Pedagogická fakulta
Faculty of Education

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH

Pedagogická fakulta

Katedra informatiky

Programování her v HTML5 pomocí knihoven jQuery a Box2D

Programming HTML5 games using libraries jQuery and Box2D

Bakalářská práce

Vypracovala: Kristýna Holečková

Vedoucí práce: PaedDr. Petr Pexa, Ph.D.

České Budějovice 2015

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
Fakulta pedagogická
Akademický rok: **2013/2014**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Kristýna HOLEČKOVÁ**
Osobní číslo:	**P12165**
Studijní program:	**B7507 Specializace v pedagogice**
Studijní obory:	**Informační technologie se zaměřením na vzdělávání** **Technická výchova se zaměřením na vzdělávání**
Název tématu:	**Programování HTML5 her s využitím knihoven jQuery a Box2D**
Zadávající katedra:	**Katedra informatiky**

Zásady pro vypracování:

Cílem bakalářské práce bude zpracovat možnosti tvorby počítačových her s využitím HTML5, JavaScriptové knihovny jQuery a engine pro simulaci fyziky Box2D. V bakalářské práci bude popsán princip vytváření her pomocí těchto technologií a postup při jejich následném publikování na internetu. Součástí práce bude také porovnání výhod a nevýhod vývoje a nasazení HTML5 herních aplikací oproti obdobným hrám, realizovaných pomocí tradiční technologie Flash, která není podporována v iOS (tedy na zařízeních firmy Apple). V praktické části práce bude vytvořeno několik online her, na kterých bude otestována podpora této nové technologie v dostupných verzích webových prohlížečů a zpracována podrobná dokumentace a tutoriál

Rozsah grafických prací: **CD ROM**

Rozsah pracovní zprávy: **40**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

1. **MAKZAN. HTML5 games development by example: beginner's guide ; create six fun games using the latest HTML5, Canvas, CSS, and JavaScript techniques. Birmingham, U.K: Packt Pub, 2011, ix, 331 s. ISBN 978-1-849691-26-0.**
2. **FERONATO, Emanuele. Flash game development by example: build 9 classic Flash games and learn game development along the way. Olton, Birmingham, 2011, 311 s. Community experience distilled. ISBN 978-184-9690-904.**
3. **THAU. Velký průvodce JavaScriptem: tvorba interaktivních webových stránek v praxi. 1. vyd. Praha: Grada, 2009, 516 s. ISBN 978-80-247-2211-5.**
4. **Html5.cz: vše co potřebujete vědět o HTML5. [online]. 2011 [cit. 2014-03-25]. Dostupné z: http://www.html5.cz/**
5. **HTML5 Games Development. [online]. 2011, 2013 [cit. 2014-03-25]. Dostupné z: http://www.html5gamedevelopment.com/**
6. **Box2D: A 2D Physics Engine for Games. [online]. 2011, 23.3.2014 [cit. 2014-03-25]. Dostupné z: http://box2d.org/**
7. **Box2DFlash: A 2D Physics Engine for Games. [online]. 2010 [cit. 2014-03-25]. Dostupné z: http://www.box2dflash.org/**

Vedoucí bakalářské práce: **PaedDr. Petr Pexa, Ph.D.**
Katedra informatiky

Datum zadání bakalářské práce: **27. března 2014**

Termín odevzdání bakalářské práce: **30. dubna 2015**

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 27. března 2014

Prohlášení

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

Prohlašuji, že v souladu s S 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne 23. dubna 2014

Kristýna Holečková

Abstrakt

Cílem bakalářské práce je zpracovat možnosti tvorby počítačových her s využitím HTML5, JavaScriptové knihovny jQuery a engine pro simulaci fyziky Box2D. Hry využívající HTML5 fungují na počítačích, chytrých telefonech i tabletech, včetně iPhonů a iPadů. Jsou psané pomocí JavaScriptu, kaskádových stylů CSS3 a elementů HTML5. Díky této formě her se vyřešil problém s kompatibilitou u webových prohlížečů Applu. Tyto prohlížeče (včetně u mobilních zařízení) nepodporují dosavadní a tradiční technologii Flash. Součástí této práce bude popis principu vytváření her pomocí těchto technologií a postup při jejich následném publikování na internetu. Dále v bakalářské práci bude porovnání výhod a nevýhod vývoje a nasazení HTML5 herních aplikací oproti obdobným hrám, realizovaných pomocí tradiční technologie Flash. Praktická část bude obsahovat několik online her, na kterých bude otestována podpora této nové technologie v dostupných verzích webových prohlížečů a zpracována podrobná dokumentace a tutoriál.

Klíčová slova

HTML5, hry, jQuery, Box2D, JavaScript, CSS3, technologie Flash

Abstract

The aim of this bachelor thesis is to process possibilities of creating computer games with application of HTML 5, JQuery (JavaScript library), and Box2D (an engine for physics simulation) . Games using HTML 5 work on computers, smartphones and tablets, including IPhones, and IPads. They are written in JavaScript, cascading styles CSS3 and elements of HTML5. Due to this form of games, the compatibility problem of Apple's web browsers is solved.. These web browsers (including those of mobile devices) don´t support existing and traditional Flash technology. A part of this work is a description of principle of creating games using technologies, which are mentioned above, and the pro-

cess of their later publication on the Internet. In another part of this bachelor thesis, positives and negatives of development and use of HTML 5 gaming applications are compared against those of other similar games using traditional Flash technology. The practical part of this thesis includes testing of support of this new technology in available web browser version on several online games, and also a detailed documentation and a tutorial to the testing.

Keywords

HTML5, games, jQuery, Box2D, JavaScript, CSS3, Flash technology

Obsah

1 Úvod

V současnosti se na internetu nachází velká škála různých typů her naprogramovaných jak pomocí technologie Flash, tak už i HTML5 her. Ovšem ne všechny hry fungují na současných zařízeních. Právě Flash hry se pojí s problémem přehrávání převážně na zařízeních firmy Apple, která Flash aplikace blokuje. HTML5 hry, složené převážně z JavaScriptového kódu a doplněné kaskádovými styly, prorazily barikádu právě mezi hrami na internetu a iOS zařízeními.

Samotný děj a prostředí hry tvoří JavaScriptové knihovny, které jsou určeny pro usnadnění a zrychlení práce. Těchto knihoven je celá řada a každá usnadňuje zápis nějakého kódu, ať už se jedná o přehrávání zvuků nebo zobrazování obrázků nebo jiných jiných efektů. Pro tuto práci jsem si vybrala všestrannou knihovnu jQuery, která je velmi oblíbená ze strany programátorů webů.

Pro zpestření a zvýšení kvality hry se používají různé enginy, tedy motory hry. Většina enginů se používá pro rozpohybování objektů ve hře a fyzikální zákonitostmi mezi objekty. Ovšem využití mají i u zvuků, mapových editorů apod. V této práci využívám engine Box2D pro simulaci fyziky, tedy například pro detekci kolizí při srážce objektů stejných i různých tvarů.

HTML5 hry jsou neustále aktuálním tématem, objevují se stále nové možnosti, jak hry zdokonalit právě třeba pomocí enginů nebo JavaScriptových knihoven. Dnes již existují editory přímo na vytváření těchto her bez jakéhokoliv programování, ovšem pokud chceme pochopit na jakém principu a smyslu se staví HTML5 hry, je třeba se s vývojem her seznámit více do hloubky.

Toto téma jsem si zvolila nejen díky osobnímu zájmu o HTML5 a možnostech právě s tímto značkovacím jazykem, ale i poskytnout informace o tvorbě HTML5 her, využívání JavaScriptových knihoven a enginů pohromadě pro širší veřejnost a tím rozšířit zájem o tuto problematiku v České republice.

1.1 Cíle práce

Cílem bakalářské práce je přiblížit a zpracovat možnosti tvorby her programovaných v HTML5 s využitím knihoven jQuery a Box2D. Detailně popsat princip vytváření těchto her, včetně využití kaskádových stylů tak, aby dokázal téměř úplný laik tento princip pochopit a naprogramovat svou vlastní hru včetně webového rozhraní. Součástí popisu bude přesný postup nahrání her na internet.

Porovnat a popsat, jaké jsou výhody a nevýhody herních aplikací v HTML5 oproti hrám realizovaných pomocí technologie Flash.

Dále bude vytvořeno několik her, u kterých bude podrobně vysvětleno, jakým způsobem byly vytvořeny a budou publikovány na internetu, kde bude součástí i tutoriál.

1.2 Východiska práce

S vývojem nových technologií a zařízení je třeba nezůstávat pozadu ani s vývojem kompatibilních aplikací. Mezi tyto aplikace se řadí i HTML5 hry, které zajišťují funkčnost her na všech počítačích, tabletech, chytrých telefonech, a to včetně iOS. Před vývojem značkovacího jazyka HTML5 vládly na internetu hry vytvořené pomocí technologie Flash. Ovšem tyto technologie nebyly a nejsou kompatibilní se zařízeními firmy Apple.

HTML5 hry jsou psané pomocí JavaScriptu, kaskádových stylů a elementů HTML5. V této práci budu využívat knihovnu jQuery, což je JavaScriptová knihovna, která urychluje a usnadňuje práci a dále fyzikální engine založený na sytému Box2D.

Téma programování her v HTML5 je známé už pár let, jsou již vytvořeny i editory, které umožňují vytvářet hry v HTML5 bez jakéhokoliv programování. Ovšem doposud nebyla napsaná práce v českém jazyce, která by seznamovala s vývojem her v HTML5 s využitím knihovny jQuery a enginu Box2D a zároveň poukazovala na výhody této technologie oproti hrám psaných pomocí technologie Flash, ke které je třeba zvlášť program, ve kterém se aplikace vytvoří a teprve poté se implementuje do webového prostředí. Zároveň tato

práce bude užitečná pro začátečníky programování, kteří si celou hru budou chtít naprogramovat sami bez použití editorů na HTML5 hry.

1.3 Metoda práce

V úvodu práce stručně popíšu programovací jazyky, kterými jsou hry psané. Dále se zaměřím přímo na možnosti vytváření HTML5 her, tedy zápis v JavaScriptu, CSS a samotný značkovací jazyk HTML5. V popisu budou popsány funkce a principy využití knihoven jQuery a enginu Box2D. Dále rozeberu největší výhody a nevýhody tvorby her v HTML5 a technologie Flash.

V praktické části bakalářské práce vytvořím několik online her, které budou podrobně popsané i v teoretické části. Tyto hry umístím na webové stránky, které pro tuto práci vytvořím.

2 Základní pojmy

Hry, které využívají HTML5, fungují jak na počítačích, tak na chytrých mobilních zařízeních i tabletech, včetně zařízení firmy Apple, tedy iPhonů a iPadů [1]. Jsou psané pomocí JavaScriptu, kaskádových stylů CSS3 a elementů HTML5.

2.1 HTML5

Jazyk HTML je zkratkou z anglického Hyper-Text Markup Language. Jedná se o značkovací jazyk sloužící pro tvorbu webových stránek. Určuje, jak bude vypadat výsledná struktura obsahu zobrazeného na webu a propojení s dalšími souvisejícími stránkami, případně s JavaScriptovými soubory nebo kaskádovými styly. HTML5 je nástupce HTML 4.1 a XHTML[1] 1.1 [1].

HTML5 na rozdíl od HTML značkovacích jazyků se zabývá JavaScriptovými API[2], což je rozhraní pro programování aplikací. Jedná se o například o funkce, struktury, třídy a knihoven, které se mohou využít u programování. A právě API určuje, jakým způsobem jsou ze zdrojového dokumentu (programu) funkce knihoven spouštěny. Dále se HTML5 zabývá i offline aplikacemi,kreslením v prohlížeči, které je využito i přímo v této práci a mnohými dalšími *„vychytávkami“*, které doteď HTML chyběly.[2]

Další novinkou u HTML5 jednodušší deklarace typu dokumentu *DOCTYPE* a nastavení kódování pomocí *charset*. Nové sémantické elementy jsou <*header*>, <*nav*>, <*section*>, <*footer*>, <*article*>, grafické elementy pak například <*canvas*>, což je element pro vykreslení grafiky pomocí skriptů. Tento element je použit v této práci pro vykreslení herního prostředí ve 2D i 3D. Dále je HTML5 obohaceno například o multimediální elementy <*audio*> a <*video*> 3.6[3].

[1] extensible hypertext markup language (rozšířený hypertextový značkovací jazyk
[2] Application Programming Interface

2.2 JavaScript

JavaScript je programovací jazyk, který se používá na programování webových stránek. Přípona pro tento jazyk je `.js`, což je obvyklejší přípona, ale je možné setkat se i s `.jse`. Může se zapisovat přímo do HTML kódu, nebo být samostatně napsaný a v HTML stránce na něj pouze odkazovat a přistupovat k HTML dokumentu pomocí modelu DOM [3]. Na umístění jak externího JavaScriptu, tak vnořeného nezáleží, ovšem je třeba dávat pozor na přehlednost a strukturu kódu. Funkce JavaScriptu mohou být u každého programátora jiné. JavaScript se může použít pro různé animace a efekty objektů, přehrávání zvuků, nebo třeba zobrazení datumu, aktualizace stránek, ale i například zobrazení kalendáře a mnoho dalšího.

```
1  <!DOCTYPE html>
2  <html lang="cs">
3
4  <head>
5     <title>Ukázka vložení JavaScriptu do HTML5</title>
6  </head>
7
8  <body>
9     <script> alert('Vložený skript'); </script> <!-- vložený
          JavaScript v těle dokumentu bez odkazování -->
10 </body>
11
12    <script src="jquery-2.1.3.min.js"></script> <!-- vložený
          externí JavaScriptový soubor -->
13
14 </html>
```

Příklad 1: Ukázka zápisu JavaScriptu v HTML5

[3] Document Object Model (objektový model dokumentu)

2.2.1 JavaScriptové knihovny

V dnešní době je již mnoho vytvořených knihoven, které mohou být součástí webových stránek. Jak je již zmíněno na straně 10, tyto knihovny jsou určeny pro urychlení a usnadnění práce s programováním webové stránky. Informace o nejvyužívanějších knihovnách se poměrně liší. Pro porovnání můžeme využít tři různé webové stránky W3Techs, JavaScripting a W3Schools. V této práci je využita knihovna jQuery, která je podle různých zdrojů jedna z nepoužívanějších.

2.2.2 jQuery

JavaScriptová knihovna jQuery je všestranná knihovna, která je volně šiřitelná a přístupná pro každého. Tuto knihovnu je možné stáhnout na stránkách jQuery.com a zároveň zkontrolovat, zda pracujete s nejnovější verzí. Vložení knihoven do HTML dokumentu je stejný, jako u klasických externích JavaScriptových kódů 2.2.

Funkce této knihovny jsou například efekty, animace, manipulace s kaskádovými styly, práci s JavaScriptovými pluginy, které rozšíří knihovnu třeba o nové metody. Další funkcí jQuery je Ajax, který zajišťuje aktualizaci stránek, aniž by uživatel musel ručně aktualizovat stránky (příkladem může být aktualizace emailů, kde dříve bylo třeba ručně aktualizovat stránku s doručenými emaily, abychom zjistili, zda nám nepřišla nová zpráva - dnes je aktualizace těchto stránek automatická) [4].

2.3 Herní enginy

Herní enginy slouží pro vytváření a rozvoj her. Základní funkce těchto enginů je především tzv. rendering, což je vykreslování objektů. Jedná se o proces generování obrazu pro 2D a 3D modely, detekci kolizí, fyzikální vlastnosti, zvuk a mnoho dalšího. Herní enginy ulehčují programování her tím, že obsahují nějaký kód (funkci), kterou je třeba použít na více objektů nebo částí hry.

Pro vývoj her je v současnosti vytvořeno mnoho enginů, které se většinou

zaměřují různé části her. Může jít právě třeba o multimediální enginy pro zvuk a obraz hry, 3D nebo 2D enginy, dále enginy pro vykreslování map nebo geometrie. Na těchto stránkách HTML5 Game Engines je přehled nejpoužívanějších a nepopulárnějších enginů v současnosti včetně zaměření těchto enginů, licence a dalších zajímavých informací.

2.3.1 Box2D

V této práci je použit engine pro simulaci tuhých těles ve 2D, tzv. Box2D. Tento engin je Open Source[4] vytvořený v C ++, jehož autorem je Erin Catto. Box2D je zaměřen na detekci kolizí po srážce objektů, které mohou být různých tvarů. Zároveň je možné spojit více objektů a tím zajistit, aby se chovaly jako jeden objekt. Jako příklad je možné uvést dopravní prostředek, který může být složen z těla a z kol. Dalšími funkcemi tohoto enginu je aplikování fyziky, např. tření, gravitace, pohyb objektů plynutím času [6].

Dostačující ukázka, jak tento engine pracuje, je na těchto stránkách lib.ivank.net. Zároveň je zde možné prohlédnout si, jak se chovají objekty s jinými enginy.

2.4 CSS3

CSS[5] neboli kaskádové styly jsou nezbytnou součástí každého webu. Určujeme jimi vzhled stránky, tedy například pozadí stránky, vlastnosti písma, obrázků a mnoho dalších nezbytných a užitečných funkcí.

Teprve CSS3 přišly s novinkami jako animace objektů, textovými efekty, možnosti zakulacení objektů, stínů, průhledností a dalšími efekty. Soubor s kaskádovými styly se píše zvlášť a v dokumentu HTML se na něj odkazuje. Dříve se styly psaly přímo do HTML dokumentu, ale pro usnadnění práce a lepší přehlednost, se CSS začaly dávat do externích souborů. Odkazy na kaskádové styly se dávají do hlavičky dokumentu (<*head*>).

[4]software s otevřeným zdrojovým kódem
[5]Cascading Style Sheets

```
1  <!DOCTYPE html>
2  <html lang="cs">
3
4  <head>
5     <title>Ukázka vložení kaskádových stylů do HTML5</
          title>
6     <link rel="stylesheet" href="styly.css" type="text/css"
           media="screen" />  <!-- externí soubor CSS -->
7
8     <style type="text/css"> <!-- dřívější zápis CSS
          vnořený v HTML -->
9        h1 {color: blue; font-weight: bold;}
10    </style>
11 </head>
12
13 <body>
14 </body>
15
16 </html>
```

Příklad 2: Ukázka zápisu CSS kódu v HTML5

3 Zápis kódu v HTML5

Aby bylo možné začít vytvářet HTML5 hry, je třeba znát způsob zápisu jak HTML kódu, kaskádových stylů, tak JavaScriptu. Právě správný způsob zapisování je jedna z nejdůležitějších znalostí pro vytváření her nebo jen třeba samotných webových stránek. Ovšem samotný jazyk HTML5 zajistí strukturu stránek a teprve kaskádové styly zajistí vzhled stránek, případně JavaScript.

Struktura každé HTML stránky se skládá ze začátku a konce dokumentu, dále z hlavičky (*head*) a těla (*body*). Pro zápis této struktury se používají tzv. tagy, které mohou být párové i nepárové. Příklad párového tagu je právě třeba zápis začátku a konce HTML dokumentu, kde *<html>* znamená začátek dokumentu a *</html>* je naopak konec dokumentu. Nepárový tag je například pro obrázek, kde zápis je *<img />*. Každý tag může dále obsahovat tzv. atributy, které upřesňují funkci daného tagu.

Kromě deklarování typu dokumentu (strana 19) se veškeré informace píšou právě mezi začátek a konec dokumentu.

3.1 HTML editory

K vytvoření webové stránky nebo her je nutný HTML editor. V dnešní době se nabízí celá řada různých těchto editorů avšak ne všechny mají uživatelské rozhraní shodné. HTML editory se dělí na dva typy, a to na wysiwyg[6] editory a strukturní editory. Ve wysiwygových editorech je práce mnohem snažší a není potřeba znalost jazyka HTML, jelikož se v těchto editorech skládají jednotlivé části stránky a program automaticky generuje HTML kód (ovšem je i možnost přímo zasáhnout do kódu a upravit jej). Použití tohoto editoru má i své nevýhody, a to že ne vždy je výsledná webová stránka stejná, jak se zobrazuje v editoru. Wysiwyg editory jsou dnes například Microsoft Web Expression, Microsoft FrontPage, Dreamweaver, Adobe GoLive a další [7].

Veškeré kódy, které se vztahují k této práci jsou psané ve strukturním editoru, konkrétně v editoru PSPad, což je jeden z nejlepších editorů, které jsou

[6] What you see is what you get (”Co vidíš, to dostaneš”)

k dostání. Velkou výhodou tohoto editoru je bezplatná licence a české rozhraní. Dalšími strukturními editory jsou HomeSite, Notepad++ nebo EasyPad [7].

V tomto článku jsou porovnány editory podle oblíbenosti uživatelů. Po porovnání je zřejmé, že jsou převážně oblíbenější strukturní editory. Je zde také možné porovnat výhody a nevýhody jednotlivých editorů, případně si přečíst fórum, kde je možné dozvědět se o těchto (i jiných) editorech něco víc.

3.2 Document Type Definition

Aby byl HTML dokument správně formátovaný, je třeba, aby na začátku obsahoval informace o verzi používaného HTML jazyka a DTD (Document Type Definition) neboli návod pro prohlížeče, které zpracovávají dokument [8].

Každý zápis deklarování, ať HTML5 nebo starších verzí HTML dokumentů, je umístěn na prvním řádku. Jedinou výjimkou je XHTML, kde na prvním řádku je umístěna deklarace dalšího značkovacího jazyka XML[7], který umožňuje navrhnout vlastní tagy. Dále součástí deklarace XML je kódování češtiny *windows-1250*.

Pro správný zápis deklarace dokumentu HTML5 se používá následující kód, který byl právě při vzniku jazyka HTML5 velmi zjednodušen.

```
1 <!DOCTYPE html> <!-- zápis deklarování typu dokumentu
    HTML5 -->
```

Příklad 3: Deklarace typu dokumentu v HTML5

Než vznikl jazyk HTML5, používalo se jiné deklarování, ať už pro XHTML nebo HTML 4.01 a další typy. Pro zajímavost jsou typy deklarování kódů uvedeny v následujícím příkladě [8].

```
1 <!-- zápis deklarování typu dokumentu XHTML 1.0 s
    deklarací XML a kódováním češtiny -->
2 <?xml version="1.0" encoding="windows-1250"?>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "
    http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

[7] eXtensible Markup Language

```
4
5  <!-- zápis deklarování typu dokumentu HTML 4.01 -->
6  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
7     "http://www.w3.org/TR/html4/strict.dtd">
8
9  <!-- zápis deklarování typu dokumentu HTML 3.2 -->
10 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
```

Příklad 4: Deklarace typu dokumentu v XHTML 1.0, HTML4 a HTML 3.2

3.3 Jazyk dokumentu

Nastavený jazyk dokumentu určuje, jakým jazykem je dokument napsán. Zapisuje se přímo do značky *<html>* pomocí atributu *lang*. Výhodou tohoto atributu je, že pokud celý dokument má být napsán v anglickém jazyce (*"en"*) a jen část chceme, aby byla v českém jazyce (*"cs"*), můžeme atribut *lang* napsat např. do značky * * a čeština se bude vztahovat jen na tu část textu, která je obsažena právě v tomto *divu* a tím pádem jdou jazyky v dokumentu kombinovat.

```
1  <!DOCTYPE html>
2  <html  lang="en">  <!-- jazyk dokumentu nastavený na
      angličtinu -->
3  <head></head>
4  <body>
5     <p>Zde je text nastavený pro angličtinu</p>
6     <div  lang="cs">
7        Pro tento text je nastavená čeština
8     </ div >
9     <p>Zde je text nastavený pro angličtinu</p>
10 </body>
11 < /html >
```

Příklad 5: Ukázka nastavení jazyka v HTML5

3.4 Hlavička dokumentu

Hlavička dokumentu se také značí párovým tagem *<head>* a *</head>*. Dovnitř hlavičky se píšou informace, které se na stránce nezobrazují. Například o kódování stránky (strana 21), o autorovi stránek, klíčových slovech na stránkách atd. (více na straně22).

3.4.1 Kódování

Aby počítače dokázaly rozeznat text, který máme v HTML dokumentu, je třeba použít kódování jazyka, který je obsažen v našich stránkách. Jednotlivé znaky psané na počítači se převádí do čísel. Pokud se jedná o český jazyk, používá se *ISO 8859-2*, *Windows-1250* nebo *UTF-8*. Z těchto tří typů kódování je nejčastější právě *UTF-8*, které je určeno pro všechny světové jazyky, jedná se o nejmodernější znakovou sadu používanou v právě v v dokumentu deklarovaném typem HTML5 . *Windows-1250* je preferováno na operačních systémech Windows. Toto kódování je oblíbené především proto, že většina editorů používá právě tuto znakovou sadu jako základní [9].

Kódování se zapisuje v HTML dokumentu do tzv. *meta* tagu, který je umístěn v hlavičce stránky. Další typy meta tagů jsou popsány na straně 22.

```
1 <!-- zápis kódování UTF-8 (již pro HTML5) -->
2 <meta charset="utf-8" />
3
4 <!-- zápis kódování windows-1250 -->
5 <meta charset="windows-1250">
6
7 <!-- zápis kódování iso-8859-2 -->
8 <meta charset="iso-8859-2">
```

Příklad 6: Zápis kódování češtiny

3.4.2 Meta tagy

Meta tagy představují informace o dokumentu. Může to být informace o kódování, autorovi, popis stránky nebo použití klíčových slov na stránce. Veškeré meta tagy se umisťují do tzv. hlavičky dokumentu (*<head></head>*). Jedná se o nepárový tag, který obsahuje atribut *name* nebo *http-equiv*. Atribut *name* nastavuje meta tag, zda jde právě o informaci o autorovi, popisu stránky, klíčových slovech, kódování a dalších. Pokud místo atributu *name* je použit *http-equiv*, může se jedna o informaci o jazyce dokumentu, přesměrování na jinou stránku po uplynutí určitého času, kompatibilitu webových prohlížečů atd.

```
1  <!-- zápis kódování UTF-8 (již pro HTML5) -->
2  <meta charset="utf-8" />
3
4  <!-- informace o autorovi stránek -->
5   <meta name="author" content="Jméno; email@email.cz" />
6
7  <!-- popis stránek -->
8  <meta name="description" content="Čeho se stránky týkají"
       />
9
10 <!-- nastavení klíčových slov stránek -->
11 <meta name="keywords" content="klíč1, klíč2, ..." />
12
13 <!-- po 5 sekundách dojde k přesměrování na www.jine-
     stranky.cz -->
14 <meta http-equiv="refresh" content="5;URL=http://www.jine-
     stranky.cz">
```

Příklad 7: Zápis meta tagů

3.4.3 Link tagy

Link tagy jsou nepárové tagy, které spojují HTML dokument s jinými soubory. Nejčastěji se jedná o vložení externího souboru s kaskádovými styly nebo tzv. *favikony*, což je ikona stránky. Atributem těchto tagů je atribut *rel*, který určuje vztah k externímu linkovanému souboru. V případě CSS souborů se jedná o *"stylesheet"* a u ikony stránky o *"shortcut icon"*. Dále atribut *href* nastavuje cestu k souboru, atribut *type* zase specifikuje typ souboru (např. u kaskádových stylů *"text/css"*) a atribut *media*, který určuje omezení na výstupu, což znamená, kdy se bude např. CSS soubor zobrazovat. Hodnoty atributu *media* mohou být *"all"*, *"screen"*, *"print"* a další.

```
1 <!-- připojení CSS souboru k HTML dokumentu -->
2 <link rel="stylesheet" href="styly.css" type="text/css"
     media="screen" />
3
4 <!-- připojení favikony k HTML dokumentu -->
5 <link rel="shortcut icon" type="image/x-icon" href="
    favicon.ico" />
```

Příklad 8: Zápis link tagů

Ikona stránky se zobrazuje v prohlížeči nad adresou webové stránky (viz Obrázek 1). Jako ikonu si lidé volí logo webových stránek nebo pokud logo příliš dlouhé a jako ikona by nevyniklo, používá se zkrácená verze, například dvě písmena nebo jen nějaký symbol. Pro vytvoření *favikony* se může použít generátor, který je umístěn na webu. Například tento generátor, po vložení souboru (připraveného obrázku), automaticky vytvoří *favikonu*, kterou poté stačí umístit do složky s připravenými stránkami a nahrát ji pomocí *link* tagu.

Obrázek 1: Zobrazení ikony stránky

3.4.4 Titulek stránky

Titulek stránky by být posledním kódem v hlavičce dokumentu. Jedná se o párový tag značený *<title>* a *</title>*. Titulek se používá především kvůli vyhledávačům, aby rozeznaly, o čem konkrétní stránky jsou. Právě titulky stránek se zobrazují při vyhledávání konkrétních stránek v prohlížečích. Stejně tak když si uživatel ukládá stránku do záložek, jako název stránky se mu nabídne právě titulek [10].

```
1 <!-- titulek na stránce -->
2 <title>Titulek stránky</ /title >
```

Příklad 9: Zápis titulku stránky

Důležité je, aby byl výstižný a aby podával konkrétní informaci. Konkrétní příklad může být u webových stránek html5-hry.cz, kde na úvodní stránce je titulek HTML5 HRY. V tomto případě titulek podává konkrétní informaci o stránkách, na které se uživatel dostal. Pokud by uživatel přešel na nějakou hru, která se mu na těchto stránkách nabízí, měl by se mu zobrazit titulek konkrétní hry. Titulek by neměl být HRA, protože her je na tomto webu víc a není to konkrétní informace. Proto stačí titulek doplnit o název té hry - například HRA PEXESO nebo HRA AUTO nebo místo HRY napsat jen název dané hry.

Obrázek 2: Titulek webové stránky

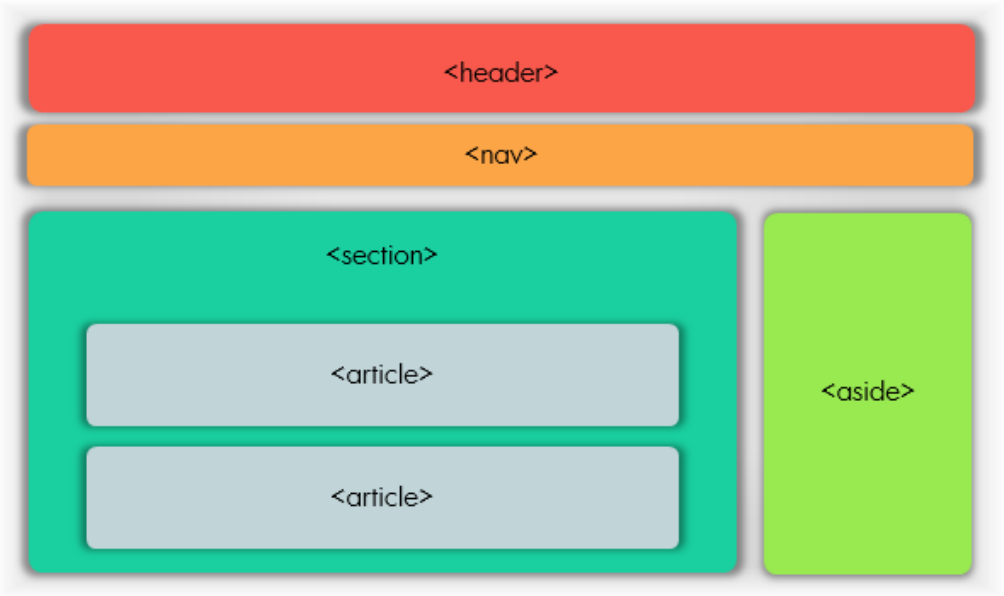
3.5 Tělo dokumentu

Stejně jako hlavička dokumentu se značí i tělo dokumentu párovým tagem *<body>* a *</body>*. Do těla dokumentu se zapisuje vše, co má být na stránce zobrazeno. Tedy text, obrázky, odkazy apod.

Dříve se v těle dokumentu jednotlivé části oddělovaly pomocí divů, ovšem v

dnešním HTML5 dokumentu má každá část (sekce) svůj název, podle kterého se vyhledávačům lépe orientuje v obsahu stránky [11].

Struktura a rozložení částí na stránce je zobrazeno v následujícím obrázku (Obrázek 3). Jednotlivé sekce jsou popsány v dalších podkapitolách.



Obrázek 3: Struktura těla webové stránky

3.5.1 Header

Header neboli hlavička stránky (nikoli hlavička dokumentu) je zpravidla první sekcí, která je v těle použita. Jedná se o párový tag, který se značí *<header>* a *</header>*.

V této sekci by měl být umístěn hlavní nadpis (*h1*) případně spolu s dalším nadpisem (ovšem už by to neměl být *h1* ale nějaký podnadpis např. *h2*) a oba dva nadpisy sloučené v párovém tagu *hgroup*. Dále do hlavičky stránky patří obrázek loga stránky, případně vyhledávací pole nebo výběr jazyka [11].

```
1 <!-- začátek hlavičky stránky -->
2 < header >
3    <!-- načtení obrázku loga stránky -->
```

```
4     <img  src="logo.png"  alt="logo" />
5
6     <!-- sloučení nadpisů -->
7     <hgroup>
8         <h1>Hlavní nadpis</h1>
9         <h3>Podnadpis</h3>
10    </hgroup>
11 <!-- konec hlavičky stránky -->
12 </header>
```

Příklad 10: Zápis hlavičky v HTML5

3.5.2 Nav

Nav neboli navigační menu je určeno ke zobrazení odkazů podstránek konkrétního webu. Opět se se jedná o párový atribut, který se značí *<nav>* a *</nav>*. Může být součástí hlavičky stránky nebo být umístěn zvlášť ve stránce. Ovšem nejčastěji bývá právě pod hlavičkou stránky.

Navigační menu je většinou tvořeno z *ul* seznamu nebo pomocí tabulky, která se značí párovým atributem *<table>*.

```
1  <!-- začátek navigačního menu -->
2  <nav>
3     <!-- začátek seznamu s odkazy na podstránky -->
4     <ul>
5        <li>Odkaz1</li>
6        <li>Odkaz2</li>
7        <li>Odkaz3</li>
8     </ul>
9  <!-- konec navigačního menu -->
10 </nav>
```

Příklad 11: Zápis navigačního menu v HTML5

3.5.3 Section

Section, je ta část stránky, kde se zobrazuje hlavní text na stránce. Stejně jako předchozí sekce stránky i section je párovým atributem, který se značí *<section>* a *</section>*. Může obsahovat články (*article*), které obsahují další section [11].

Příklad správného zápisu je v podkapitole 3.5.4.

3.5.4 Article

Tag article je prostor na stránce pro umístění konkrétního článku. Článek se značí *<article>* a *</article>*, což znamená, že se opět jedná o párový tag těla dokumentu. Sekce article může obsahovat další hlavičku stránky (*header*) i tělo stránky (*section*). Použití sekcí článků je výhodné především pro vyhledávače, které mají poté možnost zaměřit se na konkrétní část textu [11].

```
1 < article >    <!-- začátek článku -->
2     < header >
3         <h1 > Nadpis </h1 >
4     < /header >
5     < section >    <!-- začátek těla stránky (článku)-->
6         <p>Konkrétní text</p>
7     < /section >  <!-- konec těla stránky (článku) -->
8 < /article >  <!-- konec článku -->
```

Příklad 12: Zápis článku v HTML5

3.5.5 Aside

Aside v HTML5 představuje párový tag pro postranní panel, který se značí *<aside>* a *</aside>*. Především je brán jako doplňující součást článku (*article*) na odkazy podobných článků, dalších informací apod. [11].

```
1 <!-- začátek postranního panelu -->
2 < aside >
3     <h1>Užitečné odkazy</h1>
```

```
4        <ul>
5            <li>Odkaz1</li>
6            <li>Odkaz2</li>
7        </ul>
8    <h1>Více čtěte na:</h1>
9        <ul>
10           <li>Odkaz4</li>
11       </ul>
12 <!-- konec postranního panelu -->
13 </aside>
```

Příklad 13: Zápis postranního panelu v HTML5

3.5.6 Footer

Footer, neboli patička stránky, se umisťuje na konec těla dokumentu (body). Opět se jedná o párový tag, který se značí <footer> a </footer>. Do patičky se nejčastěji umisťuje copyright, datum aktualizace stránky, také odkaz na email autora stránky atd.

```
1 <!-- začátek patičky -->
2 <footer>
3    Copyright (c) 2015
4    web master : <a title="e-mail" href="mailto:autor-stranek
         @email.cz">Jméno autora</a>
5 <!-- konec patičky -->
6 </footer>
```

Příklad 14: Zápis patičky v HTML5

3.6 Shrnutí

Na závěr této kapitoly je zde příklad s celkovou (základní) strukturou stránky napsané v HTML5 jazyce.

```
<!DOCTYPE html> <!-- zápis deklarování typu dokumentu -->
<html lang="cs">
<head>   <!-- hlavička dokumentu -->
   <meta charset="utf-8" /> <!-- zápis kódování -->

   <!-- informace o autorovi stránek -->
   <meta name="author" content="Jméno; email@email.cz" />

   <!-- nastavení klíčových slov stránek -->
   <meta name="keywords" content="klíč1, klíč2, ..." />

   <title>Titulek</title>
</head>

<body>   <!-- tělo dokumentu -->
   <header> <!-- hlavička zobrazené stránky -->
      <h1>Nadpis 1</h1>
   </header>

   <nav> <!-- prostor pro navigační odkazy -->
      <a href="odkaz1.html">Odkaz1</a>
      <a href="odkaz2.html">Odkaz2</a>
   </nav>

   <section>   <!-- tělo zobrazeného stránky -->
      <article>   <!-- začátek článku -->
         <header>
         <h1>Nadpis</h1>
         </header>

         <section>   <!-- tělo článku -->
            <p>Konkrétní text</p>
         </section>
```

```
34          </article>
35
36          <aside>  <!-- začátek postranního panelu -->
37              <h1>Více čtěte na:</h1>
38                  <ul>
39                      <li>Odkaz4</li>
40                  </ul>
41          </aside>
42      </section>
43
44      <footer> <!-- patička stránky -->
45          Copyright (c) 2015
46          web master : <a title="e-mail" href="mailto:autor-
                stranek@email.cz">Jméno autora</a>
47      </footer>
48  </body>
49  </html>
```

Příklad 15: Ukázka zápisu kódu v HTML5

4 Zápis kódu v CSS3

Tato kapitola seznamuje s použitím a správným zápisem kaskádových stylů. Jelikož v této práci je pro vzhled HTML5 her využito CSS3, jsou na něj následující podkapitoly zaměřeny. Všechny hry mají kaskádové styly napsané v samostatném souboru s příponou .css, proto i zde jsou veškeré příklady uváděny pro samostatný soubor CSS3. Příklad s externím CSS souborem je na straně 16.

4.1 Nastavení stylů pro tagy

Stylování tagů se používá tehdy, když chceme pro všechny tagy stejného typu stejný vzhled. Například pokud by bylo na stránce více obrázků, pro které je tag <img />, ve stylech by se pro tag nastavilo, že šířka *img* má být 100px, všechny obrázky budou široké 100px. Proto je dobré promyslet, jestli opravdu nastavený styl má být pro všechny objekty stejného tagu. Pokud by každý obrázek měl být jinak široký, poté je dobré přidat ke konkrétnímu obrázku atribut *id* (str. 38) nebo *class* (str. 33) například s názvem obrázku a poté pro tento název nastavit nějaký styl. Pokud je na stránce například jediná tabulka, je zbytečné k ní přidávat další atributy, které by upřesňovaly konkrétní objekt a styly se mohou nastavit přímo na tag tabulky (*table*). Ovšem pro jednotlivé buňky v tabulce (pokud chceme například jinak barevné nebo velké) už by bylo potřeba nastylovat každou zvlášť stejným způsobem jako v předchozím příkladu s obrázky.

Veškeré tagy psané do CSS dokumentu se zapisují v podstatě stejně, jen bez veškerých znaků (<,> apod.). Tedy jen např. *h1*, *p*, *img*, *div*, *section* atd. Dále se jen připojí složené závorky {}, do kterých se zapisuje veškeré stylování (*height, width, border, color, backgroung, ...*) a za dvojtečku se napíše například daná hodnota, barva nebo kombinace hodnot. Pokud se zadává barva (color) ať se jedná o barvu pozadí, rámečku nebo písma, píše se buď slovy jako *white*, *black*, *blue* nebo pomocí rgb modelu, kde se zadává číslo odstínu červené barvy (***r****ed*), zelené (***g****reen*) a modré (***b****lue*). Ve většině editorů je možnost si danou barvu navolit podle jakési paletky s různými odstíny barev a po vybrání dané

barvy se číslo odstínů zapíše automaticky k danému příkazu (v PSPadu se pro výběr těchto možností, ale i příkazů používá klávesová zkratka CTRL + mezerník).

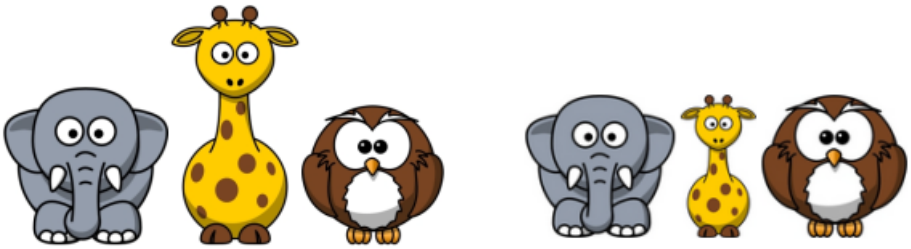
Příklad: V HTML dokumentu jsou načteny tři obrázky a každý z nich má ve skutečnosti jinou velikost. Přesto se mají zobrazit tak, aby byly stejně vysoké.

```
<!DOCTYPE html>
<html lang="cs">
<head>
    <meta charset="utf-8" />

    <link rel="stylesheet" href="styly.css" type="text/css"
        media="screen" />

    <title>Nastavení stejné výšky pro tag img</title>
</head>
<body>
    <section>
        <img src="slon.jpg" alt="slon" title="slon" />
        <img src="zirafa.jpg" alt="žirafa" title="žirafa" />
        <img src="sova.jpg" alt="sova" title="sova" />
    </section>
</body>
</html>
```

Příklad 16: Načtení obrázků v HTML dokumentu

```
/* CSS document */
// volání tagu obrázku z HTML dokumentu
img {
        height: 100px; // nastavení výšky v pixelech
    }
```

Příklad 17: Upravení velikosti obrázku v CSS dokumentu



(a) Obrázky bez nastavení velikosti (b) Obrázky s nastavením velikosti

Stejně tak se upravují například nadpisy (*h1, h2, h3, ...*), kde se mění font písma, barva, velikost nebo i části těla dokumentu (*header, section, article, ...*).

4.2 Nastavení stylů pro class

Class, neboli třída elementu, se používá tehdy, pokud chci nějaký konkrétní objekt odlišit od druhého, například právě kvůli odlišného nastylování od ostatních objektů se stejným tagem. Zápis do CSS dokumentu je podobný jako u tagů (str. 31). Jen s tím rozdílem, že místo názvu tagu se používá element (jméno) třídy a zároveň se před jméno píše tečka.

Příklad: V HTML dokumentu je nadefinována tabulka s devíti buňkami. Každá buňka má být vyplněna libovolnou barvou, avšak v celé tabulce se konkrétní barva nesmí opakovat více jak dvakrát.

```
1  <!DOCTYPE html>
2  <html lang="cs">
3  <head>
4      <meta charset="utf-8" />
5
6      <link rel="stylesheet" href="styly.css" type="text/css"
           media="screen" />
7
8      <title>Barevná výplň buněk v tabulce</title>
9  </head>
10 <body>
```

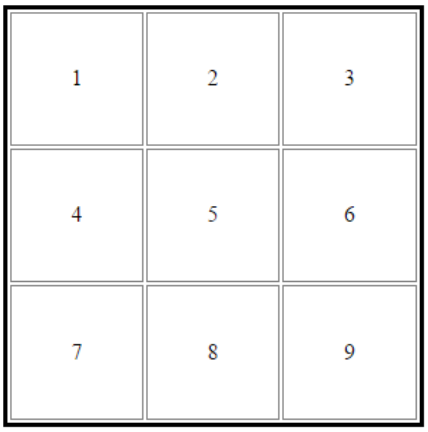
```
11      <section>
12          <table>
13              <tr>  <!-- první řádek tabulky -->
14                  <td>1</td> <!-- první buňka -->
15                  <td>2</td> <!-- druhá buňka -->
16                  <td>3</td> <!-- třetí buňka -->
17              </tr>
18              <tr><!-- druhý řádek tabulky -->
19                  <td>4</td> <!-- první buňka -->
20                  <td>5</td> <!-- druhá buňka -->
21                  <td>6</td> <!-- třetí buňka -->
22              </tr>
23              <tr>    <!-- třetí řádek tabulky -->
24                  <td>7</td> <!-- první buňka -->
25                  <td>8</td> <!-- druhá buňka -->
26                  <td>9</td> <!-- třetí buňka -->
27              </tr>
28          </table>
29      </section>
30  </body>
31  </html>
```

Příklad 18: Nastavení tabulky

```
1   /* CSS document */
2   // nastavení tabulky
3   table {
4       width: 300px;
5       height: 300px;
6       border: 3px solid black; // ohraničení tabulky plnou
            černou čarou
7       text-align: center;     // umístění textu na střed
8   }
9
10  // nastavení všech buněk najednou
```

```
11 td {
12     border: 1px solid gray; // ohraničení buněk plnou
           šedivou čarou
13 }
```

Příklad 19: Úprava stylů pro tagy



Obrázek 4: Tabulka bez nastylování tříd

Teď je jsou již buňky v tabulce vytvořeny, stačí každou zvlát' nastylovat pomocí tříd.

```
1  <!DOCTYPE html>
2  <html lang="cs">
3  <head>
4      <meta charset="utf-8" />
5
6      <link rel="stylesheet" href="styly.css" type="text/css"
           media="screen" />
7
8      <title>Barevná výplň buněk v tabulce</title>
9  </head>
10 <body>
11     <section>
12         <table>
```

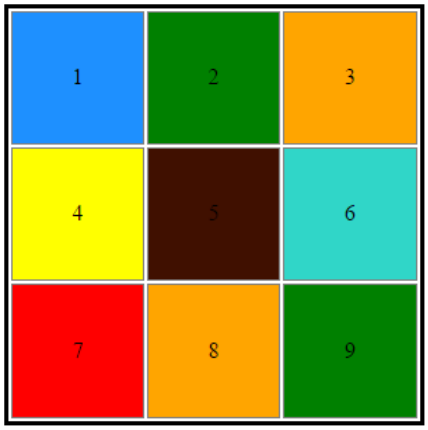
```
13          <tr>  <!-- první řádek tabulky -->
14             <td  class="modra">1</td> <!-- první buňka -->
15             <td  class="zelena">2</td> <!-- druhá buňka
                  -->
16             <td  class="oranzova">3</td> <!-- třetí buňka
                  -->
17          </tr>
18          <tr><!-- druhý řádek tabulky -->
19             <td  class="zluta">4</td> <!-- první buňka -->
20             <td  class="hneda">5</td> <!-- druhá buňka -->
21             <td  class="tyrkysova">6</td> <!-- třetí buňka
                   -->
22          </tr>
23          <tr>     <!-- třetí řádek tabulky -->
24             <td  class="cervena">7</td> <!-- první buňka
                  -->
25             <td  class="oranzova">8</td> <!-- druhá buňka
                  -->
26             <td  class="zelena">9</td> <!-- třetí buňka
                  -->
27          </tr>
28       </table>
29    </section>
30 </body>
31 </html>
```

Příklad 20: Nastavení tabulky pomocí tříd v HTML dokumentu

```
1  /* CSS document */
2  // nastavení tabulky
3  table {
4     width: 300px;
5     height: 300px;
6     border: 3px solid black;
7     text-align: center;
```

```
8  }
9  // nastavení všech buněk najednou
10 td {
11     border: 1px solid gray; // ohraničení buněk plnou
       šedivou čarou
12 }
13
14 // nastavení barev pomocí jednotlivých tříd
15  .modra {
16     background-color: rgb(30,144,255); // nastavení barvy
       pomocí RGB modelu
17 }
18  .zelena {
19     background-color: green;
20 }
21  .oranzova {
22     background-color: orange;
23 }
24  .zluta {
25     background-color: yellow;
26 }
27  .hneda {
28     background-color: rgb(64,16,0);
29 }
30  .tyrkysova {
31     background-color: rgb(48,214,200);
32 }
33  .cervena {
34     background-color: red;
35 }
```

Příklad 21: Nastavení CSS pro třídy



Obrázek 5: Tabulka s upravenými buňkami

Tento způsob úpravy objektů pomocí tříd lze samozřejmě použít na různé objekty. Například pokud je na stránce více článků (*article*) a každý má být například jinak veliký, nebo má mít jiný font nebo velikost písma.

4.3 Nastavení stylů pro id

Id neboli identifikátor se používá nejen pro kaskádové styly, ale i pro skripty. Stejně jako u tříd by mělo jméno identifikátoru být jednoznačné. Způsob použití v kaskádových stylech je téměř stejný jako u tříd (str. 33), jediná změna je ta, že místo předřazené tečky u jména třídy v CSS dokumentu je předřazený křížek #.

```
1    .
2    .
3    .
4        <table>
5            <tr>  <!-- první řádek tabulky -->
6                <td id="modra">1</td> <!-- první buňka -->
7                <td id="zelena">2</td> <!-- druhá buňka -->
8                <td id="oranzova">3</td> <!-- třetí buňka -->
9            </tr>
10           <tr><!-- druhý řádek tabulky -->
```

```
11              <td id="zluta">4</td> <!-- první buňka -->
12              <td id="hneda">5</td> <!-- druhá buňka -->
13              <td id="tyrkysova">6</td> <!-- třetí buňka
                    -->
14          </tr>
15          .
16          .
17          .
```

Příklad 22: Nastavení tabulky s id v HTML dokumentu

```
1  /* CSS document */
2  .
3  .
4  .
5  // nastavení barev pomocí jednotlivých id
6  #modra
7  {
8      background-color: rgb(30,144,255); // nastavení barvy
           pomocí RGB modelu
9  }
10 #zelena
11 {
12     background-color: green;
13 }
14 #oranzova
15 {
16     background-color: orange;
17 }
18 #zluta
19 {
20     background-color: yellow;
21 }
22 #hneda
23 {
```

```
24      background-color: rgb(64,16,0);
25  }
26  #tyrkysova
27  {
28      background-color: rgb(48,214,200);
29  }
```

Příklad 23: Nastavení stylů pro id

4.4 Upravení stylů podle typu prohlížeče

V současnosti mezi nejoblíbenější prohlížeče patří Mozilla Firefox, Google Chrome, Opera Internet Explorer a Safari. Někdy se ovšem mohou vyskytnout se shodným zobrazováním stránky v různých prohlížečích, což je převážně díky nastylování. Prohlížeče mají různá renderovací jádra, neboli jádra pro vykreslování webových stránek. Díky tomu se nemusí všechny styly na stránce zobrazovat správně nebo vůbec. Tento problém řeší tzv. prefixy s klíčovými slovy jednotlivých prohlížečů.

4.4.1 Mozilla Firefox

Mozilla Firefox má vykreslovací jádro tzv. Gecko. Stejně tak prohlížeč Netscape, který také patří k Mozille. Pro nastavení stylů právě pro tyto prohlížeče se používá následující zápis [12].

```
1  /* CSS document */
2  @-moz-keyframes {
3      .zelena {
4            background-color: green;
5        }
6      .oranzova {
7            background-color: orange;
8        }
9  }
```

Příklad 24: Nastavení stylů pro Moz prohlížeče

Výhodou tohoto zápisu je, že není třeba vytvářet zvláštní CSS dokument pro každý prohlížeč, ale může se zapsat k ostatním stylům v jednom dokumentu.

4.4.2 Google Chrome, Opera, Safari

Google Chrome, Opera i Safari mají vykreslovací jádro WebKit (nebo části tohoto jádra), což je asi nejlepší engin právě pro tuto funkci. Pro všechny tyto prohlížeče funguje následující zápis v CSS dokumentu [12].

```
1 /* CSS document */
2 @-webkit-keyframes {
3     .zelena {
4           background-color: green;
5       }
6     .oranzova {
7           background-color: orange;
8       }
9 }
```

Příklad 25: Nastavení stylů pro WebKitové prohlížeče

4.4.3 Internet Explorer

Internet Explorer využívá jako vykreslovací jádro Trident. Zápis v kaskádových stylech, jako je u výše uvedených prohlížečů, pro tento engin není. Doporučuje se vytvořit nový CSS dokument a v HTML dokumentu na něj odkazovat pomocí podmíněného komentáře.

```
1 <!DOCTYPE html>
2 <html lang="cs">
3 <head>
4    <meta charset="utf-8" />
```

```
5     <!--[if IE]> <link rel="stylesheet" href="styl-pro-
          explorer.css"> <![endif]-->
6
7     <title>Podmíněný komentář s odkazem na CSS pro
          Internet Explorer</title>
8  </head>
9  <body>
10    .
11    .
12    .
```

Příklad 26: Vymezení stylů pro Internet Explorer

4.5 Novinky v CSS3

CSS3 přišlo s několika novinkami, které zjednodušují zápis celého kódu. Právě před vývojem této verze se často na různé vychytávky jako je např. stín v pozadí používal JavaScript, kde byl kolikrát poměrně dlouhý kód. Dnes to jde napsat velmi zkráceně a jednoduše.

Novinky v CSS3 představují již uvedený stín objektu, zakulacené ohraničení, průhlednost objektu, animace, nastavování barev pomocí různých modelů barev a další [13].

Příklad: V HTML dokumentu je načtený obrázek, na které budou vyzkoušeny různé nové stylování v CSS3.

```
1  <!DOCTYPE html>
2  <html lang="cs">
3  <head>
4     <meta charset="utf-8" />
5     <link rel="stylesheet" href="styly.css" type="text/css"
          media="screen" />
6
7     <title>Barevná výplň buněk v tabulce</title>
```

```
8  </head>
9  <body>
10     <section>
11         <img class="tucnak" src="tucnak.jpg" alt="tucnak"
               title="tučňák" />
12     </section>
13 </body>
14 </html>
```

Příklad 27: Načtení obrázku v HTML dokumentu

```
1  /* CSS document */
2  // nastavení stylů pro obrázek tučňák
3  .tucnak {
4      border: 2px solid gray; // nastavení ohraničení
           obrázku
5      box shadow: 8px 8px 15px 3px gray; // nastavení stínu
           obrázku (x, y, průhlednost, velikost stínu, barva)
6      border radius: 15px; // nastavení zaoblení rohů
7      opacity: 0.5; // nastavení průhlednosti obrázku
8  }
```

Příklad 28: Využití nových vlastností ve stylech

(a) Ohraničení obrázku

(b) Stín obrázku

(c) Zaoblení rohů obrázku

(d) Průhlednost obrázku

Jelikož ne všechna nová stylování se zobrazují správně ve všech prohlížečích, mohou se dávat před danou vlastnost stylu tzv. prefixy. Označení je podobné jako v předchozí podkapitole (4.4 na straně 40) jen s tím rozdílem, že se prefix pro daný prohlížeč jen předřadí před vlastnost CSS a nemusí se dávat do složných závorek.

```
1 /* CSS document */
2 // nastavení stínu pro jednotlivé prohlížeče
3 .tucnak {
4    -webkit- box shadow: 8px 8px 15px 3px gray;   //
       nastavení stylu pro WebKitové prohlížeče
5    -moz- box shadow: 8px 8px 15px 3px gray;   // nastavení
        stylu pro Moz prohlížeče
6    -ms- box shadow: 8px 8px 15px 3px gray; // nastavení
       stylu pro Internet Explorer
7    box shadow: 8px 8px 15px 3px gray;  // obecné
       nastavení stylu
```

```
8 }
```

Příklad 29: Použití prefixů v CSS3

Poznámka: Obecné nastavení stylu by mělo být umístěno vždy nakonec po přidaných prefixech.

Další novinky s ukázkami jsou v tomto článku.

4.6 Shrnutí

Vlastností v kaskádových stylech je velké množství, proto jsou v této kapitole pomocí příkladů představeny jen některé. Cílem je pochopit způsob zápisu CSS a co se stane a změní při nastylování nějakého objektu na stránce. Přehled vlastností CSS s názornými ukázkami je například zde.

5 Zápis kódu v JavaScriptu

Ke znalosti zápisu JavaScriptového kódu, je třeba ovládat i HTML kód. A to především proto, že právě JavaScript s HTML kódem spolupracuje. Ať už třeba jde o vypisování nějakého textu nebo upravování stylu pomocí objektu z HTML. Stejně jako CSS i JavaScript je možné zapisovat přímo do HTML dokumentu, nebo ho psát zvlášť do externího souboru `.js` (příklad na straně 14). Pokud se jedná o jednoduchý příklad, není třeba vytvářet samostatný dokument, ale pokud už se jedná o delší kód, tak pro přehlednost a omezení chyb se již nový skriptový dokument používá. V této práci se používá externí dokument, jelikož pro vytvoření her je třeba poměrně obsáhlý JavaScriptový kód.

Pro zjednodušení zápisu se v JavaScriptu používají tzv. aritmetické operátory [15], které upravují hodnoty proměnných (kapitola 5.1).

Operátor	**Funkce**
x++	x se zvětší o 1
x–	x se zmenší o 1
x += y	k x se přičte hodnota y
x -= y	od x se odečte hodnota y
x *= y	x se vynásobí s hodnotou y
x /= y	x se vydělí hodnotou y

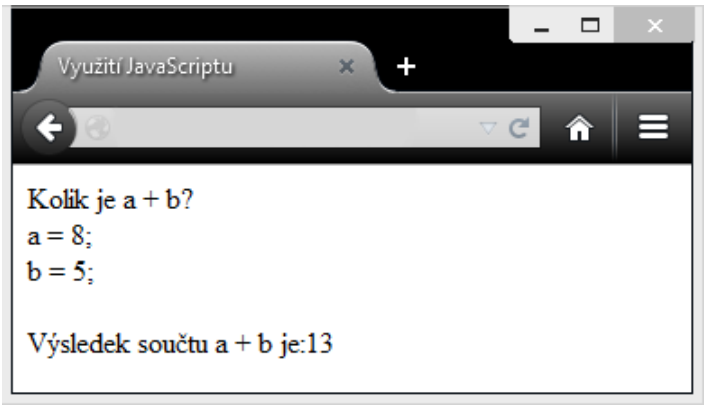
Tabulka 1: Aritmetické operátory

5.1 Proměnné

Aby bylo možné začít programovat v JavaScriptu, je třeba znát základní typy a klíčová slova kódu. Jedním ze základních znalostí je právě práce s proměnnou a její deklarování. Na deklarování proměnné slouží klíčové slovo *var*, za které se píše název proměnné a ta se může rovnat nějaké hodnotě, ale to již není podmínkou.

```
1  <!DOCTYPE html>
2  <html lang="cs">
3  <head>
4          <meta charset="utf-8" />
5      <title>Využití JavaScriptu</title>
6  </head>
7  <body>
8          Kolik je a + b?
9                  <br /> <!-- odřádkování -->
10         a = 8;
11                 <br />
12         b = 5;
13                 <br />
14                 <br />
15 </body>
16 <script language="JavaScript" type="text/javascript">
17         // deklarování proměnných
18         var a = 8;
19         var b = 5;
20         var vysledek;
21
22         // pokud jsou již proměnné nadeklarované, může se
               s nimi dále pracovat
23         vysledek = a + b;        // práce s proměnnými
24
25         document.write("Výsledek součtu a + b je:" +
               vysledek)
26 </script>
27 </html>
```

Příklad 30: Deklarace proměnných



Obrázek 6: Využití proměnných v JavaScriptu

5.2 Funkce

Funkce všeobecně v programování se používají pro jednodušší zápis kódu, aby se daný kód, který se bude opakovat častěji, nemusel psát víckrát a tím se mohou ušetřit chyby i čas. Stačí potom jen použít jméno té funkce.

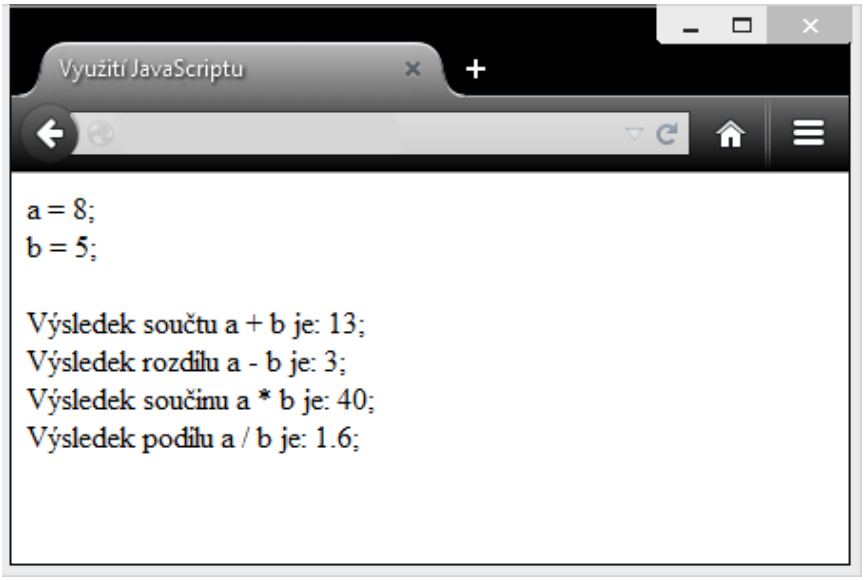
Funkce se deklaruje slovem *function*, za které se píše jméno dané funkce a kulaté závorky *( )*, do nichž se mohou psát parametry, ale také nemusí. Poté následuje tělo funkce, které je ohraničeno složenými závorkami {}. Do těla funkce se píšou veškeré příkazy, které má daná funkce obsahovat.

```
1  .
2  .
3  .
4  < body >
5            a = 8;
6                     <br />
7            b = 5;
8                     <br />
9                     <br />
10 < /body >
11 < script  language= "JavaScript"  type= "text/ javascript ">
12           function  Soucet(a,b)     // funkce s názvem Součet
               a parametry a, b
```

```
13          {
14                  var soucet = a + b;      // tělo funkce
15                  return soucet;
16          }
17
18          function  Rozdil(a,b)
19          {
20                  var rozdil = a - b;
21                  return rozdil;
22          }
23
24          function  Soucin(a,b)
25          {
26                  var soucin = a * b;
27                  return soucin;
28          }
29
30          function  Podil(a,b)
31          {
32                  var podil = a / b;
33                  return podil;
34          }
35
36          var a = 8;
37          var b = 5;
38
39          // zde probíhá volání funkcí
40          document.write("Výsledek součtu a + b je: " +
               Soucet(a,b) + ";<br>");
41          document.write("Výsledek rozdílu a - b je: " +
               Rozdil(a,b)+ ";<br>");
42          document.write("Výsledek součinu a * b je: " +
               Soucin(a,b)+ ";<br>");
43          document.write("Výsledek podílu a / b je: " +
```

```
            Podil(a,b)+ ";<br>");
44  </script>
45  </html>
```

Příklad 31: Deklarace funkcí a jejich využití



Obrázek 7: Využití funkcí v JavaScriptu

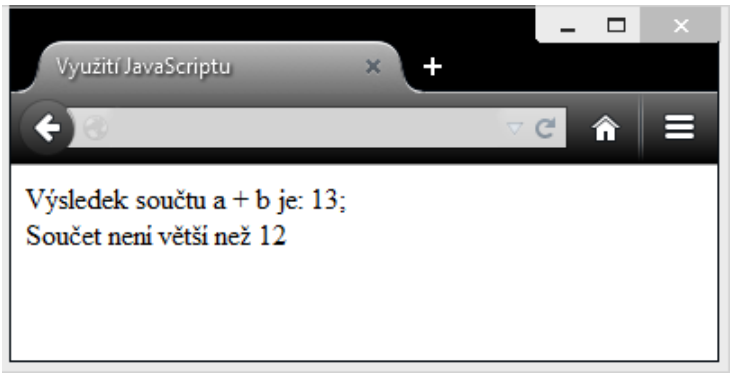
5.3 Podmínky

Podmínky se využívají tehdy, pokud se má vykonat nějaký příkaz jen tehdy, pokud jsou splněný nějaké parametry (podmínky). Začátek podmínky se značí slovem *if* („jestliže“), za kterým následuje v kulatých závorkách daná podmínka. Většinou za podmínku se píšou složené závorky, do kterých se zadávají příkazy, ale pokud je pouze jeden příkaz, složené závorky nejsou třeba. Dále se zapisuje, co se má stát, pokud podmínka nevyhovuje, a to příkazem *else* („jinak“), za který se může a nemusí psát (stejně jako u *if*) složené závorky, kde jsou umístěny příkazy nebo příkaz.

```
1  .
2  .
```

```
3  .
4  <body>
5  </body>
6  <script language="JavaScript" type="text/javascript">
7     function Soucet(a,b)
8           {
9                 var soucet = a + b;
10                return soucet;
11          }
12
13          var a = 8
14          var b = 5;
15          document.write("Výsledek součtu a + b je: " +
               Soucet(a,b) + ";<br>");
16
17          if(Soucet(a,b) < 12)
18                document.write("Součet je větší než 12");
19          else
20                document.write("Součet není větší než 12"
                  );
21 </script>
22 </html>
```

Příklad 32: Využití podmínky v JavaScriptu



Obrázek 8: Využití podmínky v JavaScriptu

5.4 Cykly

Cykly jsou velmi častým kódem, který se běžně v programování používá, a to na opakování určité činnosti několikrát za sebou.

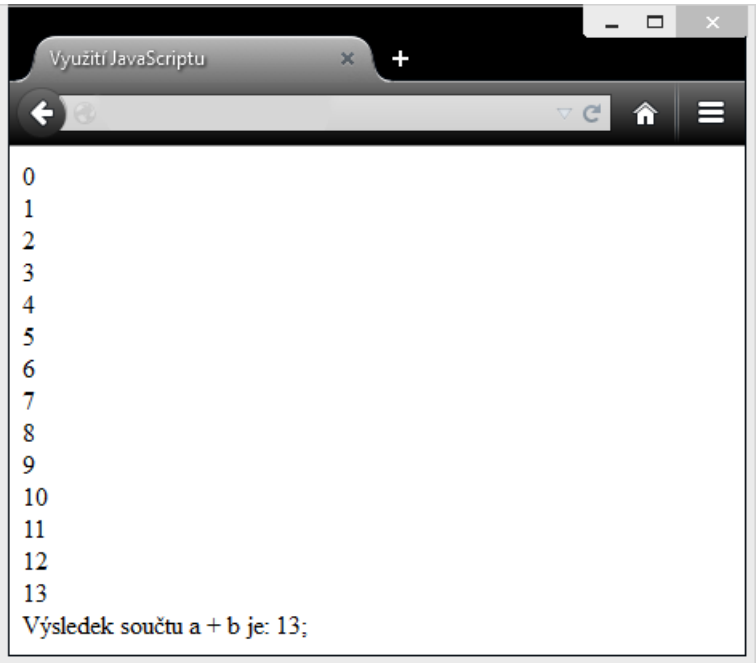
5.4.1 For cyklus

For cyklus se používá převážně tehdy, když počet opakování je daný. Zápis tohoto cyklu začíná slovem *for*, za které se píše do kulatých závorek nastavení proměnné, podmínka, přičtení a poté již do složených závorek příkazy, které mají být v cyklu.

```
1  .
2  .
3  .
4  < body >
5  < /body >
6  < script  language= "JavaScript"  type= "text/ javascript ">
7     function  Soucet(a,b)
8           {
9                   var  soucet  =  a  +  b;
10                  return  soucet;
11          }
12
```

```
13          var a = 8;
14          var b = 5;
15
16          for(i=0; i<=Soucet(a,b); i++ )  // vypisování
               čísel od 0 vždy o jedno větší (i++), dokud
               číslo (i) bude menší nebo rovno funkci Soucet(
               a,b)
17          {
18                  document.write(+ i + "<br>");
19          }
20
21          document.write("Výsledek součtu a + b je: " +
               Soucet(a,b) + ";<br>");
22  </script>
23  </html>
```

Příklad 33: Využití for cyklu



Obrázek 9: Využití for cyklu v JavaScriptu

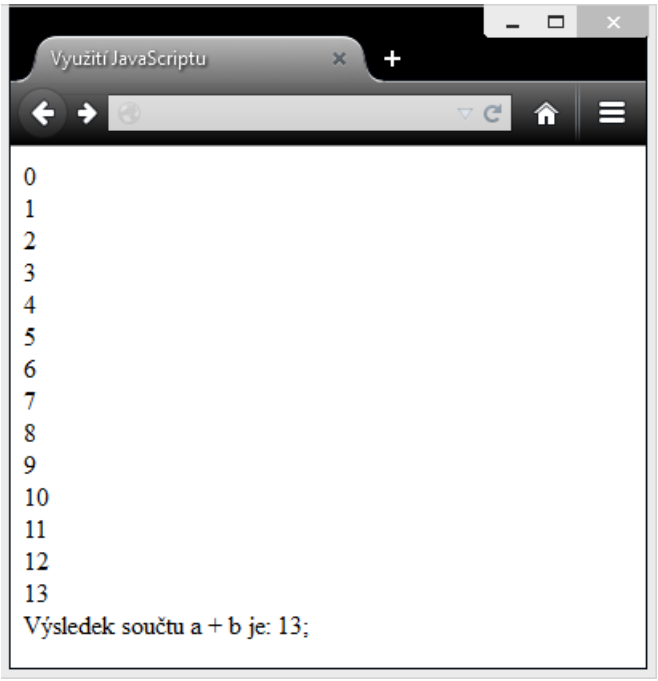
5.4.2 While cyklus

While cyklus je cyklus, který má podmínku na začátku. Příkazy napsané v těle cyklu se opakují tak dlouho, dokud není dané podmínka splněna. Zápis začíná slovem while, následují kulaté závorky, ve kterých je zapsaná podmínka a poté ve složených závorkách jsou příkazy.

```
1  .
2  .
3  .
4  <body>
5  </body>
6  <script language="JavaScript" type="text/javascript">
7     function Soucet(a,b)
8           {
9                   var soucet = a + b;
10                  return soucet;
11          }
12
13          var a = 8;
14          var b = 5;
15          var vysledek = 0;
16
17            document.write("0 <br>");
18
19          while(vysledek < Soucet(a,b))
20          {
21                  vysledek ++;     // dokud vysledek bude
                       menší než funkce Soucet(a,b), bude se
                       zvětšovat o 1
22                  document.write(+ vysledek + "<br>");
23          }
24
25          document.write("Výsledek součtu a + b je: " +
               Soucet(a,b) + ";<br>");
```

```
26
27 < /script >
28 < /html >
```

Příklad 34: Využití while cyklu



Obrázek 10: Využití while cyklu v JavaScriptu

5.5 Shrnutí

Možností zápisu v JavaScriptu je mnoho, proto v kapitolách výše jsou uvedeny jen ty nejdůležitější zápisy kódu. V popisu her (strana 59) je JavaScript popsán více, a to přímo na konkrétních příkladech z ukázek HTML5 her.

6 Výhody a nevýhody vytváření her v HTML 5 oproti technologii Flash

V posledních letech dochází ke spekulacím, jestli HTML5 kompletně nahradí stále populární a držící se technologii Flash, která je využívána na reklamy, hry ale třeba i na přehrávání videí. Ovšem jak HTML5 tak i Flash mají své výhody i nevýhody.

6.1 Adobe Flash

Flash technologie vznikla již v roce 1994, tehdy byla známá jako Macromedia Flash. V roce 2005 si Flash osvojila společnost Adobe Systems, která ho přejmenovala na Adobe Flash.

Jedná se o grafický vektorový program, který slouží k tvorbě reklam, interaktivním animacím, prezentacím a her. Dříve velkou výhodou Flashe byla malá velikost souborů používaných pro hry jak internetových, tak klasických a zároveň možnost hraní v okně prohlížeče bez stahování hry do počítače [16].

6.2 Mobilní zařízení a Flash Player

Jednou z největších nevýhod Flash her je nefunkčnost na mobilním operačním systému jak u Applu na iOS tak již i na Android systému.

6.2.1 iOS zařízení

V roce 2007 společnost Apple představila první řadu iPhonů, které technologii Flash nepodporovaly (v té dobře zřejmě kvůli výkonu mobilního zařízení). Ovšem v roce 2010 Steve Jobs oznámil, že tuto technologii nebude firma Apple podporovat ani v budoucnosti jak u iPhonů, iPodů i iPadů. Prvním důvod, proč Apple nechce využívat Flash, je otevřenost kódu. Do kódu může zasahovat pouze společnost Adobe, tudíž veškeré vylepšení, ceny atd. je v rukou právě této společnosti. Druhým důvodem je, že Flash oznámil, že Apple nemůže získat plný přístup k webovým aplikacím, protože 75 % videí je na webu ve Flash

technologii. Stejně tak hry. Třetím důvodem je bezpečnost, výkon a spolehlivost. Adobe Flash v roce 2009 bylo uvedeno jako jedno z nejméně zabezpečených aplikací, proto Apple uvedl, že nechce snižovat bezpečnost a spolehlivost u svých zařízení. Čtvrtým důvodem je životnost baterie při přehrávání Flash videa, kdy zařízení musí dekódovat video v softwaru, a to značně ubírá energii. Pátý důvod je ovládání dotykových obrazovek, kde Flash je programovaný na ovládání myší a ne dotykem. A pro Apple nejdůležitějším důvodem je zjištění, že Flash chtěl nechat vývojářům (třetí straně) upravovat aplikace pro iPhony, iPody a iPady [17].

6.2.2 Android zařízení

Do roku 2012 Android jako jeden z mála mobilních operačních systémů neměl problém s Adobe Flash Playerem. Ovšem kromě verze Android Jelly Bean, která tuto technologii nikdy podporovala a neexistují k ní ani instalační soubory pro tuto technologii. Ovšem další aplikace Adobe společnosti na Androidu jsou stále k dispozici např. Adobe Reader.

6.2.3 Konec vývoje Flash Playeru pro mobilní zařízení

Společnost Adobe ukončila vývoj Flash Playeru 15.8.2012. Od tohoto data nejsou k dispozici žádné verze tohoto pluginu. Důvodem je kompatibilita Flashe na mobilních zařízeních, kterou Flash nedokázal zajistit. Především šlo o problematiku s rozlišením u dotykových zařízení, kde nastal problém se shodným zobrazením jedné stránky na různých zařízeních s libovolným rozlišením [18].

6.3 Vývoj HTML5

Finální specifikace HTML5 byla vydána 28.10.2014, s čímž přišla poměrně velká změna ohledně samotného jazyka HTML a přehrávání multimédií. Do té doby byl potřeba k přehrávání například právě Flash, ale HTML5 již pracuje s tagy <*audio*> a <*video*>. Jediná možná malá nevýhoda je, že je třeba na-

hrát více formátů jak videí tak hudby, protože každý prohlížeč primárně může podporovat jiný formát.

V současnosti se velmi zlepšila a rozšířila podpora právě HTML5 u webových prohlížečů. Důkazem je toho i například server YouTube společnosti Google. Ačkoliv YouTube využívalo HTML5 už od roku 2010, byla to jen druhá varianta spouštění videí, ale na prvním místě byl Flash. Dnes je to přesně naopak. Od roku 2015 se videa pouští již ve formě HTML5 a Flash je druhou variantou. Důvodem je především rychlost načítání videí a také úspora energie (využití procesoru).

Další výhodou je vykreslovací prostředí se značkou <*canvas*>, které umí zobrazit 2D a 3D grafiku (s využitím WebGL).

6.4 Shrnutí

Pro shrnutí této kapitoly jsou zde vypsány nejdůležitější výhody vytváření her v HTML5 oproti Flashi.

Výhody HTML5 oproti Flash technologii

- asi největší výhodou HTML5 oproti Flashi je podpora na všech mobilních zařízeních i tabletech, což platí nejen pro samotné prohlížení webových stránek, ale samozřejmě i pro hraní HTML5 her
- nižší hardwarové nároky
- HTML5 není softwarem třetí strany

Nevýhody HTML5 oproti Flash technologii

- možné problémy s novými kaskádovými styly díky různé podpoře prohlížečů
- horší podpora ve starších prohlížečích (Internet Explorer 8, ...)

7 Popis a ukázky her z praktické části

7.1 Připojení knihoven jQuery a Box2D

JavaScriptové knihovny se připojují stejným způsobem jako externí JavaScriptový dokument (strana 14).

7.1.1 jQuery

Pro použití a stažení nejnovější verze jQuery je nejlepší sledovat oficiální stránky pro tuto knihovnu, kde jsou informace o aktuální verzi a možnost stažení i starších verzí. V této práci je použity verze jQuery 1.6 a jQuery 2.1.3.

```
1 <script src="jquery-2.1.3.min.js"></script>
```

Příklad 35: Připojení knihovny jQuery

7.1.2 Box2D

Tato knihovna je také volně dostupná na stránkách Box2D pro JavaScript spolu s informacemi k této knihovně. Jelikož se jedná o knihovnu s mnoha funkcemi a možnostmi využití, nejedná se pouze o jeden JavaScriptový soubor. Proto je lepší vytvořit si složku zvlášť s JavaScriptovými soubory, ve které bude ještě k tomu další složka s JavaScriptovými soubory Box2D. V této práci jsou využity tyto soubory:

```
1 <!--JavaScriptové soubory jsou umístěny v hlavní složce "
     box2d" a dále se dělí do dalších složek "common", "
     collision", "dynamics" a další -->
2 <script src='box2d/common/b2Settings.js'></script>
3 <script src='box2d/common/math/b2Vec2.js'></script>
4 <script src='box2d/common/math/b2Mat22.js'></script>
5 <script src='box2d/common/math/b2Math.js'></script>
6 <script src='box2d/collision/b2AABB.js'></script>
7 <script src='box2d/collision/b2Bound.js'></script>
8 <script src='box2d/collision/b2BoundValues.js'></script>
```

```
9  <script src='box2d/collision/b2Pair.js'></script>
10 <script src='box2d/collision/b2PairCallback.js'></script>
11 <script src='box2d/collision/b2BufferedPair.js'></script>
12 <script src='box2d/collision/b2PairManager.js'></script>
13 <script src='box2d/collision/b2BroadPhase.js'></script>
14 <script src='box2d/collision/b2Collision.js'></script>
15 <script src='box2d/collision/Features.js'></script>
16 <script src='box2d/collision/b2ContactID.js'></script>
17 <script src='box2d/collision/b2ContactPoint.js'></script>
18 <script src='box2d/collision/b2Distance.js'></script>
19 <script src='box2d/collision/b2Manifold.js'></script>
20 <script src='box2d/collision/b2OBB.js'></script>
21 <script src='box2d/collision/b2Proxy.js'></script>
22 <script src='box2d/collision/ClipVertex.js'></script>
23 <script src='box2d/collision/shapes/b2Shape.js'></script>
24 <script src='box2d/collision/shapes/b2ShapeDef.js'>
     </script>
25 <script src='box2d/collision/shapes/b2BoxDef.js'></script
     >
26 <script src='box2d/collision/shapes/b2CircleDef.js'>
     </script>
27 <script src='box2d/collision/shapes/b2CircleShape.js'>
     </script>
28 <script src='box2d/collision/shapes/b2MassData.js'>
     </script>
29 <script src='box2d/collision/shapes/b2PolyDef.js'>
     </script>
30 <script src='box2d/collision/shapes/b2PolyShape.js'>
     </script>
31 <script src='box2d/dynamics/b2Body.js'></script>
32 <script src='box2d/dynamics/b2BodyDef.js'></script>
33 <script src='box2d/dynamics/b2CollisionFilter.js'>
     </script>
34 <script src='box2d/dynamics/b2Island.js'></script>
```

```
35 <script src='box2d/dynamics/b2TimeStep.js'></script>
36 <script src='box2d/dynamics/contacts/b2ContactNode.js'>
       </script>
37 <script src='box2d/dynamics/contacts/b2Contact.js'>
       </script>
38 <script src='box2d/dynamics/contacts/b2ContactConstraint.
       js'></script>
39 <script src='box2d/dynamics/contacts/
       b2ContactConstraintPoint.js'></script>
40 <script src='box2d/dynamics/contacts/b2ContactRegister.js
       '></script>
41 <script src='box2d/dynamics/contacts/b2ContactSolver.js
       '></script>
42 <script src='box2d/dynamics/contacts/b2CircleContact.js
       '></script>
43 <script src='box2d/dynamics/contacts/b2Conservative.js'>
       </script>
44 <script src='box2d/dynamics/contacts/b2NullContact.js'>
       </script>
45 <script src='box2d/dynamics/contacts/
       b2PolyAndCircleContact.js'></script>
46 <script src='box2d/dynamics/contacts/b2PolyContact.js'>
       </script>
47 <script src='box2d/dynamics/b2ContactManager.js'></script
       >
48 <script src='box2d/dynamics/b2World.js'></script>
49 <script src='box2d/dynamics/b2WorldListener.js'></script>
50 <script src='box2d/dynamics/joints/b2JointNode.js'>
       </script>
51 <script src='box2d/dynamics/joints/b2Joint.js'></script>
52 <script src='box2d/dynamics/joints/b2JointDef.js'>
       </script>
53 <script src='box2d/dynamics/joints/b2DistanceJoint.js'>
       </script>
```

```
54 <script src='box2d/dynamics/joints/b2DistanceJointDef.js
   '></script>
55 <script src='box2d/dynamics/joints/b2Jacobian.js'><
   /script>
56 <script src='box2d/dynamics/joints/b2GearJoint.js'><
   /script>
57 <script src='box2d/dynamics/joints/b2GearJointDef.js'><
   /script>
58 <script src='box2d/dynamics/joints/b2MouseJoint.js'><
   /script>
59 <script src='box2d/dynamics/joints/b2MouseJointDef.js'><
   /script>
60 <script src='box2d/dynamics/joints/b2PrismaticJoint.js'><
   /script>
61 <script src='box2d/dynamics/joints/b2PrismaticJointDef.js
   '></script>
62 <script src='box2d/dynamics/joints/b2PulleyJoint.js'><
   /script>
63 <script src='box2d/dynamics/joints/b2PulleyJointDef.js'><
   /script>
64 <script src='box2d/dynamics/joints/b2RevoluteJoint.js'><
   /script>
65 <script src='box2d/dynamics/joints/b2RevoluteJointDef.js
   '></script>
```

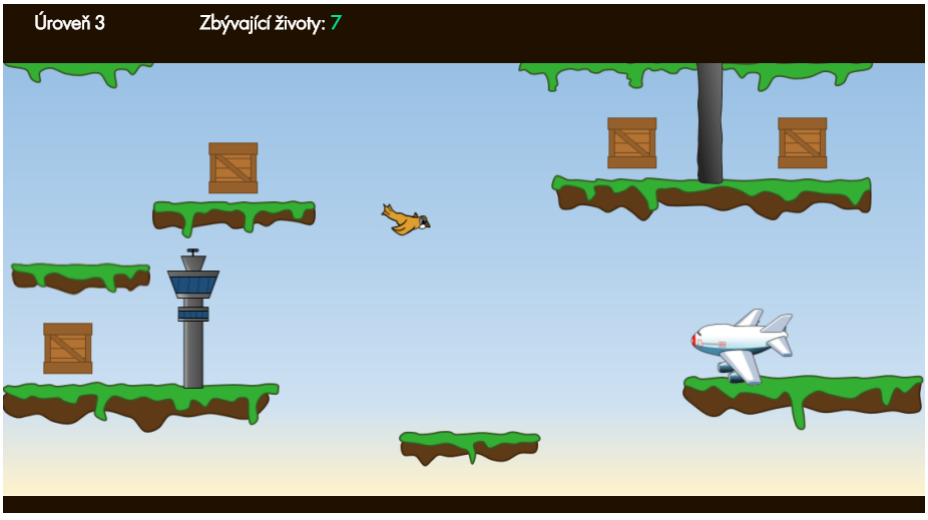
Příklad 36: Připojení knihovny Box2D

7.2 Letadlo

Tato hra závisí na pohybu letadla. Celý děj hry je řízený enginem Box2D, který nastavuje různé typy objektů. Letadlo, pták, kamenná brána a pohybující se ostrov jsou nastaveny jako pohybující se objekty a ostatní ostrovy, věž a bedny jsou nastaveny jako pevné objekty. Dále tento engine zajišťuje gravitaci ve světě (herním poli) této hry, detekci srážek objektů a také třeba sílu zrychlování objektů. Pro tuto hru jsou nastaveny tři úrovně.

Letadlo se ovládá šipkami nahoru, doleva a doprava. Pokud letadlo spadne na hranu dolní hrací plochy, přijde o život a daná úrovneň hry se opakuje. Dále se letadlo nesmí dotknout ptáka, kamenné brány a létajícího ostrova, jinak opět letadlo přijde o život a daná úrovneň se bude opakovat.

Letadlo má nastaveno sedm životů, které když hráč vyčerpá, zobrazí se závěrečné okno se zprávou, že hráč prohrál. Ovšem když se hráč dostane do cíle, také se zobrazí závěrečné okno s výsledným časem a zároveň pokud má hráč lepší čas než v předchozí hře, zobrazí se stuha s informací, že hráč dosáhl nového rekordu.



Obrázek 11: Náhled hry Letadlo

7.2.1 Definování objektů

Veškeré objekty jsou definovány pomocí enginu Box2D. Je nastavena například i hustota objektu a tření. Pokud se nenastavuje velikost a pozice objektu pro každý level zvlášť, mohou se i při definování objektu nastavit tyto konkrétní údaje.

```
1  function  letadlo(x, y)
2  {
3  // definice tvaru auta
4          var boxSd = new b2BoxDef();
5          boxSd.density = 1.0;
6          boxSd.friction = 1.5;
7          boxSd.restitution = .4;
8          boxSd.extents.Set(50, 30);
9          boxSd.userData = document.getElementById("letadlo
               ");
10 // definice objektu auta
11         var boxBd = new b2BodyDef();
12         boxBd.AddShape(boxSd);
13         boxBd.position.Set(x,y);
14         var letadlo = letadloHra.world.CreateBody(boxBd);
15         return letadlo;
16 }
```

Příklad 37: Definování objektu pomocí enginu Box2D

7.2.2 Animace objektů

Ve hře je použito několik animací. Jedna z nich je animace bedny, která se neustále točí dokola. To je uděláno pomocí kódu v JavaScriptu, který je uvedený v následujícím příkladu. Jsou načteny čtyři obrázky, které se mění podle velikosti `i`. Pokud `i` je větší než počet obrázků, nastaví se na nulu a vše běží od znova. Příkaz *setTimeout()* je časový interval použitý na rychlost měnění obrázků (otáčení bedny).

```
var quotes = new Array ("box1.PNG", "box2.PNG", "box3.PNG
   ", "box4.PNG");
var i = 0;
function animaceBedna()
{
   document.getElementById("bedna").src = ( "images/" +
      quotes[i] );
     if (i<3)
       i++;
     else
       i = 0;
   setTimeout("animaceBedna()", 300);
}
```

Příklad 38: Animace objektu

Stejně tak je napsaná animace pro ptáka a řídící věž. Dále pro hýbaci objekty je nastavený pohyb. Pro ptáka je nastavený pohyb po ose X a pro ostrov a kamennou bránu je nastavený pohyb po ose Y. Vše je naprogramováno pomocí enginu Box2D v JavaScriptu.

```
function PohybPtak()
{
  if(letadloHra.pohyb < 100)
  {
    if(letadloHra.pohyb < 51 )
    {
      var vel = new b2Vec2(50,-16);
      letadloHra.ptak.SetLinearVelocity (vel);
      letadloHra.pohyb++;
    }
    else
    {
      var vel = new b2Vec2(-50, -16);
      letadloHra.ptak.SetLinearVelocity (vel);
```

```
15          letadloHra.pohyb++;
16        }
17    }
18    else
19     letadloHra.pohyb = 0;
20  }
```

Příklad 39: Pohyb objektu

Pozice pro každý level je nastavený stejně jako ve hře Auto (strana 76).

7.2.3 Detekce srážek objektů

Pro funkci step je nastaven časový interval, ve kterém se tato funkce opakuje každých 10 milisekund. Právě v této funkci jsou nastaveny podmínky ve kterých se zjišťuje, zda letadlo narazilo do nějakého objektu nebo ne.

Jeden typ podmínek je pro srážku letadla s bednami. Pokud letadlo narazí do bedny, přičte se i a hra pokračuje dál. Pokud jsou posbírané všechny bedny a letadlo narazí do řídící věže, tak se hra přesouvá do další úrovně (v poslední úrovni se zobrazí závěrečné okno). Pokud se ale letadlo dotkne řídící věže a nejsou posbírány všechny bedny, neprovede se nic.

```
1  if ((body1 == letadloHra.letadlo && body2 == letadloHra.
       bedna5)
2          || (body2 == letadloHra.letadlo && body1 ==
              letadloHra.bedna5))
3  {
4          letadloHra.world.DestroyBody(letadloHra.bedna5);
5          letadloHra.i++;
6          bednaMusic();    // hudba po doteku letadla a
              bedny
7  }
```

Příklad 40: Detekce srážky objektů

Dále je pro hru nastavaná hudba pomocí HTML5 tagu <audio>, která se ovládá pomocí ikony Hudba a také tlačítka pro ovládání hry na dotykovém za-

řízení. Tlačítka slouží místo klávesnice a zobrazují se pouze tehdy, pokud hráč klikne na ikonu Mobil. Hra také obsahuje informace o ovládání a pravidlech hry, které jsou skyté a po kliknutí na ikonu Informace se popis zobrazí.

7.2.4 Shrnutí

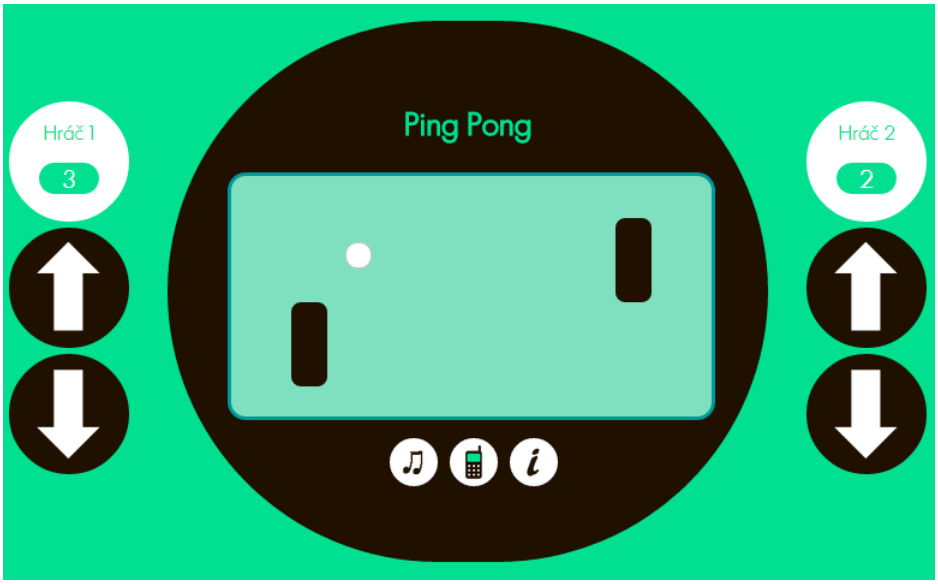
Celý kód hry Letadlo je možné si prohlédnout přímo na webu HTML5-hry.cz zde.

7.3 Ping Pong

Celý průběh hry závisí na pohybu míčku, který má být odrážen pálkami, se kterými pohybují hráči pomocí klávesnice. Ve hře jsou právě pro míček nastaveny podmínky, co se má stát, když míček narazí buď do hran herního pole nebo do pálek. Pokud míček narazí do bočních hran herního mole, přesouvá se do výchozí polohy a přičte se skóre hráči s pálkou na druhé straně. Dále pokud míček narazí do horní nebo dolní hrany herního pole, má se pouze odrazit (tzn. otočit směr). To samé se má stát při nárazu míčku do pálek.

Pálky jsou nastaveny tak, že se jejich pozice mění o pět pixelů nahoru nebo dolu podle toho, jaká klávesa je stisknuta. Konkrétně se jedná o klávesy W (nahoru), S (dolu) pro levou pálku a šipka nahoru, šipka dolů pro pravou pálku. Dále je také nastavena klávesa ENTER pro spuštění hry. Pohyb pálek je dále přizpůsoben i pro dotyková zařízení pomocí příkazu *touch*, který je nastavený pro tlačítka se šipkami nahoru a dolů pro obě pálky na každé straně. Tlačítka jsou nastavená tak, aby se zobrazovala pouze pokud si hráč přímo zvolí (pomocí ikony Mobil), že chce hrát hru právě pomocí tlačítek.

Součástí hry Ping Pong je i hudba, která je vložena pomocí HTML5 tagu <*audio*>, do kterého byly vloženy tři formáty pro přehrání (MP3, OGG, WAV). Více formátů je použito z toho důvodu, že každý prohlížeč upřednostňuje jiný formát. Například Internet Explorer přehrává převážně MP3 formát a ještě k tomu je potřeba, aby tento formát byl umístěn před těmi ostatními.



Obrázek 12: Náhled hry se zobrazenými tlačítky pro dotyková zařízení

7.3.1 Ukázka kódu hry Ping Pong

V následujícím příkladu je ukázka s detekcí srážky míčku a pravé hrany herního pole. Pokud míček narazí do pravé hrany, přičte se hráči skóre a dál se zjišťuje, zda skóre se již nerovná hodnotě 10. Pokud se rovná hodnotě 10, volá se funkce s koncem hry, kde se zobrazí závěrečné okno, že Hráč A vyhrál. Pokud ale skóre je stále menší než 10, míček se přesouvá do původní polohy a hra s přičteným skórem pokračuje dál.

```
1 // pravá hrana herního pole
2 if (micVyska+micVlevo+mic.speed*mic.directionX >
     herniPoleSirka)
3 {
4         pingpong.scoreA++;
5         // podmínka, kdy hráč B prohraje
6         if(pingpong.scoreA == 10)
7         {
8           konecHryHracA();
9         }
```

```
10          else
11          {
12            $("#scoreA").html(pingpong.scoreA);
13
14            // přemístění míčku do výchozí polohy
15            $("#mic").css({
16            "left":"250px",
17            "top":"100px",
18            });
19
20            // načtení původní pozice míčku a směru
21            micNahore = parseInt($("#mic").css("top"));
22            micVlevo = parseInt($("#mic").css("left"));
23            mic.directionX = -1;
24          }
25  }
```

Příklad 41: Ukázka detekce dotyku míčku a pravé hrany herního pole

7.3.2 Shrnutí

Celý kód hry Ping Pong je možné si prohlédnout přímo na webu HTML5-hry.cz zde.

7.4 Pexeso

Princip hry Pexesa spočívá v nalezení všech párů karet v co nejkratší dobu. V této hře k otočení každé karty byla použita animace pomocí kaskádových stylů, která je volána z JavaScriptového dokumentu. Na začátku každé hry se volá funkce se zamícháním karet, aby se v nové hře jejich pozice neopakovaly a nastavuje se rozložení karet určitého počtu řádků a sloupců.

Pro pozadí jednotlivých karet je použit jediný obrázek, ve kterém jsou umístěny všechna zvířata zobrazující se na jednotlivých kartách. To je možné díky nastylování pomocí pozicování pozadí (*background-position*) a udání velikosti každé karty. Tím se najde přímo pozice daného zvířete a není třeba mít několik obrázků najednou (načítání méně dat).

Hra se dá ovládat bez dalších nastavení i na dotykových zařízeních. Dále je pro hru nastavená hudba, která se začne přehrávat automaticky po spuštění hry a dále se ovládá pomocí ikony Hudba.

Obrázek 13: Zvířáta v jednom obrázku pro hru Pexeso

7.4.1 Ukázka kódu hry Pexeso

V následujícím příkladu je kód nastavený styl pro všechny karty s obrázkem zvířete (tedy lícem karty). Zároveň je pro každou takovou kartu nastavená

rotace o 180 stupňů. Jelikož jde o animaci v CSS3, je třeba nastavit prefixy pro každý typ prohlížeče zvlášť.

Dále je v příkladu znázorněno pozicování pozadí pro každý pár karet zvlášť. Jedná se o zvířata v obrázku 13, ze kterého se určují jednotlivé pozice zvířat.

```
1  .licKarty {
2          background: #efefef url(images/karty.png);
3          // animace - přetočení karty
4          -moz- transform : rotate3d(0,1,0,-180deg);
5          -webkit- transform : rotate3d(0,1,0,-180deg);
6           transform : rotate3d(0,1,0,-180deg);
7          z-index: 8;
8  }
9  // nastavení pozice obrázků jednotlivých zvířat z obrázku
       karty.png
10 .medved {background- position: 0px 0px;}
11 .zelva {background- position: -90px 0px;}
12 .kocka {background- position: -180px 0px;}
13 .pes {background- position: -270px 0px;}
14 .slon {background- position: -360px 0px;}
15 .sob {background- position: -450px 0px;}
16 .zirafa {background- position: 0px -90px;}
17 .koza {background- position: -90px -90px;}
18 .klokan {background- position: -180px -90px;}
19 .sova {background- position: -270px -90px;}
20  .tucnak {background- position: -360px -90px;}
21 .ovce {background- position: -450px -90px;}
```

Příklad 42: Ukázka detekce dotyku míčku a pravé hrany herního pole

Pokud se otočí dvě karty se stejným obrázkem, karty se odstraní opět pomocí stylů (*opacity: 0*) a zároveň se hlídá, jestli byl odstraněn poslední pár karet. Pokud byl odstraněn, volá se funkce konec hry a zobrazí se závěrečné okno s uplynulým časem a zároveň se vyhodnocuje, zda má hráč lepší čas než v předchozí hře.

Obrázek 14: Závěrečné okno ve hře Pexeso

7.4.2 Shrnutí

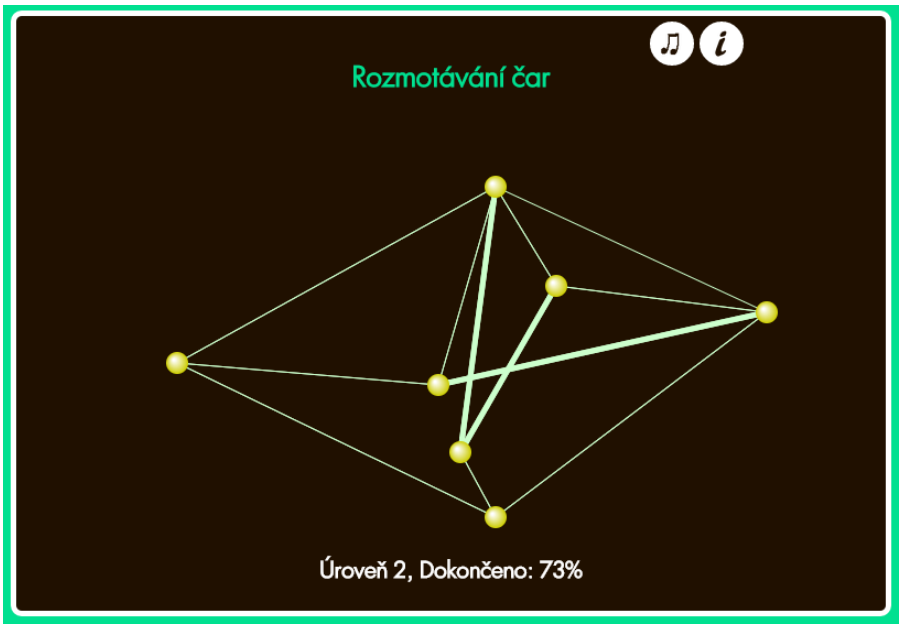
Celý kód hry Pexeso je možné si prohlédnout přímo na webu HTML5-hry.cz zde.

7.5 Rozmotávání čar

V této hře je úkolem rozmotat čáry pomocí tahu s kruhy tak, aby se čáry nepřekrývaly. Pro tuto hru je použito plátno <*canvas*>, což je novinka HTML5. Do něj jsou pomocí JavaScriptu právě objekty typu kruh a čára, pro které je dál vytvořené pole, ze kterého se poté tvoří jednolivé obrazce pro danou úroveň.

Pro tuto hru je nastaveno pět úrovní s různými obrazci. Aby se hráč dostal do další úrovně, musí posouvat žluté kruhy tak, aby čáry, které je spojují se navzájem nepřekrývaly (čáry, které se překrývají, jsou vyznačené větší šířkou, naopak čáry, které se nepřekrývají se ztenčí). Po splnění páté úrovně se zobrazí závěrečné okno, že hráč dokončil všechny úrovně.

K této hře je také hudba, která jse vložena (stejně jako u ostatních her) pomocí HTML5 tagu <*audio*>.



Obrázek 15: Náhler hry Rozmotávání čar

7.5.1 Ukázka kódu hry Rozmotávání čar

V následujícím příkladě je znázorněno vypočítávání, kolik je z dané úrovně dokončeno procent.

```
1  function  dokonceniUrovne()
2  {
3    // určení postupu v rámci úrovně
4    var dokonceno = 0;
5    for (var i=0;i<rozmotavani.cary.length;i++)
6    {
7      if (rozmotavani.cary[i].thickness == rozmotavani.
         tloustkaTenkeCary)
8      {
9        dokonceno++;
10     }
11   }
12   var dokoncenoProcenta = Math.floor(dokonceno/
       rozmotavani.cary.length*100);
13   $("#dokonceno").html(dokoncenoProcenta);
14   // zobrazení aktuální úrovně
15   $("#level").html(untangle.aktualniLevel);
16 }
```

Příklad 43: Ukázka nastavení pozice objektů ve hře Auto

7.5.2 Shrnutí

Celý kód hry Rozmotávání čar je možné si prohlédnout přímo na webu HTML5-hry.cz zde.

7.6 Auto-hra

Tato hra je založena převážně na vlastnostech fyzikálního enginu Box2D. Právě tento engine zajišťuje například gravitaci světa, ve kterém se auto pohybuje. Dále například detekci kolizí, a to když auto narazí do jiného objektu. V této hře jsou nastavené jako pevné objekty ostrovy, po kterých se pohybuje jako druhý typ objektu auto. Dalšími objekty je cíl, dolní a pravá hrana herní plochy. Pozice veškerých objektů, které se ve hře objevují, jsou nastavovány pomocí souřadnic X a Y.

Auto, které se ovládá pomocí klávesnice (konkrétně šipky dolů a šipky nahoru) je poháněno nastavenou silou, díky které auto postupně zrychluje. V této hře je nastaveno pět úrovní, kterými musí auto projet a dostat se pokaždé k objektu cíl. Pokud auto spadne z ostrovů a spadne na konec spodní hrací plochy, daná úroveň se restartuje. To samé se stane, pokud by auto narazilo do pravé hrany hrací plochy. Dále je pro auto nastavené určité množství paliva, které se odebírá podle stisku klávesy, kterou se auto pohybuje.

V této hře je opět nastaveno měření času, kdy na konci hry po splnění všech úrovní se opět vyhodnotí a zobrazí v závěrečném okně, zda je čas lepší, než v předchozí hře.

Herní pole je opět nastaveno pomocí tagu <*canvas*> v HTML dokumentu a dále pomocí tagů <*div*> jsou nastaveny veškeré typy objektů (cíl a auto).

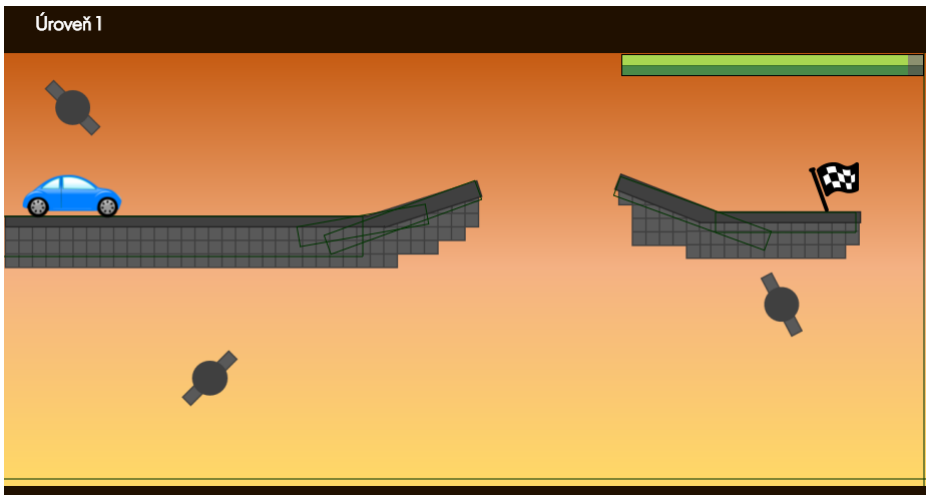
Pro ovládání hry dotykovým zařízením jsou nastavená tlačítka, která se zobrazí po kliknutí na ikonu Mobil. Dále je také nastavená hudba pomocí tagu <*audio*>, která začne hrát automaticky po spuštění hry a informace o možnostech ovládání hry.

7.6.1 Ukázka kódu Auto-hry

V následujícím příkladě je znázorněno umístění objektů v poli v první úrovni hry a odebírání paliva (tmavší část je odebrané palivo). Pokud se jedná o pevný objekt (ostrov - box, cíl - cil, zed - zed' a dno), musí se nastavovat pozice X a Y, výška a šířka objektu a rotace objektu. U objektu auto je nastavena pozice X , Y a množství paliva (to vše viz obrázek 16).

```
1 autoHra.levely[0] = [{"type":"auto","x":50,"y":138,"fuel":20},
2 {"type":"box","x":108, "y":183, "width":250, "height":20, "rotation":0},
3 {"type":"box","x":358,"y":172,"width":65,"height":10,"rotation":-10}, {"type":"box","x":398,"y":164,"width":80,"height":10,"rotation":-20}, {"type":"box","x":687,"y":160,"width":80,"height":10,"rotation":20}, {"type":"box","x":780,"y":169,"width":70,"height":10,"rotation":0}, {"type":"cil","x":828,"y":134,"width":15,"height":20,"rotation":0}, {"type":"zed","x":0,"y":450,"width":1300,"height":25,"rotation":0}, {"type":"dno","x":938,"y":400,"width":20,"height":600,"rotation":0}];
```

Příklad 44: Ukázka nastavení pozice objektů ve hře Auto



Obrázek 16: Náhled nastavení pozice objektů

7.6.2 Shrnutí

Celý kód hry Auto je možné si prohlédnout přímo na webu HTML5-hry.cz zde.

8 Způsob nahrání her na internet

Aby bylo možné zveřejnit hry nebo jakékoliv jiné dokumenty určené pro web, je třeba zajistit doménu, webhosting a způsob, jakým se soubor nebo soubory na daný server od webhostingu nahrají.

8.1 Doména

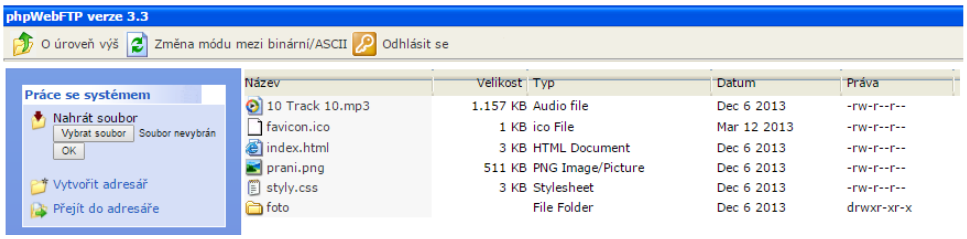
Doména je v podstatě název dané stránky, kterou chceme na internetu zveřejnit. Například doména pro hry k této práci je html5-hry.cz z čehož samotný název html5-hry je doménou druhého řádu a koncovka .cz je doménou prvního řádu a slouží převážně k označení země.

Název domény může být jakýkoliv, ale samozřejmě by měl naznačovat, co je obsahem daných webových stránek a také aby byl název domény zapamatovatelný pro ostatní uživatele, kteří by dané stránky chtěli více navštěvovat. Každá doména musí být jedinečná a nemohou existovat domény se stejným názvem. Proto když si uživatel chce pořídit doménu, měl by si nejdřív zjistit, zda název dané domény není již používán. To se dá nejlépe ověřit na stránkách nic.cz, což je přímo správce domén, nebo by měla být možnost ověření přímo i u konkrétního poskytovatele webhostingu.

8.2 Webhosting

Webhosting je prostor pro soubory, které tvoří webové stránky. Může být placený ale i zdarma.

Webhosting zdarma má omezenější služby než placený webhosting a je určený spíš jen pro pokusné stránky nebo stránky, které tvoří malé soubory. Navíc díky tomu, že se za prostor neplatí, jsou vkládány do obsahu webových stránek uživatele různé reklamy a domény jsou třetího řádu (html5-hry.nazevwebhostingu.cz). Příkladem takového webhostingu je webzdarma.cz, kde nahrávání a aktualizace souborů je zajištěno grafickým webovým rozhraním.

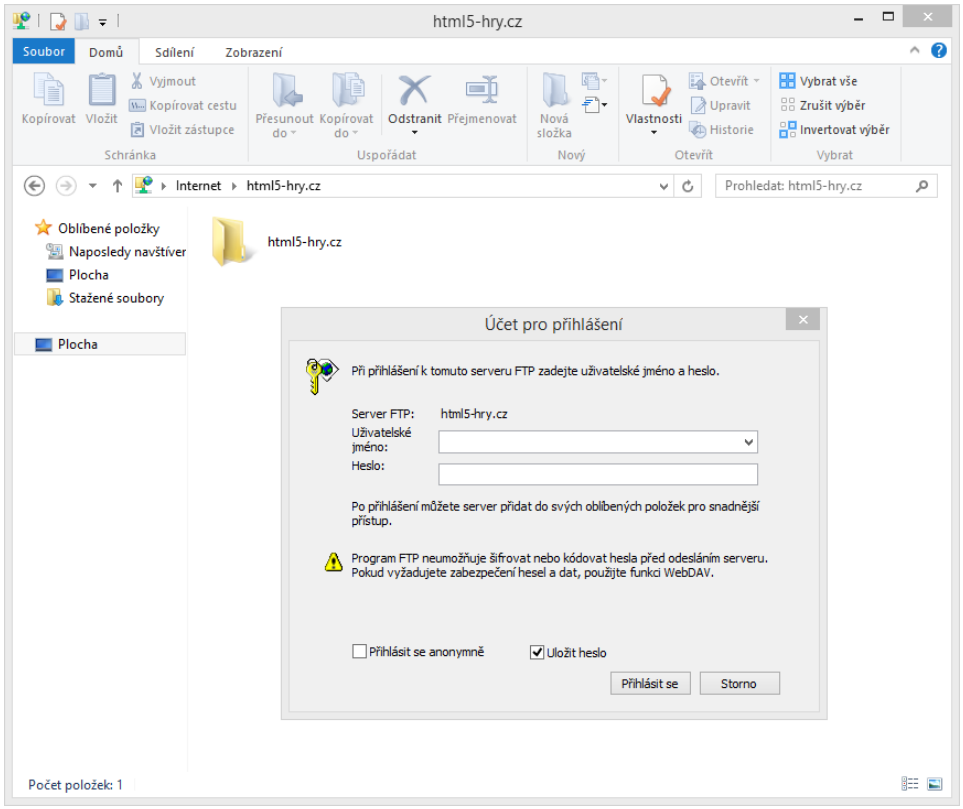


Obrázek 17: Grafické webové rozhraní pro nahrávání a aktualizaci souborů na Webzdarma.cz

Placený webhosting má obrovskou výhodu v kapacitě úložiště na daném serveru a možnost komunikace s pracovníky dané společnosti, kteří uživatelům pomáhají s případnými problémy atd. Dalšími výhodami je doména druhého řádu (bez názvu webhostingu), vlastní email (např. info@html5-hry.cz) a další. Veškeré služby se odvíjí od ceny daného balíčku webhostingu. Příkladem placeného webhostingu je active24.cz, který je využit právě pro stránky html5-hry.cz.

8.3 Nahrávání souborů na server webhostingu přes FTP

Používání FTP (File Transfer Protocol) je asi nejčastější způsob nahrávání a aktualizace souborů na server. K použití FTP je třeba znát FTP adresu na svůj server, login a heslo k přístupu. To vše je poskytnuto webhostingem uživateli, který si zaplatí dané služby v konkrétním balíčku.



Obrázek 18: Přihlašování k vlastnímu serveru přes FTP

9 Závěr

Ačkoliv značkovací jazyk HTML5 nemá ještě mezi programátory webových stránek stoprocentní důvěru, přesto jsou HTML5 hry současností i budoucností webu. Možnosti tohoto značkovacího jazyka se pořád rozrůstají a díky různým knihovnám se na webových stránkách dá dnes celkem jednoduše vytvořit téměř cokoliv.

Díky HTML5 se vyřešil problém s podporou webových stránek a her využívajících technologii Flash, jelikož právě společnost Adobe přestala pracovat a vyvíjet Flash Player pro mobilní zařízení díky neustálým problémům například s bezpečností.

Bakalářská práce obsahuje teoretickou část, která popisuje podrobně způsob zápisu kódu v HTML5 dokumentu, kaskádových stylech včetně novinek v CSS3 a JavaScriptu a dále jsou popsány největší výhody a nevýhody tvorby her v HTML5 oproti Flashi. Další část bakalářské práce je praktická, kde jsou popisy a ukázky her vytvořených právě pro tuto práci a popsán způsob nahrání her na internet. Veškeré hry, které jsou součástí této práce jsou k dispozici na stránkách html5-hry.cz.

Tato práce má sloužit i začátečníkům, kteří by se chtěli naučit vytvořit základní HTML5 hry a nechtějí používat jen editory, které vytvářejí hry bez jakéhokoliv programování ze strany uživatele.

Cíle, které byly stanoveny v zadání bakalářské práce (tedy přiblížit a zpracovat možnosti tvorby her programovaných v HTML5 s využitím knihoven jQuery a Box2D, detailně popsat princip vytváření těchto her, včetně využití kaskádových stylů, porovnat a popsat výhody a nevýhody herních aplikací v HTML5 oproti hrám realizovaných pomocí technologie Flash, vytvořit a popsat několik online her a popsat postup nahrání her na internet), jsou splněny.

Literatura

[1] MAKZAN. HTML5 games development by example: beginner's guide; create six fun games using the latest HTML5, Canvas, CSS, and JavaScript techniques. Birmingham, U.K: Packt Pub, 2011, ix, 331 s. ISBN 978-1-849691-26-0.

[2] SLÁDEK, Honza. Zdroják.cz: Webdesignérův průvodce po HTML5 – díl nultý. ILLICH, Michal a Martin HASSSMAN. Zdroják.cz [online]. 2003, 25.5.2010 [cit. 2015-01-26]. Dostupné z: http://www.zdrojak.cz/clanky/webdesigneruv-pruvodce-po-html5-dil-nulty/

[3] HTML5 Introduction: What is New in HTML5?. W3Schools.com [online]. 1999, 28.8.2014 [cit. 2015-02-26]. Dostupné z: http://www.w3schools.com/html/html5_intro.asp

[4] JQuery: Ajax. JQuery [online]. 2005, 26.1.2015 [cit. 2015-02-27]. Dostupné z: http://learn.jquery.com/ajax/

[5] SMOLA, Martin. Novinky v CSS3: animace. Root.cz [online]. 2010, 5.9.2012 [cit. 2015-02-27]. Dostupné z: http://www.root.cz/clanky/novinky-v-css3-animace/

[6] Box2D: A 2D Physics Engine for Games. [online]. 2011, 23.3.2014 [cit. 2014-03-25]. Dostupné z: http://box2d.org/

[7] JANOVSKÝ, Dušan Yuhů. Editory HTML stránek. Jak psát web: o tvorbě, údržbě a zlepšování internetových stránek [online]. 2003, 2009 [cit. 2015-02-22]. Dostupné z: http://www.jakpsatweb.cz/editory.html

[8] JANOVSKÝ, Dušan Yuhů. Document Type Definition. Jak psát web: o tvorbě, údržbě a zlepšování internetových stránek [online]. 2003, 16.12.2014 [cit. 2015-03-10]. Dostupné z: http://www.jakpsatweb.cz/archiv/doctype.html

[9] JANOVSKÝ, Dušan Yuhů. Čeština / cestina: Diakritika na HTML stránkách. Jak psát web: o tvorbě, údržbě a zlepšování internetových stránek [online]. 2003, 16.12.2014 [cit. 2015-03-10]. Dostupné z: http://www.jakpsatweb.cz/cestina.html

[10] JANOVSKÝ, Dušan Yuhů. Titulek stránky: a proč je důležitý. Jak psát web: o tvorbě, údržbě a zlepšování internetových stránek [online]. 2003, 16.12.2014 [cit. 2015-03-10]. Dostupné z: http://www.jakpsatweb.cz/titulek.html

[11] Český HTML 5 manuál: Layout (rozložení stránky). Itnetwork.cz: Sociální síť pro IT profesionály [online]. 5.12.2012 [cit. 2015-02-22]. Dostupné z: http://www.itnetwork.cz/html-layout-rozlozeni-stranky-cesky-manual/all#komentare

[12] Encyclopedia: browser rendering engine. PCMag Digital Group [online]. 1996, 2015 [cit. 2015-03-26]. Dostupné z: http://www.pcmag.com/encyclopedia/term/61819/browser-rendering-engine

[13] ŠIMEČEK, Martin. CSS3 - držte krok s dobou (nové vlastnosti). Programujte.com [online]. 2003 - 2015, 2. 9. 2010 [cit. 2015-03-26]. Dostupné z: http://programujte.com/clanek/2010070801-css3-drzte-krok-s-dobou-nove-vlastnosti/

[14] JANOVSKÝ, Dušan Yuhů. Funkce v Javascriptu: pro začátečníky. Jak psát web: o tvorbě, údržbě a zlepšování internetových stránek [online]. 2003, 16.12.2014 [cit. 2015-03-10]. Dostupné z: http://www.jakpsatweb.cz/javascript/funkce-prolamy.html

[15] JavaScript - Proměnné: Aritmetické operátory. Tvorba-webu.cz: o webdesignu, grafice a reklamě [online]. 2003 - 2008 [cit. 2015-03-29]. Dostupné z: http://www.tvorba-webu.cz/javascript/vars.php

[16] MIKLÁŠ, Michal. Úvod do programu Adobe Flash: Historie, vývoj a popis. Informatika na Gymnáziu a Jazykové škole s právem

státní jazykové zkoušky Zlín [online]. [cit. 2015-04-03]. Dostupné z: http://www.gjszlin.cz/ivt/esf/flash/flash-uvod-do-programu.php

[17] JOBS, Steve. Thoughts on Flash. APPLE INC. Apple [online]. April, 2010 [cit. 2015-04-03]. Dostupné z: https://www.apple.com/hotnews/thoughts-on-flash/

[18] MALÝ, Martin. „Flash je mrtev, HTML5 není připravené, co teď?“. Zdroják: o tvorbě webových stránek a aplikací [online]. 16.11.2011 [cit. 2015-04-03]. Dostupné z: http://www.zdrojak.cz/clanky/flash-je-mrtev-html5-neni-pripravene-co-ted/

Seznam obrázků

Seznam příkladů

Seznam tabulek

Přílohy

1. CD - součástí práce je CD, na kterém jsou ve složce html5-hry veškeré kódy jednotlivých her (html, js, css) včetně obrázků a knihoven jQuery a Box2D, dále se na něm nachází tato bakalářká práce pod názvem bakalarska_prace.pdf.