://www.dash-player.com/blog/2014/11/mpeg-dash-content-generation-using-mp4box-and-x264/

4 CSS3

CSS neboli Cascading Style Sheet je jazyk používaný ke stylování HTML prezentací. Aktuální verze CSS3 je třetí specifikací tohoto jazyka. Výhodou CSS je separace vizuální a obsahové části webové stránky. Toto oddělení umožňuje sdílení formátování mezi více stránkami a tím redukci stejného kódu na každé stránce. Další výhodou je možnost vytváření různých stylů pro různá zařízení. Z historického pohledu prošly kaskádové těmito důležitými milníky:

- 1996 – uvolněna specifikace CSS1
- 1998 – uvolněna specifikace CSS2
- 1999 – započaty práce na CSS3
- 2011 – uvolněna specifikace CSS2.1
- 2006 – až do současnosti probíhá dokončování CSS3[35]

CSS3 k funkcionalitě z předchozích verzí přidává nové vlastnosti. Tyto nové vlastnosti jsou rozděleny do takzvaných modulů a jsou přidávány postupně v různých stádiích vývoje od pracovního návrhu (Working Draft) po doporučení (Recommendation). Protože popis všech nových prvků by byl mimo rozsah této práce, tak se v této kapitole budeme zabývat zejména vlastnostmi, které mohou najít využití u multimediálních elementů, zejména při vytváření vlastních ovládacích prvků.

4.1 Nové prvky CSS3

Rámečky

Prakticky každý HTML prvek má svůj rámeček, který je bez použití CSS skrytý. Pomocí CSS vlastností můžeme volit jeho velikost, barvu, tloušťku a styl čáry. CSS3 nově přináší možnost tvorby zakulacených rohů, stínů a styl rámů tvořený pomocí obrázků.

[35] Vývoj CSS. KULHÁNEK, Tomáš, Lukáš LEDVINKA a Jakub VOŘÍŠEK. *Samizdatová skripta* [online]. 2009 [cit. 2014-02-24]. Dostupné z: http://skripta.lmsoft.cz/index.php?id=44

Border-radius

Slouží k zaoblení rohů. Jednoduchým zápisem můžeme zaoblit rohy vybraného boxu o 10 px: `box {border-radius: 10px}`

Další možností zápisu je zápis, kde udáváme poloměr zaoblení pro každý roh zvlášť a to v pořadí levý horní roh, pravý horní, pravý dolní a levý dolní roh:

```
box {border-radius: 10px 15px 20px 30px}
```

Pro jednotlivé rohy můžeme zadávat dokonce dva poloměry zakřivení:

```
box {
border-top-left-radius: 10px 15px;
border-top-right-radius: 20px 30px;
border-bottom-right-radius: 60px 30px;
border-bottom-left-radius: 30px 60px}
```

Box-shadow

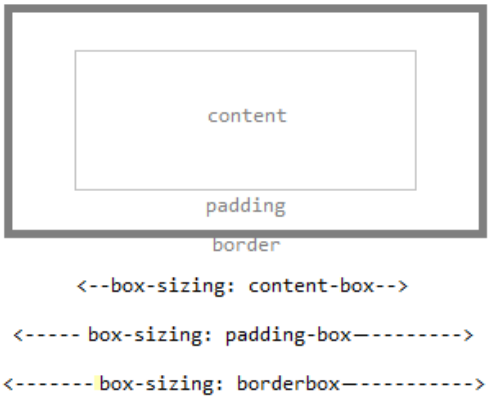
Tato vlastnost slouží k vytvoření jednoho či dokonce více barevných stínů za rámečkem. Je dán následujícími parametry: `box-shadow: inset x y radius tloušťka barva`. *Inset* je nepovinný parametr, který slouží pro vytvoření vnitřního stínu. Hodnoty *x* a *y* udávají horizontální a vertikální délku stínu. Rádius udává poloměr rozmazání stínu a parametr *color* samozřejmě barvu.

Border-image

Border-image umožňuje boxu přidat ohraničení, které je tvořeno obrázkem. Pro stejný efekt bylo dříve nutno obalit element elementy div. Vlastnost má následující zápis: `border-image: source slice width outset repeat`. Parametr *source* určuje cestu ke zdrojovému obrázku. Parametr *slice* pomyslně umístí čtyři přímky podle, kterých se rozřeže obrázek. Parametr *width* určuje tloušťku rámečku a *outse*t určuje posunutí rámečku oproti základní pozici ven z boxu. Parametr *repeated* můžeme nastavit na hodnotu *stretch* pro roztáhnutí obrázku na šířku boxu nebo na hodnotu *repea*t pro opakování obrázku.

Box model

Tradiční box model počítá šířku či výšku elementu jako viditelnou šířku či výšku obsahu. CSS3 přidává vlastnost *box-sizing*, která umožňuje změnit způsob výpočtu velikosti elementu. Může nabývat hodnot *content-box, border-box* a *padding-box*. *Content-box* je již uvedený tradiční způsob výpočtu. *Padding-box* počítá do velikosti elementu i hodnoty pro *padding*. Poslední možnost *border-box* počítá do velikosti elementu i hodnoty pro *padding* i *border*. Uvedené skutečnosti ilustruje obrázek níže.



Obr. 8 – Způsoby výpočtu velikosti elementu
Zdroj: vlastní

Příklad použití zakulacených rohů a stínů je na obrázcích níže na elementech videa. Levý obrázek má nastavenou navíc vlastnost *box-sizing* na hodnotu *border-box* a zobrazený rámeček.

Obr. 9 Ukázka použití zakulacených rohů a stínů u elementu video
Zdroj: vlastní

Průhlednost a gradienty

Průhlednost

Tuto vlastnost bylo možno využít již dříve, ale až s příchodem CSS3 byla oficiálně standardizována. Průhlednost je definována jako hodnota mezi 0.0 a 1.0, hodnota 0.0 znamená úplnou průhlednost a hodnota 1.0 neprůhlednost. Zápis může být např. následující: `opacity: 0.7.`

Gradienty

Gradienty slouží k zobrazování přechodů mezi několika barvami. Většinou slouží pro stylování pozadí stránky nebo jednotlivých boxů. V minulosti bylo jejich alternativou použití obrázků, které jsou datově náročnější.

Linear gradient

Lineární gradienty jsou rovné přechody a jsou určeny osou přechodu. Sklon osy přechodu je určen buď klíčovými slovy *to bottom left*, *to right* nebo úhly sklonu např. *30 deg*. Úhel 0 znamená směr ze zdola nahoru, úhel 180 shora dolů a úhel 90 zleva doprava. Pomocí zarážek barev můžeme určit místo přechodu. Zarážky můžeme zadávat v procentech, pixelech nebo relativních jednotkách.

Můžeme také tvořit opakující se přechody s následující syntaxí:

```
repeating-linear-gradient (red, orange 20px, red 40px)
```

Obr. 10 – Ukázka použití lineárních gradientů pro stylovaní progress baru
Zdroj: The HTML5 progress Element. CSS-Tricks [online]. [cit. 2015-07-04]. Dostupné z: https://css-tricks.com/html5-progress-element/

Na obrázku výše je znázorněno možné použití gradientů u elementu *progress*. K výslednému efektu byly využity tři lineární gradienty, viz kód:

```
background-image:
linear-gradient(-45deg, transparent 33%, rgba(0, 0, 0, .1) 33%, rgba(0,0,
0, .1) 66%, transparent 66%),
```

```
linear-gradient(top, rgba(255, 255, 255, .25), rgba(0, 0, 0, .25)),
linear-gradient(left, #09c, #f44);
border-radius: 2px;
background-size: 35px 20px, 100% 100%, 100% 100%;
```
[36]

Radial gradient

Radial gradient je přechod ve tvaru kruhu nebo elipsy. Zápis kruhového gradientu může vypadat následovně: `background: radial-gradient(yellow, orange 80%).`

Zápis pro přechod ve tvaru elipsy: `background: radial-gradient(ellipse 70px 40px, red, orange 20px, red 40px)`

Obdobně jako u lineárního gradientu můžeme vytvořit také opakující se přechod: `repeating-radial-gradient (red, orange 20px, red 40px)`

Transitions, transform a animace

Transitions

Běžně pokud měníme CSS vlastnost, tak je tato změna aplikována okamžitě např. změna barvy pozadí tlačítka po najetí myší. Pomocí přechoců můžeme tyto změny stavů plynule animovat. Syntaxe je následující: `transition: property duration timing-function`

Property určuje, na jaké vlastnosti se mají přechody aplikovat. Hodnota může být *all*, tedy na všechny vlastnosti nebo můžeme použít výčet jednotlivých vlastností.

Vlastnost *duration* určuje dobu trvání přechodu v sekundách nebo v milisekundách. *Delay* specifikuje zpoždění, s jakým se má přechod spustit. Poslední vlastnost *timing-function* udává styl průběhu přechodu. Může nabývat těchto hodnot: *linear, ease, ease-in, ease-out, ease-in-out*.[37]

Transform

Pomocí vlastnosti *transform* můžeme manipulovat s elementem ve 2D i 3D prostoru. Ve 2D jsou k dispozici následující funkce: *translate, rotate, scale, skew* a *matrix*.

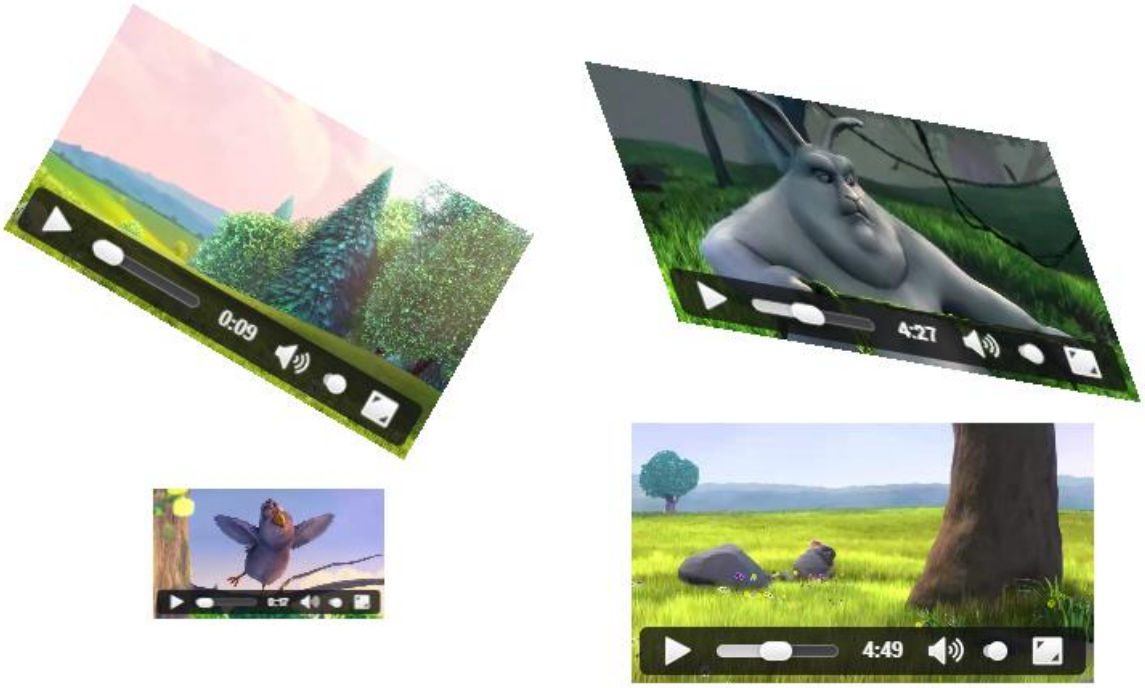
---

[36] The HTML5 progress Element. *CSS-Tricks* [online]. [cit. 2015-07-04]. Dostupné z: https://css-tricks.com/html5-progress-element/

[37] MAKZAN a Jakub ZEMÁNEK. *Programujeme hry v HTML5*. 1. vyd. Brno: Computer Press, 2012. ISBN 978-80-251-3731-4.

Pomocí funkce *translate* můžeme elementy přesouvat na nové pozice, které jsou určeny pomocí nových souřadnic *x* a *y*. Zadávají se v pixelech nebo procentech: `transform:translate(50px,100px)`. *Rotate* otáčí elementem o zadaný počet stupňů: `transform:rotate(35deg)`. Velikost elementu můžeme měnit pomocí funkce *scale*, kde zadané hodnoty udávají násobek původní šířky a výšky: `transform:scale(2,3)`.[38]

Pro manipulaci ve 3D prostoru slouží obdobné funkce: *translate3d*, *rotate3d* a *scale3d*. *Translate3d* obsahuje třetí parametr pro posun elementu i ve směru osy *z*: `transform: translate3d(20px,20px,50px)`. *Rotate3d* může s elementem otáčet podle všech tří os: `transform: rotateX(1deg) rotateY(85deg) rotateZ(20deg)`. Funkce *scale3d* umožňuje měnit i hloubku elementu: `transform:scale3d(2,3,5)`



Obr. 11 Ukázka použití transformačních funkcí na video elementu (rotate, skew, scale a transform) Zdroj: vlastní

Transformace doprovázejí další vlastnosti. *Transform-origin* slouží k umístění transformovaného elementu od výchozího bodu. U 3D efektů máme navíc vlastnost

[38] CSS3 Transforms. *W3Schools.com* [online]. [cit. 2015-03-15]. Dostupné z: http://www.w3schools.com/css/css3_2dtransforms.asp

perspective, která specifikujeme, z jaké perspektivy na element nahlížíme. Vlastnost *backface-visibilit*y slouží ke schování elementů u 3D transformace.

Animace

Animace jsou obdobou přechodů s tím rozdílem, že nejsou dány pouze počátečním a koncovým stavem, ale je zde přidána navíc řada mezistavů. Tyto mezistavy označujeme jako klíčové snímky a jsou definovány ve speciálním *@keyframes* selektoru. Tento selektor definuje různé stavy animace od 0 % do 100 %.

Deklarace animace vypadá následovně: `animation: name duration timing-function delay iteration-count direction play-state`

Položka *name* vytváří spojení s námi definovaným *@keyframes* selektorem, který chceme na element aplikovat. *Duration* udává dobu trvání animace v sekundách či milisekundách. *Timing-function* udává průběh mezi snímky a nabývá stejných hodnot jako *timing-function* u přechodů. *Delay* slouží ke zpožděnému spuštění animace a je udáváno v sekundách nebo milisekundách. *Iteration-count* udává počet opakování animace. Může být nastaveno na hodnotu *infinite*, která zajistí nekonečné přehrávání animace. Pomocí *direction* můžeme měnit směr animace. Hodnota *reverse* způsobí přehrávání od konce animace. Hodnota *alternate* zajistí přehrávání každé sudé animace od konce. Výchozí hodnota je *normal*. Poslední vlastnost *play-state* umožnuje animaci pozastavit pomocí hodnoty *pause* nebo opět spustit hodnotou *running*. Výchozí je hodnota *running*.[39]

Pomocí CSS animací můžeme vytvářet jednoduché animace a vyvarovat se tak použití GIF obrázků, Flash animací a také použití JavaScriptu. Tedy především k tvorbě bannerů, navigačních prvků a načítacích obrazovek.

Webfonts

Dříve byli možnosti vývojářů ve výběru fontů omezené, protože byli limitováni na použití fontů, které mají uživatelé naistalovány v systému. Pomocí pravidla *@font-face* můžeme přidávat fonty vlastní. Tyto fonty bývají na serveru uloženy v několika formátech, aby byla zajištěna podpora ve všech prohlížečích. Mezi nejpoužívanější formáty patří: SVG, EOT

[39] GOLDSTEINOVÁ, Alexis, Louis LAZARIS a Estelle WEYLOVÁ. *HTML5 a CSS3 pro webové designéry*. 1. vydání. Jan Pokorný. Brno: Zoner Press, 2011, 286 s. ISBN 978-80-7413-166-0

(Embedded OpenType), WOFF (Web Open Font Format) a TTF (TrueType). Pomocí zápisu níže je možné stáhnout fonty v jednotlivých formátech do prohlížeče.

```
@font-face {
    font-family: "Flaticon";
    src: url("fonty/flaticon.eot") format("embedded-opentype"),
    url("icons/flaticon.woff") format("woff"),
    url("icons/flaticon.ttf") format("truetype"),
    url("icons/flaticon.svg") format("svg");
}
```

Fonty pak používáme běžným způsobem s využitím vlastnosti *font-family*.

Mediaqueries

Uživatelé nepřistupují na stránky pouze z osobních počítačů a notebooků, ale také z různých mobilních zařízení, jako jsou mobilní telefony, tablety a elektronické čtečky. Proto vznikly *mediaqueries*, které slouží ke zjištění zobrazovacích možností cílového zařízení pomocí CSS. Díky tomu může být aplikováno stylování vhodné pro určité zařízení.

Dotazy můžeme zapisovat buď do HTML kódu a každému zvolenému typu přiřadit vlastní CSS soubor: `<link rel="stylesheet" media="(max-width: 850px)" href="styl1.css">`

Další možností je vkládat dotaz do CSS, kdy se zadané vlastnosti aplikují pouze pří splnění určitých podmínek: `@media (max-width: 600px) { /*kód, který se aplikuje, pouze pokud se splní podmínky*/}`

V příkladu vidíme u vlastnosti uvedený prefix *max*, který stanovuje maximální hodnotu. Prefix pro minimální hodnotu je *min*. Operátor and se používá pro řetězení podmínek a operátor not pro jejich negaci. Operátor *only* zabraňuje starším prohlížečům bez podpory *mediaqueries* použití stylování s vlastnostmi *mediaqueries*.[40] Jednotlivé vlastnosti jsou uvedeny v tabulce níže.

[40] CSS media queries. MOZILLA DEVELOPER NETWORK [online]. 2015 [cit. 2015-03-05]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Media_queries

Tab. 10 – Přehled media queries vlastností a jejich funkcí

Vlastnost	Funkce
width	šířka dostupná k zobrazení webové stránky
height	výška dostupná k zobrazení webové stránky
aspect-ratio	poměr mezi výškou a šířkou
orientation	orientace obrazovky na výšku nebo na šířku
resolution	hustota pixelů výstupního zařízení
scan	způsob vykreslování: progressive nebo interlaced
color	počet bitů, které definují barvy, 0 když nedisponuje barvami
color-index	určuje kolik barev je schopno zřízení zobrazit
monochrome	zda je zařízení schopno zobrazit pouze odstíny šedi a kolika bity na pixel
hover	zda vstupní zařízení umožňuje vznášení se nad prvky (myš)
scripting	zda je dostupné skriptování (př. JavaScript)

Zdroj: vlastní zpracování podle CSS media queries. MOZILLA DEVELOPER NETWORK [online]. 2015 [cit. 2015-03-05]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Media_queries

5 Situace na trhu

V této kapitole se nejprve podíváme na statistiku rozdělení přístupů na webové stránky z klasických PC a mobilních zařízení. Následně se již budeme věnovat tomu, z jakých mobilních zařízení uživatelé konzumují webový obsah a jaké mají dostupné technologie. Pro vývojáře je důležité mít přehled o tom, jaká zařízení mají uživatelé k dispozici, aby na ně mohli optimalizovat svoje webové stránky. Budou nás hlavně zajímat především tyto parametry: rozlišení displeje, výpočetní výkon a v neposlední řadě dostupné rychlosti datového připojení.

Statistika přístupů na webové stránky

Důkazem o vzestupu konzumace webového obsahu z mobilních zařízení jsou data naměřená v rámci projektu NetMonitor. Cílem tohoto projektu je poskytnout informace o návštěvnosti webových stránek a sociodemografickém profilu uživatelů internetu v České republice. Projekt sleduje řadu provozovatelů webových stránek, včetně těch nejvýznamnějších v ČR: Seznam.cz (seznam.cz, novinky.cz), Mediální skupina Mafra (idnes.cz, lidovky.cz) a Economia (centrum.cz, aktualne.cz). V tabulce níže jsou shrnuty naměřené výsledky za měsíc únor v letech 2013 až 2015.

Tab. 11 – přístupy na webové stránky podle zařízení

Ukazatel	Únor 2013	Únor 2014	Únor 2015
Reální uživatelé (RU)	6 357 896	6 557 548	6 850 546
Shlédnuté stránky (PV)	8 001 889 341	7 765 911 913	7 767 294 274
RU mobilní zařízení	1 451 301	2 655 293	4 062 113
PV mobilní zařízení	452 381 737	865 282 657	1 257 881 319
RU streaming	2 986 652	3 399 241	3 685 528

Zdroj: Vlastní zpracování podle dat poskytovaných společností NetMonitor [online]. 2015. [cit. 2015-03-04]. Dostupné z: http://online.netmonitor.cz/

** Reální uživatelé z ČR - počet uživatelů internetu z České republiky, kteří navštívili alespoň jeden z webů zapojených do výzkumu NetMonitor během daného měsíce.*

Zatímco meziroční nárůst reálných uživatelů internetu se pohybuje v jednotkách procent. Meziroční růst RU přistupujících na internet z mobilních zařízení vzrostl v roce 2014 o více než 80 %, respektive o více než 50 % v roce 2015. V únoru 2015 internetové stránky z mobilních zařízení již navštěvuje téměř 60 % uživatelů. Poslední statistický údaj se týká

počtu uživatelů sledujících streamovaný obsah. V únoru 2015 sledovalo streamovaný obsah 3 685 528 reálných uživatelů.

Používaná rozlišení

Než se pustíme do samotných zařízení, tak si ukážeme, jaká jsou běžně používaná rozlišení u telefonů, tabletů, notebooků a stolních monitorů. V tabulce níže jsou uvedena rozlišení obrazovek notebooků a desktopových PC. Podle globální statistiky je nejpoužívanějším rozlišením HD rozlišení, následováno FHD rozlišením. Uvedená statistika odpovídá rovněž přibližně situaci v ČR. V tabulce je uvedeno také rozlišení UHD, které je morálním nástupcem rozlišení FHD. Nesprávně je někdy označováno jako 4K, které má více bodů na šířku a používá se zejména u kamer a také jej můžeme nalézt u televizí.

Tab. 12 – Procentuální zastoupení jednotlivých rozlišení obrazovek u notebooků a stolních počítačů

Označení	Poměr stran	Šířka (px)	Výška (px)	Zastoupení (%)
HD	16:9	1366	768	31
FHD	16:9	1920	1080	16
SXGA	5:4	1280	1024	8
WXGA	16:10	1280	800	7
WXGA+	16:10	1440	900	7
XGA	4:3	1024	768	6
HD+	16:9	1600	900	6
WSXGA+	16:10	1680	1050	5
WUXGA	16:10	1920	1200	3
WQHD	16:9	2560	1600	1
UHD (4K)	16:9	3840 (4096)	2160	-

Zdroj: vlastní zpracování s použitím dat z Screen Resolution Statistics. 2014. W3Schools.com *[online]. [cit. 2015-05-04]. Dostupné z: http://www.w3schools.com/browsers/browsers_display.asp*

Tabulka sleduje situaci zastoupení jednotlivých rozlišení u mobilních zařízení, tedy tabletů a smartphonů. Nejpoužívanější rozlišení je standard 720p. Podíl zařízení s tímto rozlišením se bude zvětšovat, protože se postupně dostává do nižších kategorií telefonů.

Tab. 13 – Procentuální zastoupení jednotlivých rozlišení obrazovek u mobilních zařízení

Standart	Poměr stran	Výška (px)	Šířka (px)	Zastoupení (%)
720p	16:9	1280	720	15,5
WVGA	5:3	800	480	14,6
FHD	16:9	1920	1080	10,6
iPhone 5	16:9	1136	640	7
WXGA	8:5	1280	800	6,4
WSWGA	16:9	1024	600	5,9
XGA (iPad)	4:3	1024	768	5,3
qHD	16:9	960	540	5,2
FWVGA	16:9	854	480	5,1
iPhone 4S	3:2	960	640	4,2
HVGA	3:2	480	320	4,1

Zdroj: vlastní zpracování s použitím dat z MOBILE HARDWARE STATS. 2015. Unity *[online]. [cit. 2015-03-04]. Dostupné z: http://stats.unity3d.com/mobile/display.html*

U mobilních telefonů a tabletů by nás rovněž měla zajímat uhlopříčka displeje. Nejčastější je u telefonů úhlopříčka v rozmezí 4 – 5 palců. U tabletů je to potom hodnota 10 palců.

Aby si uživatelé mohli přehrávat multimediální obsah našich stránek plynule, musí být vybaveni dostatečně rychlým připojením a naopak my bychom měli poskytnout tyto multimédia v takové formě, aby odpovídalo možnostem uživatelů. Z těchto důvodů je důležité disponovat informacemi o tom, jaké připojení mají uživatelé k dispozici. Jediným vhodným mobilním datovým připojením pro plnohodnotné sledování video obsahu je 4G LTE, které v České republice již disponuje vysokým pokrytím. Nicméně vzhledem k měsíčním datovým omezením operátorů si uživatelé sledování videí ve vyšší kvalitě příliš neužijí. Z tohoto důvodu budou nejčastěji uživatelé konzumovat video obsah, budou-li připojeni k síti Wi-Fi, která nabízí zpravidla stabilní připojení s vysokým datovým tokem.

Dále se již podíváme na samotná mobilní zařízení, která jsou aktuálně na trhu. Zařízení budou rozděleny do kategorií tabletů a smartphonů. Kategorie telefonů je rozdělená do tří kategorií a v každé třídě jsou uvedeni tři zástupci. V kategorii tabletů jsou uvedeni pouze tři zástupci, jelikož používají obdobný HW jako telefony. Účelem tohoto srovnání je

abychom měli přehled pro jak výkonná zařízení a s jakými zobrazovacími schopnostmi webové stránky vyvíjíme.

Kategorie smartphonů

High-End

Tato kategorie představuje ty nejlepší dostupné produkty na trhu. Tyto produkty nabízejí ty nejnovější dostupné technologie a utvářejí image značky. I přes vysokou cenu zařízení jsou prodeje vysoké. Příkladem může být prodejnost poslední novinky od Apple, modelu iPhone 6, kterého se prodalo za první tři dny od uvedení na trh více než 10 miliónů.[41]

Tab. 14 – Vybrané telefony nejvyšší třídy

	 **iPhone 6 (iPhone 6 plus)**	 **Samsung Galaxy S6**	 **HTC One M9**
Úhlopříčka	4.7" (5.5")	5.1"	5"
Rozlišení	1334 x 750 (1920 x 1080)	2560 x 1440	1920 x 1080
Hustota bodů	326 ppi (401ppi)	577 ppi	441 ppi
Procesor	2-jádrový, 64-bit, 1.4 Ghz	8-jádrový (2.5GHz quad + 2.1GHz quad)	8-jádrový (2.0GHz quad Cortex A-57 + 1.5GHz quad Cortex A-53)
RAM	2 GB	3 GB	3 GB
Video	H.264: až 1080p, 60 FPS, High Profile level 4.2 s audiem AAC-LC až 160 Kbps, 48 KHz, stereo MPEG-4: až 2.5Mbps, 640 x 480, s audiem AAC-LC až 160 Kbps, 48 KHz, stereo	HW kodek: H.264, H.265, VP9 (až 1440p)	HW kodování/dekodování: H.264, H.265 (4K až 30fps, 1080p až 120fps)
Data	LTE 150 Mbps, 802.11ac	LTE 300 Mbps, 802.11ac	LTE 450 Mbps, 802.11ac
Operační systém	iOS 8	Andorid 5.0	Android 5.0

Zdroj: vlastní zpracování podle údajů výrobců zařízení

[41] First Weekend iPhone Sales Top 10 Million, Set New Record. *Apple* [online]. [cit. 2015-04-17]. Dostupné z: http://www.apple.com/pr/library/2014/09/22First-Weekend-iPhone-Sales-Top-10-Million-Set-New-Record.html

Mainstream

Smartphony spadající do střední kategorie patří dlouhodobě mezi neprodávanější. Střední kategorie telefonů vyniká nepřekonatelným poměrem cena/výkon což znamená, že zákazník dostává za svoje peníze to nejlepší. Do střední kategorie se často přesouvají modely nejvyšších řad, které již mají svého nástupce.

Tab. 15 – Vybrané telefony střední třídy

	 **Motorola moto G**	 **Nokia Lumia 730**	 **LG G3s**
Úhlopříčka	5.0"	4.7"	5"
Rozlišení	1280 x 720	1280 x 720	1280 x 720
Hustota bodů	294 ppi	316 ppi	294 ppi
Procesor	4-jádrový 1.2 GHz Cortex-A7	4-jádrový 1.2 GHz Cortex-A7	4-jádrový 1.2 Ghz
RAM	1 GB	1 GB	1GB
Video	H.264: až 1080p (HW kodek) H.265: přehrávání až do 1080p30	H.264/AVC, MPEG-4, VC-1	H.264: až 1080p (HW kodek) H.265: přehrávání až do 1080p30
Data	LTE 150 Mbps, 802.11bgn	HSDPA, HSUPA, 802.11bgn	HSDPA, HSUPA, 802.11bgn
Operační systém	Android 4.4.4	Windows Phone 8	Android 4.3

Zdroj: vlastní, zpracování podle údajů výrobců zařízení

Low-End

Do této kategorie náleží nejlevnější produkty na trhu. Nejlevnější telefony disponují pouze zastaralými technologiemi a hardwarem. Tyto základní modely často již disponují displeji s HD rozlišením a 2-jádrovými procesory, což jsou specifikace plně dostačující na běžné prohlížení webu. Na úplný konec spadají modely s nyní již poněkud archaickým rozlišením pouhých 480 x 320 px.

Tab. 16 – Vybrané telefony nejnižší třídy

	 **Lenovo A328**	 **Xiaomi RedMi 1S**	 **ASUS ZenFone 5**
Úhlopříčka	4.5"	4.7"	5"
Rozlišení	854 x 480	1280 x 720	1280 x 720
Hustota bodů		316 ppi	
Procesor	4-jádrový 1.3 GHz	4-jádrový 1.6 Ghz	2-jádrový Intel Atom Z2560 (1,6 GHz)
RAM	1 GB	1 GB	1 GB
Video	H.264: až 1080p (HW kodek)	H.264: až 1080p (HW kodek) H.265: přehrávání až do 1080p30	H.264: až 1080p (HW kodek)
Operační systém	Android 4.4 KitKat	Android 4.3	Android 4.3

Zdroj: vlastní zpracování podle údajů výrobců zařízení

Kategorie tabletů

V nejnižší třídě tabletů jsou nejčastěji dostupná zařízení disponující displejem o úhlopříčce 7 palců a rozlišení 1024 x 600. O výpočetní výkon se nejčastěji stará procesor s dvěma jádry o frekvenci 1,2 Ghz. Zařízení má zpravidla naistalován operační systém Android.

Zařízení spadající do střední třídy jsou již vybaveny displejem o úhlopříčce 10,1 palců a s rozlišením 1280 x 800. Procesor již zpravidla bývá vybaven čtyřmi jádry. V této kategorii jsou dostupná také zařízení s operačním systém Windows 8.1.

V nejvyšší kategorii tabletů se setkáváme se zařízeními vybavenými displeji o úhlopříčce v rozmezí 8“ až 10,1“. Tuto kategorii okupují zejména zařízení s rozlišením 2048 x 1536, kterým disponují především zařízení Apple iPad. Další typické rozlišení je 1920 x 1200, na vrcholu jsou pak zařízení s rozlišením 2560 x 1600 px. V této kategorii se můžeme setkat také s modely, které mají možnost mobilního datového připojení.

Tab. 17 – Typické tablety v jednotlivých třídách - zleva nejnižší, střední a nejvyšší třída

	 **GOCLEVER Inisgnia 700 PRO**	 **PRESTIGIO MultiPad Wize**	 **APPLE iPad Mini 3**
Úhlopříčka	7"	10,1"	7,9" (9,7")
Rozlišení	1024 x 600	1280 x 800	2048 x 1536
Hustota bodů	169 ppi	150 ppi	326 ppi
Procesor	Intel Atom Z2520, 2-jádrový, 64-bit, 1.2 Ghz	4-jádrový MediaTek MTK8127 (1.3 GHz)	Apple A7 - 2-jádrový, 1.3 Ghz, 64-bit
RAM	2 GB	1 GB	3 GB
Video	HW kodek: H.263, H.264, VC1 (až 1080p30)	HW dékodování: H.265, H.264 až 1080p30	HW kodek: H.264 až 1080p60, High Profile Level 4.2 s audiem AAC-LC až 160 kb/s
Data	802.11 b/g/n	802.11 b/g/n	LTE , 802.11 b/g/n
Operační systém	Android 4.4	Andorid 4.4	iOS 8

Zdroj: vlastní zpracování podle údajů výrobců zařízení

6 Praktické využití HTML5 a multimediálních prvků na webové stránce

Praktická ukázka bude spočívat ve vytvoření nové webové prezentace pro regionální autoklub Krakonoš Jilemnice. Cílem je vytvoření webových stránek s moderním a intuitivním designem. Stránky budou obsahovat video galerii, která by měla poskytnout uživatelům video obsah ve vysoké kvalitě bez použití nástrojů třetích stran. Dalším cílem je vytvořit verzi pro mobilní zařízení, protože stále více uživatelů přistupuje na web také z mobilních zařízení. Původní stránky byli z tohoto hlediska již nevyhovující.

6.1 Použité nástroje a zařízení

K zápisu html kódu i kaskádových stylů byl použit známý textový editor Notepad++. Otestování správného zobrazení stránek v aktuálních verzích prohlížečů bylo provedeno na notebooku vybaveným operačním systémem Windows 7 a displejem o rozlišení 1440 x 900. K otestování starších prohlížečů a prohlížeče Safari byl použit online nástroj CrossBrowserTesting, který používá reálná zařízení a ne pouze jejich emulaci. Mobilní verze webu byla testována na telefonu Sony Xperia Tipo vybaveným OS Android 4.04 a tabletu Asus MemoPad s OS Android 4.4.2. Dále byl využit emulátor, který je součástí prohlížeče Chrome.

6.2 Uspořádání stránky

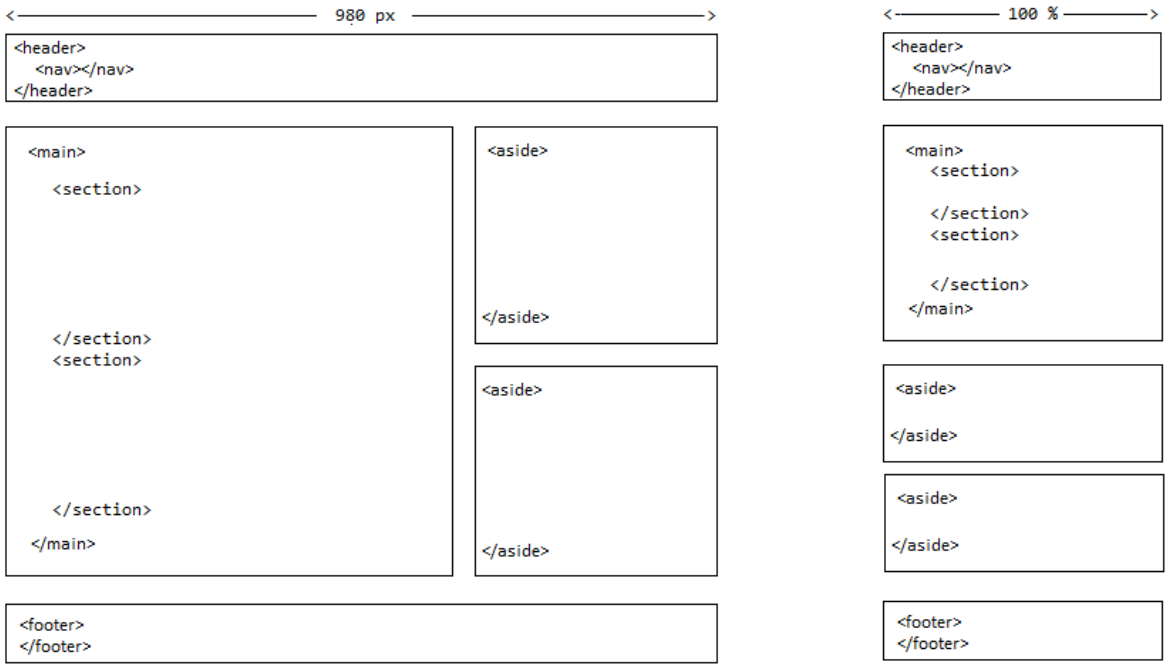
Hlavní části stránky jsou tvořeny s využitím nových HTML5 elementů: *header*, *nav*, *main*, *aside* a *footer*. V horní části se nachází hlavička – element *header*, která obaluje název webu a hlavní navigační menu tedy element *nav*. Následuje element *main*, ve kterém je umístěn hlavní obsah stránek rozčleněný na sekce s využitím elementů *section*. Na pravé straně se nacházejí elementy *aside*, které slouží k zobrazení doplňkových informací, např. rozpis závodů. Na konci stránky se nachází patička stránky.

Pro stylování byli vytvořeny dva soubory kaskádových stylů. Jeden pro mobilní zařízení s maximální úhlopříčkou 7 palců a druhý jako výchozí. Na každé stránce jsou pak

v hlavičce umístěny odkazy na jednotlivé kaskádové styly. Verze pro mobilní zařízení je s příslušným mediaquery dotazem:

```
<link rel="stylesheet" href="styl_m.css" media="(max-width: 7in)">
```

U výchozího stylování je nastavena šířka na pevnou hodnotu 980px, u mobilní verze je nastavena na hodnotu 100%. Obdobně je to i s ostatními prvky, aby docházelo k automatickému přizpůsobování prvků velikosti obrazovky. Schematické uspořádání obou verzí je ilustrováno na obrázku níže.



Obr. 12 – Schematické rozložení navržených webových stránek pro desktopovou a mobilní verzi
Zdroj: vlastní

Důležitým prvkem každých stránek je menu. U výchozího stylování je použito klasické horizontální menu. Mobilní verze má implementováno menu, které je vyvoláno jediným tlačítkem umístěným v hlavičce stránky.

Obr. 13 – Ukázka menu u mobilní verze webu
Zdroj: vlastni

Protože se práce zabývá spíše multimediálním obsahem stránek, tak zde dále nebude podrobněji popisováno, jak bylo docíleno finálního vzhledu stránek. Podrobněji bude popsána problematika umístění videa jako pozadí stránek a tvorba vlastního HTML5 video přehrávače.

Pozadí stránek

Jednotlivé části zmíněné výše jsou stylovány do červené barvy s 75% průhledností. Průhlednost je důležitá, aby skrz bloky prosvítalo pozadí celé stránky. První variantou pozadí je cyklické přehrávání videa po celé viditelné zobrazovací ploše okna prohlížeče. Tato možnost bude využívána pouze u hlavní stránky především jako upoutávka na následující závody nebo jako ukázka ze sestřihů závodů uplynulých. U ostatních stránek je pozadí tvořeno z několika se střídajících fotografií.

K dosažení požadovaného efektu bylo dosaženo užitím JavaScriptu a *mediaqueries*. Cílem použití *mediaqueries* je zamezení stáhnutí videa a jeho přehrávání na pozadí u mobilních zařízení. Minimální požadovaná šířka pro zobrazení videa byla zvolena na 11 palců. Díky JavaScriptu je video element přidán do kódu stránky dynamicky. JavaScriptový kód vypadá následovně:

```
var mq = window.matchMedia( "(min-width: 11in)" );
  if (mq.matches){
  window.addEventListener('load', eventWindowLoaded, false);
  var videoElement;
  }else{
    return;
  }
```

```
  function eventWindowLoaded(){
    videoElement = document.createElement('video');
    document.body.appendChild(videoElement);
    videoElement.id = "bgvid"
    videoElement.setAttribute("loop", "loop");
    videoElement.addEventListener("canplaythrough", videoLoaded, false);
    var videoFormat = supportedVideoFormats(videoElement);
    if (videoFormat == ""){
    bgGalery()
    }
    videoElement.setAttribute("src", "video/vp9." + videoFormat);

  }
  function supportedVideoFormats(video){
  var extension = "";
  if (video.canPlayType("video/webm") == "probably" ||
      video.canPlayType("video/webm") == "maybe"){
      extension = "webm";
  }else if (video.canPlayType("video/mp4") == "probably" ||
      video.canPlayType("video/mp4") == "maybe"){
      extension = "mp4";
      return extension
  }
  function videoLoaded(event){
  videoElement.play()
 }
```

V kódu jsou dva *event listenery*. První zachytává načtení celé stránky. Po načtení stránky je vytvořen element video a připojen jako potomek k elementu body. Dále následuje přidání atributů pro přehrávání ve smyčce a pro pozadí videa než bude načteno. Další *event listener* slouží k zachycení události, zdali bylo video již načteno. Ta pak následně vyvolává funkci k jeho spuštění. Pomocí funkce *supportedVideoFormats* je testováno jaké formáty prohlížeč zvládne přehrát. Když prohlížeč není schopen přehrát žádný z formátu je na pozadí místo videa spuštěna galerie z fotografií.

Další část kódu ukazuje funkce, které zajišťují změnu fotografie na pozadí každých 60 s:

```
function bgGalery (){
  setInterval(loopImages, 60000);
  }
  function loopImages (){
  if (i > bgImgCount) i = 1;
  document.body.style.backgroundImage = "url(" + 'bg/bg' + i + '.jpg' +
")";
  i++;
  }
```

K zobrazení videa na celém pozadí slouží následující CSS stylování:

```
video#bgvid {
```

```
    position: fixed;
    top: 0;
    left: 0;
    min-width: 100%;
    min-height: 100%;
    z-index: -100;
}
```

Vlastnost *postiton* je nastavena na hodnotu *fixed,* což zajišťuje stálou pozici videa při *scrolování*. Aby se video roztáhlo po celé stránce, byly vlastnosti *min-width* a *min-height* nastaveny na hodnotu 100 %.

Pro galerii fotografií na pozadí je použita tato sada CSS pravidel:

```
body {
  background: url(bg.jpg);
  background-repeat: no-repeat;
  background-size: cover;
  background-position: center center;
  background-attachment: fixed;
  transition: background 1.5s;
```

Stálou pozice pozadí při *scrolování* zajišťuje vlastnost *background-attachment* nastavená na hodnotu *fixed*. Pomocí vlastnosti *background-size* nastavené na hodnotu *cover* se obrázek rozprostře do okna prohlížeče. Aby byl přechod mezi fotografiemi plynulý byl aplikován přechod na vlastnost background s délkou trvání 1.5 s.

6.3 Tvorba vlastního HTML5 video přehrávače

Stránky autoklubu mají jednoduchou video galerii, kde jsou jednotlivá videa seřazená pod sebou. Cílem tvorby vlastního video přehrávače je vytvořit přehrávač pro tuto galerii s designem, který bude korespondovat s celkovým vzhledem stránek. Výhodou vlastního přehrávače je rovněž sjednocení vzhledu napříč prohlížeči. Dalším požadavkem bylo, aby design přehrávače byl responzivní.

Uživatelské rozhraní

Požadavky na možnosti uživatelského rozhraní přehrávače bylo zajištění základní funkcionality pro ovládání videa a zobrazení informací o přehrávání. To tedy obnášelo implementaci tlačítka na spuštění a pozastavení přehrávání, implementaci ukazatele průběhu přehrávání, ukazatele aktuálního času přehrávání a délky videa. Důležitá byla také implementace tlačítka pro zobrazení v režimu celé obrazovky.

Celý video přehrávač je tvořen elementem div, který obaluje video element spolu s dalšími elementy, které tvoří uživatelské rozhraní. Celý zápis v HTML kódu vypadá následovně:

```
<div class="player-wrap">
      <video          width="480px"          title="Titulek          videa">
      <source src="video/ECC.mp4"/>
      </video>
      <div class="controls-bar">
        <div class="button play"></div>
        <div class="video-progress">
        <div class="video-progress-bar"></div>
        <div class="details">
            <span class="title">Titulek videa</span>
            <span class="timing"></span>
        </div>
        </div>
        <div class="button bigger"></div>
    </div>
</div>
```

Celé uživatelské rozhraní je obaleno div elementem , ve kterém se nacházejí dva div elementy představující tlačítka pro spuštění/pozastavení a pro přechod do zobrazení na celou obrazovku. Element div s identifikátorem *video_progress* obaluje div pro zobrazení průběhu přehrávání a div, ve kterém jsou umístěny elementy span pro zobrazení titulku videa a informací o době přehrávání. Při „pokliku“ na tento element je video přehráváno od příslušné pozice v časové ose, tvoří tak funkcionalitu pro tzv. seeking.

Ikony obou tlačítek jsou ve vektorovém formátu, aby nedocházelo k jejich „rozmazání“ u velkých rozlišení a přehrávač, tak splnil požadavek na responzivní design. Konkrétně se jedná o fonty Entypo[42], které jsou importovány ve vektorovém formátu SVG. Tyto použité fonty jsou volně dostupné. Jejich import je zajištěn pomocí následujícího zápisu kaskádových stylů:

```
@font-face {
    font-family: "Entypo";
    url("icons/entypo.svg") format("svg");
    font-weight: normal;
    font-style: normal;
}
```

Do jednotlivých tlačítek jsou ikony vloženy využitím pseudotřídy *:after*. Zápis níže ukazuje konkrétní implementaci pro tlačítko zajišťující spuštění přehrávání.

```
.play::after {
```

[42] *Entypo* [online]. 2015. [cit. 2015-04]. Dostupné z: http://www.entypo.com/index.php

```
font-family: Entypo;
content: "\e001";
}
```

Celé uživatelské rozhraní lze jednoduše barevně modifikovat pomocí kaskádových stylů. Ve třídě *button* je možné změnit barvu pro pozadí tlačítek a barvu samotných ikon tlačítek:

```
.button {
    position: absolute;
    width: 64px;
    height: 100%;
    font-size: 32px; /* definuje velikost ikon tlacitek */
    line-height: 64px;
    text-align: center;
    color: #fff; /* definuje barvu ikon tlacitek */
    background-color: #00aef6;  /* definuje barvu pozadi tlacitek*/
}
```

Ve třídě *video-progress* je možné změnit barvu pro pozadí *progessbaru*:

```
.video-progress {
    height: 100%;
    position: absolute;
    bottom: 0;
    left: 64px;
    right: 64px;
    background-color: black; /* definuje barvu pozadi progressbaru */
}
```

Ve třídě *video-progress-bar* vlastnost *background-color* určuje barvu samotného *progressbaru*:

```
.video-progress-bar {
    height: 100%;
    width: 0%;
    position: relative;
    transition: width .5s;
    background-color: aqua; /* definuje barvu progressbaru */
}
```

Obrázek níže znázorňuje příklad možných barevných variant ovládacích prvků.

Obr. 14 – Příklad možných barevných variant ovládacího panelu přehrávače
Zdroj: vlastní

Při nečinnosti uživatele se menu schová. Tato funkcionalita byla implementována s použitím animací a průhlednosti:

```
.hide-controls {
    animation: hideAnimation 2s ease-in 2s forwards;}
@keyframes hideAnimation {
    to {opacity: 0;}
}
```

Obrázek Obr. 15 ilustruje konkrétní použitý motiv na webových stránkách autoklubu. Přehrávač je zde ve výchozím stavu a používá atribut poster pro náhled videa.

Obr. 15 – Ukázka vzhledu přehrávače v galerii
Zdroj: vlastní

Na dalším obrázku je zobrazen při přehrávání.

Obr. 16 – Ukázka přehrávače během přehrávání
Zdroj: vlastní

Programování funkcí přehrávače

Html kód přehrávače uvedený výše není vytvářen ručně, ale je vytvářen dynamicky pomocí JavaScriptu. Při načtení stránek je zavolána inicializační funkce, která vyhledá všechny video elementy na stránce a ke každému přidá potřebné elementy. Tento postup usnadňuje implementaci přehrávače a jeho přenositelnost. Následně jsou uživatelsky interaktivním prvkům přidány *event listenery*, které volají příslušné funkce. Přidání jednotlivých *event listenerů* je zobrazen v kódu níže.

```
btnPlayPause.addEventListener('click', playPause, false);
btnFullScreen.addEventListener('click', toggleFullScreen, false);
progressPosition.addEventListener('click', changePosition , false);
myVideo.ontimeupdate = function () { updateControlBar()};
myVideo.onloadedmetadata = function(){ updateTimeLabel()};
```

První tři *event listenery* slouží k zachytávání kliků uživatele. První zachytává stisky tlačítka pro přehrávání a pozastavení. *Event listener* pak vyvolá funkci *playPause* s následujícím kódem:

```
function playPause() {
```

```
    if (video.paused){
        video.play();
        btnPlayPause.classList.remove("play");
        btnPlayPause.classList.add("pause");
    }else {
        video.pause();
        btnPlayPause.classList.remove("pause");
        btnPlayPause.classList.add("play");
        }
```

Funkce zajistí v případě, kdy je video zastavené jeho spuštění a tlačítku změní ikonu na znázorňující pozastavení. Záměna ikonek je realizována pomocí funkcí pro přidání a odstranění tříd kaskádových stylů. Pokud je video spuštěné, tak naopak přehrávání pozastaví a rovněž změní ikonu tlačítka.

Druhý *event listener* zachytává stisky tlačítka pro přechod do režimu celé obrazovky. Při stisku je volána funkce *toggleFullScreen*, která používá stejný princip záměny ikonek:

```
function toggleFullScreen() {
  if (!document.fullscreenElement) {
    if (player.requestFullscreen) {
      player.requestFullscreen();
      btnFullScreen.classList.remove("bigger");
      btnFullScreen.classList.add("smaller");
    }
  } else if (document.exitFullscreen) {
      document.exitFullscreen();
      btnFullScre.classList.remove("smaller");
      btnFullScre.classList.add("bigger");
  }
  }
}
```

Funkce využívá tzv. FullScreen API, které nemá zcela uzavřenou specifikaci a proto je ve skutečnosti celá funkce složitější o prefixi k jednotlivým prohlížečům. Zobrazený kód pro jednoduchost počítá již s uzavřenou specifikací. Princip funkce je jednoduchý: pokud není prohlížeč v režimu celé obrazovky, tak je přehrávač zobrazen na celou obrazovku a naopak pokud se prohlížeč nachází v režimu celé obrazovky je tento režim ukončen.

Po kliknutí na tzv. *progressbar*, který ukazuje procentuální stav přehrávání, dojde k vyvolání funkce, která zajistí přetočení videa na zvolené místo:

```
function changePosition() {
    var position = getClickPosition (progressBar);
        video.currentTime = (position.x / progress.offsetWidth) *
video.duration;
    }
```

Pomocí funkce *getClickPosition* je zjištěna pozice kliku v elementu. Z podílu mezi pozicí kliku v rámci osy x a šířkou elementu násobeného délkou videa je určen čas na, který má být video přetočeno.

Další *event listenery* jsou přiřazeny samotnému video elementu, kterému je v kódu přidána reference *myVideo*:

```
myVideo.onloadedmetadata = function(){ updateTimeLabel()};
myVideo.ontimeupdate = function () { updateControlBar()};
```

První volá funkci *updateTimeLabel*, když jsou načtena *metadata* videa. Funkce pak aktualizuje informaci o délce videa na panelu.

```
function updateTimeLabel(){
    timeLabel.innerHTML = toTime(myVideo.currentTime) + '/' +
toTime(myVideo.duration) ;
    }
```

Druhý slouží k volání funkce updateControlBar, která atualizuje aktuální čas přehrávání a také aktualizuje progessbar.

```
function updateControlBar (){
    progressBar.style.width = 100 * myVideo.currentTime /
myVideo.duration + "%";
    timeLabel.innerHTML = toTime(myVideo.currentTime) + '/' +
toTime(myVideo.duration) ;
    }
```

Další dva *event listenery* jsou přidány při zahájení přehrávání.

```
player.addEventListener("mousemove", resetTimer, false);
player.addEventListener("touchmove", resetTimer, false);
```

Oba slouží k zachytávání aktivity uživatele v okně přehrávače. První zachytává aktivitu myši a druhý pohybů po dotykové obrazovce. Pokud není uživatel aktivní po dobu 2 s, která je nastavena v časovači je volána funkce *hideControls* sloužící k přidání třídy *hide-controls* k referenci na element představující uživatelské rozhraní.

```
timeout = window.setTimeout(hideContols, 2000);
}
function hideContols() {
    controlBar.classList.add("hide-controls")
}
```

Pokud je uživatel aktivní je zavolána funkce *resetTimer*, která obnovuje časovač a dále

volá funkci *showControls*. Funkce *showControls*, která naopak odstraňuje třídu *hide-controls*.

```
function resetTimer(e) {
    window.clearTimeout(timeoutID);
    showControls();
}
function showControls() {
    controlBar.classList.remove("hide-controls")
    startTimer();
}
```

Celá implementace přehrávač se skládá ze dvou souborů. První soubor je media-player.js, který obsahuje inicializační funkci a funkce pro obsluhu přehrávače. Druhý soubor obsahuje kaskádové styly přehrávače a jmenuje se media-player.css.

Následujícím příklad ukazuje vložení přehrávače do stránek.

```
<html>
<head>
      <meta charset="UTF-8">
      <title>Videogalerie</title>
      <link rel="stylesheet" href="media-player.css">
      <script src="media-player.js"></script>
</head>
<body onload="init()">
      <video title="Titulek videa 1"><source src="video1.mp4"/></video>
      <video title="Titulek videa 2"><source src="video2.mp4"/></video>
</body>
</html>
```

Do hlavičky html kódu je nutné vložit odkazy na oba soubory a do elementu body je přidán parametr *onload* s inicializační funkcí. Inicializační funkce se postará, aby na všechny elementy video byl aplikován náš naprogramovaný přehrávač. U video elementů je uvedený argument *title*, který je následně zobrazen v přehrávači.

Do přehrávače byla také implementována referenční DASH knihovna. Aby přehrávač přehrával video pomocí této knihovny adaptivně je nutné uvedení MPD souboru ve zdrojích na první pozici. Jinak bude načteno video na první pozici. V případě, kdy prohlížeč nepodporuje Media Source Extensions je zdroj z MPD souboru ignorován. Přehráván bude první podporovaný formát uvedený v elementu source.

Videogalerie na stránkách autoklubu, zatím nevyužívá funkcionality adaptivního streamování, protože podpora Media Source Extensions v prohlížečích je v současnosti na

nízké úrovni. V praxi by to znamenalo zbytečným plýtváním úložným prostorem na webovém serveru a také časem, který je potřebný ke kódování do různým datových toků a k vytváření MPD souborů.

6.4 Ekonomické zhodnocení

Tako kapitola má za úkol shrnout časovou i finanční náročnost projektu dalším cílem bylo najít alternativních řešení a jejich náklady.

Zhodnocení navrženého řešení

Napsání samotných webových stránek si vyžádalo časovou dotaci cca 70 h, z toho bylo přibližně 10 h věnováno testování v různých prohlížečích a na různých zařízeních. Naprogramování prohlížeče zabralo cca 25 h a dalších 5 h bylo věnováno ladění kódu a odstraňování problémů. Jelikož je autor úzkým spolupracovníkem klubu, pro který byl projekt vypracován, tak celý projekt vytvářel bez nároku na odměnu.

Dodatečné náklady budou náklady na hosting. Společnost Comanet[43], která je dceřinou společností jednoho ze sponzorů autoklubu, je ochotna nabídnout hosting za 300 Kč/ rok s 15 GB prostorem pro data a neomezeným datovým provozem. Tyto parametry jsou pro web daného rozsahu zcela dostačující. Hosting u konkurenční společnosti Endora.cz[44] je nabízen s prostorem 10 GB za 200 Kč ročně. Dalších 199 Kč si společnost účtuje na registraci a provoz domény.

Alternativní řešení

Alternativním řešením je realizace stránek na zakázku. Řada společností nabízí grafický návrh včetně napojení na redakční systém Joomla či Wordpress v cenové relaci 10 000 Kč až 12 000 Kč. Společnost Sympact[45] by si za realizaci projektu účtovala částku 12 000 Kč. Další náklady by představoval hosting, jehož možné varianty jsou již zmíněny u vlastního řešení.

[43] *Comanet* [online]. [cit. 2015-05-01]. Dostupné z: http://www.comanet.cz/
[44] *Endora* [online]. [cit. 2015-05-01]. Dostupné z: http://www.endora.cz/
[45] *Sympact* [online]. [cit. 2015-05-01]. Dostupné z: http://www.sympact.cz/

6.5 Shrnutí vlastností HTML5 v oblasti multimédií

HTML5 přineslo nové elementy <video> a <audio>, které umožňují přehrávat video a audio obsah přímo v prohlížeči bez nutnosti použití externích pluginů. Díky tomuto přístupu zvládnou na stránky přidávat multimediální obsah i méně zkušení vývojáři. Naopak zkušenější vývojáři mohou k těmto elementům vytvářet pomocí kaskádových stylů a JavaScriptu vlastní ovládací prvky. Dříve by pro obdobný efekt musel vývojář disponovat znalostmi další technologie, např. Flash. Hlavním přínosem je umožnění přehrávání multimédiálního obsahu na mobilních zařízeních, které již nedisponují podporou pro Flash.

Dalším významným krokem je uvedení API Media Source Extensions, pomocí kterého je umožněno v JavaScriptu naprogramovat funkcionalitu pro adaptivní streaming. Tímto krokem je smazána výhoda konkurenčního Adobe Flash, který pro adaptivní streaming využívá své proprietární technologie Adobe Smooth Streaming.

Další výhoda Adobe Flash v oblasti distribuce chráněného obsahu byla smazána uvedením API Encrypted Media Extension. Nicméně v současnosti ne všechny prohlížeče již integrují tato rozšíření.

Hlavní nevýhodou HTML5 v oblasti multimédií je neexistence jednoho společného video či audio formátu podporovaného v prohlížečích. Z toho vyplývá nutnost poskytnout obsah ve více formátech, aby byla zajištěna kompatibilita se všemi prohlížeči. Nicméně neexistence jednotného formátu sebou přináší i jisté výhody. Tuto výhodu představuje zejména integrace podpory pro nové efektivnější video a audio formáty do prohlížečů. Příkladem může být podpora nového formátu VP9 v celé řadě prohlížečů.

Závěr

Hlavním cílem této práce bylo shrnout situaci v oblasti vývoje HTML5 a jeho nových elementů, zejména pak elementů zaměřených na distribuci multimediálního obsahu. Práce se zabývá i související technologií kaskádových stylů CSS3, která tvoří neméně důležitou složku v oblasti distribuce multimediálního obsahu a především přináší nové možnosti pro vytváření obsahu pro mobilní zařízení.

V práci byly popsány jednotlivé způsoby distribuce multimediálního obsahu s využitím multimediálních elementů a rovněž bylo provedeno porovnání s alternativami. Hlavní alternativou pro multimediální elementy je technologie Adobe Flash, která již není podporována na mobilních zařízeních. Nicméně její výhoda spočívá zejména v distribuci autorsky chráněného obsahu a proto je v této oblasti ještě hojně využívána. HTML5 přineslo nové API Encrypted Media Extensions, které rovněž slouží pro distribuci chráněného obsahu, podpora v prohlížečích však ještě nedosáhla uspokojivé úrovně. Z těchto důvodů by vývojáři neměli ani jednu z technologií zcela ignorovat a k zajištění zpětné kompatibility použít obě.

Dalším cílem práce byl průzkum trhu, ve kterém bylo zjištěno, že stále více uživatelů přistupuje na webové stránky z mobilních zařízení. Z toho důvodu by webové stránky měli být optimalizovány rovněž pro zobrazování na těchto zařízeních. Rovněž by na tyto zařízení měl být distribuován multimediální obsah odpovídající jejich možnostem. Průzkum ukazuje, že i ta nejlevnější zařízení na trhu jsou schopna přehrát video o vysokém rozlišení kódované ve formátu H.264. Limitací je tedy spíše rychlost datového připojení, kterým uživatel disponuje.

Poslední část práce spočívala v praktické ukázce využití HTML5 pro distribuci multimediálního obsahu. Jako ukázka byly vytvořeny nové webové stránky pro Autoklub Krakonoš Jilemnice. Stěžejní částí pak bylo naprogramování vlastního HTML5 video přehrávače, který slouží pro přehrávání videí ve videogalerii. V praktické ukázce byla využita většina v práci popsaných nových HTML5 elementů i CSS3 vlastností. Kompletní webové stránky, pak byli otestovány na několika zařízeních s různými prohlížeči a i přes použití nových prvků nebyli zaznamenány problémy s kompatibilitou.

Seznam použité literatury

Citace

CASTRO, Elizabeth a Bruce HYSLOP. *HTML5 a CSS3: názorný průvodce tvorbou WWW stránek*. 1. vyd. Brno: Computer Press, 2012, 439 s. ISBN 978-80-251-3733-8

CSS3 Transforms. *W3Schools.com* [online]. [cit. 2015-03-15]. Dostupné z: http://www.w3schools.com/css/css3_2dtransforms.asp

DUCKETT, Jon. *HTML: design and build websites*. Indianapolis, IN: Wiley, c2011, 490 p. ISBN 11-180-0818-9.

FLV and F4V File Format Specification (Version 10.1). 2010. *Adobe.com* [online]. [cit. 2015-05-02]. Dostupné z: http://download.macromedia.com/f4v/video_file_format_spec_v10_1.pdf

First Weekend iPhone Sales Top 10 Million, Set New Record. *Apple* [online]. [cit. 2015-04-17]. Dostupné z: http://www.apple.com/pr/library/2014/09/22First-Weekend-iPhone-Sales-Top-10-Million-Set-New-Record.html

FULTON, Steve a Jeff FULTON. 2011. *HTML5 Canvas: native interactivity and animation for the Web*. 1st ed. Sebastopol, CA: O'Reilly, xix, 628 p. ISBN 978-1-449-39390-8.

GERBER, Jan. 2009. *OGG THEORA COOK BOOK* [online]. [cit. 2015-05-03]. Dostupné z: http://en.flossmanuals.net/ogg-theora/encoding/_booki/ogg-theora/ogg-theora.pdf

GOLDSTEINOVÁ, Alexis, Louis LAZARIS a Estelle WEYLOVÁ. *HTML5 a CSS3 pro webové designéry*. 1. vydání. Jan Pokorný. Brno: Zoner Press, 2011, 286 s. ISBN 978-80-7413-166-0

Historie a vývoj HTML. In: *Http://htmlguru.cz* [online]. 2013 [cit. 2015-05-02]. Dostupné z: http://htmlguru.cz/uvod-historie.html

HTML5: A vocabulary and associated APIs for HTML and XHTML Editor's Draft 13 November 2014. *W3C* [online]. 2014 [cit. 2015-05-02]. Dostupné z: http://www.w3.org/html/wg/drafts/html/CR/

HTML5 Specifications. 2014. *W3C* [online]. [cit. 2015-02-01]. Dostupné z: http://www.w3.org/TR/html5/

JOBS, Steve. Steve Jobs explains why Apple's not a fan of Flash. In: *NETWORKWORLD* [online]. 2010 [cit. 2015-02-11]. Dostupné z: http://www.networkworld.com/article/2230600/steve-jobs-explains-why-apple-s-not-a-fan-of-flash.html

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

MAKZAN,. 2012. *Programujeme hry v HTML5*. 1. vyd. Brno: Computer Press, 320 s. ISBN 978-80-251-3731-4

MOBILE HARDWARE STATS. 2015. *Unity* [online]. [cit. 2015-03-04]. Dostupné z: http://stats.unity3d.com/mobile/display.html

MPEG-DASH CONTENT GENERATION USING MP4BOX AND X264. 2014. *Bitdash* [online]. [cit. 2015-23-04]. Dostupné z: https://www.dash-player.com/blog/2014/11/mpeg-dash-content-generation-using-mp4box-and-x264/

Opus Interactive Audio Codec. 2014. *Opus* [online]. [cit. 2015-05-04]. Dostupné z: http://www.opus-codec.org

Our Story. 2014. *RealNetworks* [online]. [cit. 2015-05-02]. Dostupné z: http://www.realnetworks.com/our-story/

OZER, By Jan. *Producing streaming video for multiple screen delivery*. Galax, Va: Doceo Publishing, 2013. ISBN 09-762-5954-0.

OZER, By Jan. *Video compression for Flash, Apple devices and HTML5*. Galax, VA: Doceo Publ, 2011. ISBN 09-762-5950-8.

PFEIFFER, Silvia. *The definitive guide to HTML5 video*. Distributed to the book trade worldwide by Springer Science Business Media, xiv, 321 p. ISBN 14-302-3090-8.

PILGRIM, Mark, Brian ALBERS a Frank SALIM. 2010. *HTML5: up and running*. 1st ed. Sebastopol, CA: O'Reilly, 304 s. ISBN 05-968-0602-7

Screen Resolution Statistics. 2014. *W3Schools.com* [online]. [cit. 2015-05-04]. Dostupné z: http://www.w3schools.com/browsers/browsers_display.asp

SIMPSON, Wes. 2013. Adaptive Streaming Adoption. *Video edge* [online]. (3) [cit. 2015-03-04]. ISSN 2372-9244. Dostupné z: http://search.proquest.com/docview/1637940701/fulltext/17D080027EBB4355PQ/1?accountid=17116

The HTML5 progress Element. *CSS-Tricks* [online]. [cit. 2015-07-04]. Dostupné z: https://css-tricks.com/html5-progress-element/

The State of HTML5 Video. In: *JWPlayer* [online]. 2015 [cit. 2015-02-11]. Dostupné z: http://www.jwplayer.com/html5/

The ultimate SVG guide. 2014. *PSDtoWORDPRESS* [online]. [cit. 2015-05-03]. Dostupné z: https://psdtowp.net/svg.html

Theora benefits. 2011. *Theora.org* [online]. [cit. 2015-03]. Dostupné z: http://www.theora.org/benefits/

Top JavaScript Technologies. In: *SimilarTech* [online]. 2014 [cit. 2015-02-05]. Dostupné z: https://www.similartech.com/categories/javascript

WebM Video for Microsoft Internet Explorer 9+. 2014. *Tools Google* [online]. [cit. 2015-03-04]. Dostupné z: https://tools.google.com/dlpage/webmmf/

WIEGAND, T., G.J. SULLIVAN, G. BJONTEGAARD a A. LUTHRA. 2003. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology* [online]. **13**(7): 560-576 [cit. 2015-08-03]. DOI: 10.1109/tcsvt.2003.815165. Dostupné z: http://ip.hhi.de/imagecom_G1/assets/pdfs/csvt_overview_0305.pdf