Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Tvorba webových aplikací s využitím HTML5 a CSS3

Bakalářská práce

Autor: Tomáš Andrys
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Monika Borkovcová

Hradec Králové duben 2015

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

……

V Hradci Králové dne 26.4.2015 Tomáš Andrys

Poděkování

Rád bych poděkoval vedoucí mé bakalářské práce Ing. Monice Borkovcové za její čas a ochotu. Dále bych rád poděkoval celé své rodině za podporu po celou dobu studia.

Anotace

Tato bakalářská práce je zaměřena na značkovací jazyk HTML5. Práce je rozdělena na část teoretickou a část praktickou. Teoretická část se zabývá přehledem historie a současnosti značkovacího jazyka HTML, popisem nových elementů a způsobu jejich použití. V další části je popsán způsob vývoje aplikací nativní, webové a hybridní. V neposlední řadě popisuje optimalizaci vzhledu stránky a uživatelského rozhraní.

V praktické části je popsána aplikace založená na prvcích rozhraní Web Storage, konkrétně na localStorage, jazyce JavaScript a frameworku jQuery Mobile. V práci je dále popsáno otestování aplikace v různých webových prohlížečích a na různých zařízeních. Tato aplikace napomůže ke zhodnocení přínosu této nové specifikace.

Annotation

Title: Development of web applications using HTML5 and CSS3

This bachelor thesis is focused on HTML5 markup language. This thesis is divided into theoretical and practical part. Theoretical part of this thesis deals with an overview of the history and present of HTML language, via describing new elements and how they are used. The next section describes how native, web and hybrid applications are developed. Finally, describes optimalization of appereance of the page and the user interface.

In the practical part, application based on Web Storage interface, specifically on localStorage, JavaScript and framework jQuery Mobile is described. Also testing of applications on different web browsers and different devices is described. This application will help to evaluate the benefits of this new specification.

Obsah

1 Úvod

Většina uživatelů webu, každý den přistupuje k webovým stránkám různého obsahu, ale málo kdo si uvědomuje skutečnou sílu prohlížečů, které se nyní používají. Možná si někteří všimnou změny vzhledu uživatelského rozhraní poté, co dojde k aktualizaci prohlížeče, ale často nemají ani potuchy o tom, že tato nová verze umožňuje komunikaci v reálném čase, volné kreslení na stránce nebo přehrávání audia a videa bez přidaných pluginů. Web již delší dobu není pouze pro počítačové nadšence, ale poskytuje mnoho hodin zábavy i velké množství informací pro běžné uživatele.

Toto všechno by nebylo možné, pokud by nedocházelo k neustálému vývoji technologie tvorby webových stránek. S příchodem HTML5 se tvorba webových stránek výrazně zjednodušila. Například s příchodem webového úložiště je možné uchovat data v paměti prohlížeče efektivnějším způsobem. Další novinkou je přizpůsobení webové stránky na velikost displeje daného zařízení bez vytvoření mobilní verze stránky.

Tato práce je zaměřena na značkovací jazyk HTML5 a zabývá se přehledem historie a současnosti značkovacího jazyka HTML, popisem nových elementů a způsobu jejich použití. Dále popisuje způsob vývoje aplikací nativní, webové a hybridní a v neposlední řadě se zabývá problematikou optimalizace vzhledu stránky a uživatelského rozhraní.

V závěru práce je popsána aplikace založená na prvcích rozhraní Web Storage, konkrétně na localStorage, jazyce JavaScript a frameworku jQuery Mobile, která napomůže ke zhodnocení přínosu této nové specifikace.

2 HTML5 a CSS3

2.1 Co je to vlastně HTML a CSS?

HTML (z anglického HyperText Markup Language) je značkovací jazyk, pomocí kterého se vytvářejí webové stránky. Je poměrně jednoduchý a tedy i vhodný pro začátečníky. Pomocí něho lze vytvářet webové stránky i bez wysiwyg editorů. Protože se v jazyce HTML využívají i další jazyky, jako např. CSS, JAVA, JavaScript a další, můžou i zde vznikat problémy. Samotné www stránky jsou pak jen textové soubory s koncovkou .html nebo .htm, ve kterých jsou k textu přidávány tagy, které určují význam jednotlivých částí textu. Zdrojový kód HTML není nijak překládán, a proto je dále šířen jako samostatný spustitelný soubor. Celou dobu vývoje webových stránek zůstává v textové podobě. Díky tomu je možné webové stránky vyvíjet v jednoduchém textovém editoru.

Termíny používané v HTML:

- TAG – základní značka jazyka HTML. Zapisuje se do špičatých závorek <> a může být párový nebo nepárový.
- ATRIBUT – zapisuje se přímo do tagu a upřesňuje nějakou vlastnost daného elementu, např. atribut `id` specifikuje unikátní id hodnotu.
- ELEMENT – je to obsah mezi úvodním a ukončovacím tagem včetně tagů a atributů, např. `<div id="id_1"></div>`.

CSS (z anglického Cascading Style Sheets) používají se ke stylování výsledného vzhledu stránky. Určuje, jak bude daný HTML element vypadat, např. nadpis stránky bude zobrazen tučným písmem Tahoma o velikosti 18 bodů a v barvě modré). Používání CSS se prosazovalo poměrně pomalu, ale v současné době asi neexistuje nikdo, kdo by při tvorbě webových stránek nepoužil stylování pomocí CSS [1].

2.2 Historie HTML

V roce 1991 vyvinul Tim Berners-Lee a Robert Caillau první verzi jazyka HTML, odvozeného od syntaxe SGML. Hlavním důvodem bylo, umožnit vědcům sdílet výsledky svých prací po celém světě. To, že oba vědci pracovali v Evropském centru jaderného výzkumu, zkráceně CERN, nebyla náhoda. Tak jak se časem zvyšovaly požadavky uživatelů na WWW stránku, Tim Bernes-Lee ve spolupráci s IETF vyvinul novou verzi jazyka HTML 2.0, která definovala práci s formuláři. Další verzí je HTML 3.0, jejímž autorem je Dave Raggett z laboratoří Hawlett-Packard. Ragget se k verzi HTML 3.0 dopracoval přes verzi HTML+, která obsahovala popis tvorby tabulek a prvky, které zajišťují lepší vzhled dokumentů. Verze HTML 3.0 se nikdy nedočkala implementace. Od této chvíle přechází vývoj jazyka HTML pod správu W3C (World Wide Web Consortium), které roku 1997 uvádí na trh verzi HTML 4.0 a přijímá ji jako standart. Tato verze bohužel obsahovala mnoho chyb, proto na konci roku 1999 představilo W3C opravnou verzi HTML 4.01 a vývoj jazyka se od této chvíle zastavil.

Již v roce 1998, W3C vydává další standard a to jazyk XML. Tento jazyk se ve světě používá k uložení a výměně dat. Vydáním tohoto jazyka se W3C začíná ubírat jiným směrem. V roce 2000 vydává W3C novou specifikaci vycházející z XML s názvem XHTML 1.0. Tato specifikace na rozdíl od HTML nevychází z jazyka SGML. Časem se však ukázalo, že vývoj jazyka XHTML byla slepá ulička.

Jelikož tvůrci prohlížečů (Mozilla Foundation, Opera Software či Apple) nesouhlasili s dalším vývojem XHTML, vzniká v roce 2004 pracovní skupina WHATWG. Jejímž cílem bylo připravit specifikace pro novou verzi HTML. V roce 2007 založilo W3C novou pracovní skupinu HTML Working Group jejímž cílem bylo vydání verze HTML5 založené na specifikacích Web Applications 1.0 a Web Forms 2.0 [3; 16].

2.3 Vývoj specifikace HTML5

V roce 2008 zveřejnila skupina WHATWG první veřejný pracovní návrh (First Public Working Draft) specifikace. Části HTML5 byly implementovány do prohlížečů navzdory tomu, že specifikace ještě nedosáhla konečného doporučení. V roce 2011 posunula pracovní skupina HTML5 do fáze Last Call. Již následující rok v červenci se WHATWG a W3C rozhodli, že WHATWG bude dále pracovat na HTML5 jako na tzv. Living Standard. Pojem Living Standard znamená, že nikdy není úplný a je neustále aktualizovaný a vylepšovaný. Nové funkce mohou být přidány, ale stávající nebudou odstraněny. Na konci roku 2012 W3C označilo HTML5 jako kandidáta na doporučení (Candidate Recommendation. Podle aktuálního plánu byla konečná specifikace (W3C Recommendation) HTML5 schválena v posledním čtvrtletí roku 2014 a budoucí verze HTML 5.1 by měla být schválena na konci roku 2016 [17].

Obrázek 1 - Přehled vývoje HTML5. Zdroj:[17]

	2012	2013	2014	2015	2016
HTML 5.0	Candidate Rec	Call for Review	Recommendation		
HTML 5.1	1st Working Draft		Last Call	Candidate Rec	Recommendation
HTML 5.2[29]				1st Working Draft	

2.4 Zjednodušení HTML5 oproti HTML 4.01

Oproti HTML 4.01 přichází HTML5 se značným zjednodušením zápisu. Příkladem může být zápis deklarace znakové sady v hlavičce stránky, tzv. `charset`. Místo složitého a zdlouhavého zápisu (Příklad 1),

Příklad 1 - Deklarace znakové sady v HTML 4.01. Zdroj:[autor]

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

stačí pouze tento zápis (Příklad 2).

Příklad 2 - Deklarace znakové sady v HTML5. Zdroj:[autor]

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

Dalším příkladem zjednodušení může být zápis deklarace hlavičky dokumentu, tzv. `DOCTYPE`. Dřívější zdlouhavý zápis (Příklad 3),

Příklad 3 – Deklarace DOCTYPE v HTML 4.01. Zdroj:[autor]

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

nahradil následující zápis (Příklad 4).

Příklad 4 – Deklarace DOCTYPE v HTML5. Zdroj:[autor]

```
<!DOCTYPE html>
```

Je to opravdové zjednodušení, které si každý zapamatuje.

2.5 Podpora HTML5 a CSS3 v prohlížečích

Vlastnosti HTML5 a CSS3 velmi usnadňují programátorovi jeho práci. Bohužel ne všechny moderní prohlížeče HTML5 a CSS3 podporují v plném rozsahu. Téměř maximální podporu má jazyk HTML5 v nejnovější verzi prohlížeče Google Chrome a Opera, jen o něco hůře je na tom Mozilla Firefox. Proti tomu u prohlížeče Internet Explorer, hlavně u starších verzí, je zcela mizivá (Obrázek 2). U CSS3 je situace velmi podobná. Pro programátora jsou velmi užitečné online nástroje, kde může snadno zjistit informace o podpoře jednotlivých vlastností HTML5 nebo CSS3 u jednotlivých prohlížečů. Níže uvádím několik z nich.

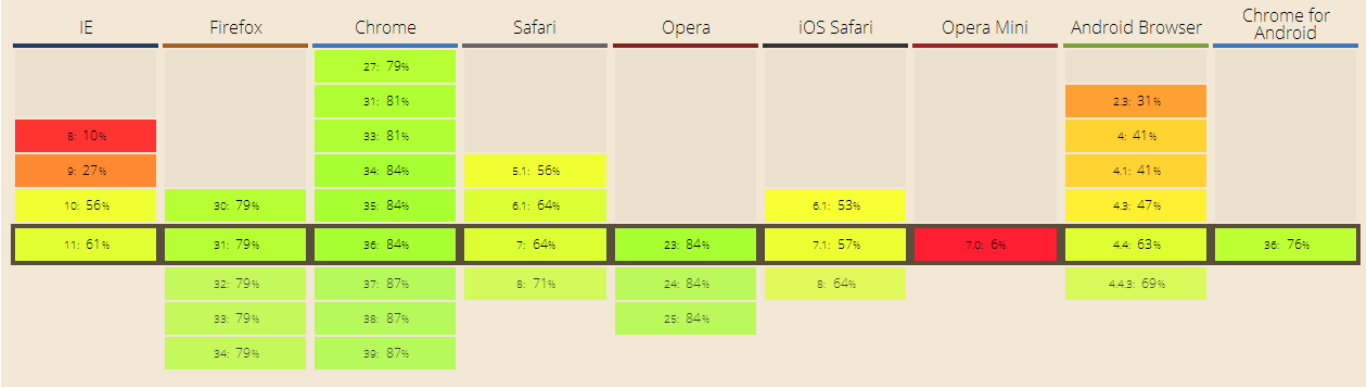
i. **Can I Use** - Stránky umožňují zobrazit podporu jednotlivých HTML5 elementů a CSS3 vlastností v různých verzích prohlížečů, ať už desktopových nebo mobilních, které si můžeme v nastavení vybrat.
   Dostupné na http://caniuse.com

ii. **HTML5 Test** - Jednotlivé prohlížeče jsou hodnoceny pomocí bodů od 0 do 555 a body získávají za podporu HTML5 elementů a jejich funkcí.
   Dostupné na http://html5test.com.

iii. **Find Me By IP** - Stránky zobrazují verzi použitého prohlížeče a jeho podporu HTML5 a CSS3 elementů.

Dostupné na http://fmbip.com.

Obrázek 2 - Přehled podpory HTML5 v prohlížečích. Zdroj:[7].



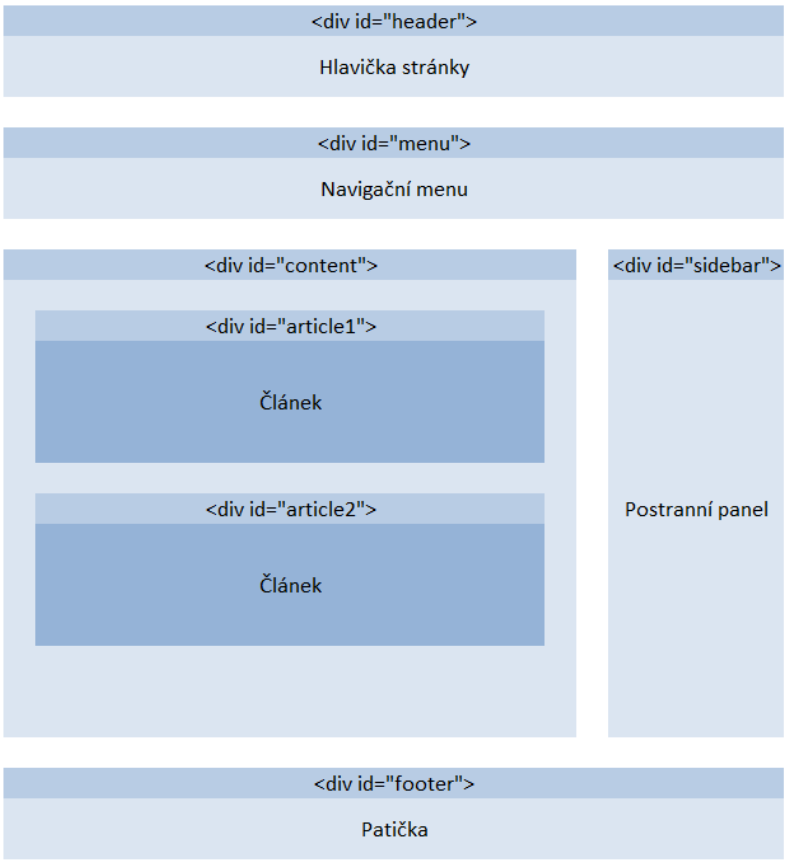
2.6 Základní stavební prvky stránky

Jeden z nejpoužívanějších elementů v HTML 4.01 je element `div`. `Div` je jeden ze základních stavebních prvků stránky. Pomocí `divů` se vytváří struktura stránky, jejíž součástí je hlavička, menu, obsah stránky a patička. Jednotlivé `divy` se rozlišují pomocí atributu `id`, který se nejčastěji používá u elementů, které jsou na stránce jen jednou. Název a účel jednotlivých `divů` je symbolizován hodnotou atributu `id`. Následující ukázka (Příklad 5) představuje zápis zdrojového kódu pomocí `div` elementů s odlišnou hodnotou `id`.

Příklad 5 - Stručný zápis zdrojového kódu v HTML 4.01. Zdroj:[autor]

```
<div id="header"></div><!-- Hlavička stránky -->
    <div id="menu"></div><!--Menu-->
<div id="content"><!-- Hlavní panel -->
    <div id="article1"></div><!-- Obsahový box 1 -->
    <div id="article2"></div><!-- Obsahový box 2 -->
</div>
<div id="sidebar"><!-- Pravý panel -->
</div>
<div id="footer"></div><!-- Patička stránky -->
```

Obrázek 3 - Rozložení stránky na jednotlivé sekce v HTML 4.01. Zdroj:[autor]

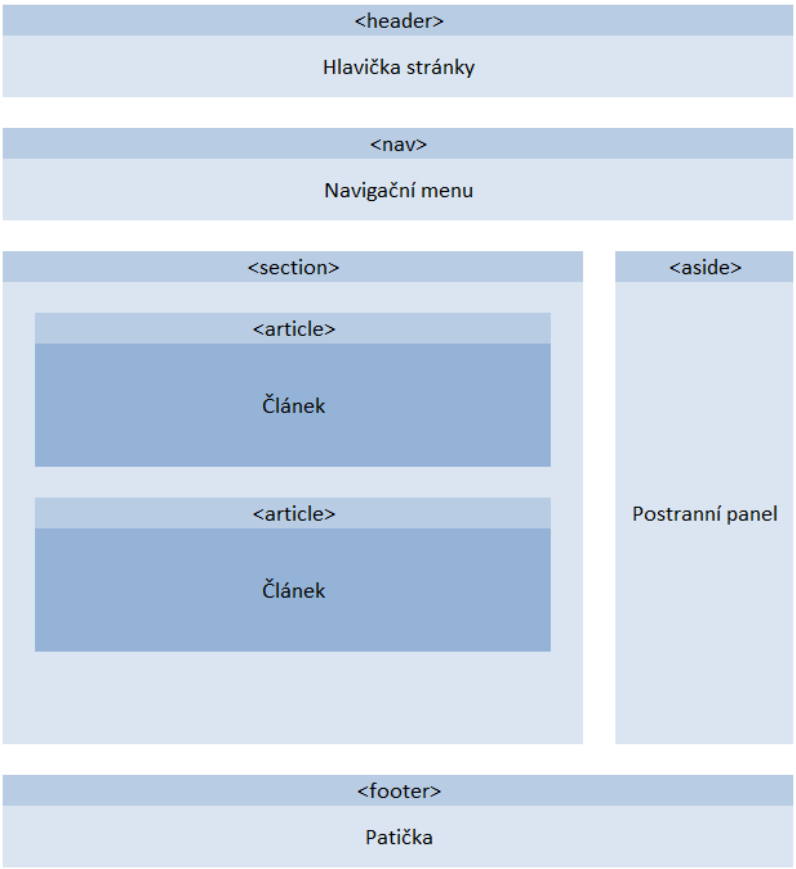


Zápis kódu pomocí `div` elementů není příliš přehledný a proto HTML5 přináší nové elementy, které mají název, podle toho k čemu jsou určené a co obsahují. Předešlý kód přepsaný do HTML5 je znázorněn níže (Příklad 6).

Příklad 6 - Stručný zápis zdrojového kódu v HTML5. Zdroj:[autor]

```
<header></header><!-- Hlavička stránky -->
    <nav></nav><!--Menu-->
<section><!-- Hlavní panel -->
    <article></article><!-- Obsahový box 1 -->
    <article></article><!-- Obsahový box 2 -->
</section>
<aside><!-- Pravý panel -->
</aside>
<footer></footer><!-- Patička stránky -->
```

Obrázek 4 - Rozložení stránky na jednotlivé sekce v HTML 5. Zdroj:[autor]



Nové označení elementů zlepšuje přehlednost kódu, jednotlivé `divy` jsou tím pádem lépe srozumitelné. Protože u některých elementů odpadá nutnost použití atributu `id`, výsledný kód je tím pádem čistší.

2.7 Podpora elementů HTML5 a CSS3

Pokud programátor použije při programování nové elementy HTML5 místo starých `div` elementů s atributem `id`, nastává u starších prohlížečů problém s jejich podporou (např. IE 8 a starší). Tento problém je možné odstranit pomocí speciálních JavaScriptových pluginů, které dovedou simulovat podporu HTML5 a CSS3. Jedním z nich je HTML5Shiv. Je to JavaScriptové řešení, vynalezené Sjoerdem Visscherem, které umožňuje zobrazení HTML5 prvků ve verzích aplikace Internet Explorer před verzí 9 [18]. Pro použití tohoto pluginu stačí vložit do elementu `head` následující kód (Obrázek 5).

Obrázek 5 - Příklad použití HTML5Shiv. Zdroj:[18].

```
<!DOCTYPE html>
<html>
 <head>
  <!--[if lt IE 9]>
  <script src="//cdnjs.cloudflare.com/ajax/libs/html5shiv/r29/html5.min.js"></script>
  <![endif]-->
 </head>
 <body>
 </body>
</html>
```

Dalším JavaScriptovým pluginem je Modernizr, který na rozdíl od HTML5Shiv je určen k detekci funkcí HTML5, ale i vlastností CSS3 v různých prohlížečích. Uživatel si na stránce projektu http://modernizr.com/download/ může nadefinovat, které vlastnosti bude chtít testovat a tak nemusí stahovat celý plugin. Po stažení souboru je nutné vložit odkaz na tento soubor do elementu `head`. Tento zápis je ilustrován v příkladu (Příklad 7).

Příklad 7 - Detekce vlastností v prohlížeči. Zdroj:[autor]

```
< script src="cesta_ke_staženému_souboru/modernizr.custom.43788.js"></script>
```

Modernizr se spustí automaticky a vytvoří globální objekt Modernizr, který pro každou detekovanou vlastnost má uloženou hodnotu `TRUE` pokud prohlížeč tuto vlastnost podporuje nebo `FALSE` pokud jí nepodporuje. Pomocí jednoduchého zápisu (Příklad 8) lze definovat požadovaná chování.

Příklad 8 - Detekce vlastností v prohlížeči. Zdroj:[autor]

```
if (Modernizr.canvas){
//Zdrojový kód, který se provede v případě, že prohlížeč podporuje vlastnost canvas.
} else {
//Zdrojový kód, který se provede v případě, že prohlížeč nepodporuje vlastnost canvas.
}
```

Pokud prohlížeč podporuje danou vlastnost, Modernizr nastaví elementu HTML do atributu `class` hodnoty odpovídající názvům daných vlastností. Jinak Modernizr před název přidá předponu "no-".

Příklad 9 - Podpora vlastností v IE9 zjištěná pomocí Modernizr. Zdroj:[10]

```
<html class="js no-flexbox canvas canvastext no-webgl no-touch geolocation draganddrop no-websockets rgba hsla multiplebgs backgroundsize no-borderimage borderradius boxshadow no-textshadow opacity no-rgba no-cssanimations no-csscolumns no-cssgradients no-cssreflections csstransforms no-csstransforms3d no-csstransitions fontface generatedcontent video audio ">
```

Ve výše uvedeném příkladu (Příklad 9) je patrné, že Internet Explorer 9 podporuje např. canvas, geolokaci nebo video a audio, ale zároveň nepodporuje zápis barev pomocí RGBA, tedy možnost nastavit jednotlivé barevné kanály včetně kanálu alfa, který udává průhlednost barvy. Pomocí hodnot atributu `class` je možné definovat vzhled webových stránek pouze v případě, že dané vlastnosti jsou v prohlížečích podporovány. Tento zápis by mohl vypadat následovně (Příklad 10).

Příklad 10 – Zápis CSS stylu při podporování a nepodporování rgba. Zdroj:[autor]

```
.no-rgba { color: #3C3C3C; // První CSS pravidlo pro prohlížeče, které NEPODPORUJÍ rgba.
}
.rgba{color: rgba(60,60,60,0.9); // Druhé CSS pravidlo pro prohlížeče podporující rgba.
}
```

Jestliže prohlížeč nepodporuje RGBA, tak atribut `class` obsahuje třídu `no-rgba` a na prvek se aplikuje první CSS pravidlo. Pokud prohlížeč RGBA podporuje, aplikuje se druhé pravidlo (Příklad 10).

Každý by však měl myslet i na případ, že uživatel nebude mít zapnutou podporu JavaScriptu v prohlížeči, protože v tomto případě nebude knihovna Modernizr fungovat. Tvůrci Modernizru na tento problém mysleli a tak stačí do úvodního tagu HTML vložit do atributu `class` hodnotu `no-js` (Příklad 11).

Příklad 11 – Zápis hodnoty no-js do atributu class. Zdroj:[autor]

```
<html class = “no-js“>
```

Tento stav zachycuje standardní nastavení bez zapnutého JavaScriptu. V případě, že uživatel bude mít zapnutý JavaScript, Modernizr sám přepíše hodnotu `no-js` na hodnotu `js`.

2.7.1 Alternativní řešení

Jinou možností než je knihovna Modernizr, která je spíše vhodnější pro větší HTML5 a CSS3 projekty, je u menších projektů, využívajících jen některé vlastnosti CSS3, alternativní způsob zápisu (Příklad 12) CSS stylu. Ten se aplikuje podle podpory daného prohlížeče.

Příklad 12 – Alternativní způsob zápisu CSS stylu. Zdroj:[autor]

```
.rgba { // První zápis
        color: #3C3C3C;
        // Druhý zápis
        color: rgba(60,60,60,0.9);
        }
```

Druhému zápisu (Příklad 12) starší prohlížeče nerozumí, proto jej ignorují a nastaví barvu podle prvního zápisu. Kdežto novější prohlížeče, které tomuto zápisu rozumí, a jelikož ho dostaly jako druhý, přenastaví barvu podle tohoto zápisu.

2.8 Funkce HTML5 a související technologie

2.8.1 Nové HTML5 elementy

Internetové stránky se v HTML4 skládají z různých elementů, jako jsou např. `<div>`, `<p>`, `<table>`, `<ul>`, `<li>` a jiné. Jestliže se chce vývojář odkázat na určitý element v JavaScriptovém kódu nebo v CSS stylech, stačí mu přidat k danému elementu atribut `id` nebo `class` s danou hodnotu, např. `<div id="header"></div>` nebo `<table class="calendar_table"></table>`. Na rozdíl od člověka, který tomuto zápisu rozumí a umí si představit, co bude obsahem daného elementu, počítače a zejména vyhledávací roboty tomuto zápisu nerozumí. Z tohoto důvodu navrhli vývojáři HTML5 nové elementy, u kterých je vidět jejich význam na první pohled. Tyto elementy jsou srozumitelnější a přehlednější nejen pro webové vývojáře a designery, ale i pro vyhledávací roboty. V následující tabulce (Tabulka 1) jsou uvedeny některé elementy a jejich význam.

Tabulka 1 – Nové HTML5 elementy. Zdroj:[autor]

Název elementu	Význam
<header>	Hlavička stránky
<footer>	Zápatí stránky
<nav>	Navigace webu
<article>	Hlavní obsah
<section>	Jednotlivé sekce, může obsahovat další <article>, <header> nebo <footer>
<aside>	Související obsah
<hgroup>	Seskupení nadpisů

V průběhu vývoje HTML5, editor Ian Hickson za použití nástrojů společnosti Google provedl průzkum jaké názvy `id` a `class` jsou nejpoužívanější. Nejčastější názvy `id` a `class` byli použity jako samotné názvy jednotlivých elementů.

2.8.2 Rozhraní Canvas

Princip plátna jako první představila společnost Apple ve svém WebKitu pro Mac OS X pro vytvoření widgetů pracovní plochy. Než bylo vytvořeno plátno, realizovalo se malování v prohlížečích pomocí zásuvných modulů, např. Flash, SVG nebo VML.

Co je to vlastně plátno? Plátno je obdélníková oblast vytvořená pomocí elementu `canvas`. Ve výchozím stavu je 300 px široká a 150 px vysoká. Velikost stejně jako jiné atributy elementu `canvas` je možné libovolně měnit. Příklad (Příklad 13) znázorňuje nejzákladnější zápis elementu `canvas` do stránky.

Příklad 13 – Základní použití elementu canvas. Zdroj:[4]

```
<canvas> </canvas>
```

Pokud se do stránky přidá element `canvas` může se pomocí JavaScriptů do něj přidat grafika, čáry, texty, vkládat pokročilé animace a je možné v něm malovat. Aby bylo možné začít do plátna malovat, je nutné přidat elementu `canvas` atribut `ID`, k němuž můžeme přistupovat pomocí nativní JavaScriptové funkce `dokument.getElementById()`. Dále je nutné na tomto plátně zavolat metodu `getContext()`, která vrátí daný kontext plátna, kde je možné aplikovat další metody pro vykreslení nejrůznějších grafických prvků. V příkladu (Příklad 14) je znázorněn grafický přechod barev [4].

Příklad 14 – Kód pro vykreslení barevného přechodu v canvasu. Zdroj:[autor]

```
<canvas id="myCanvas" width="300" height="150" style="border:2px solid #d3d3d3;">
</canvas>

<script>
var canvas = document.getElementById("myCanvas");
var context = canvas.getContext("2d");

var gradient = context.createLinearGradient(0, 0, 300, 0);
gradient.addColorStop(0, "#336699");
gradient.addColorStop(1, "#FFFFFF");

context.fillStyle = gradient;
context.fillRect(0, 0, 300, 300);
</script>
```

Výsledek jednoduchého kódu ilustruje Obrázek 6.

Obrázek 6 – Vykreslený obrázek v canvasu. Zdroj:[autor]

2.8.3 Audio a Video

V této kapitole se seznámíme s dalšími novými elementy HTML5, `audio` a `video`. Namísto dřívějších verzí HTML kde se k přehrávání audia a videa využívaly zásuvné moduly, (např. Flash, QuickTime a Windows Media) usnadňují nové elementy život jak vývojářům, tak uživatelům. I když se některé moduly těší velké oblibě, někteří uživatelé je deaktivují z důvodu častého obtěžování všudypřítomnou reklamou, i když si tím znemožní přehrávání médií. Elementy `audio` a `video` tyto překážky odstraňují díky tomu, že jsou součástí prostředí prohlížeče [4].

Multimediální soubory jsou ve skutečnosti pouze kontejnery podobně jako Zip. Protože je jejich velikost v původním stavu enormní, nebylo by možné je z důvodu velkého množství dat v přiměřeném čase přenášet po internetu, proto je nutné je pomocí kodeku zmenšit. Cílový přehrávač musí obsahovat správný dekodér, aby byl schopen zakódovanou podobu multimediálního souboru přehrát. Mezi nejznámější audiokodeky patří např. AAC (.aac), MPEG-3 (.mp3), OggVorbis (.ogg). V závorkách jsou uvedeny formáty souborů používající daný kodek. Následující tabulka (Tabulka 2) znázorňuje podporované kodeky v jednotlivých prohlížečích [4].

Tabulka 2 – Přehled podpory kodeků elementu audio v prohlížečích. Zdroj:[autor]

Prohlížeč	MP3	WAV	Ogg	AAC
Internet Explorer	Ano	Ne	Ne	Ano
Chrome	Ano	Ano	Ano	Ano
Firefox	Ano	Ano	Ano	Ano
Safari	Ano	Ano	Ne	Ano
Opera	Ano	Ano	Ano	Ano

Zápis elementu `audio` se liší podle toho, v kolika formátech je daný soubor k dispozici. Kód pro soubor v jednom formátu je znázorněn v následujícím příkladu (Příklad 15).

Příklad 15 – Použití elementu audio s jedním zdrojem. Zdroj:[autor]

```
<audio controls src="Metallica_The_Unforgiven.ogg" type="audio/ogg"></audio>
```

Aby se předešlo problému, že některý prohlížeč daný kodek nepodporuje, jako u výše uvedeného příkladu (Příklad 15), není podporován Internet Explorerem, je nutné použít element `source` a vložit více souborů, aby si přehrávač vybral ten, ke kterému má odpovídající kodek (Příklad 16).

Příklad 16 – Použití elementu audio s více zdroji. Zdroj:[autor]

```
<audio controls>
        <source src="Metallica_The_Unforgiven.ogg" type="audio/ogg">
        <source src="Metallica_The_Unforgiven.mp3" type="audio/mpeg">
</audio>
```

Atribut `controls` slouží k zobrazení ovládacích prvků, např. tlačítka play, stop, posun ve skladbě, ovládání hlasitosti, které můžeme vidět na následujícím Obrázku 7.

Obrázek 7 – Zobrazení elementu audio v prohlížeči Opera (vlevo) a IE (vpravo) Zdroj:[autor]

Podobně jako u elementu `audio` i elementu `video` je rozdílná podpora kodeků ze strany prohlížečů. Přehled podpory jednotlivých kodeků v prohlížečích je znázorněn v následující tabulce (Tabulka 3).

Tabulka 3 – Přehled podpory kodeků elementu video v prohlížečích. Zdroj:[autor]

Prohlížeč	MP4	WebM	Ogg
Internet Explorer	Ano	Ne	Ne
Chrome	Ano	Ano	Ano
Firefox	Ano	Ano	Ano
Safari	Ano	Ne	Ne
Opera	Ano	Ano	Ano

Nejen rozdílnost podpory kodeků je zde velmi podobná, ale i samotný zápis elementu `video` je velmi podobný jak zachycuje následující příklad (Příklad 17).

Příklad 18 – Použití elementu video s jedním zdrojem. Zdroj:[autor]

```
<video controls src="Metallica_The_Unforgiven_videoklip.ogg" type="video/ogg"></video>
```

I zde je lepší použít video ve více formátech s použitím elementu `source` (Příklad 18).

Příklad 17 – Použití elementu video s více zdroji. Zdroj:[autor]

```
<video controls poster=“obrazek.jpg“>
        <source src="Metallica_The_Unforgiven_videoklip.ogg" type="video/ogg">
        <source src="Metallica_The_Unforgiven_videoklip.mp4" type="video/mp4">
</video>
```

Atribut `poster` zapsaný v příkladu (Příklad 18) slouží k zobrazení obrázku, který reprezentuje obsah videa před jeho spuštěním. Názorný příklad, jak může vypadat element `video` s atributem `poster`, je zachycen v obrázku (Obrázek 8).

Obrázek 8 – Zobrazení elementu video v prohlížeči Opera s použitím atributu poster. Zdroj:[autor]

2.8.4 Geolokační rozhraní

Další funkcí, kterou je možné využít při tvorbě webových aplikací je geolokační rozhraní HTML5. Jeho použití není nijak složité. Aplikace si od uživatele vyžádá jeho aktuální polohu a dá-li k tomu uživatel souhlas, prohlížeč vrátí informace o poloze. Polohu prohlížeči sdělí zařízení, např. notebook, tablet nebo mobilní telefon, na kterém prohlížeč, který podporuje geolokační rozhraní HTML5, běží. Polohu udává zeměpisná šířka (číselná hodnota udávající vzdálenost severně či jižně od rovníku) a zeměpisná délka (číselná hodnota udávající vzdálenost východně či západně od nultého poledníku v Greenwichi v Anglii). Souřadnice zeměpisné šířky a délky mohou být zapsány dvěma způsoby.

- **Desetinný formát** (50.163, 16.280)
- **Formát DMS** (50° 9' 50", 16° 16' 48")

Geolokační rozhraní HTML5 vždy používá pouze desetinný formát souřadnic [4].

Jak může geolokační rozhraní zpříjemnit konečnému uživateli život? Snadno, například pokud uživatel hledá nejbližší kino, nenabídne mu aplikace seznam všech kin v celé republice, ale díky aktuální poloze uživatele, mu nabídne pouze seznam kin v okolí, podle toho jaké parametry uživatel zadá (např. rozsah km, jednotlivé kraje nebo přímo město). Kromě zeměpisné šířky a délky udává geolokační rozhraní také informace o přesnosti uvedených souřadnic, nadmořskou výšku, směr a rychlost pohybu. Údaje o směru a rychlosti využívají zejména sportovní aplikace určené pro mobilní zařízení, které uživateli poskytnou informace, o vzdálenosti kterou urazil a za jak dlouhou dobu, přičemž z těchto údajů jsou schopné vyhodnotit např. přibližnou spotřebu kalorií, průměrnou a maximální rychlost nebo dobu trvání.

2.8.4.1 Metody zjištění polohy

Uživatel není schopen rozhodnout o tom, jakou metodou má zařízení určit jeho polohu, protože způsob pro získání informací o poloze si zvolí samotné zařízení. Mnohá zařízení používají kombinaci jednoho či více zdrojů polohy za účelem dosažení co největší přesnosti. Jednotlivé metody a jejich výhody a nevýhody jsou popsány v následující tabulce (Tabulka 4).

Tabulka 4 – Přehled metod zjištění polohy. Zdroj:[4]

Metody	Pro	Proti
IP Adresa	Dostupná kdekoli	Ne příliš přesná (často uvádí zcela mylné informace, jinak přesná pouze na úrovni měst)
	Zpracování na straně serveru	Může se jednat o nákladnou operaci
GPS	Velmi přesná	Fixace může trvat dlouhou dobu, což může vyčerpat baterie zařízení uživatele
		Nefunguje v budovách
		Může vyžadovat dodatečný hardware
Wi-Fi	Přesná	Nepříliš dobrá v oblastech mimo město s minimem přístupových bodů
	Funguje v budovách	
	Rychlé a nenákladné určení polohy	
Mobilní síť	Poměrně přesná	Vyžaduje zařízení s přístupem k mobilní síti jako je mobilní telefon nebo modem
	Funguje v budovách	Nepřesná v oblastech mimo město s nižším počtem základových stanic
	Rychlé a nenákladné určení polohy	
Ručně zadané informace	Uživatelé mohou mít přesnější informace o své poloze, než jaké mohou poskytnout jiné služby	Může být velmi nepřesná, obzvláště pokud se poloha změnila
	Umožňuje lokaci i v jinak nedosažitelných místech	
	Zadání uživatelem může být rychlejší než detekce	

2.8.5 Offline režim aplikace

V současné době většina uživatelů využívá přístup k Internetu nejen doma na počítači, ale i na mobilních zařízeních mimo domácí síť. Nadále existují místa, kde internetové připojení není k dispozici (např. v letadle, ve vlaku a podobně). V těchto případech musí uživatelé využívající zařízení s přístupem na internet přerušit svou činnost. Pomocí offline režimu aplikace HTML5 je možné prohlížet si již dříve navštívené stránky. Toto je možné díky tomu, že si prohlížeč při první návštěvě uloží obsah stránek do paměti cache a následně z ní čerpá. Dalším využitím offline režimu je práce s online seznamy úkolů či procházení doručených i odeslaných e-mailů a pomocí JavaScriptu je mazat psát a odesílat. Následné odeslání e-mailů se uskuteční až po připojení k Internetu [14].

Offline webové aplikace využívají k ukládání důležitých souborů pro fungování aplikací aplikační cache. Aby aplikace fungovala v offline režimu musí být nejdříve vytvořen soubor manifest, kde je uveden seznam souborů aplikací, které musí existovat pro případ práce ve stavu offline, např. offline.manifest. Soubor manifest se skládá ze tří částí, `cache`, `network` a `fallback`. Obsah tohoto souboru a popis jednotlivých částí je představen v následujícím příkladu (Příklad 19).

Příklad 19 – Obsah souboru manifest. Zdroj:[14]

```
#komentáře se uvozují symbolem #
CACHE MANIFEST
#soubor o řádek níže je přiřazen sekci CACHE
 page.html

NETWORK:
#Soubory využívané výhradně v online režimu.
#Neukládají se do aplikační cache.
 imageA.png
 styleA.css
 scriptA.js

CACHE:
#Zde jsou soubory, které nejsou (ale mohou být)
#v online režimu použity, přesto je
#prohlížeč poslušně uloží do aplikační cache.
 imageB.png
 styleB.css
 scriptB.js

FALLBACK:
#Pokud soubory imageA.png, styleA.css a scriptA.js
#nelze načíst ze serveru a nejsou ani v aplikační
#cache, musí se PO UPLYNUTÍ LHŮTY pro načtení
#původních souborů použít ty alternativní.
 imageA.png imageB.png
 styleA.css styleB.css
```

Nakonec je ještě nutné připojit soubor manifest jako atribut k webové stránce přes element html.

Příklad 20 – Připojení souboru manifestu k webové stránce. Zdoj:[14]

```
<!DOCTYPE html>
<html manifest = “offline.manifest”>
```

Soubor může mít různé přípony (např. `*.manifest` nebo `*.appcache`), ale musí mít správný MIME TYPE `text/cache-manifest`. Zajištění správného typu MIME-TYPE se provádí pomocí souboru `.htaccess`, který obsahuje jeden řádek kódu (Příklad 21).

Příklad 21 – Ukázka souboru .htacces. Zdroj:[autor]

```
AddType text/cache-manifest .manifest
```

Pomocí vlastnosti `applicationCache.status` lze zjistit stav aplikační cache pro aktivní dokument. Numerické hodnoty, které vrací vlastnost status, jsou v následující tabulce (Tabulka 5) [14].

Tabulka 5 – Numerické hodnoty vlastnosti status. Zdroj:[14]

Hodnota	Popis hodnoty
0 UNCACHED	Stránka nemá přiřazen manifest, nebo ještě nebyla načtena
1 IDLE	Aplikační cache je aktuální a kompletní
2 CHECKING	Prohlížeč posuzuje aktuálnost manifestu
3 DOWNLOADING	Stahování nového obsahu do aplikační cache
4 UPDATEREADY	Připraveno pro update
5 OBSOLETE	Manifest nenalezen, aplikační cache bude vyprázdněna

2.8.6 **Lokální úložiště dat**

Před seznámením s rozhraním Web Storage je nejepší začít s jeho předchůdcem soubory Cookies. Tyto soubory umožňují předávat krátké textové hodnoty mezi klientem a serverem. Nevýhodou je jejich malá velikost, která zpravidla činí nanejvýš 4KB dat.

Stejného výsledku je možné docílit i bez účasti serveru a síťové komunikace. Právě toto je silná stránka webového úložiště Web Storage, které se dělí na dva různé typy úložišť. První typ se nazývá localStorage a druhý sessionStorage. Oba typy jsou stejné, pouze se liší doba, po jakou se data uchovávají. LocalStorage uchovává data, dokud nejsou smazána skriptem. Proti tomu sessionStorage uchovává data pouze po dobu trvání relace, tedy po dobu otevření dané stránky. Jakmile uživatel danou stránku zavře, uložená data se vymažou. Rozhraní Web Storage umožňuje ukládat data o velikosti až několika megabajtů. Především velikost činí z toho rozhraní ideální úložiště pro dokumenty a soubory [4].

Pro práci s daty slouží několik metod, které jsou definovány v rozhraní Storage. Jedná se o:

- `setItem(key, value)` – uloží novou položku s klíčem `key` a hodnotou `value` do úložiště, pokud již existuje položka se stejným klíčem, přepíše se hodnota na novou
- `getItem(key)` – slouží k načtení položky s klíčem `key` a pokud ještě položka není, vrátí `null`
- `removeItem(key)` – smaže položku s klíčem `key` z úložiště
- `clear()` – slouží k smazání celého úložiště
- `key(index)` – slouží k získání hodnoty `key` dané položky

Klíče jsou číslovány od nuly, to znamená, že první klíč zapíšeme s indexem nula a poslední klíč má index délka–1. Klíče si své indexy zachovávají v rámci úložiště až do doby odstranění některého z klíčů. Praktické použití všech metod je ilustrováno v následujícím příkladu (Příklad 22). V příkladu je použita i kontrola, jestli prohlížeč podporuje `localStorage`.

Příklad 22 – Ukázka použití metod rozhraní Storage. Zdroj:[autor]

```
if(window.localStorage){
        localStorage.setItem(„znacka_1“, „Suzuki“);
        localStorage[„model_1“] = „Swift“;
        var znacka = localStorage.getItem(„znacka_1“);
        var model = localStorage[„model_1“];
        alert(„Značka vozu je “ + znacka + „ a model je „ + model + „.“);
        for(var i = 0; i < localStorage.length; i++){
                alert(localStorage.key(i));
        }
        localStorage.clear();
}else{
        alert („Váš prohlížeč nepodporuje lokální úložiště. Použijte novější prohlížeč.“
}
```

Na začátku příkladu (Příklad 22) jsou nejprve vytvořeny dvě položky, které jsou dále načteny do proměnných a vypsány na obrazovku. Poté jsou pomocí cyklu za použití vlastnosti `length` vypsány jednotlivé klíče pomocí metody `key()`. Na závěr jsou všechny položky smazány. V příkladu jsou uvedeny oba způsoby pro vytváření hodnot v úložišti [2].

Kromě jednoduchých párových řetězců jde do `localStorage` ukládat i pole hodnot. Při ukládání je nutné pole převést na řetězec. Toho docílíme pomocí metody `JSON.stringify()`. Aby bylo možné získat uloženou hodnotu je nutné nejdříve načíst uložený řetězec a poté převést zpět do podoby pole za pomoci metody `JSON.parse()`. V příkladu (Příklad 23) je znázorněn celý proces.

Příklad 23 – Uložení pole do úložiště pomocí JSON. Zdroj:[autor]

```
var auto = {"znacka" : "Suzuki", "model" : "Swift"};
localStorage.setItem("vozidlo", JSON.stringify(auto));
var retezec = localStorage.getItem("vozidlo");
var pole = JSON.parse(retezec);
alert(„Značka vozu je “ + pole.znacka + „ a model je „ + pole.model + „.“);
```

2.8.7 Framework jQuery Mobile

jQuery Mobile je framework, určený pro tvorbu responsivních webových stránek a aplikací, využívající HTML5 a CSS3 vlastností. Framework je kompatibilní se všemi populárními mobilními platformami, ale i s většinou desktopových prohlížečů. Kompletní výčet podporovaných platforem je uveden na stránkách jQuery Mobile [8]. Framework jQuery Mobile je založen na jQuery a jQuery UI, dále přináší výhodu lehce přizpůsobitelného a skinovatelného designu pomocí online nástroje ThemeRoller dostupného na stránce *http://themeroller.jquerymobile.com* [5].

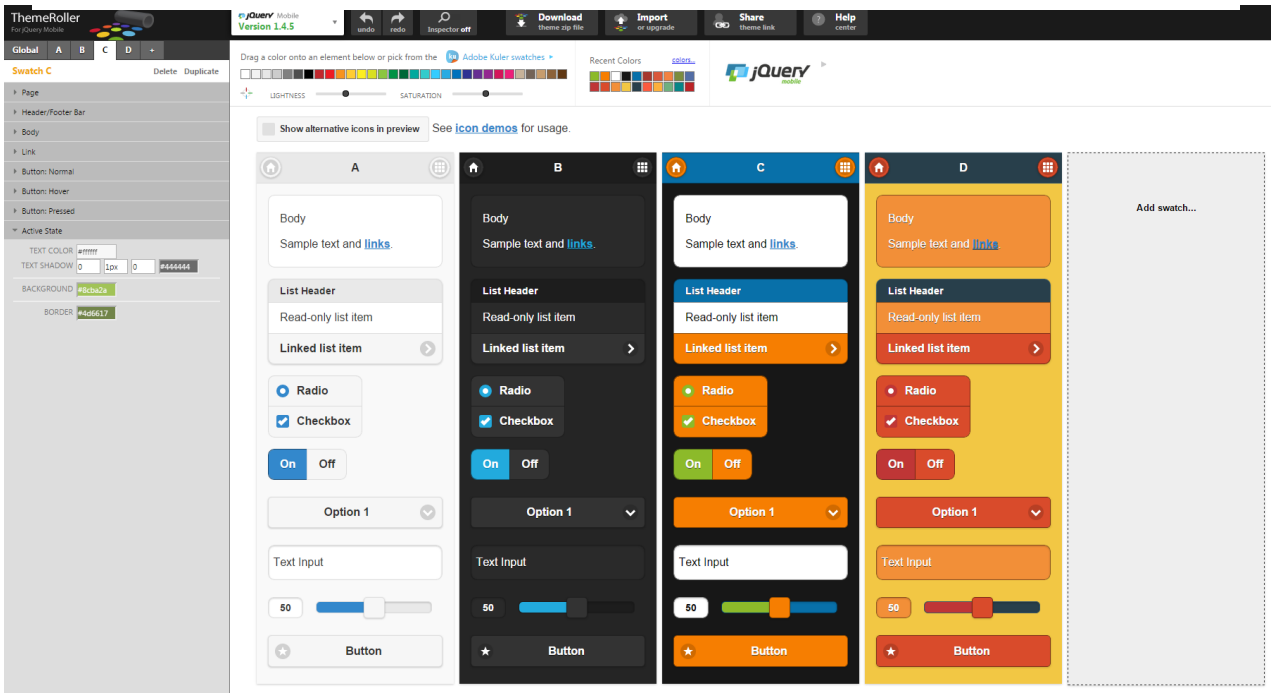
Práce s frameworkem je jednoduchá a intuitivní. Při práci s frameworkem není potřeba znát JavaScript nebo jQuery. jQuery Mobile knihovnu jQuery využívá pro své vlastní potřeby. Pomocí konkrétně zvolených `data-` atributů, se určí, jak bude s daným elementem nakládáno. Jedním ze seznamu `data-` atributů je atribut data-role, který určuje, jakou roli bude daný element zastupovat. Hodnota tohoto atributu může být například `header` (definice hlavičky), `content` (hlavní část), `footer` (patička aplikace), které rozdělí aplikaci na logické prvky stránky, nebo třeba `page`, který definuje jednu stránku aplikace. Na rozdíl od klasických statických stránek, kdy každá stránka je uložena ve vlastním html souboru a je provázána s ostatními za pomoci odkazů, tak v jQuery Mobile jsou všechny stránky uloženy v jednom html souboru a jsou rozděleny právě pomocí atributu `data-role="page"`. Při spuštění aplikace se zobrazí pouze první element s tímto atributem. Navigace mezi jednotlivými stránkami se potom provádí pomocí unikátního `id`, které zajistí provázanost mezi jednotlivými elementy. Například po kliknutí na následující odkaz (Příklad 24), se načte nový element s atributem `data-role="page"` a hodnotou `id „id-stranky"`.

Příklad 24 – Ukázka zdrojového kódu odkazu. Zdroj:[autor]

```
<a href="#id-stranky">Odkaz na jinou stránku aplikace</a>
```

Framework obsahuje i kaskádový styl, ve kterém jsou definovány dvě možné barevné varianty. Pro nastavení konkrétní varianty stačí danému elementu přidat atribut `data-theme` s hodnotou `a` nebo `b`, asi jedinou výjimku tvoří tlačítka, kterým se místo atribut `data-theme` musí přidat CSS třída `ui-btn-` a písmeno `a` nebo `b`, které značí barevnou variantu. Obě varianty lze libovolně upravit pomocí, již výše zmíněného ThemeRolleru nebo si k současným variantám vytvořit vlastní. ThemeRoller umožňuje vytvořit až 26 barevných variant označených písmeny A-Z.

Obrázek 9 – Prostředí nástroje ThemeRoller obsahující základní barevné varianty A, B a nové varianty C a D. Zdroj:[autor]

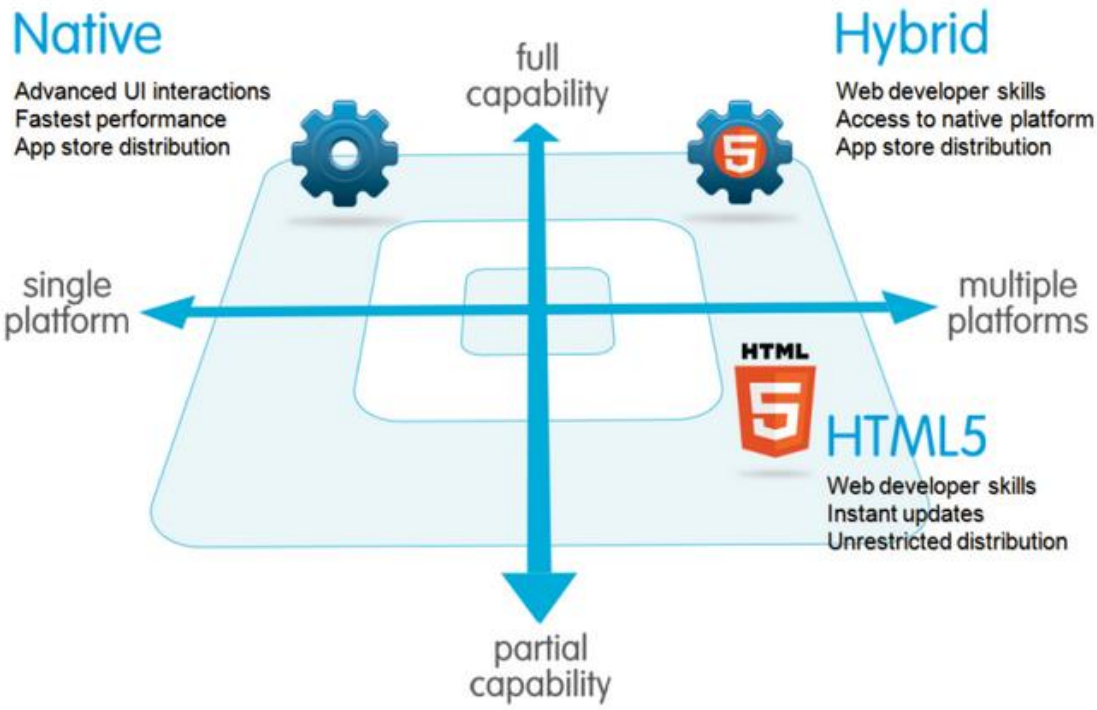


Po vytvoření schémat je možné si je stáhnout v novém css souboru, který je potřeba poté připojit v hlavičce stránky. Prostředí nástroje ThemeRoller je zobrazeno na následujícím obrázku (Obrázek 9).

3 Způsoby vývoje aplikací

Existují tři způsoby, jakými je možné přistupovat k vývoji mobilní aplikace. Jedná se o způsoby nativní, hybridní a HTML5. Každý způsob má své výhody i nevýhody, podle kterých se programátor nakonec rozhodne, jaký způsob použije. Existuje mnoho rozhodovacích faktorů např. vývojářské dovednosti, požadovaná funkcionalita zařízení, míra zabezpečení, možnosti offline režimu [9]. Na následujícím obrázku (Obrázek 10) jsou pomocí grafu názorně zobrazeny jednotlivé typy aplikací. Podél horizontální osy je znázorněna míra kompatibility mezi různými platformami. Podél vertikální osy je znázorněna míra využití hardwarových možností zařízení.

Obrázek 10 – Zobrazení aplikací v grafu. Zdroj:[9]



3.1 Nativní aplikace

Nativní aplikace jsou vyvíjeny v přirozeném kódu dané platformy. Například pro Android je to jazyk Java nebo pro platformu iOS je to jazyk Objective-C. Přehled jednotlivých platforem je uveden v následující tabulce (Tabulka 6).

Tabulka 6 – Přehled platforem. Zdroj:[11]

				
Jazyky	Obj-C, C, C++	Java	Java	C#, VB.NET
Nástroje	Xcode	Android SDK	BB Java Eclipse Plug-in	Visual Studio, Windows Phone Dev Tools
Soubory	.app	.apk	.cod	.xap
Obchody	App Store	Google Play	BlackBerry App World	Windows Phone Store

Tyto aplikace jsou díky své architektuře a uložení přímo v zařízení rychlé a umí využívat hardwarových schopností telefonu, jako jsou například vícedotykové ovládání, rychlejší vykreslování grafických prvků (u her), kamera nebo GPS. Jednou z největších výhod je uložení aplikace přímo v zařízení a tím její nezávislost na internetovém připojení. Aplikaci si může uživatel stáhnout z obchodu aplikací dané platformy (Tabulka 6). Než je možné aplikaci stáhnout z obchodu, musí aplikace projít procesem schválení. Nativní aplikace přináší i několik nevýhod. Jednou z nich je omezení aplikace pouze na jednu určitou platformu, pokud bude potřeba, aby aplikace běžela na více platformách je nutné ji celou znovu napsat v daném jazyce platformy. Další nevýhodou je náročnější správa a distribuce dané aplikace, pokud dojde v aplikaci k nějaké změně je nutné novou verzi nahrát do obchodu a uživatel si musí stáhnout aktualizaci [12].

3.2 HTML5 neboli webová aplikace

Webová aplikace je tvořena pomocí webových technologií jako je například HTML, JavaScript a CSS. Webová aplikace je v podstatě webová stránka, kterou lze otevřít pomocí všech moderních prohlížečů, nezávisle na typu zařízení či typu platformy což je její nespornou výhodou. Další velkou výhodou je snadná správa a distribuce. Pokud je nutné aplikaci aktualizovat, stačí provést změnu zdrojových souborů na serveru a změna se projeví ihned všem uživatelům.

Nevýhody této aplikace jsou například závislost na internetovém připojení, i když dnes je již možné napsat web, který bude fungovat offline, částečný přístup k hardwarovým schopnostem zařízení a omezený přístup k aplikaci pouze přes webový prohlížeč. Jelikož se tedy jedná v podstatě o web je její používání zcela zdarma [12].

3.3 Hybridní aplikace

Jak jejich název napovídá, jsou hybridní aplikace kombinací výhod (ale i nevýhod) výše zmíněných typů aplikací. Jejich kód je napsán pomocí HTML5, JavaScriptu a CSS, následně je obalen nativní slupkou pomocí emulátoru, která umožní přístup k hardwarovým schopnostem zařízení. Jedním z nejpoužívanějších emulátorů je PhoneGap.

Jednotlivé typy aplikace mají své výhody a nevýhody, tudíž nelze jednoznačně určit, která z nich je nejlepší. V následující tabulce jsou uvedeny základní vlastnosti jednotlivých typů vývoje aplikací (Tabulka 7).

Tabulka 7 – Vlastnosti jednotlivých typů vývoje. Zdroj:[6]

Vlastnost	Aplikace		
	Nativní	HTML5	Hybridní
Jazyk	Obj-C, Java, C#, atd.	HTML, JS, CSS	HTML, JS, CSS
Vývojové nástroje	Xcode, Android SDK, Eclipse, Visual Studio…	Oblíbený textový editor	Phonegap + nativní SDK
Distribuce	Nativní obchod aplikací	-	Nativní obchod aplikací
Schválení	Vyžadováno	Není vyžadováno	Vyžadováno
Zpeněžení	Přes obchod aplikací	-	Přes obchod aplikací

4 Návrh uživatelského rozhraní

Co je to uživatelské rozhraní? Je to věc, která se uživateli zobrazí po spuštění aplikace a umožňuje mu s ní komunikovat. Místo názvu se častěji uvádí označení UI (User Interface) nebo GUI (Graphical User Interface). Ve webových aplikacích se jedná především o grafické či textové prvky a jejich rozmístění. Tyto ovládací prvky umožňují uživateli práci s aplikací, získávají od něho vstupní data, na která následně reagují a zpracované výsledky pak prezentují uživateli. Moderní uživatelská rozhraní často využívají nástrojů, které jim přináší nové technologie tvorby webových stránek a aplikací založených např. na AJAXu, možností HTML5 a různých frameworcích, které usnadňují a urychlují tvorbu designu aplikace. Jedním z hlavních rysů moderních grafických uživatelských rozhraní je oddělení obsahu a grafické interpretace. Toto má pozitivní vliv především při větších úpravách aplikace, aktualizacích, ale i při vytváření nových uživatelských rozhraní pro jiné platformy. Aplikace je potom z větší části zachována, není potřeba do ní zasahovat a jen se rozšíří o další uživatelské rozhraní, což nejen uspoří čas, ale hlavně i rozpočet. Grafické uživatelské rozhraní hraje zásadní roli v tom, jak bude uživatel vnímat celou aplikaci. Správně navržené uživatelské rozhraní určuje, jak budou uživatelé aplikaci vnímat, jakým způsobem s ní budou pracovat, zda s ní budou spokojeni a budou ji pravidelně používat.

Tento způsob navrhování aplikace se zaměřením na uživatele je označován jako User Experience Design (UXD) nebo také User Centered Design (UCD). Uživatelské rozhraní je při metodice UXD navrhováno s ohledem na design, jehož cílem je co nejúčelnější propojení funkční a estetické složky navrhované aplikace a použitelnost, jejímž cílem je odstraňování nedostatků v ovládání a vylepšování na základě uživatelských zkušeností s aplikací.

Návrh kvalitního a funkčního uživatelského rozhraní aplikace není jednoduchá záležitost, nejedná se totiž pouze o vizuální vzhled a rozmístění prvků na stránce. Aby byla aplikace úspěšná, je nutný komplexní pohled na celý proces vývoje už od samotného počátku návrhu aplikace. UXD je primárně zaměřen na uživatele a jeho zážitek z používání aplikace.

Zahrnuje v sobě několik disciplín, které by měli zajistit následující vlastnosti dobré aplikace:

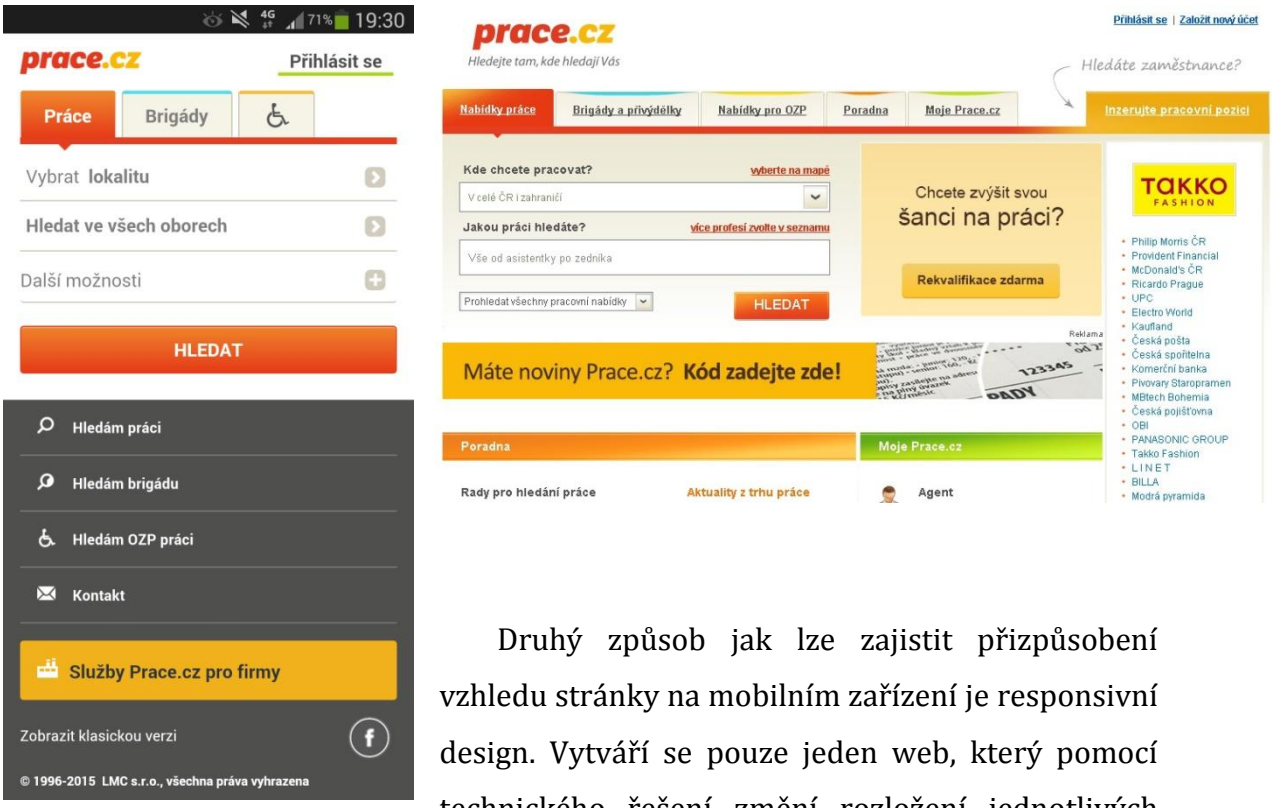
- **Funkčnost a přínos aplikace pro uživatele** – hlavní kritérium použitelnosti
- **Efektivita** – rychlost a časová náročnost na vykonání dílčího úkolu
- **Jednoduché a intuitivní rozhraní** – přizpůsobení aplikace uživatelům
- **Jednoduchost zapamatování** – nenutit uživatele přemýšlet

Celý vývoj uživatelského rozhraní se skládá z několika stádií, které na sebe navazují a staví jedno na druhém. Návrh začíná v abstraktní rovině, pomalu se stává konkrétnějším až po definitivní podobu aplikace a jejích GUI prvků [15].

4.1 Optimalizace vzhledu stránky

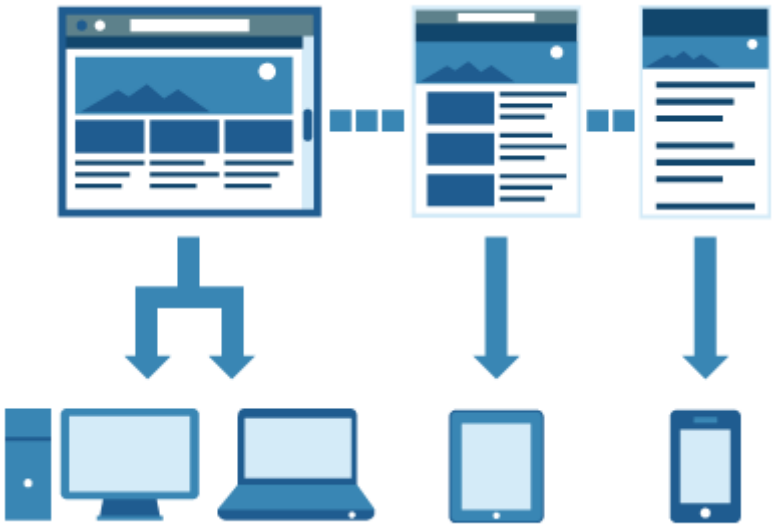
V současné době roste počet zařízení, se kterými se lze připojit na internet. Bohužel velké množství webů s tímto nepočítá a jejich ovládání na různých zařízeních je složité. Řešením tohoto problému je automatické přizpůsobení vzhledu stránky podle daného zařízení. Jedním ze způsobů, jak toho může být docíleno je vytvoření mobilní verze stránky. V případě mobilního webu se vytváří dvě oddělené verze webu, které mají vlastní nakódované šablony, styly a další zdroje. Pokud dojde ke změně na jednom webu, je nutné provést změnu i na druhé verzi. Každý web je optimalizovaný pro určitý druh zařízení. Toto řešení je časově náročné, jelikož se musí vytvořit dvě verze stránek se stejným obsahem, ale upraveným vzhledem. Jakmile uživatel zadá adresu stránky na mobilním zařízení automaticky je přesměrován na mobilní verzi (např. www.prace.cz a mobilní verze je www.m.prace.cz). Obě varianty tohoto webu jsou zobrazeny níže (Obrázek 11).

Obrázek 11 – Mobilní a desktopová verze webu práce.cz. Zdroj:[autor]



Druhý způsob jak lze zajistit přizpůsobení vzhledu stránky na mobilním zařízení je responsivní design. Vytváří se pouze jeden web, který pomocí technického řešení změní rozložení jednotlivých prvků a velikost obrázků podle možnosti daného zařízení. Vše je provedeno až v samotném cílovém zařízení. Vše je otázkou pouze kaskádových stylů a CSS3 Media Queries. Chování responsivního webu je ilustrován na následujícím obrázku (Obrázek 12).

Obrázek 12 – Znázornění responsivního webu. Zdroj:[13]



5 Výsledky práce

5.1 Popis aplikace

V podkapitole **Funkce HTML5 a související technologie** byly popsány některé funkce HTML5, které mohou z jednoduché stránky vytvořit mobilní a uživatelsky příjemný web. Pro předvedení některých z těchto prvků byla zvolena webová aplikace pro zaznamenávání úkolů, jež dostala název TaskList. Aplikace je určena všem uživatelům, kteří si nepamatují svoje úkoly, nebo si je nechtějí zapisovat na papír, ale chtějí je mít pořád po ruce, např. ve svém mobilním telefonu nebo tabletu.

Jádrem aplikace je ukládání informací, zadaných uživatelem, do aplikační paměti prohlížeče, které jsou následně dostupné i po znovu otevření stránky nebo celého prohlížeče. Je schopna fungovat i v režimu offline, tedy bez připojení k internetu. Aplikace využívá výhod JavaScriptového frameworku jQuery Mobile, který již byl blíže popsán v kapitole Funkce HTML5 a související technologie. Další funkce HTML5 zmíněné v teoretické části nejsou v aplikaci použity, jelikož nejsou předmětem praktického příkladu.

5.2 Ukázka zdrojového kódu

Zdrojový kód byl napsán za použití frameworku jQuery Mobile. Použití jQuery Mobile je velice jednoduché. Ze stránek frameworku stačí jednoduše stáhnout soubory css a js a následně přidat do hlavičky stránky následující řádky kódu, nebo se odkázat na tyto soubory uložené na CDN (Příklad 25).

Příklad 25 – Vložení CSS a JS souborů do hlavičky stránky. Zdroj:[autor]

```
<!-- Kaskádové styly -->
<link rel="stylesheet" href="css/jquery.mobile-1.4.5.min.css" />

<!-- JavaScript-->
<script src="js/jquery-1.11.2.min.js"></script>
<script src="js/jquery.mobile-1.4.5.min.js"></script>
```

Jak framework jQuery Mobile funguje, je nejjednodušší ukázat přímo na části zdrojového kódu panelu úkolů a úvodní stránky, použitého v aplikaci TaskList (Příklad 26 a 27).

Příklad 26 – Zdrojový kód panelu úkolů (Praktický příklad). Zdroj:[autor]

```
<!-- Začátek panelu -->
<div data-role="panel" id="taskPanel" data-position="left" data-display="reveal" data-dismissible="false">
    <ul data-role="listview" id="taskList" style="margin-bottom: 10px">
<!-- Položka úkolu-->
        <li id="task_1428960922325" class="ui-first-child ui-last-child"><a href="#task" data-transition="flip" class="task-item ui-btn ui-btn-e ui-btn-icon-right ui-icon-carat-r">Úkol 1 <span class="date">14.4.2015</span></a></li>
    </ul>
    <div>
      <p id="numberOfTasks">Nemáte žádné uložené úkoly.</p>
      <button id="deleteAll" class="ui-btn ui-icon-delete">Smazat vše</button>
    </div>
  </div><!-- Konec panelu -->
```

V panelu úkolů se nachází seznam uložených úkolů (Příklad 26). Každý úkol se skládá z tagu `<li>`, který obsahuje atribut `id` s unikátním názvem úkolu, tagu `<a>` s atributem `href` obsahujícím id stránky kam se má daný úkol načíst. Dále je uvnitř tagu `<a>` vložen tag `<span>`, do kterého se vypisuje deadline úkolu.

Příklad 27 – Zdrojový kód úvodní stránky (Praktický příklad). Zdroj:[autor]

```
<!-- Začátek stránky -->
<div data-role="page" id="homepage" class="ui-responsive-panel">
    <header data-role="header"><!-- Začátek hlavičky -->
      <a href="#taskPanel" id="panelBtn" class="ui-btn ui-corner-all hideFromPc"></a>
      <h1>Úvodní stránka</h1>
    </header><!-- Konec hlavičky -->

    <article data-role="content" class="ui-content"><!-- Začátek obsahové části -->
      <h1>Zdravím Vás,</h1>
      <p>nacházíte se na úvodní stránce aplikace TaskList...</p>
    </article><!-- Konec obsahové části -->

<!-- Začátek patičky -->
    <footer data-role="footer" data-position="fixed" style="text-align: center">
      <a href="#newTask" data-transition="flip" class="footerBtn ui-btn ui-corner-all ui-
shadow ui-icon-plus ui-btn-icon-left">Nový úkol</a>
    </footer><!-- Konec patičky -->
  </div><!-- Konec stránky -->
```

Kód úvodní stránky se skládá z hlavičky, obsahu a patičky (Příklad 27). Hlavička obsahuje nadpis stránky a tlačítko pro otevření a zavření panelu úkolů. Obsahovou část tvoří úvodní text. A nakonec patička, která je pomocí atributu `data-position` s hodnotou `fixed`, fixováná vždy k dolnímu okraji displeje a nikoliv stránky. V patičce se nachází tlačítko, které slouží k přesměrování na stránku pro vytvoření nového úkolu. Atribut tlačítka `data-transition` určuje, jaký efekt přechodu se má použít při zobrazení nové stránky. Defaultně je nastaven přechod fade, díky němuž dochází k postupnému zmizení stránky a objevení se nové. V případě praktického příkladu, byl použit přechod flip, kdy se daná stránka otočí ve středu podle vertikální osy a na její „rubové straně“ se objeví stránka nová. Atributy s prefixem `data-` specifikují, jak bude daný element vypadat a jak se bude chovat. Atribut `data-icon` (např. `search`, `refresh`) přidá elementu ikonu nebo atribut `data-role` (např. `navbar`, `footer`), který určí, jakou roli bude element zastupovat.

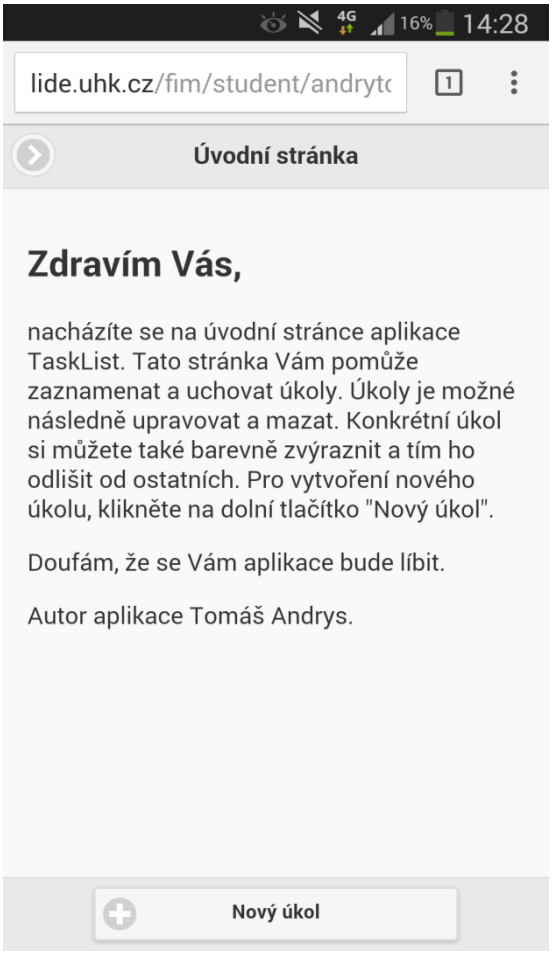
5.3 Testování hotové aplikace

Jedním z cílů projektu bylo také otestování hotové aplikace v nejpoužívanějších prohlížečích a na různých typech zařízení. Níže uvádím seznam prohlížečů a zařízení, na kterých byl test aplikace proveden.

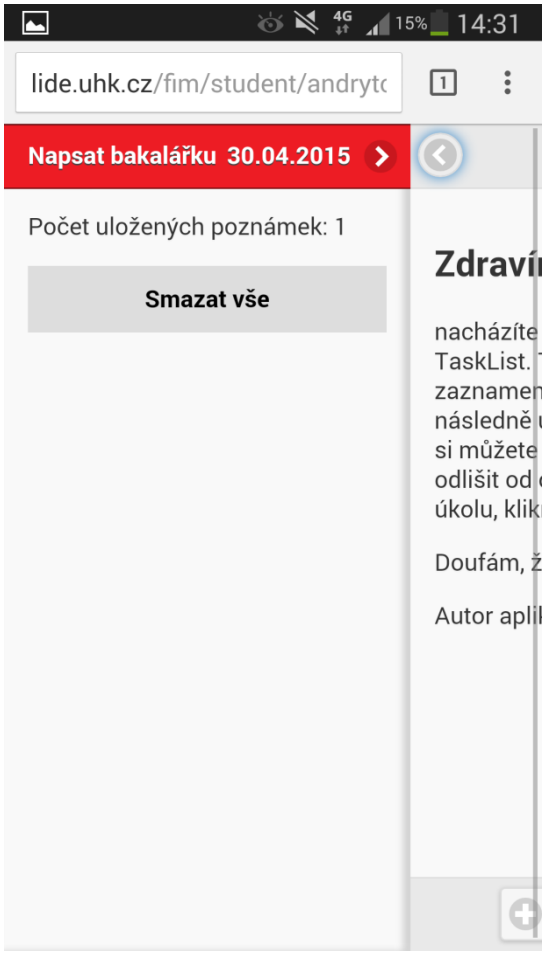
- **Notebook**, operační systém Windows 7, rozlišení obrazovky 1920x1080px
  - **Google Chrome**, verze 42.0
  - **Opera**, verze 28.0
  - **Mozilla Firefox**, verze 37.0.1
  - **Internet Explorer**, verze 11
- **Tablet Samsung Galaxy Tab2**, prohlížeč Opera Mobile 26.0, operační systém Android 4.2.2, rozlišení obrazovky 1280 x 800px
- **Mobilní telefon Samsung Galaxy S4**, prohlížeč Android Browser 4.3, operační systém Android 4.3, rozlišení obrazovky 1920x1080px

Aplikace se zobrazuje ve všech testovaných prohlížečích stejně, výjimku tvoří mobilní telefon, na kterém se díky malé velikosti displeje zobrazí aplikace jinak. Pro konkrétní představu reálného zobrazení aplikace jsou níže uvedeny snímky obrazovky z notebooku a mobilního telefonu, na kterých je vidět použití responsivního designu (Obrázky 13, 14, 15 a 16).

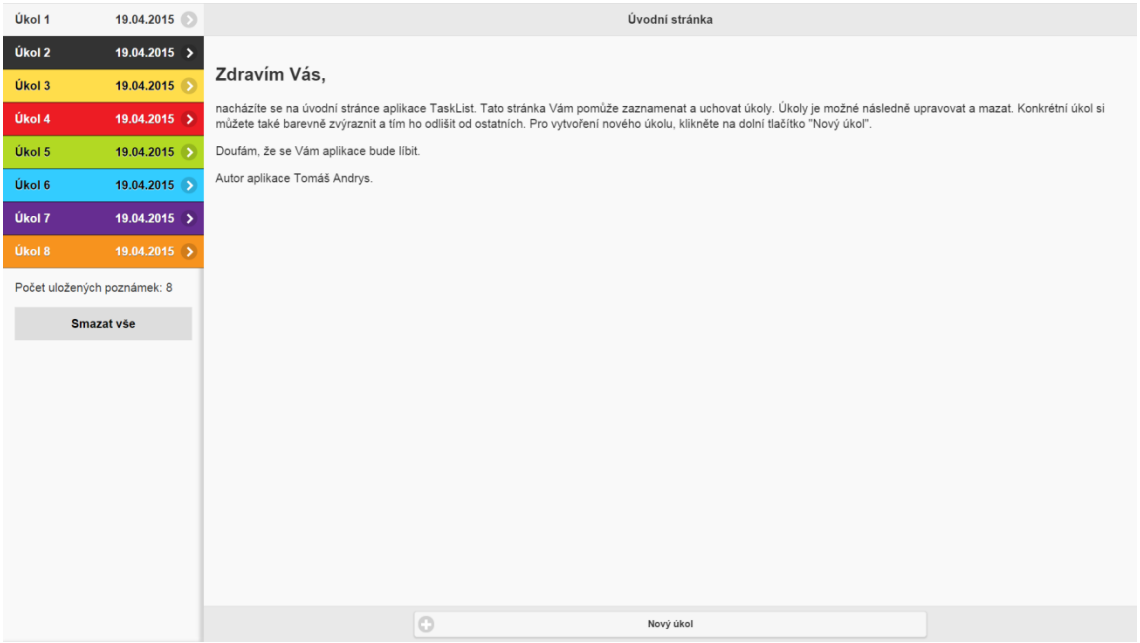
Obrázek 13 – Snímek Úvodní stránky aplikace. Zdroj:[autor]



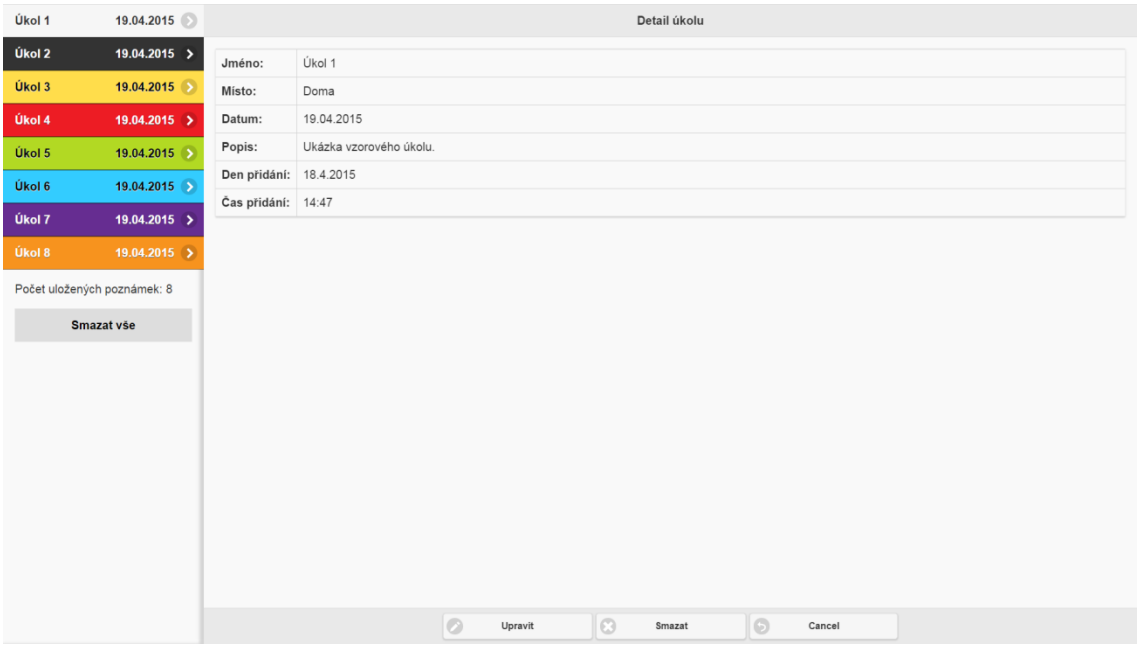
Obrázek 14 – Snímek Úvodní stránky po otevření panelu úkolů. Zdroj:[autor]



Obrázek 15 – Snímek Úvodní stránky aplikace v prohlížeči Opera. Zdroj:[autor]



Obrázek 16 – Snímek stránky Detail úkolu v prohlížeči Opera. Zdroj:[autor]



5.4 Zhodnocení aplikace TaskList

V aplikaci TaskList je demonstrován postup vývoje webových aplikací. Názorně bylo předvedeno použití některých funkcí uvedených v teoretické části práce. Nejvíce aplikace využila lokálního úložiště pro uložení dat do paměti a dále je třeba zmínit, že aplikace je připravena pro použití i v režimu offline pomocí cache manifestu.

Zdrojový kód je součástí této práce a aplikaci si je možné vyzkoušet přímo v prohlížeči na adrese http://lide.uhk.cz/andryto1/.

6 Závěry a doporučení

Z práce je zřejmé jak daleko se technologie HTML dostala za několik let. S příchodem HTML5 se práce vývojářů zjednodušila. Zkrátil se zápis `DOCTYPE`, zjednodušila se deklarace znakové sady a spousta dalších často používaných zápisů. HTML5 obsahuje několik rozhraní jako je geolokace či webové úložiště.
Práce je rozdělena do několika kapitol, které popisují problematiku HTML a CSS od jejich počátku vývoje až po současnou verzi.

Druhá kapitola se věnuje HTML a CSS, jejich historií a vývojem. Porovnává zjednodušení HTML5 oproti HTML 4.01. Dále je v kapitole řešena podpora HTML5 a CSS3 v prohlížečích jak ve starších tak i v moderních. V další části této kapitoly jsou popsány další funkce HTML5 a související technologie, jako je například offline režim aplikace nebo framework jQuery Mobile.

Další kapitola nazvaná „Způsoby vývoje aplikací“ se zabývá třemi způsoby vývoje aplikací, nativní, webovou a hybridní. Seznamuje čtenáře s hlavními výhodami a nevýhodami daných způsobů.

Čtvrtá kapitola popisuje návrh uživatelského rozhraní. Seznamuje čtenáře s tím co je to vlastně uživatelské rozhraní a co mu jako uživateli webových stránek nebo aplikací přináší. Popisuje způsob návrhu aplikace se zaměřením na uživatele nazvaný User Experience Design neboli UXD a optimalizaci vzhledu stránky na různá zařízení.

Poslední kapitola shrnuje výsledky práce, zabývá se popisem praktického příkladu aplikace, která je postavena na technologii HTML5. Obsahuje ukázku vlastního zdrojového kódu a jeho popis. Za pomocí uvedené adresy je možné si aplikaci vyzkoušet. V praktickém přikladu byl použit framework jQuery Mobile, který velmi zjednodušuje a zefektivňuje práci s HTML5. V neposlední řadě jsou součástí této kapitoly snímky hotové aplikace a popis testování hotové aplikace v různých moderních prohlížečích a na různých platformách.

7 Zdroje

7.1 Seznam literárních zdrojů

[1] DOSTÁL, Jiří. Tvorba webu pro učitele. 1. vyd. Olomouc: Univerzita Palackého v Olomouci, 2011, 65 s. ISBN 978-80-244-2780-5.

[2] FREEMAN, Eric a Elisabeth ROBSON. *Head First HTML5 programming: building web apps with Javascript*. 1st ed. Sebastopol, CA: O'Reilly, 2011, xxxiii, 573 p. Head first series. ISBN 14-493-9054-4.

[3] KOSEK, Jiří a Elisabeth ROBSON. *HTML: tvorba dokonalých www stránek : podrobný průvodce*. Vyd. 1. Praha: Grada, 1998, 291 s. Průvodce (Grada). ISBN 80-716-9608-0.

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

[5] VÁVRŮ, Jiří, Brian ALBERS a Frank SALIM. *JQuery Mobile: programujeme moderní webové aplikace*. 1. vyd. Brno: Computer Press, 2013, 247 s. Průvodce (Grada). ISBN 978-80-251-3811-3.

7.2 Seznam internetových zdrojů

[6] BOLOGNESI, Emanuele. How to build, distribute and monetize HTML5 mobile web apps. *Slideshare* [online]. 2012 [cit. 2015-04-18]. Dostupné z: http://www.slideshare.net/emabolo/how-to-build-distribute-and-monetize-html5-mobile-web-apps

[7] *Can I Use* [online]. 2014 [cit. 2014-08-18]. Dostupné z: http://caniuse.com/#cats=HTML5

[8] *JQuery Mobile* [online]. 2015 [cit. 2015-04-18]. Dostupné z: jquerymobile.com/broker-support/1.4/

[9] KORF, Mario a Eugene OKSMAN. Native, HTML5 or Hybrid: Understanding Your Mobile Applikaction Development Options. *Developer salesforce*[online]. Rok vydání neznámý [cit. 2015-04-15]. Dostupné z: https://developer.salesforce.com/page/Native,_HTML5,_or_Hybrid:_Understanding_Your_Mobile_Application_Development_Options

[10] LEADBETTER, Tom. Using Modernizr to detect HTML5 features and provide fallbacks. *Html5Doctor* [online]. 2012 [cit. 2014-08-10]. Dostupné z:http://html5doctor.com/using-modernizr-to-detect-html5-features-and-provide-fallbacks/

[11] MACHAL, Danny. Anatomy of a Hybrid Mobile GIS Application. *GeoChalkboard* [online]. 2012 [cit. 2015-04-19]. Dostupné z:http://geospatialtraining.com/blog/index.php/anatomy-of-a-hybrid-mobile-gis-application/

[12] MAŇÁK, Michal. Chcete mít vlastní mobilní aplikaci? Začněte s webovou, je levnější. *Podnikatel.cz* [online]. 2013 [cit. 2015-04-20]. Dostupné z:http://www.podnikatel.cz/clanky/chcete-mit-vlastni-mobilni-aplikaci-zacnete-s-webovou-je-levnejsi/

[13] MARTIN, Michael. Responsive Design Alone Is Not Mobile SEO. *Search Engine Land* [online]. 2012 [cit. 2015-04-20]. Dostupné z:http://searchengineland.com/responsive-design-alone-is-not-mobile-seo-124202

[14] SALVET, Pavel. HTML 5: Offline webové aplikace a aplikační cache. *Interval.cz* [online]. 2011 [cit. 2015-01-11]. Dostupné z: https://www.interval.cz/clanky/html-5-offline-webove-aplikace-a-aplikacni-cache/

[15] ŠUBRTA, Václav. Návrh uživatelského rozhraní webové aplikace. *Gml.vse.cz* [online]. 2014 [cit. 2015-04-20]. Dostupné z: http://gml.vse.cz/data/oppa-webdesign/ui.html

[16] HTML. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-04-17 [cit. 2014-08-10]. Dostupné z:http://en.wikipedia.org/wiki/HTML

[17] HTML5. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-04-14 [cit. 2014-08-15]. Dostupné z:http://en.wikipedia.org/wiki/HTML5

[18] HTML5 Shiv. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-04-16 [cit. 2014-08-15]. Dostupné z: http://en.wikipedia.org/wiki/HTML5_Shiv

[19] WYSIWYG. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2014-03-19 [cit. 2015-04-23]. Dostupné z:http://cs.wikipedia.org/wiki/WYSIWYG

[20] Standard Generalized Markup Language. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-02-06 [cit. 2015-04-23]. Dostupné z: http://cs.wikipedia.org/wiki/Standard_Generalized_Markup_Language

[21] Internet Engineering Task Force. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-02-18 [cit. 2015-04-23]. Dostupné z: http://cs.wikipedia.org/wiki/Internet_Engineering_Task_Force

[22] WebKit. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-02-24 [cit. 2015-04-23]. Dostupné z:http://cs.wikipedia.org/wiki/WebKit

[23] Framework. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2014-08-26 [cit. 2015-04-23]. Dostupné z:http://cs.wikipedia.org/wiki/Framework

[24] Content delivery network. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001-, 2015-04-08 [cit. 2015-04-23]. Dostupné z: http://cs.wikipedia.org/wiki/Content_delivery_network

7.3 Seznam použitých obrázků

7.4 Seznam použitých příkladů

7.5 Seznam použitých tabulek

7.6 Slovník pojmů

Termín	Význam[zdroj]
PLUGIN	Doplňkový modul aplikace, který rozšiřuje její funkčnost.
WYSIWYG	Česky „co vidíš, to dostaneš“. Způsob editace dokumentů v počítači. Verze zobrazená na obrazovce je vzhledově totožná s výslednou verzí dokumentu [19].
JAVA	Objektově orientovaný programovací jazyk.
JAVASCRIPT	Objektově orientovaný skriptovací jazyk.
SGML	Univerzální značkovací jazyk umožňující definovat jiné značkovací jazyky [20].
WWW	World Wide Web
IETF	Internet Engineering Task Force - organizace, která vyvíjí a podporuje internetové standardy [21].
XML	Značkovací jazyk
WHATWG	Web Hypertext Application Technology Working Group [16].
RGBA	Red Green Blue Alpha
WebKit	Název rendrovacího jádra prohlížeče [22].
Widget	Ovládací prvek
SVG	Scalable Vector Graphics
VML	Vector Markup Language
Kodek	Algoritmy používané pro kódování a dekódování audia či videa [4].
DMS	Degree Minute Second
FRAMEWORK	Slouží jako podpora při programování a vývoji a organizaci jiných softwarových projektů [23].
CDN	Content Delivery Network je síť pro doručování obsahu nebo dat [24].

8 Přílohy

8.1 Zdrojový kód praktického příkladu aplikace TaskList

Obsah této přílohy je možné si stáhnout z internetové adresy:
http://lide.uhk.cz/fim/student/andryto1/download.html

Oskenované zadání

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Rokitanského 62, 500 03 Hradec Králové, tel: 493 331 111, fax: 493 332 235

Zadání k závěrečné práci

Jméno a příjmení studenta: Tomáš Andrys

Obor studia: Aplikovaná informatika
Jméno a příjmení vedoucího práce: Monika Borkovcová

Název práce:
Tvorba webových aplikací s využitím HTML5 a CSS3

Název práce v AJ:
Development of web applications using HTML5 and CSS3

Podtitul práce:

Podtitul práce v AJ:

Cíl práce: Popis vývoje jazyka HTML a základní principy HTML5. Způsoby vývoje aplikací, jejich výhody a nevýhody. Optimalizace uživatelského rozhraní, popis vybraných funkcí HTML5 a vytvoření webové aplikace.

Osnova práce:
1. Úvod
2. Cíl a metodika práce
3. Teoretická východiska
4. HTML5 a CSS3
5. Způsoby vývoje aplikací
6. Návrh uživatelského rozhraní
7. Výsledky práce
8. Závěry a doporučení
9. Zdroje

Projednáno dne:

Podpis studenta Podpis vedoucího práce