**ŽILINSKÁ UNIVERZITA V ŽILINE**

**FAKULTA RIADENIA A INFORMATIKY**

BAKALÁRSKA PRÁCA

Študijný odbor: Informatika

Jaroslav Ondreják

**Vytvorenie podporného frameworku pre responzívny dizajn**

**Vedúci práce:** Ing. Matej Meško

Reg. Číslo: 13/2014

Žilina, 2015

Čestné prehlásenie

Čestne prehlasujem, že som bakalársku prácu vypracoval samostatne na základe vlastných teoretických a praktických poznatkov, konzultácií a štúdia odbornej literatúry.

V Žiline Dňa: ........................ Podpis: ........................

Poďakovanie

Touto cestou sa chcem poďakovať vedúcemu bakalárskej práce Ing. Matejovi Meškovi za odbornú pomoc, cenné rady a pripomienky, ktoré mi pomohli úspešne vypracovať prácu.

ABSTRAKT

ONDREJÁK, Jaroslav: Vytvorenie podporného frameworku pre responzívny dizajn. [Bakalárska práca] – Žilinská univerzita v Žiline. Fakulta riadenia a informatiky; Študijný program: Informatika. - Vedúci práce: Ing. Matej Meško. Stupeň odbornej kvalifikácie: Bakalár informatiky. Žilina: FRI ŽU v Žiline, 2015. Počet strán: X

Cieľom bakalárskej práce je navrhnúť a implementovať podporný framework pre responzívny webový dizajn. Výsledná knižnica bude použitá ako základ pre tvorbu vizuálnej časti webových stránok, ktorá bude dynamicky reagovať na zmenu veľkosti okna webového prehliadača. Výsledná knižnica dokáže správne kvalifikovať zobrazovacie zariadenie a prispôsobí webovú stránku skutočnej veľkosti displeja zariadenia.

Kľúčové slová: HTML5, CSS3, JavaScript, responzívny dizajn, webový prehliadač, webová stránka

ABSTRACT

ONDREJÁK, Jaroslav: Vytvorenie podporného frameworku pre responzívny dizajn. [Bakalárska práca] – Žilinská univerzita v Žiline. Fakulta riadenia a informatiky; Študijný program: Informatika. - Vedúci práce: Ing. Matej Meško. Stupeň odbornej kvalifikácie: Bakalár informatiky. Žilina: FRI ŽU v Žiline, 2015. Počet strán: X

Cieľom bakalárskej práce je navrhnúť a implementovať podporný framework pre responzívny webový design. Výsledná knižnica bude použitá ako základ pre tvorbu vizuálnej časti webových stránok, ktorá bude dynamicky reagovať na zmenu veľkosti okna webového prehliadača. Výsledná knižnica dokáže správne kvalifikovať zobrazovacie zariadenie a prispôsobí webovú stránku skutočnej veľkosti displeja zariadenia.

Kľúčové slová: HTML5, CSS3, JavaScript, responzívny dizajn, webový prehliadač, webová stránka

OBSAH

[ÚVOD 1](#_Toc418420722)

[1 ANALÝZA SÚČASTNÉHO STAVU 2](#_Toc418420724)

[1.1 Webová služba 2](#_Toc418420725)

[1.2 Webová stránka a webová aplikácia 3](#_Toc418420726)

[1.3 W3C 4](#_Toc418420727)

[1.3.1 HTML 5](#_Toc418420728)

[1.3.2 XHTML 6](#_Toc418420729)

[1.3.3 HTML 5 6](#_Toc418420730)

[1.3.4 CSS 8](#_Toc418420731)

[1.3.5 CSS3 10](#_Toc418420732)

[1.4 JavaScript 10](#_Toc418420733)

[1.4.1 jQuery 11](#_Toc418420734)

[1.5 Webový prehliadač 11](#_Toc418420735)

[1.5.1 Podporné nástroje vykresľovania webových prehliadačov 13](#_Toc418420736)

[1.5.2 Viewport 14](#_Toc418420737)

[1.6 Nástroje na vývoj webových stránok 14](#_Toc418420738)

[1.6.1 PSPad 15](#_Toc418420739)

[1.6.2 Sublime Text 15](#_Toc418420740)

[1.7 Webový dizajn 16](#_Toc418420741)

[1.7.1 Liquid 17](#_Toc418420742)

[1.7.2 Adaptive 17](#_Toc418420743)

[1.7.3 Responsive 17](#_Toc418420744)

[2 ZBER POŽIADAVIEK PRE NAVRHOVANÚ KNIŽNICU 19](#_Toc418420745)

[2.1 Existujúce riešenia 19](#_Toc418420746)

[2.1.1 Bootstrap 19](#_Toc418420747)

[2.1.2 Foundation 20](#_Toc418420748)

[3 ANALÝZA POŽIADAVIEK PRE KNIŽNICU 20](#_Toc418420749)

[4 IMPLEMENTÁCIA NAVRHOVANEJ KNIŽNICE 21](#_Toc418420750)

[4.1 Jadro frameworku 22](#_Toc418420751)

[4.1.1 Reset preddefinovaných štýlov 22](#_Toc418420752)

[4.1.2 Rozloženie 23](#_Toc418420753)

[4.1.3 Skrývajúce elementy 24](#_Toc418420754)

[4.1.4 Elementy s nastavenou maximálnou šírkou 25](#_Toc418420755)

[4.2 Plugin pre zobrazenie menu 25](#_Toc418420756)

[4.3 Plugin pre zobrazenie slideshow 25](#_Toc418420757)

[ZÁVER 25](#_Toc418420758)

[ZOZNAM REFERENCIÍ 26](#_Toc418420759)

[PRÍLOHY 29](#_Toc418420760)

ZOZNAM POUŽITÝCH SKRATIEK

API. Application programming interface

HTML, 5, *HyperText Markup Language*

HTTP. Protokol

W3C. *World Wide Web Consortium*, World Wide Web Consortium

WWW. World Wide Web

# ÚVOD

Používanie internetu je v posledných rokoch veľmi rozšírené. Internet je sieť z miliónov uzlov, ktoré sú poprepájané celosvetovo a poskytuje svoje miliónom používateľom. Počet uzlov v internete, zariadení z ktorých je možné pristupovať na webové stránky a počet používateľov neustále rastie a môžeme povedať, že tento rast sa nikdy nezastaví.

Internet zaznamenal najväčší rozmach zavedením služby pre zdieľanie informácií WWW. Tvorcom tejto služby je anglický počítačový expert, výskumník a profesor *Tim Berners-Lee*. Základný kameň služby WWW je webová stránka vo forme textového zápisu a môže obsahovať rôzne dáta ako napríklad text, grafiku, zvuk a video.

Používatelia sa pripájajú na webové stránky pomocou koncových zariadení ako sú napríklad počítače, notebooky a v  neposlednej rade mobilnými zariadeniami, ktoré sa stávajú „lídrom“ v prístupe na internet. Každé z týchto zariadení môže mať rôzne rozmery a veľkosť displeja.

Čo vytvára otázku: „Ako by sme mali stránku správne formátovať a členiť aby sa správne zobrazila na všetkých zariadeniach?“

Základom mojej práce je návrh a implementácia knižnice, pomocou ktorej programátor s dodržaním jednoduchej štruktúry a pravidiel, zabezpečí správne zobrazenie webovej stránky bez ohľadu na rozmery displeja zariadenia a použitého prehliadača. Knižnica bude vytvorená tak, aby správne zobrazovala stránky na 4 najpoužívanejších prehliadačoch.

# ANALÝZA SÚČASTNÉHO STAVU

Prenos a zobrazovanie webových stránok v internete je založená na architektúre klient-server. Webová stránka je uložená na webovom serveri, ku ktorému pristupujeme prostredníctvom prenosového protokolu HTTP a zobrazuje sa pomocou webového prehliadača (Obrázok 1). Webový prehliadač odošle požiadavku na webový server a odpoveďou na ňu sú dáta prezentujúce stránku. Tieto dáta webový prehliadač vizualizuje do finálnej podoby zobrazenej stránky.



Obrázok 1 – zobrazenie stránky vo webovom prehliadači

## Webová služba

Webová služba je komunikačný interfejs, respektíve API, ktorému sa posiela požiadavka prostredníctvom protokolu HTTP. Poskytuje tak spôsob kooperácie medzi softvérovými aplikáciami, pri čom nezáleží aká technológia je použitá na klientskej alebo serverovej strane. Komunikácia pomocou webových služieb je realizovaná cez sieť a nevyžaduje preto pripojenie do internetu. S webovou službou zariadenia komunikujú spôsobom, ktorý je predpísaný v popise služby.

Na popis API pre volanie procedúry a jej dát slúži protokol *SOAP (Simple Object Access Protocol)* a rozhranie webovej služby je popísané vo formáte *WSDL (Web Services Description Language)*. [1]

Ako alternatívu k použitiu dvojice technológii SOAP a WSDL pre webové služby sa v súčasnosti používa štýl softvérovej architektúry *REST (Representational State Transfer)* určený pre distribuované prostredie, ktorý na rozdiel od nich nie je štandard a je dátovo orientovaný.

Existujú aj ďalšie prístupy ako je RPC (Remote Procedure Call) z ktorého webové služby vychádzajú . Tento prístup je založený na volaní vzdialenej procedúry na serveri pomocou HTTP požiadavky, ktorá sa vykoná a vracia výsledné dáta.

## Webová stránka a webová aplikácia

Webová stránka je dokument napísaný v značkovacom jazyku HTML, určený pre službu WWW. Pôvodom bola určená pre zobrazenie statického obsahu napríklad: textov, obrázkov, tabuliek, zoznamov a iných základných prvkov ako pri klasických dokumentoch. V súčasnosti webové stránky obsahujú rôzne komplikovanejšie prvky, napríklad: navigačnú lištu pre navigovanie na iné stránky alebo časti webu, slideshow[[1]](#footnote-1) na zobrazenie obrázkov alebo fotografií, rôzne typografické prvky, nadpisy, logá. V skutočnosti nezáleží na tom, čo webová stránka obsahuje, pretože jazyk HTML definuje iba štruktúru dát.

Pre odkazovanie na iné stránky sa používajú hypertextové odkazy, čím vytvárajú medzi jednotlivými stránkami prepojenie.

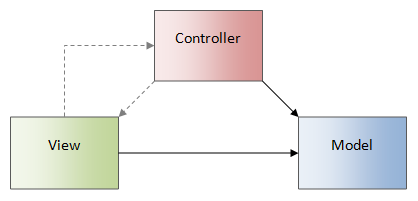
Zobrazená stránka je výsledkom odpovede na HTTP požiadavku, ktorá môže byť generovaná rôznymi technológiami na strane servera. Tým pádom je možné do stránky vložiť rôzny obsah z rôznych externých aj interných zdrojov, napríklad: dáta z relačnej databázy, iného serveru, atď. Takéto stránky sú generované dynamicky a nazývame ich dynamické stránky.

Vlastnosti dynamických webových stránok sa dajú veľmi dobre využiť ako náhrada pre klasické *desktopové* aplikácie. Dáta a celá logika webovej aplikácie je umiestnená na serveri a nesie názov **serverová** časť. Používateľské rozhranie (GUI[[2]](#footnote-2)), nazývané **klientská** časť, beží vo webovom prehliadači, ktorý sám o sebe logiku webovej aplikácie nepozná a iba posúva vstupné dáta a akcie serverovej časti.

Webové aplikácie sa dajú preto popísať pomocou MVC architektúry, ktorá delí aplikáciu na 3 logické časti: *Model*, *View* a *Controller* (Obrázok 2). Dôvodom je snaha aby tieto časti bolo možné upravovať samostatne a celkový vplyv zmien na ostatné časti bol čo najmenší.

1. *Model* reprezentuje dáta a logiku webovej aplikácie,
2. *View* zobrazuje používateľské rozhranie,
3. *Controller* je zodpovedný za riadenie udalostí v aplikácií.

Vo vývoji webových aplikácií sa používajú termíny frontend (V*iew*) a backend (*Model* a *Controller*) pre oddelenie klientskej a serverovej časti .



Obrázok 2 – diagram MVC architektúry[[3]](#footnote-3)

## W3C

*World Wide Web Consortium (W3C)* je organizácia, dohliadajúca na zjednotenie rôznych technológií, ktoré sú používané na internete pomocou vydávania štandardov.

Konzorcium vyvíja technické špecifikácie a postupy na zabezpečenie nezávislosti na prehliadačoch a iných softvérových alebo hardvérových požiadavkách. Zameriava sa na technológie, s ktorými môžeme pristupovať k webu z rôznych miest, kedykoľvek a za použitia rôznych zariadení.

Medzi známe štandardy ktoré zahŕňajú vytvorenie a správne zobrazenie webových stránok patria napríklad: HTML5, XHTML, XML, CSS, DOM, SVG a iné.

Používanie a dodržiavanie týchto štandardov je prínosom napríklad pre indexáciu vo webových vyhľadávačoch alebo pre zobrazenie zdravotne postihnutým používateľom pomocou špeciálnych webových prehliadačov.

Webové štandardy sú odporúčania a ich dodržiavanie je dobrovoľné.

### HTML

*HyperText Markup Language (HTML)* je základný značkovací jazyk, popisujúci štruktúru webových stránok. Značkovací jazyk je súbor značkovacích tagov, respektíve značiek. Tagy v jazyku HTML popisujú určitú časť obsahu webového dokumentu a väčšina z nich je definovaná v pároch (otvárací a ukončovací tag). Tým pádom môže webový prehliadač rozoznať a správne zobraziť časti dokumentu. Každý tag je určený svojím preddefinovaným názvom a je ohraničený lomenými zátvorkami.

Pre deklaráciu typu webového dokumentu sa používa tag <!DOCTYPE> umiestnený na začiatku dokumentu, so špecifickým označením použitého štandardu. Definíciou tagu <html> začína HTML dokument a končí tagom </html>. Meta informácie o dokumente, ako nadpis stránky, znaková sada použitá na kódovanie dokumentu a pripojené externé zdroje sa nachádzajú medzi párovými tagmi <head> a </head>. Časť dokumentu, ktorá bude zobrazená ako obsah webovej stránky vo webovom prehliadači sa nachádza medzi dvojicou tagov <body> a </body>.

Každý pár tagov spolu s ich obsahom sa nazýva element. Niektoré elementy nemajú ukončovací tag, preto sú reprezentované len otváracím tagom a nazývajú sa *prázdne elementy*. Elementy sa môžu do seba vnárať, respektíve dediť a môžeme ich rozdeliť do dvoch základných skupín:

* Blokové elementy – zobrazujú sa ako blok, to znamená, že zaberajú celú šírku nadradeného elementu. Vložením viacerých blokových elementov sa zobrazia pod sebou.
* Inline elementy – sú také, ktoré zaberajú len šírku svojho obsahu. Ak vložíme viac takýchto elementov za sebou a ich spoločná šírka je menšia ako šírka nadradeného elementu, potom sú zobrazené vedľa seba.

Elementy môžu obsahovať atribúty, ktoré poskytujú rozširujúce informácie o danom elemente a nachádzajú sa v otváracom tagu v tvare nazov\_atributu=“hotnota\_atributu“.

### XHTML

*Extensible HyperText Markup Language (XHTML)* je skoro identický ako HTML. Syntax XHTML zavádza prísnejšie pravidlá pre štruktúru webového dokumentu a je založená na syntaxe XML. Takýto dokument môžeme jednoducho rozšíriť o novú značku alebo atribút. [5]

Pre deklaráciu XHTML je povinné uviesť tag <!DOCTYPE> podľa použitej verzie štandardu v prvom riadku webového dokumentu. Používanie jednotlivých tagov je prísne definované nasledujúcimi predpismi:

* elementy <html>, <head>, <title> a <body>, sú povinné a musí ich obsahovať každý webový dokument,
* všetky elementy musia byť vždy uzatvorené a to aj v prípade prázdnych elementov,
* žiadny element nemôže byť ukončený skôr ako jeho detský element,
* všetky elementy a ich atribúty musia byť písané malým písmom,
* všetky atribúty musia byť v úvodzovkách.

### HTML 5

Táto špecifikácia je piatou verziu základnej revízie jadra jazyka htmlslužby WWW. V tejto verzii sú zavedené nové prvky rozširujúce základnú verziu HTML. Vo verzii HTML5 sú upravené niektoré zo základných elementov pre jednoduchšie používanie a niektoré zastarané elementy boli odstránené.

Prvotné zmeny sa týkajú tagu slúžiaceho na deklaráciu samotného štandardu HTML5. Touto zmenou je jeho najjednoduchšia možná forma, <! DOCTYPE html>. Tag na definovanie znakového kódovania je zjednodušený, píšeme ho v tvare <meta charset=”UTF-8”>.

Najzaujímavejšie pridané elementy vo verzii HTML5 sú:

* Štrukturálne elementy, ktoré upresňujú štruktúru webového dokumentu (Obrázok 3). Všetky tieto elementy sú blokové elementy. Napríklad:
  + <header> definuje úvodné informácie,
  + <section> definuje sekciu,
  + <footer> definuje pätu v dokumente,
  + <article> definuje článok,
  + <nav> definuje navigačné odkazy alebo navigačné menu,
  + <main> definuje hlavný obsah.
* Atribúty pre ovládanie formulárových prvkov:
  + number – vstupná hodnota by mala obsahovať číselnú hodnotu,
  + date – vstupná hodnota by mala obsahovať dátum,
  + time – umožňuje používateľovi vybrať čas,
  + range – vstupná hodnota by mala obsahovať hodnotu v zadanom rozmedzí. Použitie tohto atribútu vyžaduje použitie dodatočných atribútov min a max.
* Grafické elementy <svg> a <canvas>, ktoré definujú grafické vykreslenie použitím SVG alebo JavaScriptu.
* Elementy pre podporu multimédií:
  + <audio> - definuje zvukový alebo hudobný obsah,
  + <video> - definuje video alebo filmový obsah.



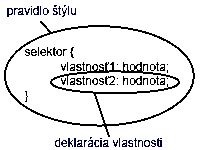
Obrázok 3 – štrukturálne elementy[[4]](#footnote-4)

Všetky moderné webové prehliadače podporujú verziu HTML5. Treba pripomenúť, že každý webový prehliadač spracúva neznáme elementy ako inline elementy.

### CSS

*Cascading Style Sheets (CSS)* je jednoduchý mechanizmus, ktorý definuje ako budú elementy zobrazené. Definuje zobrazenie textu, farieb, odsadení, okrajov a podobne. Jeho syntax a používanie je veľmi jednoduché a intuitívne. Hlavným dôvodom vzniku CSS bolo oddelenie formátovanie vzhľadu dokumentu od jeho štruktúry a dát.

Syntax CSS je založený na písaní pravidiel pre HTML elementy (Obrázok 4). Pravidlo je množina obsahujúca selector a blok deklarácie vlastností. Selector definuje HTML element, na ktorý chceme aplikovať daný štýl. Blok deklarácie vlastností, môže obsahovať viacero vlastností oddelených bodkočiarkou a je ohraničený zloženými zátvorkami. Každá vlastnosť musí obsahovať hodnotu, ktorá môže byť číselná alebo preddefinovaná pre danú vlastnosť.



Obrázok 4 – príklad CSS pravidla[[5]](#footnote-5)

Selektory umožňujú určenie HTML elementov na základe výrazov, zadefinovaných napríklad podľa ich názvu, atribútu id, atribútu class alebo mnoho ďalších. Selektory môžu byť poskladané z viacerých poprepájaných výrazov alebo ich môžeme zoskupovať.

Najpoužívanejšie CSS výrazy selektorov sú:

* Názov elementu – ako selektor použijeme názov elementu, napríklad p { … }.
* Atribút id – ako selektor použijeme názov atribútu id, pred ktorým napíšeme znak „#”, napríklad #hlavny { … }.
* Atribút class – ako selektor použijeme názov atribútu class, pred ktorým napíšeme znak „bodky“, napríklad .hlavny { … }.

Keď webový prehliadač číta CSS, aplikuje štýly na HTML elementy, čím formátuje webovú stránku. Sú tri hlavné možnosti kde môžu byť štýly definované:

* Externý súbor.
* Interne v dokumente html.
* Priamo v elemente na ktorý chceme štýl aplikovať.

Najčastejšie používaná je prvá možnosť uloženia CSS v externom súbore, čim dosiahneme to, že môžeme jednoducho zmeniť vzhľad celej webovej stránky zmenením len jedného súboru. Každá stránka, ktorá chce používať dané štýly, musí obsahovať referenciu na externý súbor. Pre tento účel slúži <link> element, umiestnený v sekcii head HTML stránky. Externý súbor s CSS musí byť ukončený koncovkou .css. Hlavnou výhodou použitia externého súboru je zníženie prenášaného objemu dát, keďže je pravdepodobné, že štýl sa nebude veľmi často meniť.

Štýl, ktorý chceme aplikovať len na jeden element nazývame *inline* štýl a definujeme ho v atribúte style daného elementu zadaním konkrétnych vlastnosti. Zmena štýlu definovaného týmto spôsobom znamená prepisovanie aplikačného kódu. Cieľom CSS je oddeliť štýl od kódu aplikácie, tak dizajnér nemusí zasahovať do kódu aplikácie, preto by sme inline štýly mali používať šetrne.

Pri kombinácií viacerých druhov uloženia CSS štýlov musíme brať do úvahy v akom poradí sú jednotlivé štýly aplikované. Najvyššiu prioritu majú štýly definované v atribúte style, potom externé a interné štýly a posledné v poradí štýly, ktoré sú preddefinované webovým prehliadačom.

### CSS3

Najnovším štandardom pre CSS je verzia CSS3, ktorý je kompletne spätne kompatibilný so skoršími verziami CSS. Sú pridané nové vlastnosti a tvorcovia sa rozhodli zjednotiť mnoho monolitických a individuálnych vlastností do celkov.

CSS3 sa rozčleňuje do modulov, ktoré obsahujú celú staršiu CSS špecifikáciu a navyše sú pridané nové moduly, ktoré poskytujú rozsiahlejšiu možnosť formátovania webovej stránky.

Niektoré najdôležitejšie CSS3 moduly sú:

* selektory,
* pozadia a okraje,
* textové efekty,
* 2D/3D transformácie,
* Animácie.

Väčšina modulov a väčšina nových vlastností sú implementované vo všetkých moderných webových prehliadačoch.

## JavaScript

JavaScript je základný programovací jazyk pre HTML a celkovo pre web. Pôvodne bol vyvýjaný *Brendanom Eichom* zo spoločnosti *Netscape Communications* pod názvom „*Mocha*“, neskôr pod názvom „*LiveScript*“. Pred uvedením verejnosti bol pomenovaný ako „JavaSript“, najmä vďaka popularite jazyka Java. Reálne neexistuje žiadny významný vzťah medzi jazykmi Java a JavaSript a ani výrazná podoba ich syntaxe. Najväčšia podobnosť je, že sú založené na spoločnom predkovi, jazyku C. JavaScript dokáže mnoho vecí, ktoré sú bežne potrebné pri tvorbe webových stránok.

Spomenieme základné, popredné možnosti JavaSriptu:

* manipulácia s DOM,
* môže validovať vstupné dáta od používateľa.

Nad programovacím jazykom JavaScript existuje naprogramovaných množstvo knižníc.

### jQuery

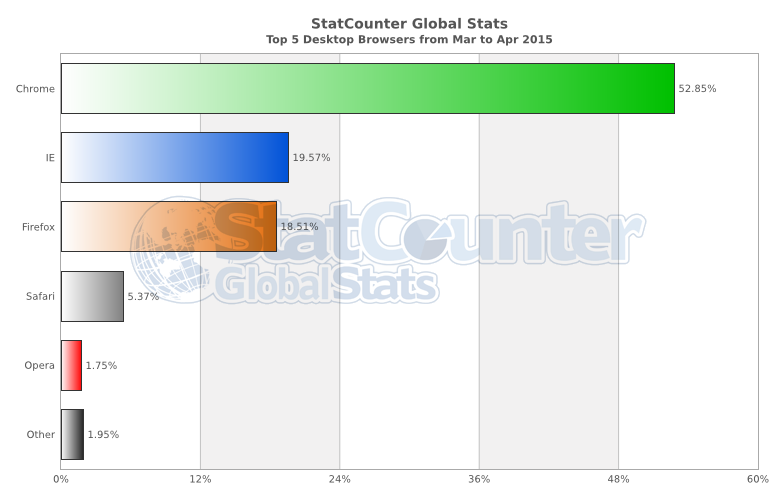
Je framework pre jazyk JavaScript. Rieši veľa úloh, ktoré si vyžadujú mnoho riadkov kódu napísaných v JavaScripte a zjednodušuje ich do metód, ktoré sa môžu volať jediným riadkom kódu. To je pre programátora veľkou výhodou, vzhľadom na rýchlosť písania aj prehľadnosti kódu.

Knižnica jQuery zjednodušuje nasledovné funkcie:

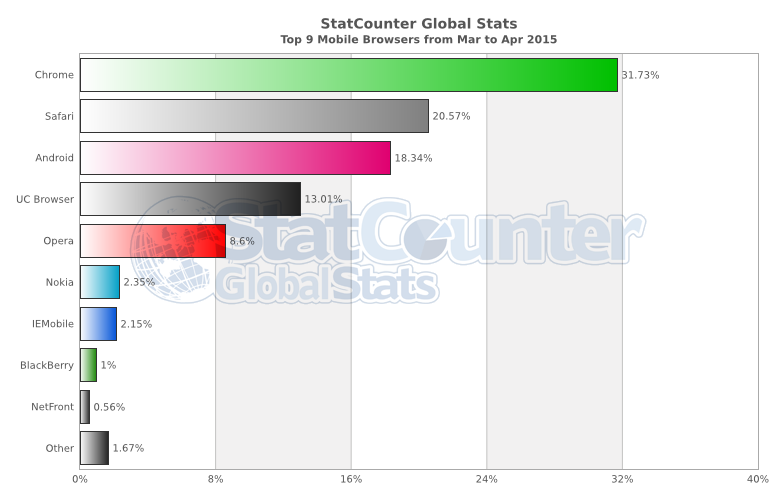
* HTML/DOM manipuláciu
* CSS manipuláciu
* HTML event metódy
* Efekty a animácie
* AJAX
* Utility

## Webový prehliadač

Webový prehliadač je program nainštalovaný v počítači alebo inom zariadení, ktorého hlavnou úlohou je zobrazenie webovej stránky na zobrazovacom zariadení. Na desktopových počítačoch sú v súčasnosti najpoužívanejšie 4 webové prehliadače Google Chrome, Mozzila Firefox, Safari a Internet Explorer. Na mobilných zariadeniach prevažne Google Chrome a Safari optimalizované pre mobilné zariadenia. Na presnejšie rozdelenie desktopových a mobilných prehliadačov podľa používania v Marci 2015 slúži Obrázok 5 a Obrázok 6.



Obrázok 5 – najpoužívanejšie prehliadače pre počítače[[6]](#footnote-6)



Obrázok 6 – najpoužívanejšie prehliadače pre mobilne zariadenia[[7]](#footnote-7)

### Podporné nástroje vykresľovania webových prehliadačov

Renderovacie jadro webového prehliadača je softvér, ktorý číta dáta webovej stránky a naformátovaný obsah vykresľuje do webového prehliadača. Vzhľadom na štatistiky nástroja *StatCounter* (Obrázok 5) je v súčasnosti najviac používaný webový prehliadač Google Chrome, ktorý používa renderovacie jadro WebKit. Toto jadro taktiež obsahuje webový prehliadač Safari.

V produkte Mozilla je použité renderovacie jadro Gecko. Pôvodne bolo vyvíjané firmou *Netscape Communication Corporation*, ale momentálne je vyvíjané firmou *Mozilla Corporation*.

Internet Explorer obsahuje renderovacie jadro Trident už od verzie 4.0, ktoré je postupne vylepšované vďaka jeho využitiu v mnohých komponentoch operačného systému Windows.

Štandard CSS umožňuje definovanie vlastnosti konkrétne pre niektoré renderovacie jadrá, použitím špeciálnych predpôn. Príklad pravidla CSS pre nastavenie veľkosti zaoblenia rohov na ohraničení elementu s hodnou atribútu class=“element“.

.element{  
 -moz-border-radius: 2em;  
 -ms-border-radius: 2em;  
 -o-border-radius: 2em;  
 -webkit-border-radius: 2em;  
 border-radius: 2em;  
}

### Viewport

Tag <meta name=viewport> dáva zmysel pri použití návrhu webu s responzívnym dizajnom, pretože webové prehliadače v mobilných zariadeniach vykresľujú webovú stránku zmenšenú, aby ju bolo celú vidieť a používateľ si ju musí sám priblížiť do čitateľnej veľkosti.

Napríklad mobilné zariadenie môže mať fyzickú šírku displeja 360 pixlov, ale rozlíšenie displeja môže mať šírku 720 pixlov. Prehliadač natiahne *okno zobrazenia* na 720 pixlov pre vykreslenie webovej stránky.

Použitím tagu meta spolu s atribútom name=“viewport“ definujeme odlišné správanie okna zobrazenia definovaním hodnoty atribútu content ako sériu parametrov oddelených čiarkou. Môžeme nastaviť šírku okna zobrazenia v prehliadači parametrom width=360 alebo width=device-width, ktorá nastaví veľkosť okna rovnú skutočnej šírke zobrazovacieho zariadenia. Ďalším parametrom je initial-scale=1, ktorý vyjadruje v akom pomere sa webová stránka zobrazí vzhľadom na skutočnú veľkosť obrazovky. Hodnota 1 definuje, že veľkosť okna zobrazenia a skutočná veľkosť obrazovky je v pomere 1:1. Menšie hodnoty pomeru znamenajú zmenšenie a vyššie hodnoty zväčšenie a stránku nie je možné zmenšiť pod stanovenú hodnotu.

## Nástroje na vývoj webových stránok

Na vývoj webových stránok nie sú potrebné žiadne špeciálne vývojové preostredia. Ako bolo spomenuté v kapitole 1.2 webová stránka je štandardný textový dokument zo špecifickou príponou. Na tvorbu a editáciu dokumentu postačí aj jednoduchý textový editor. Existujú však pokročilejšie textové editory, ktoré ponúkajú napríklad nastavenie zvýraznenia syntaxe podľa zvoleného typu dokumentu, automatické návrhy a dopĺňanie kódu, generovanie kódu.

### PSPad

http://www.pspad.com/

Editor PSPad je voľne šíriteľný univerzálny editor. Editor je užitočný pre všetkých, ktorí pracujú v operačnom systéme Windows, pracujú s obyčajným textom, tvoria webové stránky, programujú. PSPad ponúka široké spektrum funkcií, ako napríklad:

* Práca s projektami
* Práca s viacerými dokumentmi súčasne
* Porovnávanie textov s farebným zvýraznením rozdielov
* Zvýraznenie syntaxe

### Sublime Text

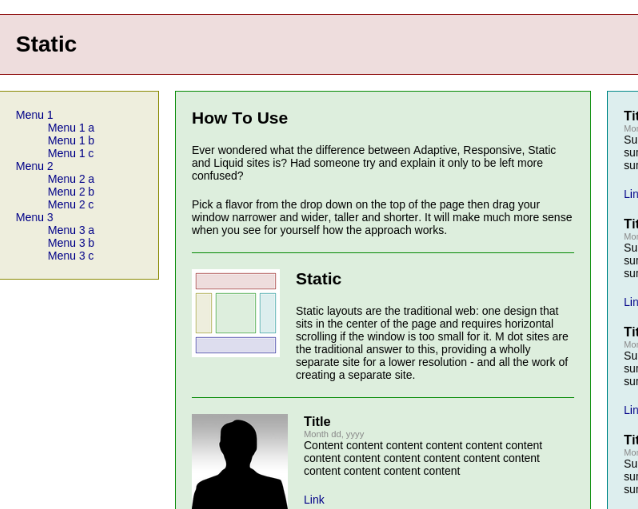
http://www.sublimetext.com/

Sublime Text je sofistikovaný textový editor pre písanie rozličného kódu. Obsahuje šikovné používateľské rozhranie, s pridanými funkciami a obrovskou výkonnosťou. Medzi popredné funkcie môžeme zaradiť:

* Reagovanie na udalosti pri používaní programu alebo manipulácií zo súborom v reálnom čase.
* Viacnásobný výber - umožňuje robiť zmeny v rovnakom čase na viacerých miestach v texte.
* Pridávanie rôznych pluginov.
* Nastavovanie editora pomocou zmeny preddefinovaných JSON objektov.
* Nastavenie zvýrazňovania syntaxe pre otvorený dokument.
* Automatické generovanie kódu alebo častí obsahu.
* Podpora konzoly.

## Webový dizajn

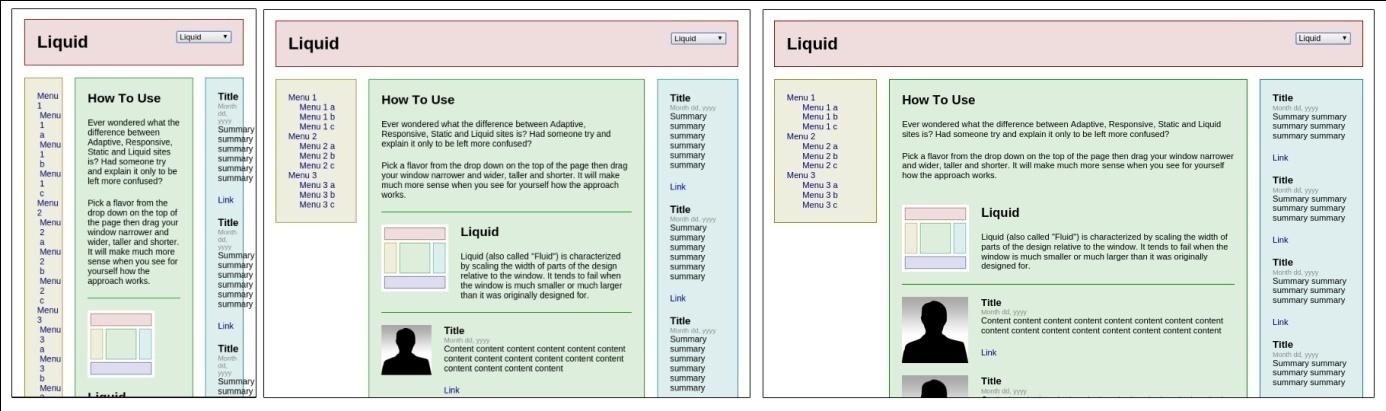
Webový dizajn (webdizajn) je funkčné riešenie vzhľadu a používateľského rozhrania webovej aplikácie. Pod pojmom funkčné myslíme zobrazenie a interpretáciu obsahu hlavne webovým prehliadačom. Webové stránky môžu byť z pohľadu návrhu dizajnu *statické* alebo *dynamicky reagujúce na zmenu šírky okna*. Statické návrhy dizajnu nijakým spôsobom nereagujú na zmenu šírky okna a ostávajú nezmenené, pozri Obrázok 7. Pri zmenšení šírky zobrazovacieho okna pod veľkosť šírky obsahu automaticky sa na webových prehliadačoch zobrazí horizontálny scrollbar[[8]](#footnote-8). V súčasnosti sa najčastejšie využíva návrh *responzívny* webdizajn, ktorý obsahuje prvky *liquid* a *adaptive* webdizajnu.



Obrázok 7 – statický webdizajn[[9]](#footnote-9)

### Liquid

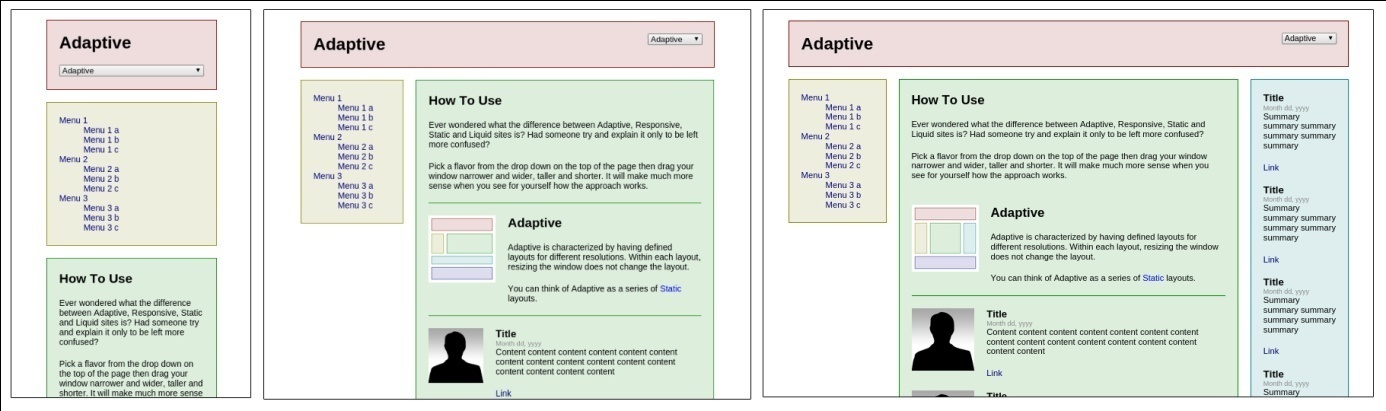
Liquid (premenlivý) alebo tiež nazývaný fluid (tečúci) je charakterizovaný naťahovaním častí webdizajnu proporcionálne šírke oknu webového prehliadača (Obrázok 8).



Obrázok 8 – liquid webdizajn, zobrazenie pre mobil, tablet, počítač9

### Adaptive

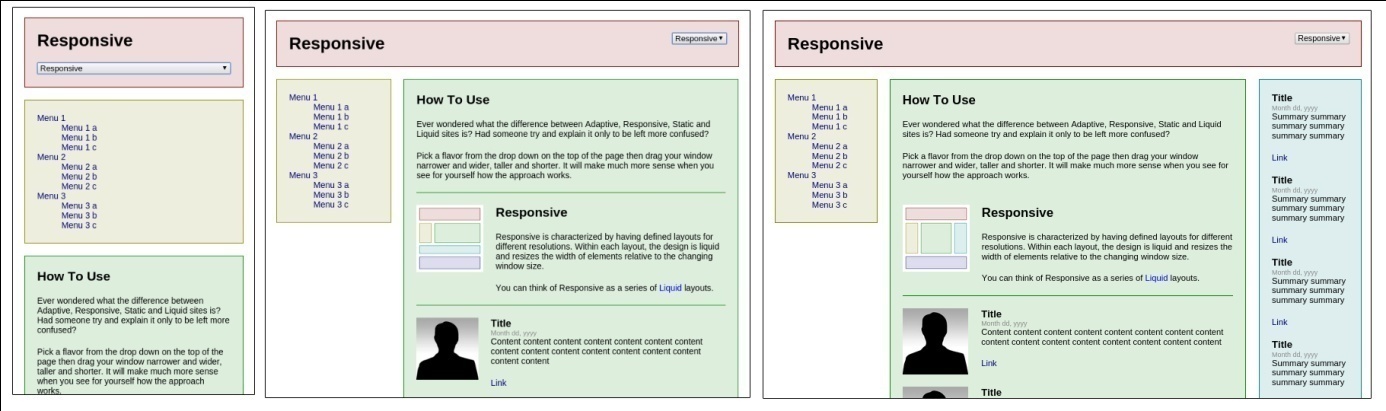
Adaptívny webdizajn má definované rozličné rozloženia (layouty) pre rozličné rozlíšenia. Môžeme si to predstaviť ako množinu statických rozložení. (Obrázok 9).



Obrázok 9 – adaptívny webdizajn, zobrazenie pre mobil, tablet, počítač9

### Responsive

Responzívny webdizajn definuje rozličné rozloženia pre rozličné rozlíšenia a zároveň každý z týchto rozložení sa prispôsobuje šírke okna webového prehliadača (Obrázok 10). Cieľ je aby jeden dizajn bol zobraziteľný na všetkých zariadeniach a nebolo potrebné vytvárať osobitné zobrazenie webu pre každé zariadenie.



Obrázok 10 – responzívny webdizajn, zobrazenie pre mobil, tablet, počítač

# ZBER POŽIADAVIEK PRE NAVRHOVANÚ KNIŽNICU

Nástupom smartfónov, tabletov a celkovo mobilných zariadení sa začal prudko zvyšovať počet používateľov, ktorí cez ne pristupujú k webovým stránkam. Tieto zariadenia majú rádovo nižšiu veľkosť displeja ako desktopové počítače.

Rovnako je dôležitá správna kvalifikácia zariadenia, či sa jedná o smartfón, tablet alebo desktopový počítač vzhľadom k tomu, že smartfón môže mať rovnaké rozlíšenie ako desktopový počítač pri menšej veľkosti displaya. Treba brať ohľad na to, že používateľ na smartfóne pracuje s dotykovou obrazovkou a preto by mala byť navrhovaná knižnica „mobile-friendly“.

Navrhovaná knižnica by mala zabezpečiť správne zobrazenie väčšine používateľom pristupujúcich na webové stránky a generovať kód podľa štandardov W3C, ktorých popis je v kapitole 1.3.

Vzhľadom na to, že knižnica bude tvoriť časť webovej stránky, rýchlosť načítania a zobrazenia webovej stránky nie je závislá od výkonu zariadenia na, ktorom beží webový prehliadač. Silná závislosť je od rýchlosti pripojenia k internetu a veľkosti prenášaných dát, prezentujúce webovú stránku, pretože sa musia preniesť do webového prehliadača.

## Existujúce riešenia

V súčasnosti existujú riešenia, ktoré dokážu zabezpečiť požiadavky pre výslednú webovú stránku. Najčastejšie používaný je framework *Bootstrap*, za ním druhý v poradí framework *Foundation*.

### Bootstrap

*Bootstrap* je voľne šíriteľná sada nástrojov pre tvorbu frontendu webových stránok, pod licenciou *MIT*. Pôvodne nazývaný *„Twitter Blueprint“* a vyvíjaný dvojicou programátorov *Mark Otto* a *Jacob Thornton*. Bootstrap je kompatibilný z poslednými verziami webových prehliadačov Google Chrome, Firefox, Internet Explorer, Opera a Safari, ale niektoré z týchto webových prehliadačov nie sú podporované na všetkých platformách. Od verzie 2.0 podporuje responzívny webdizajn a od verzie 3.0 prijal koncepciu *mobile-first*.

Štruktúra knižnice Bootstrap je modulárna a pozostáva z množiny *LESS* štýlov, ktoré implementujú rozličné komponenty. Tieto štýly sú uložené v súbore *bootstrap.less* a programátor môže prispôsobiť súbor vybratím komponentov ktoré chce vo svojom projekte použiť.

### Foundation

*Foundation* je responzívny frontend framework, ponúkajúci základné rozloženie a mnoho HTML a CSS komponentov, šablón, formulárov, tlačidiel a iných používateľských rozhraní. Okrem toho ponúka rôzne voliteľné JavaScriptové rozšírenia. Spoločnosť *ZUBR* vydala knižnicu Foundation ako *open-source* projekt pod licenciou *MIT* a v súčasnosti je vydaná jeho piata verzia.

Táto knižnica využíva návrh *mobile-first* a je postavená s pomocou preprocesoru*[[10]](#footnote-10)* *Sass/SCSS*. Od verzie 3.0 obsahuje špeciálnu možnosť v dokumentácií na webovej stránke, kde programátor pomocou formulára môže označiť komponenty, ktoré bude využívať a následne je vygenerovaný balíček, ktorý obsahuje ich CSS kód.

# ANALÝZA POŽIADAVIEK PRE KNIŽNICU

Požiadavkou je, aby bola stránka zobrazená správne na celú šírku okna webového prehliadača bez ohľadu na fyzickú veľkosť zariadenia, čo nám zabezpečí návrh responzívneho dizajnu webu. Jeho výhody sú bližšie popísané v kapitole 1.7. Vývoj pri tomto návrhu dizajnu komplikuje fakt, že ak chceme mať odlišné zobrazenie elementov na odlišnom rozlíšení webového prehliadača, musíme preddefinovať predlohu správania sa elementov na každom rozlíšení.

Funkcie navrhovanej knižnice by mali správne pracovať na 4 najpoužívanejších prehliadačoch, vzhľadom na to, že tieto prehliadače pokrývajú 96,3% používateľov pristupujúcich na internet v mesiaci marec 2015 (Obrázok 5).

# IMPLEMENTÁCIA NAVRHOVANEJ KNIŽNICE

Z analýzy požiadaviek vyplýva, že knižnicu tvoríme pre responzívny dizajn. Uľahčujeme vývoj webových stránok programátorom tým, že preddefinujeme správanie elementov pri zmene rozlíšenia. Toto správanie však má programátor pod kontrolou, správnym aplikovaním funkcií implementovaných v knižnici.

Všetky procesy zabezpečujúce správanie sa daného elementu sú viazané na určitú hodnotu atribútu class v HTML elementoch. Programátor jednoducho pri písaní HTML dokumentu dopísaním kľúčových názvov tried aplikuje na daný element preddefinovanú funkčnosť.

Navrhovaná knižnica je založená na štýle definovania CSS pravidiel, nazývaný *mobile-first*, pri ktorom začíname definíciou zobrazenia pre nízke rozlíšenia a pri väčších rozlíšeniach sú tieto pravidlá predefinované. To dosiahneme použitím CSS3 pravidla @media, slúžiaceho pre definovanie odlišných štýlov pre odlišné rozlíšenia, šírky a výšky zariadenia, orientácie mobilných zariadení a mnoho ďalších.

Na rozdiel od konkurenčných riešení definujeme len funkčnosť a správne zobrazenie elementov webovej stránky, nie grafický vzhľad. Týmto nechávame programátorom využívajúcim navrhovanú knižnicu možnosť dodefinovať grafickú podobu každého prvku webovej stránky. Rovnako tým zabezpečíme pohodlnejšiu implementáciu navrhovanej knižnice do už existujúcej webovej stránky za účelom jej modernizácie a využitiu funkčných vlastností, ktoré knižnica ponúka.

Dôležitou požiadavkou na implementáciu je optimalizácia rýchlosti načítania webovej stránky a preto sme zvolili použitie modulárneho návrhu knižnice. Programátor webových stránok si môže sám zvoliť, ktoré vlastnosti od knižnice vyžaduje a tie následne formou modulov pripojiť do frameworku. To ma za dôsledok zníženie prenášanej veľkosť dát potrebných pre zobrazenie webovej stránky a tak stránka sa načíta rýchlejšie. Knižnica bude pozostávať z jadra a prídavných pluginov.

Jednotlivé pluginy môžu byť implementované rozličným spôsobom, niektoré pozostávajú iba z kódu v CSS, iné kombináciou CSS s JavaScriptom. Popis ako plugin začleniť do webovej aplikácie a popis knižníc na ktorých je závislý, je popísaný v súbore *readme.txt* pri zdrojových súboroch každého pluginu.

## Jadro frameworku

Jadro tvorí základný kameň navrhovanej knižnice a celý jeho aplikačný kód je napísaný len v jazyku CSS3. K správnemu fungovaniu všetkých prídavných pluginov je jeho prítomnosť vo webovej aplikácií nevyhnutná. Obsahuje definíciu nasledovných funkcií:

* automatické resetovanie preddefinovaných štýlov webového prehliadača,
* kontajnery pre udržanie plávajúcich (*float*) prvkov,
* základné rozloženie pomocou stĺpcov s definovanými odsadeniami,
* základné rozloženie pomocou stĺpcov s nulovými odsadeniami,
* fixovanie definovaného rozloženia bez ohľadu na veľkosť šírky webového prehliadača,
* skrývajúce elementy pri určitých hraničných rozlíšeniach,
* nastavenie maximálnej šírky elementu, na ktorú sa môže natiahnuť.

### Reset preddefinovaných štýlov

Jadro frameworku v sebe obsahuje základné CSS pravidlá na nastavenie spoločného základného nastavenia webového dokumentu. Týmto krokom predefinujeme CSS štýly, ktoré sú integrované vo webovom prehliadači, keďže každý môže mať rôzne nastavené štýly pre základné HTML elementy.

Prvé pravidlo definuje veľkosť vnútorných a vonkajších okrajov každého elementu na hodnotu 0.

\* {  
 margin: 0;  
 padding: 0;  
}

Pravidlo, ktoré nastaví aby element html mal nastavenú vlastnosť box-sizing na hodnotu border-box, čo zaručí, že skutočná veľkosť elementu (boxu) sa bude skladať zo súčtov veľkosti elementu, veľkosti orámovania a veľkosti vnútorného okraju. Vzhľadom na to že element html je predkom takmer všetkých elementov môžeme nastaviť pravidlo, aby všetky detské elementy mali nastavenú vlastnosť box-sizing na hodnotu inherit, čo znamená zdedená.

html {  
 box-sizing: border-box;  
}

\* {  
 box-sizing: inherit;  
}

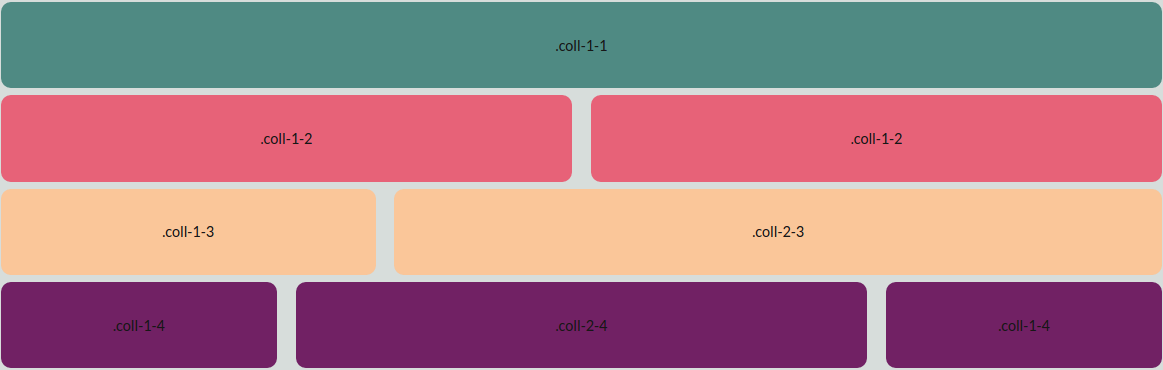
Všetkým netriedeným zoznamom vypneme zobrazovanie zarážiek jednotlivých položiek zoznamu, nastavením vlastnosti list-style-type na hodnotu none.

### Rozloženie

Základný komponent jadra navrhovanej knižnice je možnosť pre rozloženie obsahu webovej stránky. Osvedčená metóda je rozloženie obsahu do mriežky, podobne ako pri štruktúre tabuliek. Táto mriežka je určená množinami stĺpcov usporiadaných pod sebou, pri čom stĺpce v jednej množine môžu byť rôznej šírky a predstavujú jeden riadok tabuľky.

Stĺpce sú prezentované formátom CSS triedy coll-X-Y, kde hodnoty X a Y predstavujú šírku stĺpca v pomere X:Y zo šírky nadradeného elementu. Triedy tohto typu sú definované po hodnotu Y rovné 10 a hodnota X musí byť menšia alebo rovná hodnote Y. Stĺpce majú medzi sebou pevne definovanú medzeru. Veľkosť horizontálnej medzery je 1,6% a veľkosť vertikálnej medzery je 0,3% zo šírky nadradeného elementu. Táto medzera je definovaná na každom stĺpci ako veľkosť odsadenia z pravej strany, okrem posledného stĺpca v množine (Obrázok 11). Pre konkrétnu implementáciu v CSS uvádzam časť kódu:

[class\*="coll"] { margin: 0.3% 1.6% 0.3% 0; float: left; }  
[class\*="coll"]:last-child { margin-right: 0; }  
.coll-1-3 { width: 32.266667%; }  
 .coll-2-3 { width: 66.133333%; }  
 .coll-3-3 { width: 100.000000%; }



Obrázok – základné rozloženie

Rozloženie potrebuje mať elementy, ktoré slúžia na držanie iných elementov, napríklad množiny stĺpcov a nazývame ich *obaľovacie* elementy. Preto sme definovali triedy container a row, kde trieda row sa líši len tým, že má preddefinované pravé a ľavé vonkajšie odsadenie na hodnotu 1,6% zo šírky rodičovského elementu. Element s jednou z týchto CSS tried si udržiava veľkosť závislú od veľkosti obsahu, bez ohľadu či je obsah plávajúci *napravo* alebo *naľavo*. Túto vlastnosť sme dosiahli použitím *clearfixu[[11]](#footnote-11)*, ktorý vyzerá nasledovne:

.container:before,   
.container:after {  
 content: '';  
 display: table;  
}

.container:after {  
 clear: both;  
}

Zobrazenie horizontálnej aj vertikálnej medzery na všetkých stĺpcoch v jednej množine je možné potlačiť pridaním CSS triedy nospaced do rodičovského obaľovacieho elementu. Takéto správanie sme dosiahli tým, že sme definovali rozličné východiskové veľkosti šírky stĺpcov. Pre porovnanie uvádzam dve CSS pravidlá:

.coll-1-4 { width: 23.800000%; }  
.nospaced > .coll-1-4 { width: 25.000000%; }

Responzivita stĺpcových elementov je zabezpečená vyjadrením rozmerov obsahu a odsadení v percentuálnych jednotkách a aplikovaním štýlov, ktoré zmenia hodnotu šírky elementu podľa rozlíšenia. Tieto štýly sme definovali použitím CSS pravidiel @media pre šírku rozlíšenia:

* nad 400 pixlov pre stĺpce coll-1-2 a coll-2-2,
* nad 600 pixlov pre stĺpce coll-1-3, coll-2-3 a coll-3-3,
* nad 800 pixlov pre všetky zvyšné stĺpce.

Pri východiskovej veľkosti šírky rozlíšenia, čo v tomto prípade znamená pod 400px, všetky stĺpce nadobúdajú 100%-nú šírku nadradeného elementu a sú zobrazené pod sebou.

Rovnako ako pri odsadeniach jednotlivých stĺpcov od seba, aj v tomto prípade definujeme kľúčovú CSS triedu fixed aplikovateľnú na rodičovský obaľovací element stĺpcov, ktorá zabezpečí, že všetky stĺpce v množine sú zobrazené v riadku, nie pod sebou.



Obrázok – správanie sa stĺpcov pri šírke rozlíšenia menšej ako 400 pixlov

### Skrývajúce a viditeľné elementy

Častou požiadavkou programátorov webových stránok je možnosť skryť menej dôležitý obsah napríklad pri zobrazení na mobilných zariadeniach kde zaberajú mnoho miesta. Elementy, ktoré nechceme zobrazovať pri určitých šírkach rozlíšenia môžeme skryť pridaním triedy:

* hidden-small, ktorá skrýva obsah pri šírke rozlíšenia menšej ako 400 pixlov,
* hidden-medium , ktorá skrýva obsah pri šírke rozlíšenia menšej ako 800 pixlov,
* hidden-large, ktorá skrýva obsah pri šírke rozlíšenia menšej ako 1200 pixlov.

Pre dosiahnutie opačného správania, teda zobrazenia elementov pri určitých šírkach rozlíšenia slúži trojica CSS tried:

* visible-small, ktorá zobrazí obsah pri šírke rozlíšenia menšej ako 400 pixlov,
* visible-medium, ktorá zobrazí obsah pri šírke rozlíšenia menšej ako 800 pixlov,
* visible-large, ktorá zobrazí obsah pri šírke rozlíšenia menšej ako 1200 pixlov.

### Elementy s nastavenou maximálnou šírkou

Uvažujme, že návštevníci webových stránok môžu k nim pristupovať zo zariadení, ktoré majú veľkú fyzickú veľkosť zobrazovacej jednotky ako napríklad televízory. Vzhľadom na to, že veľkosti šírky blokových elementov nadobúdajú hodnotu 100% okna prehliadača, môže sa stať že obsah webovej stránky bude pôsobiť „roztiahnutý“. Pre zamedzenie tejto skutočnosti sme definovali CSS triedy, ktoré nastavia blokovým elementom maximálnu šírku. Sú to triedy max-size-X, kde X môže nadobúdať hodnoty 800, 1000 a 1200, ktoré sú ekvivalentné hraničným hodnotám v pixloch. K implementácii týchto tried sme definovali aj vycentrovanie elementov na stred okna webového prehliadača pomocou CSS vlastnosti margin na hodnotu auto.

## Plugin pre zobrazenie navigačného menu

Jeden z komponentov je plugin pre zobrazenie navigačného menu, ktorý je možné do webovej stránky začleniť pomocou jednoduchej štruktúry. Zdrojový kód je zložený z CSS a JavaScript súborov a je závislý na JavaScriptovej knižnici jQuery. Plugin je plne responzívny a pri šírke zobrazovacieho okna menšej ako 800 pixelov celý jeho obsah skryje a zobrazí rozklikávaci prvok (*roller*) pre zobrazenie tohto obsahu. To má za následok uvoľnenie zobrazovacej plochy pre hlavný obsah webovej stránky. Plugin je zložený z dvoch základných a k nim niekoľko doplnkových CSS tried.

* Základná obaľovacia trieda je reprezentovaná kľúčovým názvom plug-menu, ktorá má nasledovnú jednoduchú štruktúru. Obsahuje jednotlivé položky menu plug-menu-item kde každá obsahuje záhlavie plug-menu-head do ktorého môžeme vložiť požadovaný obsah, ktorý chceme zobraziť. Napríklad text alebo hypertextový odkaz reprezentujúci danú položku. Navyše je možnosť v tomto záhlaví zobraziť doplnkovú ikonu pomocou triedy plug-menu-icon.
* Druhá základná trieda je plug-menu-roller, ktorá reprezentuje vyššie spomínaný rozklikávací prvok.

Každá položka menu môže obsahovať rozklikávacie *sub-menu (pod-menu)* použitím jednotnej štruktúry plug-menu vloženú pod záhlavie. Zobrazenie sub-menu je *absolútne*, čo znamená, že sa zobrazuje prekrytím zvyšného obsahu. Toto správanie je dosiahnuté CSS vlastnosťou position a jeho hodnotou absolute.

Pre možnosť identifikácie stlačenia tlačidla je najvhodnejšie použiť predefinovanú funkciu click( handler ) ponúkanú knižnicou jQuery, na elementy triedy plug-menu-head. Ako *handler* určíme funkciu, ktorá dynamicky manipuluje s pridávaním a odstraňovaním triedy clicked všetkých elementov triedy plug-menu-item následovne:

1. Ak je daný prvok stlačený, potom odstráni triedu clicked rodičovskému elementu (plug-menu-item)
2. Ak nie je daný prvok stlačený, potom odstráni triedu clicked všetkým elementov triedy plug-menu-item. Sebe a všetkým rodičovským triedam tohto typu triedu clicked nastaví.

Trieda clicked ovplyvňuje to, či sa daný prvok zobrazí alebo nie pomocou CSS vlastnosti display.

Na rovnakom zjednodušenom princípe manipulácie s triedami pracuje rozklikávací prvok *roller*.

## Plugin pre zobrazenie slideshow

# ZÁVER

Zámerom práce bolo zefektívniť vývoj používateľského rozhrania pri tvorbe webových aplikácií a zabezpečiť správne zobrazenie webovej stránky bez ohľadu na rozmery displeja a použitého prehliadača používateľov, čo sa nám úspešne podarilo. Poskytujeme možnosť programátorom využívať pokročilé funkcie knižnice jednoduchým písaním zdrojového kódu webovej stránky v jazyku HTML, čo odľahčí celý proces vývoja webových stránok.

Počas jej tvorby sme získali cenné teoretické aj praktické vedomosti v rôznych oblastiach vývoja webových aplikácií. Zoznámili sme sa z najpoužívanejšími návrhmi webových architektúr, webových dizajnov a prístupov k organizácií zdrojového kódu.

Knižnica je vhodne optimalizovaná pre rýchlosť načítania webovej stránky, vzhľadom na to že použité technológie nie sú závislé od výkonu zariadenia na ktorom je webová stránka zobrazovaná. Ďalšou pridanou hodnotou je zvolená štruktúra knižnice, ktorá je členená do modulov.

Pri návrhu sme kládli zvýšenú pozornosť súčasnému trendu prístupu k webovým stránkam pomocou mobilných zariadení. Preto všetky súčasti knižnice podporujú ovládanie pomocou dotykovej obrazovky.

Hlavným zámerom bol dôraz na funkčnosť a preto nie je v knižnici nejakým spôsobom preddefinovaný grafický vzhľad jednotlivých komponentov. To dáva programátorom možnosť jednoduchého zasadenia knižnice do už existujúcich webových stránok, ktoré sú navrhnuté iným, starším návrhom webového dizajnu neoptimalizovaným pre mobilné zariadenia. Priaznivý vplyv to má aj pre indexáciu vo webovom vyhľadávači od spoločnosti *Google*, ktorý pri vyhľadávaní na mobilných zariadeniach zobrazuje len výsledky odkazujúce na web optimalizovaný pre mobilné zariadenia, ktorej efekt je zvýšenie návštevnosti danej stránky.

Testovanie knižnice prebiehalo priebežne počas vývoju na šiestich rôznych webových prehliadačoch určených pre stolové počítače a mobilných zariadeniach. Pri testovaní sme narazili na nekonzistentné správanie webových prehliadačov a museli sme aplikovať patričné zaopatrenia na vyriešenie problému.

Používanie knižnice v praxi si vyskúšalo niekoľko ľudí, ktorí jej používanie odporúčajú. V budúcnosti by som chcel knižnicu rozširovať o ďalšie možnosti a spraviť tak vývoj webových stránok triviálnou záležitosťou pre každého.

# ZOZNAM REFERENCIÍ

|  |  |
| --- | --- |
| [1] | W3C, „Web Services Architecture,“ 2015. [Online]. Available: http://www.w3.org/TR/ws-arch/. |
| [2] | IBM, „RESTful Web services,“ 2015. [Online]. Available: https://www.ibm.com/developerworks/webservices/library/ws-restful/. |
| [3] | W3C, „World Wide Web Consortium,“ 2015. [Online]. Available: http://www.w3.org/Consortium/. |
| [4] | W3C, „HTML & CSS,“ 2015. [Online]. Available: http://www.w3.org/standards/webdesign/htmlcss. |
| [5] | W. Schools, „HTML Introduction,“ 2015. [Online]. Available: http://www.w3schools.com/html/html\_intro.asp. |
| [6] | W. Schools, „HTML and XHTML,“ 2015. [Online]. Available: http://www.w3schools.com/html/html\_xhtml.asp. |
| [7] | W3C, „HTML5,“ 2015. [Online]. Available: http://www.w3.org/TR/html5/. |
| [8] | W3C, „Cascading Style Sheets,“ 2015. [Online]. Available: http://www.w3.org/Style/CSS/. |
| [9] | W. Schools, „CSS Introduction,“ 2015. [Online]. Available: http://www.w3schools.com/css/css\_intro.asp. |
| [10] | W3C, „Introduction to CSS3,“ 2015. [Online]. Available: http://www.w3.org/TR/2001/WD-css3-roadmap-20010523/. |
| [11] | W. Schools, „CSS3 Introduction,“ 2015. [Online]. Available: http://www.w3schools.com/css/css3\_intro.asp. |

ZOZNAM ILUSTRÁCIÍ

Obrázok 1 – zobrazenie stránky vo webovom prehliadači 2

Obrázok 2 – diagram MVC architektúry 4

Obrázok 3 – štrukturálne elementy 7

Obrázok 4 – príklad CSS pravidla 8

Obrázok 5 – najpoužívanejšie prehliadače pre počítače 11

Obrázok 6 – najpoužívanejšie prehliadače pre mobilne zariadenia 12

Obrázok 7 – statický webdizajn 15

Obrázok 8 – liquid webdizajn, zobrazenie pre mobil, tablet, počítač9 16

Obrázok 9 – adaptívny webdizajn, zobrazenie pre mobil, tablet, počítač9 16

Obrázok 10 – responzívny webdizajn, zobrazenie pre mobil, tablet, počítač9 17

Obrázok 11 – Základné rozloženie 23

Obrázok 12 – správanie sa stĺpcov pri šírke rozlíšenia menšej ako 400 pixlov 24

# PRÍLOHY

K práci prikladám CD médium, na ktorom je uložená vypracovávaná knižnica, spolu s príkladovým webom využívajúci túto knižnicu pre návrh dizajnu.

1. **Slideshow** je funkčný prvok webu, ktorý slúži na prezeranie položiek v množine. [↑](#footnote-ref-1)
2. GUI je grafické používateľské rozhranie. [↑](#footnote-ref-2)
3. [online] dostupné na: http://www.zdrojak.cz/wp-content/uploads/2009/05/02-MVC-zakladni-vztahy-124164090266884.png [↑](#footnote-ref-3)
4. [online] dostupné na: http://www.w3schools.com/html/html5\_semantic\_elements.asp [↑](#footnote-ref-4)
5. [online] dostupné na: http://ta3d.hys.cz/css\_kapitola2.php [↑](#footnote-ref-5)
6. [online] dostupné na: http://gs.statcounter.com/#desktop-browser-ww-monthly-201503-201504-bar [↑](#footnote-ref-6)
7. [online] dostupné na: http://gs.statcounter.com/#mobile\_browser-ww-monthly-201503-201504-bar [↑](#footnote-ref-7)
8. **Scrollbar** slúži pre posun zobrazenia webovej stránky vzhľadom na okno prehliadača po horizontálne osi. [↑](#footnote-ref-8)
9. [online] Ukážka reálneho správania sa v prehliadači dostupné na: http://www.liquidapsive.com/ [↑](#footnote-ref-9)
10. Preprocesor je program, ktorý zo vstupných dát produkuje výstupný kód na použitie v inom programe. [↑](#footnote-ref-10)
11. [online] podrobnejšie vysvetlené v článku na adrese: https://css-tricks.com/snippets/css/clear-fix/ [↑](#footnote-ref-11)