ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE
PROVOZNĚ EKONOMICKÁ FAKULTA

PROVOZNĚ EKONOMICKÁ FAKULTA

BAKALÁŘSKÁ PRÁCE

Využití CSS a JavaScript frameworku pro tvorbu pokročilé webové aplikace

Autor: Marek Tůma
Vedoucí práce: Ing. Petr Benda

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačních technologií

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Tůma Marek

Informatika

Název práce

Využití CSS a JavaScript frameworku pro tvorbu pokročilé webové aplikace

Anglický název

Usage of CSS and JavaScript framework for advanced web application development

Cíle práce

Bakalářská práce bude zaměřena na výběr javascriptového a css frameworku pro tvorbu pokročilé webové aplikace. Hlavním cílem práce bude výběr vhodného css a javascript frameworku pro tvorbu bankovní aplikace a srovnání navrženého řešení s řešením nevyužívajícím frameworky.
Dílčí cíle bakalářské práce jsou:
- analyzovat odborné zdroje a zjištěné poznatky použít při výběru technologií pro vlastní řešení tvorby aplikace,
- srovnat nejpoužívanější javascript a css frameworky z hlediska vhodnosti pro danou aplikaci,
- vybrat řešení, které bude použito k realizaci.
- implementovat vybrané řešení.
- demonstrovat na konkrétních případech vhodnost využití vybraného řešení.

Metodika

Metodika řešené problematiky bakalářské práce bude založena na studiu a analýze odborných informačních zdrojů a konzultacích s vedením společnosti, které požaduje realizaci dané aplikace.
Vlastní řešení bude spočívat v návrhu aplikace na základě technologií používaných zadavatelem a jejich doplnění o řešení založené na klientských technologiích, které budou vybrány na základě rešeršní analýzy a provedených hodnocení.
Hlavními hledisky při výběru vhodné technologie budou znovupoužitelnost a rozšiřitelnost vybraného řešení.
Na základě syntézy teoretických poznatků a poznatků získaných během řešení vlastní aplikace budou formulovány závěry bakalářské práce.

Harmonogram zpracování

1) Příprava a studium odborných informačních zdrojů, upřesnění dílčích cílů práce a volba postupu řešení: 06/2011
2) Zpracování přehledu řešené problematiky dle informačních zdrojů: 07/2011 - 08/2011
3) Vypracování vlastního řešení, diskuze a zhodnocení výsledků: 09/2011 - 10/2011
4) Tvorba finálního dokumentu bakalářské práce: 11/2011 - 01/2012
5) Odevzdání bakalářské práce a teze: 02/2012

Rozsah textové části

30 - 40 stran

Klíčová slova

css, javascript, framework, web, jQuery, cofeescript, MooTools, Blueprint, 960 CSS, jquery, UI css framework, Internet

Doporučené zdroje informací

ORCHARD, Leslie M.; PEHLIVANIAN, Ara; KOON, Scott. Professional JavaScript Frameworks : Prototype,YUI, ExtJS, Dojo and MooTools (Wrox Programmer to Programmer). 1 edition. [s.l.] : Wrox, 2008. 888 s. ISBN 047038459X.

BUDD, Andy; COLLISON, Simon; MOLL, Cameron. CSS Mastery : Advanced Web Standards Solutions. 2nd ed. edition. [s.l.] : Apress, 2009. 366 s. ISBN 1430223979.

SURHONE, Lambert M.; TENNOE, Mariam T.; HENSSONOW, Susan F. Blueprint : CSS Framework. [s.l.] : BETASCRIPT PUBLISHING, [2010]. 80 s. ISBN 6131494487.

Build better sites with CSS frameworks. Practical Web Design Magazine. 2010, July, s. 1-4. ASIN B003XCZC8M.

RUTTER, Jake. Smashing jQuery. 1 edition. [s.l.] : Wiley, 2011. 336 s. ISBN 047097723X.

CHAFFER, Jonathan; SWEDBERG, Karl. Learning jQuery : Better Interaction Design and Web Development with Simple JavaScript Techniques. 1st edition. [s.l.] : Packt Publishing, 2007. 380 s. ISBN 1847192505.

MEYER, Eric. Smashing CSS : Professional Techniques for Modern Layout. 1 edition. [s.l.] : Wiley, 2010. 304 s. ISBN 047068416X.

Comparison of JavaScript frameworks. In Wikipedia : the free encyclopedia [online]. St. Petersburg (Florida) : Wikipedia Foundation, 31.3.2008, last modified on 6.6.2011 [cit. 2011-06-25]. Dostupné z WWW: <http://en.wikipedia.org/wiki/Comparison_of_JavaScript_frameworks>.

Noupe.com : curiosity for finding what's new &better [online]. 1.2.2011 [cit. 2011-06-25]. 5 popular CSS Frameworks + Tutorials & Tools for Getting Started. Dostupné z WWW: <http://www.noupe.com/css/5-popular-css-frameworks-tutorials-tools-for-getting-started.html>.

The jQuery Project. JQuery : write less do more [online]. [200?] [cit. 2011-06-25]. Dostupné z WWW: <jquery.com>.

Vedoucí práce

Benda Petr, Ing.

Konzultant práce

Ing. Václav Lohr

Termín odevzdání

březen 2012

doc. Ing. Zdeněk Havlíček, CSc.
Vedoucí katedry

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE
Provozně ekonomická fakulta

prof. Ing. Jan Hron, DrSc., dr.h.c.
Děkan fakulty

V Praze dne 21.11.2011

Prohlášení

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

V Praze dne 20. 3. 2012 Marek Tůma

Poděkování patří Ing. Pavlu Jelínkovi, majiteli společnosti Astra Systems, s.r.o. za umožnění participace na projektu pokročilých webových formulářů a
Ing. Petru Bendovi za vedení práce.

Název práce: Využití CSS a JavaScript frameworku pro tvorbu pokročilé webové aplikace
Autor: Marek Tůma
Katedra (ústav): Provozně ekonomická fakulta
Vedoucí bakalářské práce: Ing. Petr Benda
e-mail vedoucího: benda@pef.czu.cz

Abstrakt: Bakalářská práce se zabývá analýzou nasazení JavaScriptového a CSS frameworku na modelovou aplikaci – formulářové žádosti z bankovní oblasti. Na začátku je čtenář uveden do problematiky, jsou definovány cíle a metodika. Poté se práce zabývá protokolem http ve vztahu k událostem na stránce, které mohou ovlivňovat funkčnost těchto frameworků. Dále jsou v práci popsány zvažované frameworky, stanovena kritéria pro výběr vhodných řešení a tato řešení vybrána. V rámci JavaScriptu je vybrán framework jQuery, v rámci CSS technologií Blueprint.
Dále je na konkrétních příkladech ukázána vhodnost použití těchto řešení pro modelovou aplikaci, naznačen ekonomický přínos a jsou formulovány závěry.

Klíčová slova: CSS, JavaScript, framework, web, jQuery, MooTools, Blueprint, 960 CSS, jQuey UI css framework, Internet

---

Title: Usage of CSS and JavaScript framework for advanced web application development
Author: Marek Tůma
Department: Faculty of Economics and Management
Supervisor: Ing. Petr Benda
Supervisor's e-mail: benda@pef.czu.cz

Summary: This thesis is based on analysis of possibility of JavaScript and CSS framework usage on model website – bank form application. At the beginning the problem of this thesis are shown to the reader, and then comes setting of the aims and methods. After that HTTP protocol is explained in details focused on showing browser events, which can affect framework behavior. Considered solutions are listed, criteria are set and two technologies are chosen. jQuery is chosen JavaScript solution and Blueprint is the CSS one. Thereafter suitability of using these two solutions on model application is discussed and proved using specific examples. Economy of using such technologies is discussed and conclusions are made.

Keywords: CSS, JavaScript, framework, web, jQuery, MooTools, Blueprint, 960 CSS, jQuey UI css framework, Internet

Obsah

Seznam tabulek

Seznam obrázků

1 Úvod

Bakalářská práce se zabývá výběrem CSS a javascript frameworku pro tvorbu pokročilé webové aplikace. Modelem této pokročilé webové aplikace bude bankovní formulář pro složitější projekty, jako je například životní pojištění, pojištění budov, žádosti o hypotéku a další bankovní produkty.

Takovéto formuláře jsou obvykle stavěny jako vícestránkové, musí mít přívětivou validací vkládaných dat a měly by být snadno rozšiřitelné. V průběhu výběru technologií pro tuto aplikaci je tedy nutno držet se těchto požadavků. (1)

Serverovou část aplikace předpokládejme naprogramovanou v jazyce PHP ve spojení s některou z mnoha SQL databází. Uživatel odesílá vyplnění jednotlivých stránek formuláře na server separátním http requestem, a jako odpověď dostává skript pro příslušnou změnu stránky, tj. buď přepnutí na další stranu formuláře, označení chyb ve vyplnění stávající stránky anebo přesunutí na poděkování za vyplnění a další pokyny. Pro takovéto standardizované úkoly se často využívají různá předpřipravená řešení nazývané v frameworky. A právě srovnáním, výběrem a příklady použití takovýchto technologií se bude tato práce zabývat.

2 Cíl práce a metodika

2.1 Hlavní cíl

Bakalářská práce bude zaměřena na výběr javascriptového a css frameworku pro tvorbu pokročilé webové aplikace.

Hlavním cílem práce bude analýza vhodnosti nasazení css a javascript frameworku, pro tvorbu bankovní aplikace, výběr případného řešení a srovnání navrženého řešení s řešením nevyužívajícím frameworky.

2.2 Dílčí cíle

- Analyzovat odborné zdroje a zjištěné poznatky použít při výběru technologií pro vlastní řešení tvorby aplikace,
- srovnat nejpoužívanější javascript a css frameworky z hlediska vhodnosti pro danou aplikaci,
- vybrat řešení, které bude použito k realizaci,
- implementovat vybrané řešení,
- demonstrovat na konkrétních případech vhodnost využití vybraného řešení

2.3 Metodika práce

Metodika řešené problematiky bakalářské práce bude založena na studiu a analýze odborných informačních zdrojů a konzultacích s vedením společnosti, které požaduje realizaci dané aplikace.

Vlastní řešení bude spočívat v návrhu aplikace na základě technologií používaných zadavatelem a jejich doplnění o řešení založené na klientských technologiích, které budou vybrány na základě rešeršní analýzy a provedených hodnocení.

Hlavními hledisky při výběru vhodné technologie budou znuvupoužitelnost a rozšiřitelnost vybraného řešení.

Na základě syntézy teoretických poznatků a poznatků získaných během řešení vlastní aplikace budou formulovány závěry bakalářské práce.

3 Přehled řešené problematiky

Pod pojmem framework rozumíme určitou softwarovou strukturu navrženou ke zjednodušení vývoje standartních aplikací či jejich částí.

V současné době (psáno 2011) zažívají tyto frameworky velký rozmach spojený s postupným rozšířením broadband internetového připojení po celém světě. Například dle měření společnosti Akamai (2), což je jeden z největších CDN operátorů na světě, se průměrná rychlost internetového připojení v ČR zvedla ze 4269 kbps (Q3 2007) na 7377 kbsp (Q2 2011).

Javascriptové i CSS frameworky jsou k uživateli standardně distribuovány jako separátní soubory přenášené dalším http požadavkem, což v minulosti odrazovalo od jejich použití. Css framework se obvykle zavádí do html dokumentu v rámci hlavičky pomocí tagu <link>, javascriptový pomocí tagu <script>.

Dalším obecným trendem podporujícím nasazení frameworku je zlevňování procesorového času. Firemní snahy o co nejnižší náklady se tudíž přenáší z co nejméně náročného kódu na co nejrychlejší a nejlevnější vývoj.

Metodikou výběru frontendových technologií se na internetu zaobírá velké množství specializovaných serverů, za všechny uveďme například Smashing magazine (http://www.smashingmagazine.com/) nebo český Root (root.cz) a dále velké množství bloggerů i jiných autorů, nicméně v této práci je potřeba přihlédnout k charakteru požadované aplikace (viz úvod) a vybrat z dostupných možností dobré a rozumně implementovatelné řešení. U této problematiky se také setkáváme s problémem rychlého stárnutí informací, protože webové technologie jsou velmi dynamický a rychle se rozvíjející obor. Řešení, které bude popsané v této práci, není v tomto ohledu výjimkou.

4 Vlastní práce

4.1 Události v prohlížeči a spuštění frameworku

Od prvního vyslání http požadavku klienta na server po úplné vykreslení stránky a její prezentace uživateli probíhá v prohlížeči mnoho různých událostí. Tato kapitola představuje určité „nahlédnutí pod pokličku“ a popisuje základní filosofii tohoto procesu nutnou pro pochopení některých dějů ve frameworcích.

4.1.1 Http protokol

Valná většina komunikace, kterou dnes internetový prohlížeč realizuje, probíhá pomocí http protokolu.

Hypertext Transfer Protocol (http) je protokol pracující v aplikačním levelu ISO-OSI modelu sítě pro distribuované, kolaborativní (tj. vytvářené společně), hypermediové informační systémy. Je to obecný, bezestavový protokol, který může být použit na různé úkoly i mimo hypertext, jako jsou name servery, distribuované object management systémy, a to díky extenzím požadavkových metod, chybových stavů a hlaviček. Jeho výhodou je snadná úprava formy reprezentace dat, díky čemuž lze systémy stavět nezávisle na přenášených datech.

HTTP je používán W3C iniciativou již od roku 1990. Jeho původní specifikace je navržena v RFC 2068. (3)

4.1.2 Synchronní a asynchronní volání

Historicky je http protokol jako takový vždy synchronní, ve smyslu, že klient (zpravidla prohlížeč) pošle požadavek a čeká na odpověď serveru. (Toto paradigma začíná být bouráno technologií websockets, kdy prohlížeč umí zpracovat i data poslaná ze serveru, aniž by jim nutně musel předcházet požadavek (4), tato technologie ovšem není zatím finálně standardizovaná ani nijak rozšířená, proto se jí tato práce dále nezabývá.)

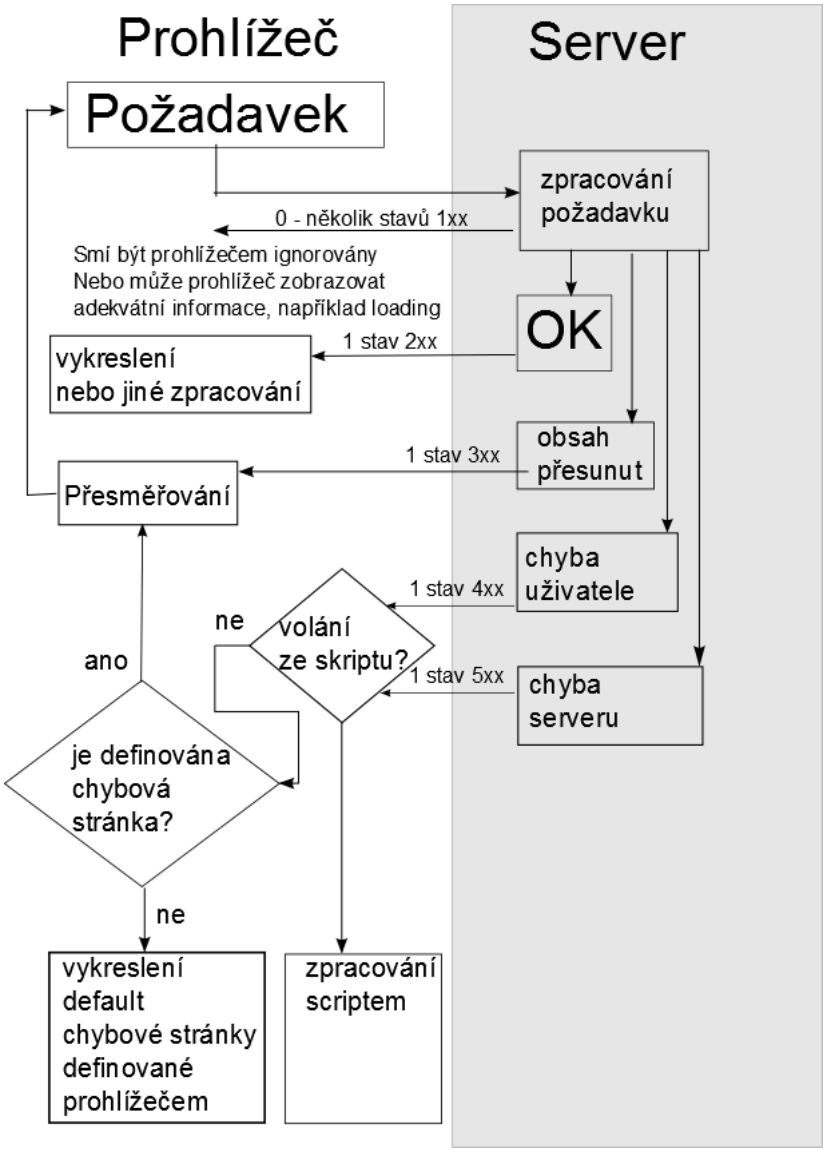
Ve smyslu použití na vykreslované stránce ovšem mohou být http požadavky (a může jich být v rámci webové aplikace téměř nekonečné množství) volány synchronně i

asynchronně. V takovémto případě synchronní volání znamená zastavení procesu vykreslení, případně provádění skriptu, dokud není požadavek dokončen a zpracován.

Příkladem takovéhoto synchronního volání je i vložení externího skriptu (pomocí tagu <script>), nebo stylopisu(pomocí tagu <link>) do hlavičky dokumentu. Vkládají se takto zdroje nutné pro správnou funkci webové stránky.

Pokud je takovéto vložení provedeno až v těle (tag <body>) dokumentu, provede se volání asynchronně, prohlížeč nečeká na jeho dokončení a dále zpracovává stránku či skript. Některé skripty, zejména pokud nejsou důležité pro vlastní běh stránky (uveďme například google analytics, nebo reklamy) je zvykem tímto způsobem vkládat do kódu stránky co nejpozději, často až před ukončení tagu </body>, aby uživateli neprodlužoval čekání na vizuální prezentaci stránky (5).

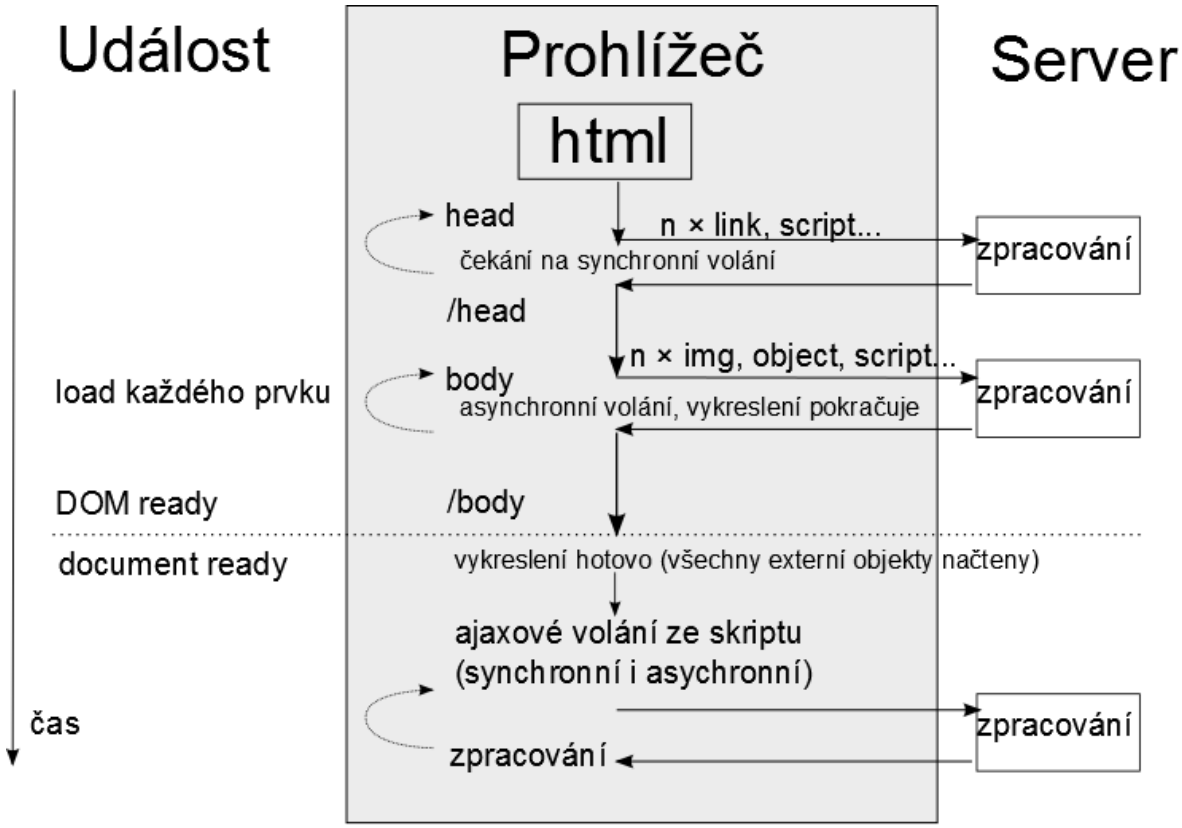
4.1.3 http požadavek – život http požadavku



Obrázek 1 Život http požadavku, zdroj: autor

Na Obr. 1 - (3) zpracováno dle sekce 10 - je zobrazen život a rozhodovací strom na straně prohlížeče u volání typického http požadavku. Požadavek je vyslán klientem, serverem zpracován a vrácen klientu ke zpracování. Cesta, kterou by mělo projít nejvíce http požadavků (je časově a zdrojově nejkratší) je prohlížeč->Server–>stav 200 OK->zpracování prohlížečem.

4.1.4 http požadavek – vykreslení html



Obrázek 2 Vykreslení html v prohlížeči, zdroj: autor

Obr 2. – zpracováno dle (6)- Popisuje jaké události (zachytitelné skriptovacím jazykem) nastávají v jakém okamžiku v závislosti na zpracování html kódu webové stránky. Zpracování těla http response (neplést s body v html dokumentu) začíná zpracováním hlavičky tj. tagu head.

V rámci tohoto tagu jsou zpravidla zavedeny externí zdroje nutné pro správné vykreslení, či správnou funkci webové stránky, takovéto zdroje jsou zaváděny synchronně, tj. prohlížeč nepřistoupí ke zpracování následujícího kódu, dokud není stávající kód zpracován. Toto je jedním z důkazů, že i pořadí prvků v XML dokumentu může mít sémantický význam. Například následující kód obsahující zavedení javascriptového frameworku jQuery a jeden z jeho modulárních doplňků jQuey UI může fungovat jen díky této vlastnosti. Skript obsažený v jquery.ui.js je totiž přímo závislý na zinicializovaném javascriptovém objektu $, či jQuery reprezentujícího jQuery framework.

```
<script type="text/javascript‟ src="jquery.js" />
<script type="text/javascript" src="jquey.ui.js" />
```

Ve chvíli, kdy je obsah tagu head zpracovaný, přichází na řadu zpracování tagu body.

I v průběhu tohoto zpracování prohlížeč vysílá http požadavky, zejména pro získání externích obrázků, objektů (např. typu flash), nebo scriptů. Tyto požadavky se však již provádí asynchronně a další zpracování kódu dokumentu nečeká na jejich výsledek.

Každý z prvků, u nějž je dokončeno vykreslení spouští událost load, na niž je možno klientským skriptem reagovat. Ve chvíli, kdy je dokončen load posledního potomka, je zavolána i událost load elementu body. Na tuto chvíli například s činností čeká framework Prototype. Modernější řešení spočívá v detekci okamžiku, kdy je zavedený celý DOM, ale nečeká na donačtení všech externích zdrojů. Tohoto přístupu využívá například jQuery, detekuje kdy je DOM takzvaně traverzovatelný a v tu chvíli spouští událost $(document).ready, kdy je již možné s DOM pracovat. Vzhledem k tomu, že se nečeká na načtení jiných zdrojů, nastává událost DOM ready zpravidla dříve, než body onload.

Po načtení stránky je také možno vysílat na server další požadavky pomocí technologie AJAX, viz dále.

obrázky a jiné objekty

Obrázky a jiné objekty se, jak již bylo řečeno, zpravidla zavádí speciálními tagy <img> a <object> v rámci html dokumentu. Při načtení volají událost (on)load, při chybě - například http status 404: obsah nenalezen - událost (on)error. Protože v okamžiku zavádění takovéhoto objektu do stránky není známá odpověď serveru, bývá dobrým zvykem explicitně uvádět jeho rozměry. Ať už starším způsobem, tj. použitím atributů width a height přímo v tagu, nebo použitím stejnojmenných CSS vlastností. Prohlížeč si pak vyhradí prázdné místo, které doplní získaným zdrojem a zavolá událost load.

Je třeba zde ještě vysvětlit slůvko „zpravidla“ použité v první větě této podkapitoly. Málo známou možností je zavádění obrázků přímo jejich bitovou reprezentací zpravidla zakódovanou pomocí technologie Base64. Například v (7) je pomocí tohoto postupu zavedena ikonka takto:

```
<img
src="
mApmZmZtZfYmdakyH5BAQUAP8ALAAAAABQAA8AAAPb
WLrc/jDKSVe4OOvNu/9gqARDSRBHegyGMahqO4R0bQcjIQ8E4BMCQc930Jluy
GRmdAAcdiigMLVr
ApTYWy5FKM1IQe+Mp+L4rphz+qIOBAUYeCY4p2tGrJZeH9y79mZsawFoaIRxF
3JyiYxuHiMGb5KT
kpFvZj4ZbYeCiXaOiKBwnxh4fnt9e3ktgZyHhrChinONs3cFAShFF2JhvCZlG
5uchYNun5eedRxM
AF15XEFRXgZWWdciuM8GCmdSQ84lLQfY5R14wDB5Lyon4ubwS7jx9NcV9/j5+
g4JADs=
" alt="British Blog Directory" width="80" height="15" />
```

Výhodou je omezení počtu http volání na server a snížení serverové zátěže, nevýhodou je, že si prohlížeče neumí takto získaný obsah cachovat a musí ho pokaždé zpracovat znovu.

Specialitou mezi těmito objekty je tag <iframe> který umožňuje do webové stránky vložit celou další webovou stránku. Nastává zde ovšem problém, že událost onload v rodičovské stránce čeká na donačtení potomka (8)

AJAXové požadavky v rámci skriptu

Dokud uživatel stránku v prohlížeči nezavře, může na ní docházet k vykonávání skritů. Zpravidla se dnes jedná o javascript, založený na syntaxi jazyka C, pro starší prohlížeče (dnes už jen Internet explorer) byl k disposici i VB script, VB značí visual basic, založený na syntaxi jazyka basic. Tento script si může vyžádat další http požadavky a zpracovat jejich odpovědi. Těmto požadavkům se říká AJAX – asynchrounious javascript and XML, ačkoliv ve skutečnosti nemusí tyto požadavky být ani asynchronní, ani nemusí přenášet XML (časté je využití pro přenos částí html dokumentu), vžil se tento název. Při studiu odborné literatury je nutno tedy zkoumat, jak autor daný výraz používá.

K vyslání požadavku ze scriptu se používají speciální objekty ActiveX, nebo XMLHttpRequest (v závislosti na prohlížeči). A práce s nimi je poměrně obtížná.

Javascriptový framework abstrahuje programátora od nutnosti znát jednotlivé implementace a to na různých stupních abstrakce. Jako příklad zde opět poslouží jQuery.

jQuery obsahuje základní volání (9)

```
$.ajax({objekt_parametru});
```

Ve kterém lze uživatelsky řídit množství parametrů, například lze nastavit request jako synchronní, řidit metodu, který se pro volání použije a podobně.

O něco méně komplexní jsou funkce $.get a $.post, až na nejvíce specializované úrovni najdeme funkci $(selector).load(), která slouží k naplnění elementu určeného selektorem obsahem získaným ajaxovým požadavkem.

Cache

Výraz cache se obvykle používá pro vyrovnávací (mezi)paměť. Kvůli zrychlení načítání html stránek si pohlížeč často cachuje obsah odpovědí na určité adrese, zejména pak právě externích obrázků, scriptů a stylopisů. Takováto cache se může řídit speciálními http hlavičkami (například Cache-Control, Last-Modified a podobně). V případě, že je požadavek vyhodnocen tak, že je možný načíst z cache, prohlížeč dále nečeká na přenesení těla http odpovědi, ale načte si ho právě z této paměti. Prohlížeč si může cachovat nejen samotný obsah, ale i jeho zpracovanou verzi, tato problematika je ovšem velmi obsáhlá a přesahuje rámec této práce. Důležité je zde jen uvést, že při práci s ajaxovými i jinými požadavky ve scriptu je nutno s cachováním počítat.

4.2 Javascriptový framework

Javascriptový framework slouží uživateli k ulehčení práce s javascriptovým kodem. Obvykle obsahuje zkratky k nejběžněji používaným funkcím. Například referenci na objekt s id="id_elementu" standartně referencovaný jako

```
document.getElementById("id_elementu");
```

lze v MooTools odkázat takto

```
document.id("id_elementu");
```

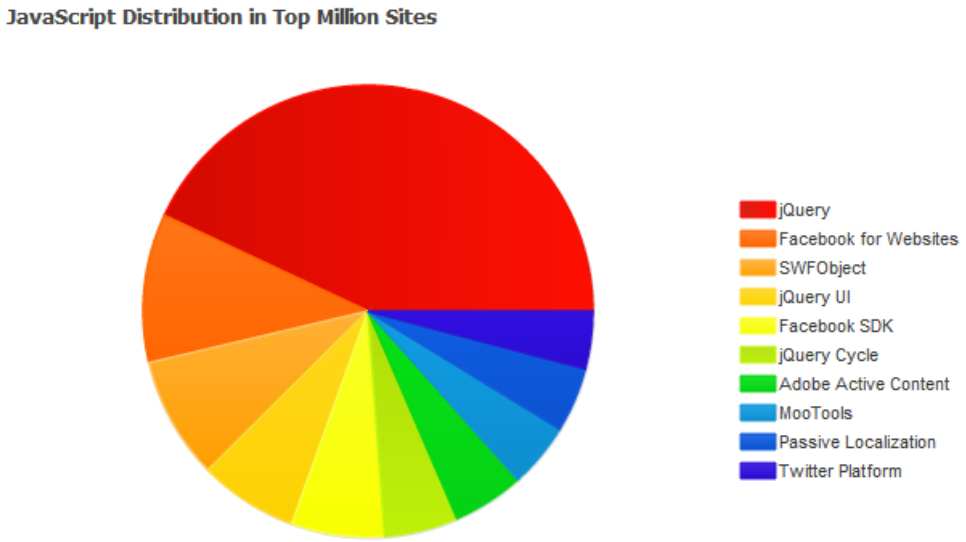
a v jQuery takto

```
$("#id_elementu");
```

Pozor ovšem, oba tyto případy nevrací přímo DOM element, ale nějakým způsobem „vylepšený“ objekt. jQuery používá technologii DOM wrapping, MooTools DOM extension, tyto technologie jsou popsány dále v kapitole 4.2.2, která se DOM wrappingem zabývá. Samotný DOM element je potom v jQuery dostupný jako atribut vráceného objektu

```
$("id_elementu")[0];
```

Vzhledem k tomu, že se tato práce zaobírá použitím javascriptu jako klientského jazyka pro práci s výše uvedenými formuláři, bude se, soustředit především na práci s DOM a http požadavky. Je tedy nutno si stanovit, mezi jakými frameworky bude proveden výběr. Z charakteru aplikace vyplývá, že odpadají úzce specializované frameworky určené například pro pokročilou práci s animací html 5 prvků (tedy k nahrazení technologie flash), frameworky určené pro snazší vývoj na mobilní zařízení, wysiwig editory, vizuální efekty, napojení na sociální sítě a podobně. Na Obr. 3 lze vidět, že toto omezení má smysl, protože dle měření společnosti Builtwith se do top 10 řešení dostaly jen 2 skutečné obecné frameworky a to jQuery a MooTools.



Obrázek 3 Zastoupení JavasScriptových technologií v top milionu stránkách, zdroj: (10)

4.2.1 Zavedení Javascriptového frameworku do website

Javascriptové frameworky se do website vkládají zpravidla pomocí tagu <script> umístěného v hlavičce (<head>) dokumentu. Viz výše (kapitola 4.1.2).

Dalším možným způsobem je načtení pomocí scriptu. Například jQuery používá následující syntax:

```
jQuery.getScript( url [, success(script, textStatus, jqXHR)]
);
```

kde parametr url představuje adresu dalšího skriptu a success je callbacková funkce, která se volá po načtení a vykonání získaného scriptu. Vzhledem k tomu, že se takto volaný script spouští v globálním kontextu, lze takto dobře do stránky dobře vkládat pluginy pro jQuery, jelikož volaný script je přímo závislý na globálním objektu jQuery.

Příkladem takto volaných pluginů může být například jQuey UI, spinner a další.

Výhodou načítání pomocí scriptu je zejména nezpomalení úvodního načtení stránky.

4.2.2 Požadavky na JS framework

Pro každý výběr je nutné stanovit kritéria hodnocení, každému kritériu bude dále věnovaná samostatná podkapitola. Požadavky jsou vybrány na základě studia příslušné literatury a konzultace ve společnosti zhotovitele.

Uživatelská základna/komunita

Jako základní kritérium výběru byla stanovena silná uživatelská základna, protože pro zadanou aplikaci je nutné, aby framework přežil, i pokud původní autor na jeho vývoj nemá čas, či prostředky. Javascriptové frameworky jsou ve většině případů vyvíjeny jako open source komunitní projekty, velikost a stálost komunity je tedy klíčovým kritériem.

Existuje zde kladná zpětná vazba mezi počtem nasazení frameworku a jeho stálostí na trhu (velikostí a aktivitě komunity). Jinak řečeno, pokud se framework aktivně používá, komunita ho vyvíjí a drží krok s novými prohlížeči a technologiemi. A čím lépe komunita kolem frameworku drží krok, tím více se používá na nových projektech.

Do prvního předvýběru jsou tedy vybrány frameworky, které mají dle spolehlivého měření největší zastoupení na nejnavštěvovanějších webových stránkách. Pozor, součet je větší, než 100%, protože existuje nemalá množina stránek, které používají více, než jeden framework.

JS Framework	Podíl	Podíl mezi frameworky
Žádný	48.2%	--
JQuery	43.9%	84.6%
MooTools	5.2%	10.0%
Prototype	4.3%	8.3%
Script.aculo.us	3.2%	6.2%
ASP.NET Ajax	2.9%	5.6%
YUI Library	2.8%	5.5%
Spry	0.6%	1.2%
Dojo	0.2%	0.3%
Ext JS	0.1%	0.1%

Tabulka 1 Zastoupení javascriptových frameworků na webových stránkách, zdroj: W3C

Použitelnost se serverovou částí

Dalším vhodným kritériem je možnost nasazení na dané serverové řešení.

U většiny frameworků nezávisí klientské řešení na serverovém, problém zde nastává u ASP.NET knihoven, které se instalují jako zásuvné moduly do .NET frameworku. V uvažované aplikaci má být jako serverový jazyk použito PHP a tyto technologie zde jsou nepoužitelné.

Další těžko použitelnou technologií se stává Spry, který svým zaměřením na web-designery a používáním proprietárních rozšíření samotného html odrazuje od použití v klasicky stavěné aplikaci, kde má vývojář pod kontrolou všechna ajaxová volání.

Feature detection

Feature detection je schopnost řešení detekovat v prohlížečích určitou funkčnost. Zároveň se toto označení používá pro přístup k psaní kódu, který tuto vlastnost využívá. Opakem tohoto přístupu je browser detection, kdy se dotazue přímo na prohlížeč. V javascriptu existuje globální objekt navigator, který zprostředkuje informace o použitém prohlížeči (má vlastnosti jako například navigator.appName).

Různé prohlížeče se často liší ve zpracování určitých funkčností nebo postupů. Za všechny uveďme například AJAXové volání, tedy asynchronní požadavek na server, který nezpůsobí přenačtení stránky.

V Internet Exploreru se objekt tohoto požadavku zavádí takto:

```
Request = new ActiveXObject("Microsoft.XMLHTTP");
```

zatímco v jiných prohlížečích zpravidla takto:

```
Request = new XMLHttpRequest();
```

Při použití feature detection potom vypadá kód takto:

```
    if (window.XMLHttpRequest) {  //většina prohlížečů
Request = new XMLHttpRequest();
    }else if (window.ActiveXObject) { //IE
        Request = new ActiveXObject("Microsoft.XMLHTTP");
    }
```

Při browser detection potom například takto:

```
if (navigator.appName == "Microsoft Internet Explorer") {
    Request = new ActiveXObject("Microsoft.XMLHTTP");
}
```

Javascriptové frameworky často řeší funkčnost, která je v budoucnu plánovaná přímo pro samotný javascript, ale ještě není implementovaná do prohlížečů. Je tedy vhodné, když si sám framework pomocí feature detection zjišťuje, zda prohlížeč už tuto funkci neobsahuje a případně volá přimo ji.

Z uvedených frameworků feature detection podporují jQuery, MooTools, a YUI.

Trend

Dalším výběrovým kritériem je trend vývoje nasazování jednotlivých frameworků na webové aplikace. Toto kritérium je požadováno vedoucími pracovníky u zhotovitele. Důležité je vybrat takový framework, který má určitou historii a v čase vykazuje růst, nebo stagnaci, tedy nesmí se od jeho nasazování obecně dlouhodobě upouštět.

Zhotovitel poskytuje smluvní záruku na poskytnuté řešení a je nutné použít takové technologie, které v horizontu životnosti takovéto aplikace (předpoklad je kolem 5 let) budou stále podporovány jejich tvůrci a/nebo komunitou. Je zde vysloven předpoklad, že pokud neklesá oblíbenost frameworku, je jeho vývoji věnována dostatečná pozornost, zaručená zpětná kompatibilita nových verzí (které bude pravděpodobně nutné na aplikaci nasadit s příchodem nových verzí prohlížečů) a případné opravy chyb.

V níže použitých srovnáních (obr. 7-15) je opět použito měření z builtwith.com, na svislé ose jsou vynesena procenta nasazení, vodorovná osa ukazuje čas.

DOM Wrapping

DOM Wrapping je podobně jako feature detection určitou filosofií psaní frameworku. Spočívá v obalení DOM objektu do nativního javascriptového objektu, kterému jsou teprve poté přidávány funkčnosti, samotný DOM element bývá zpravidla nezměněn. Opakem tohoto přístupu je DOM Extension, který skriptem rozšiřuje standardní vlastnosti a metody DOM objektu. Tento přístup představuje opět do budoucna riziko,

protože v novějších verzích prohlížečů může být časem implementováno rozšíření DOM, které bude obsahovat metody pojmenované stejně jako metody použitého frameworku a dojde ke kolizi.

Jako takový nevhodný příklad je zde možné použít framework MooTools, který elementu rozšířenému pomocí funkce document.id přidává metodu setStyle.

DOM jako takový je „*interface nezávislý na platformě a použitém jazyku, který dovoluje programům a skriptům dynamicky přistupovat a měnit obsah, strukturu a styl dokumentů. Dokument může být dále zpracováván a výsledek tohoto zpracování může být zabudován zpět do prezentované stránky.“* (11)

Použití technologie DOM Extension tedy porušuje tuto filosofii, protože rozšiřuje DOM javascriptovými metodami a tím ho činí jazykově závislým.

Příklad potenciálně nebezpečného kódu:

```
var element = document.getElementById('id_elementu');
element.setStyle = function (style_name, style_value) {
    //vlastní funkce
};
```

používaném takto

```
var dom_element = document.getElementById('id_elementu');
dom_element.setStyle('float', 'left');
```

Lze s pomocí technologie DOM Wrapping psát například takto

```
var Wrapper = function(element){
            // inicializace wrapperu
}
Wrapper.prototype.setStyle = function (style_name, style_value) {
   //vlastní funkce
};
```

A ve vlastním javasriptovém kódu použít takto:

```
var dom_element = document.getElementById('id_elementu');
var wrappedElement = new Wrapper(dom_element);
wrappedElement.setStyle('float', 'left');
```

Firemní/osobní znalosti

Jedním z dalších kritérií je úroveň znalostí daného frameworku ve společnosti, která aplikaci zhotovuje, případně osobní znalosti programátora, který o použité technologii rozhoduje. Pokud už bylo v minulosti s daným frameworkem něco realizováno, je jeho další nasazení zpravidla méně časově náročné, než zaučování na nové technologie.

4.2.3 O zvažovaných frameworcích

Tato kapitola shrnuje základní informace o některých zvažovaných frameworcích.

jQuery

jQuery je novým druhem javascriptové knihovny.

jQuery je rychlá a stručná JavaScriptová knihovna, která zjednodušuje traverzování v HTML dokumentu, zpracovávání událostí, animace a ajaxové interakce určená k rychlému vývoji webových stránek. jQuery je navržená tak, aby změnila způsob, kterým píšete javascript. (12)

jQuery je javascriptová knihovna, která je vytvořena, aby pomohla Web designerům a developerům rychle psát a zlepšovat JavaScriptové interakce díky použití definovaného setu metod vybudovaných nad nativními JavaScriptovými funkcemi. jQuery nepřináší novou funkcionalitu, ale bere stávající těžko pochopitelná a aplikovatelná JavaScriptová API a zpřístupňuje je širšímu publiku pomocí jednoduše uchopitelné syntaxe, se kterou se snadno vyvíjí. (13)

jQuery je dnes nejvíce nasazovaným javascriptovým frameworkem, který vykazuje setrvalý nárůst v používání na webových stránkách. Jeho hlavní výhodou je škálovatelnost, protože je pro ni k disposici množství pluginů. Díky tomu nemusí vývojář používat funkce, které nepotřebuje, naopak si nalinkuje jen ty doplňky, které jsou pro jeho projekt relevantní. Odpadá tak jedna z nevýhod použití vělkých a hodně obecných frameworků.

Vývojáři jQuery na svých webových stránkách přímo vytváření nových doplňků podporují (14).

jQuery využívá pro usnadění vývoje pokročilé CSS 3 selectory, což mnohdy velmi zjednodušuje zápis kódu.

MooTools

MooTools je kompaktní modulární objektově orienovaný JavaScriptový framework určený středně a více pokročilým JavaScriptovým vývojářům. Dovoluje Vám psát výkonné, flexibilní a cross-browser kompatibilní kód díky svému elegantnímu, dobře zdokumentovanému a koherentnímu API.

Zdrojový kód MooTools respektuje striktní standardy a neprodukuje žádné warningy. Je vysoce dokumentován a má srozumitelné pojmenování proměnných: Radost pohledět a snadno porozumět. (15)

Díky způsobu, jakým zavádí funkcionalitu do prototypů nativních objektů je MooTools často přirovnáván k Prototype. Ale na rozdíl od pouhých klonů, MooTools poskytuje čisté a mocné API, které je vystavené se silným důrazem na konzistenci. Navíc MooTools nabízí animace a další prvky uživatelského rozhraní podobně jako jsou do Prototype zaváděny pomocí scriptaculous – a přesto si stále zachovává menší celkovou velikost. (16)

MooTools je standartně distribuován s některými linuxovými distribucemi (17), díky čemuž se zřejmě stal druhým nejpoužívanějším javascriptovým frameworkem. Podobně jako jQuery obsahuje silný engine pro CSS3 selectory.

Prototype

Prototype je JavaScript Framework, který se zaměřuje na ulehčení vývoje dynamických webových aplikací.

Obsahuje unikátní, jednoduše použitelnou sadu nástrojů pro class-driven vývoj a nejhezčí Ajaxovou knihovnu, která je k disposici. Prototype se rychle stává základní součástí vývoje webů pro vývojáře po celém světě. (18)

Prototype byl jednou z prvních JavaScriptových knihoven, které se dostaly na výsluní ve vlně webu 2.0. Když se v roce 2005 poprvé začalo mluvit o Ajaxu (který se zakrátko stal jistým minovým polem mezi prohlížeči díky specifickému zavedení v každém z nich), Prototype začal asistovat s cross browser kompatibilitou a začal poskytovat společný interface pro volání AJAX requestů. Také poskytuje cross-browser kompatibilní cesty k manipulaci s DOM a umožňuje se vývojáři soustředit jen na psaní kódu bez toho, aby se musel zabývat if-else bloky kódu specifického pro jednotlivé prohlížeče. (16)

Mimo výše popsané vlastnosti je zajímavým přínosem Prototype zavedení tříd a jejich dědičnosti (19), což pomáhá vývojářům zvyklým na určitý typ vývoje, ale odporuje paradigmatu javascriptu jako prototypového (neplést s názvem frameworku prototype) jazyka. Prototype se také dostal do distribuce Ruby on Rails.

Script.aculo.us

Script.aculo.us vám přináší jednoduše a cross-browser použitelné UI JavaScriptoví knihovny, které pomohou vašim webovým stránkám a aplikacím doslova létat.

Co je uvnitř? Animační framework, drag and drop, Ajaxové technologie, DOM nástroje a unit testing.

Je přídavkem k fantastickému frameworku Prototype. (20)

Script.aculo.us je rozšířením frameworku Prototype. Jeho nejvýznamnějším úspěchem je, že je distribuován s Ruby on Rails, ale samozřejmě je k disposici ke stažení a spolupráci i s ostatními řešeními.

YUI

YUI je zdarma poskytovaný opensource JavaScript a CSS framework pro vývoj vysoce internaktivních webových aplikací. YUI je poskytován pod BSD licencí a je k disposici na GitHubu pro forkování a přispívání.

YUI je prověřený škálovatelný rychlý a robustní. Naprogramován frontendovými inženýry z Yahoo! a přispěvateli z celého světa přináší JavaScriptovou knihovnu, za kterou stojí silná společnost, pro profesionály, kteří milují JavaScript (21)

Framework YUI těží ze zázemí silné mezinárodní firmy. Je k němu k disposici velmi dobrá dokumentace (jen instalační balík má 50 MB a jsou v něm k disposici příklady, dokumentace a podklady pro testování).

4.2.4 Volba použitého řešení

V předchozích kapitolách byly představeny zvažované frameworky i kritéria pro výběr. Tabulka 2 obsahuje srovnání zvažovaných technologií pomocí těchto kritérií. Grafy trendu použití jednotlivých technologií jsou uvedeny v přílohách. Zeleně zvýrazněny jsou nejlepší hodnoty v rámci každého kritéria.

FW	Uživatelská základna	Použitelnost se serverovou částí	Feature detection	Trend	DOM Wrapping
JQuery	84,6 %	Ano	Ano	Rostoucí	Ano
MooTools	10 %	Ano	Ano	Setrvalý	Ne
Prototype	8,3 %	Ano	Ne	Klesající	Ne
Script.aculo.us	6,2 %	Ano	Ne	Klesající	Ne
ASP.NET Ajax	5,6 %	Ne	Ne	Rostoucí	--
YUI Library	5,5 %	Ano	Ano	Setrvalý	Ano
Spry	1,2 %	Ano	Ne	Setrvalý	--
Dojo	0,3 %	Ano	Ne	Setrvalý	Ano
Ext JS	0,1 %	Ano	Ne	Rostoucí	Ano

Tabulka 2 Výběr použitého JavaScript frameworku metodou aspiračních úrovní, zdroj: autor

Použitím metody aspiračních úrovní došlo k výběru frameworku jQuery, protože jde o ostře dominantní varianta. V každém z hodnocených kritérií je stejná nebo lepší, než ostatní varianty a neexistuje jiná varianta se stejně dobrým hodnocením.

4.3 CSS Framework

CSS frameworky jsou zpravidla hotovými řešeními určenými zpravidla k standardizaci kódu a urychlení vývojářské práce. Podobně jako u Javascriptových frameworků (kapitola 4.2) je k disposici velké množství různých řešení a nástrojů. Od široce zasahujících řešení po velmi úzce specializované frameworky.

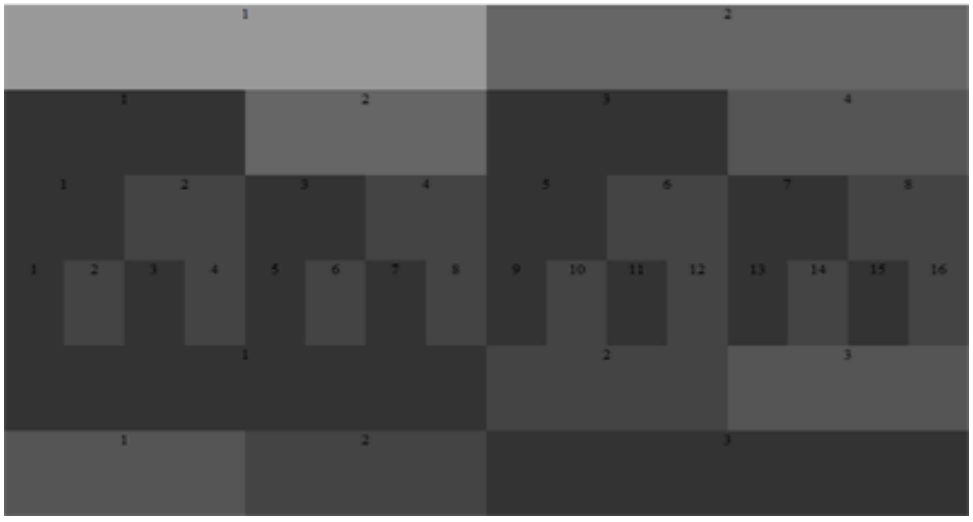
Příkladem úzce specializovaných frameworků může být například framework pro elastický layout Emastic, formulářový framework Formy a nebo tiskový framework Hartij - všechny od itlaského vývojáře Vladimira Carrera (22), frameworky řešící mobilní aplikace jako Less Framework, nebo frameworky závislé na javascriptovém řešení (například jQuery UI CSS).

Jako specialitu zde lze uvést CSS framework .dp50 (znamená div 50 percent) opět od Vladimira Carrera, který se sestává pouze z popisu pouze jedné třídy prvků.

Pro jeho malou velikost zde lze uvést celý zdrojový kód.

```
.dp50 {width:50%; float:left; display: inline; *margin-
right:-1px; }
```

A s jeho pomocí lze dosáhnout například následujícího rozvržení sloupců na webové stránce



Obrázek 4 Ukázka použití nejkratšího CSS frameeworku .dp50, zdroj: zdrojak.root.cz

Nevýhodou při takovémto použití je samozřejmě nutnost použití velkého množství zanořených prvků. Jedná se tedy spíše o „proof of concept“.

4.3.1 Vrstvy CSS frameworku

Obecně lze funkčnost CSS frameworku dle (23) rozdělit do následujících vrstev:

Vrstva	Nižší obecnost
Ornamenty	
Vzhled prvků	
Layout webu	
Alternativní media	
Typografie	
Reset	Vyšší Obecnost

Tabulka 3 Komponenty CSS Frameworků ve vrstvách designu webu, zdroj: zrojak.root.cz

Podívejme se na jednotlivé vrstvy blíže:

***Ornamenty** — sady ikonek, vzhled tlačítek a další sady CSS pravidel, sloužící dekorativním nebo vysoce specializovaným účelům…*

***Vzhled prvků** — typickým příkladem je systém pro definici vzhledu formulářů….*

***Layout webu** — systém pro definici rozvržení stránky. U frameworků jde většinou o fixní nebo elastický layout postavený na gridu. Tato část je ze všech současných CSS frameworků nejpopulárnější, ale také nejvíce přeceňovaná.*

***Alternativní média** — některé frameworky integrují obecně platná pravidla pro vzhled tiskových sestav…*

***Typografie** — pravidla, jež přiřazují zresetovaným prvkům nový vzhled. Tato část určí jak dobře se nám bude výsledný text číst. Jakkoliv je vertikální rytmus důležitý, ne všechny frameworky na něj kladou patřičný důraz…*

***Reset výchozích nastavení prohlížečů.** První a zásadní úkon pro úspěšné vykročení při stavbě webu. Frameworky většinou využívají modifikovaný reset ERICA MEYERA.*

Typografická vrstva dále zajišťuje stejný vzhled stránek v různých prohlížečích.

4.3.2 Zavedení do website

Podobně jako javascriptový framework (kapitola 4.2) je CSS framework zpravidla zaváděn do website v hlavičce html dokumentu. K tomuto účelu se používá tag <link>.

Další možností je zavedení pomocí scriptu například

```
var link = '<link rel="stylesheet" type="text/css" href="soubor.css">';

$('head').append(link);
```

Nebo o něco lépe vypadající

```
$('<link>').attr('rel','stylesheet').attr('type','text/css').attr('href',soubor.css').appendTo('head');
```

Další velmi široce využívanou možností je vložit jen část kódu vybraného frameworku do vlastních CSS souborů, nebo přilinkovat ke stránce separátní část frameworku. Naneštěstí kvůli této metodě, kterou podporují i sami vývojáři daných řešení, není možné měřit zastoupení nasazení různých frameworků. Například Blueprint CSS framework se distribuuje buď jako jeden css soubor, nebo několik rozdělených souborů s obecnými názvy jako je „grid.css“, „typography.css“ nebo „forms.css“.

Některé frameworky(blueprint, 960grid…) nabízí také možnost generování vlastního rozložení mřížky. Tyto možnosti jsou přehledně shrnuty v (24). Takto vygenerovaný kód je však jedinečný v rámci zadaných údajů a činí jakékoliv měření, či sledování ještě obtížnějším.

4.3.3 Výhody a nevýhody použití CSS frameworku

O tom, zda použít, či nepoužít při vývoji stránek CSS frameworky se mezi webovými vývojáři a kodéry vedou dlouhé debaty. V debatách jsou často vyzdvihovány možnosti rychlého prototypování, díky čemuž lze zákazníkovi rychleji ukázat alespoň zhruba funkční produkt, naopak jsou zatracovány pro produkční prostředí. Zastánci často také uvádí argument, že každý kodér si stejně dříve, nebo později vytvoří vlastní určitý standard (dá se říci framework) minimálně pro vrstvy jako je resetovací, nebo typografická, proč tedy nepoužívat něco více rozšířeného a v rámci jednoho produktu i

standardizovaného. Jisté je, že marketingová oddělení firem většinou rychlé naprototypování ocení, protože to činí produkt předveditelným a lépe zpeněžitelným.

Některé z výhod použití CSS frameworku:

Zvýšená produktivita: Znovupoužití stejných struktur je produktivnější (25) než kódování každého projektu od začátku.

Standardizované názvosloví: Když se na více projektech používá stejný framework, vede jeho použití k určíté standardizaci názvosloví. Díky tomu se lépe udržují stávající projekty, protože kodér nemusí znovu rozmýšlet, jak je právě udržovaná aplikace postavena.

Zlepšení práce v týmu: Podobně jako v předchozím bodě, nejen mezi různými projekty, ale i mezi různými lidmi pomáhá standardizace rychlosti a efektivitě..

Poskytují více organizovaný přístup k CSS: Díky abstrakci kódu a desihgnových bloků pro různé typy použití se kód stává organizovanějším.

Neustálé zlepšování komunitou: Každý veřejně vydaný framework je používán mnoho vývojáři, kteří ho používají, testují a vylepšují a poskytují zpětnou vazbu pro další vývoj frameworku, či odstranění chyb.

Méně chyb: Když vývojář znovupoužívá vyzkoušené a otestované bloky stejného kódu, zanáší se do výsledného produktu méně chyb.

Cross browser kompatibilita a důvěryhodnost: Vývojáři obecných řešení si zpravidla dávají záležet, aby tato řešení fungovala bezchybně a zobrazovala stejné věci stejně v různých prohlížečích.

Dají se použít jako výukový nástroj: Když vývojář zkoumá, proč jsou některé věci ve frameworku udělané tak, jak jsou, často se naučí nové metody přístupu, či jiné vlastnosti, které do té doby neznal.

A některé nevýhody:

Nutnost učení: Každý nový nástroj zabere určitý čas (a v případě zaměstnávání lidí to znamená i vydané peníze). Tato nevýhoda je ovšem kompenzovaná rychlostí vývoje, když už daný nástroj vývojář ovládá a je produktivnější. Zpravidla se tedy jedná o dobrou investici.

Možnost zavlečení cizích chyb: Kvůli použití cizího kódu může často nastat zavlečení cizích chyb do vlastních projektů. Proto je dobré použít Open source řešení, kterému se věnuje komunita.

Ztráta přehledu: Dlouhodobé používání hotových frameworků může vést ke ztrátě přehledu, co se vlastně děje „uvnitř". Případně k tomu, že se začínající kodér upne na jeden framework, aniž by se naučil základy.

Zbytečný kód: Při použití hotového obecného vícevrstevného řešení, dochází k zavádění částí kódu (nebo i celých vrstev), které nejsou nijak využity, přesto je prohlížeč musí zpracovat a musí se k uživateli přenést. Tím se zhoršuje rychlost dané aplikace.

Omezení vývojáře: Když vývojář používá framework, je svázán jeho možnostmi a omezeními.

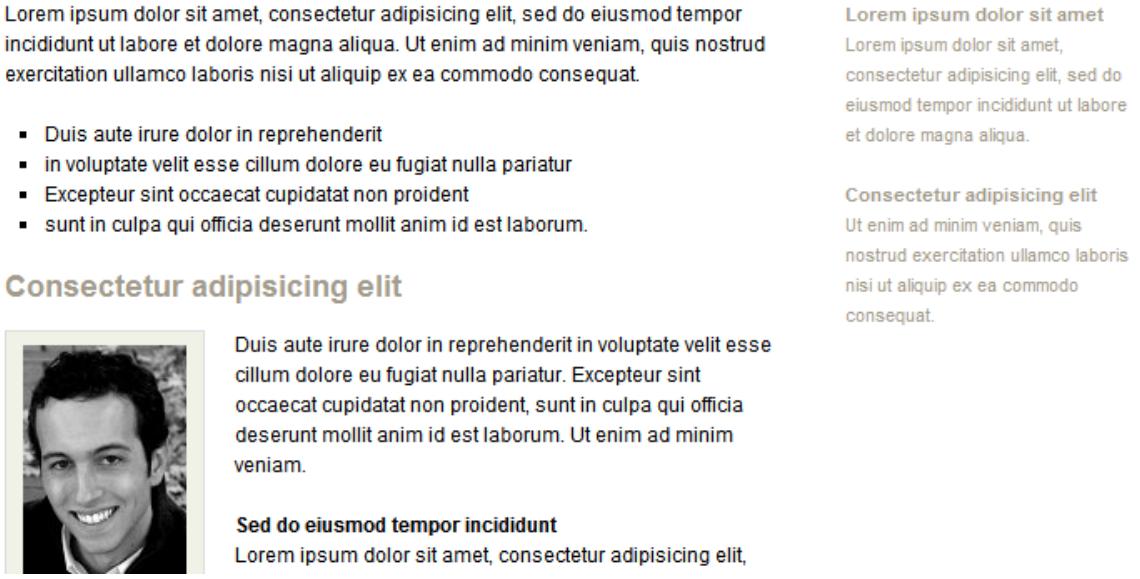
Nedosatatečná sémantika kódu: Vzhledem k abstrakci kódu frameworku pro široké použití, nemůže z pricnipu framework používat popisné (sémantické) názvy pro třídy a ID. Proto tyto názvy často zní například .column-4 místo popisnějšího „.pravy_sloupec".

4.3.4 Kritéria pro výběr CSS frameworku

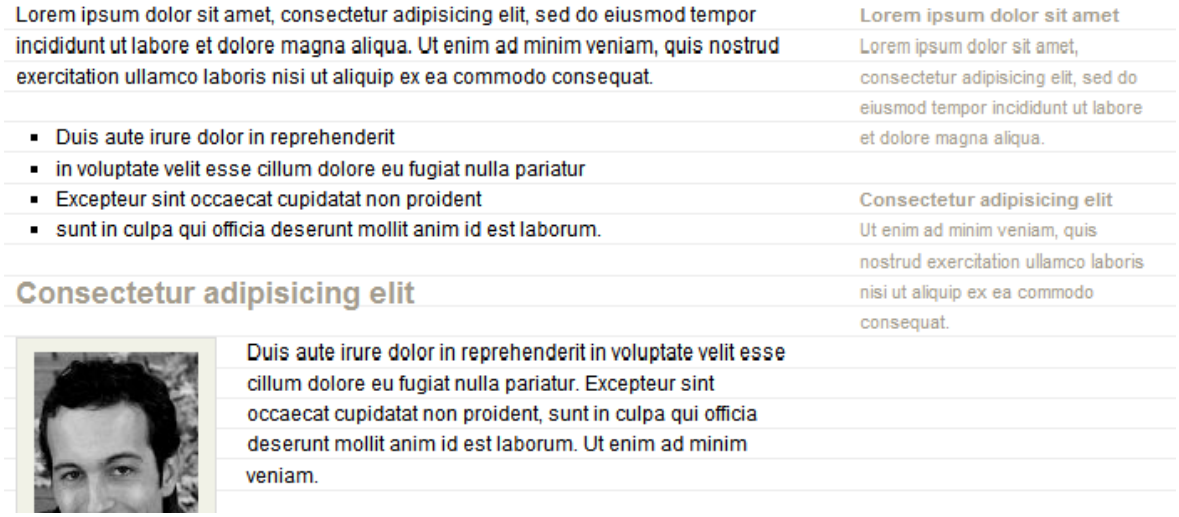
Následující kapitola popisuje kritéria pro výběr CSS frameworku. Tato kritéria jsou stanovena rešerší zejména internetových zdrojů a jsou velmi špatně definovatelná, protože se jimi autoři téměř nezabývají, designéři a kodéři využívající snadné prototypizace se často rozhodují podle vizuálního stylu, nebo osobních preferencí.

Mřížka (grid)

Při stavění formulářových aplikací je velkou výhodou mít frameworkem řešený vertikální flow dokumentu. (26) (27).Uživatelům se dobře typograficky zarovnaný text [Obr 5 a 6, zdroj (27)] lépe čte a snadněji se v něm orientují. Ze zvažovaných frameworků tuto funkčnost nabízí všechny.



Obrázek 5 Příklad správně zarovnaného textu do mřížky



Obrázek 6 Tentýž příklad se zvýrazněnou mřížkou

Hezké formuláře

Vzhledem k tomu, že modelová aplikace je formulářového typu, je potřeba vybrat takový framework, který dobře zachází s formuláři. Screenshoty formulářů stylovaných pomocí jednotlivých frameworků jsou umístěny v přílohách (Obr. 16-21) a byly pořízeny v prohlížeči Mozilla Firefox. Zdroj: (28)

Lze se přesvědčit, že element <fieldset>, který je v rozsáhlejších formulářových aplikacích potřebný, zavádí vizuálně jen frameworky Blueprint a Bluetrip (vycházející z Blueprintu).

Na obr. 21 je také vidět chyba frameworku Baseline ve vykreslování formulářového html – select s výběrem více možností je podivně odskočený.

4.3.5 O Zvažovaných frameworcích

Níže popsané frameworky byly do výběru zařazeny rešerší používaných technologií na internetových stránkách věnujících se problematice stylování webu a studiem příslušné literatury. Jak už bylo řečeno, je nemožné kvantifikovat jejich celkové využití, proto byly do výběru zařazeny ty frameworky, o kterých se nejvíce píše.

960 Grid Systém

960 Grid Systém je snahou o zpřehlednění pracovního procesu vývoje webu tím, že nabídne všeobecně přijímané rozměrové prvky založené na stránce šířky 960 pixelů. Existují dvě varianty – 12 a 16 sloupcová, které se dají použít samostatně nebo i dohromady. (29)

Baseline

Baseline je postaven s důrazem na typografickou stránku webu, čímž ulehčuje vývoj webových stránek. Baseline začíná několika soubory, které resetují výchozí chování prohlížečů a postaví základní typografické rozvržení – včetně stylů pro formuláře a nové HTML 5 elementy – a umožňuje vytvořit systém jednoduché mřížky. Baseline byl vyvinut, aby se stal nástrojem pro rychlé prototypování, ale přerostl v plně typografický framework, který využívá skutečně správně zarovnanou mřížku jako základ své funkčnosti. (30)

Blueprint

Bluerpint je CSS framework, který se snaží urychlit čas strávený vývojem. Poskytuje solidní základ, na kterém se dá stavět. Přináší snadno použitelnou mřížku, rozumnou typografii, použitelné pluginy a dokonce stylopis pro tisk. (31)

Bluetrip

Plně vybavený a pěkný CSS framework, který původně kombinoval to nejlepší z Blueprintu, Tripoli (tak dostal jméno), Hartija, 960.gs a Elements, ale nyní už si našel vlastní cestu životem.

Co Vám Bluetrip může nabídnout?

Bluetrip Vám nabízí smysluplný soubor stylů a ustálené cesty, jak postavit webovou aplikaci, díky čemuž můžete vynechat „špinavou práci" a můžete se věnovat přímo desingu. (32)

YAML

Modulární CSS framework, pro skutečně flexibilní, přístupné a interaktivní webové stránky. (33)

YUI

YUI je open-source JavaScriptový a CSS framework pro budování vysoce internaktivních webových aplikací. (34)

4.3.6 Volba řešení

Díky rozumnému nastylování formulářů byl pro modelovou aplikaci vybrán framework Blueprint. Má dobrou podporu v tištných i internetových zdrojích a obsahuje všechny vrstvy definované v kapitole 4.3.1. Zde je nutno ovšem dodat, že výběr není zdaleka tak ostrý jako v případě JavaScriptového frameworku, dá se říci, že podobnou službu vykonají všechny zmíněné CSS frameworky, u některých by ovšem bylo potřeba ještě dodefinovat vhled některých prvků (například <fieldset>).

Někteří CSS vývojáři se navíc kloní k názoru, že pro produkční prostředí není vhodné použít framework vůbec, nebo použít jen resetovací vrstvu. (35)

4.4 Vhodnost použití vybraných technologií pro modelovou aplikaci

V předchozích kapitolách byl vybrán jQuery framework pro javascriptovou část, a Blueprint framework pro stylovou část aplikace. Tato kapitola se bude zabývat vhodností jejich použití.

JavaScript

Jedním ze základních úkolů, které jsou ve větší webové aplikaci potřeba, je skrývání a zobrazování částí dokumentu.

V prostém javascriptu by skrytí elementu provedlo například takto

```
document.getElementById('element_id').style.display='none';
```

a znovuzobrazení takto:

```
document.getElementById('element_id').style.display='block';
```

Tento kód nevypadá příliš složitě, neřeší ovšem některé problémy. Co když prvek před skrytím neměl hodnotu css vlastnosti display block, ale například, inline, inline-block, nebo například i table-cell? (Tyto možné hodnoty podstatným způsobem ovlivňují zobrazení prvků na stránce). Řešením by bylo buď hodnotu vynulovat a spolehnout se na hodnotu, se kterou byl prvek definován například takto

```
document.getElementById('element_id').style.display='';
```

nebo si poli/globálním objektu držet předchozí hodnotu této vlastnosti v závislosti na ukazateli na element.

Všechny tyto problémy řeší framework jQuery při použití specializovaných zobrazovacích metod. Z těch základních to jsou například metody hide, show a toggle.

Výše popsaný příklad skrytí a znovu zobrazení prvku by pak v jQuery vypadal takto (36) (37):

```
$('#element_id').hide();
```

```
$('#element_id').show();
```

jQuery také velmi jednoduchým a efektním způsobem umožňuje toto skrývání a zobrazení animovat, čímž lze často velmi vylepšit uživatelské rozhraní.

Přidání jednoduché animace je v jQuery záležitostí prvního nepovinného parametru ať už slovně:

```
$('#element_id').hide('slow');
```

nebo pomocí počtu milisekund – jak rychle má animace proběhnout (zde za 2 s):

```
$('#element_id').hide(2000);
```

Další dva nepovinné parametry jsou takzvaný easing a callback. Easing je používán k přidání dalších efektních prvků k animaci např. různých poskakování.

Callback je potom funkce, která se vykoná po dokončení animace. Využívá se například pro odstranění již nepotřebného prvku z DOM po jeho skrytí.

Tímto způsobem je možné vytvářet velmi efektní animace, jejichž napsání v nativním javascriptu by bylo velmi časovně náročné. Příklady například (36)

CSS

Pro modelovou aplikaci bude je vhodné použít resetovací a typografickou vrstvu pro sjednocení prvků vzhledu v různých prohlížečích. Jedná se zejména o velikost a typ písma, vzhled seznamů, tabulky a zrušení výchozích hodnot některých atributů pro některá formulářová pole.

Modelová aplikace zahrnuje velké množství vyplňovaných polí uspořádaných přehledně v mřížce. Mějme například takto vypadající část formuláře (prázdné buňky představují pole pro zadání hodnot):

Základní údaje:					
Jméno		Příjmení		RČ	
Adresa trvalého pobytu					
Ulice a ČP		Obec		PSČ	

Tabulka 4 Příklad formuláře pro modelovou aplikaci

Bez CSS frameworku by se na toto uspořádání dala použít tabulka zapsaná v html takto:

```
<table>
<tr><td colspan='6'>Základní údaje</td></tr>
<tr><td>Jméno</td><td></td><td>Příjmení</td><td></td><td>RČ</td><td></td></tr>
<tr><td colspan='6'>Adresa trvalého pobytu</td></tr>
<tr><td>Ulice a ČP</td><td></td><td>Obec</td><td></td><td>PSČ</td><td></td></tr>
</table>
```

Nasadí-li se i vrstva Blueprint frameworku obsahující grid (standardně má 24 sloupců), může kód vypadat například takto:

```
<div id='container'>
<div class='span-24 last'>Základní údaje</div>
<div class='span-4'>Jméno</div>
<div class='span-4'></div>
<div class='span-4'>Příjmení</div>
<div class='span-4'></div>
<div class='span-4'>RČ</div>
<div class='span-4 last'></div>
<div class='span-24 last'>Adresa trvalého pobytu</div>
<div class='span-4'>Ulice a ČP</div>
<div class='span-4'></div>
```

```
<div class='span-4'>Obec</div>
<div class='span-4'></div>
<div class='span-4'>PSČ</div>
<div class='span-4 last'></div>
</div>
```

Kód zapsaný s pomocí frameworku je v tomto případě delší (480 znaků versus 270 znaků u tabulky), přináší ovšem několik výhod.

- Při využití elementů div,span a podobných se stránka vykresluje v prohlížeči postupně tak, jak je přenášen zdrojový kód, zatímco tabulka s vykreslením čeká, než bude přenesena celá (38).
- V případě, že se později zadavatel aplikace rozhodne, že zažádá úpravu, kód je lépe spravovatelný. Jako příklad je možno uvést rozdělení buňky „Ulice a čp“ na dvě samostatná zadávací pole.

Základní údaje:							
Jméno			Příjmení			RČ	
Adresa trvalého pobytu							
Ulice		ČP		Obec		PSČ	

Tabulka 5 Příklad formuláře pro modelovou aplikaci po vyžádané změně.

Při použití původní tabulkové koncepce se musí změnit počet buněk ze šesti na osm i v řádcích, kterých se změna nedotkla, atributy colspan a případné další možné nutné prvky rozložení (například tagy col, thead…), předchozí kód by se tedy změnil takto:

```
<table>
<tr><td colspan='8'>Základní údaje</td></tr>
<tr><td colspan='2'>Jméno</td><td> colspan='2'>
</td><td>Příjmení </td> <td></td><td>RČ </td><td></td></tr>
<tr><td colspan='8'>Adresa trvalého pobytu</td></tr>
<tr><td>Ulice</td><td></td><td>ČP</td><td></td><td>Obec</td><
td></td> <td>PSČ</td><td></td></tr>
</table>
```

Zatímco u frameworkového kódu by stačilo provést změnu pouze na řádku, kterého se změna týká a to jen u těch elementů, kterých se týká. Kód příkladu by se změnil takto:

```
<div id='container'>
<div class='span-24 last'>Základní údaje</div>
<div class='span-4'>Jméno</div>
<div class='span-4'></div>
<div class='span-4'>Příjmení</div>
<div class='span-4'></div>
<div class='span-4'>RČ</div>
<div class='span-4 last'></div>
<div class='span-24 last'>Adresa trvalého pobytu</div>
<div class='span-2'>Ulice</div>
<div class='span-2'></div>
<div class='span-2'>ČP</div>
<div class='span-2'></div>
<div class='span-4'>Obec</div>
<div class='span-4'></div>
<div class='span-4'>PSČ</div>
<div class='span-4 last'></div>
</div>
```

Kvůli udržovatelnosti aplikace je tedy vhodné použít pro modelovou aplikaci navrhovaný framework.

4.5 Slovníček

CDN operátor – CDN označuje Content distribuing network - tedy obsah distribuující síť. Tito operátoři se zaměřují na poskytování sítě kopírující obsah na různé servery umístěné po celém světě. Tím je zrychleno načítání a zároveň rozložena zátěž na serverové části velkých aplikací.

CSS – Cascading Style Sheets – je jazyk popisující vzhled a formátování dokumentů a napsaných ve značkovacím jazyce a jejich prvků.

DOM – document object model – je multiplatformní stromová jazykově nezávislá konvence pro reprezentaci objektů popsaných ve značkovacím jazyce a manipulaci s nimi.

HTTP(S) - Hypertext transfer protokol (secured) – nejpoužívanější protokol pro přenos internetových stránek k uživateli.

HTML – značkovací jazyk pro tvorbu webových stránek

JavaScript (někdy také EcmaScript) – dynamický slabě typovaný programovací jazyk zejména pro použití v rámci webových stránek.

Klient – aplikace, nebo zařízení, které se dotazuje na server (zpravidla používá porty 80 a 443) a čeká na jeho odpověď. V této práci je jím myšlen internetový prohlížeč.

Klientské technologie – technologie používané na straně klienta. Často slouží k vizualizaci, nebo úpravě zobrazovaných informací

XML – eXtended Markup Language – značkovací jazyk sloužící pro popis a přenos strukturovaných dat.

5 Výsledky a diskuse

V předchozích kapitolách bylo prokázáno, že je technologicky výhodné pro modelovou aplikaci použít javascriptový framework a alespoň některé vrstvy CSS frameworku. Kromě technologického hlediska je ovšem nutné zmínit i hledisko ekonomické.

v IT sektoru tvoří náklady na mzdy a služby odborníků zdaleka nejvyšší položku v rozpočtech podniků (39)

jQuery se dnes, v roce 2012, stává jedním z průmyslových standardů. V současné chvíli je nasazen cca na 45 % internetových stránek. Častěji se tak stává, že není potřeba nově přijaté vývojáře na tento framework proškolovat a roste i srozumitelnost a znovupoužitelnost kódu, snižuje celkový čas nutný k testování a tím se také sníží celková pracnost na projektu a mzdové náklady.

U CSS frameworku se využije zejména jejich snadná rozšiřitelnost, ovšem kvůli tomu, že na trhu neexistuje jasně dominující framework, vývojáři využijí spíše obecné principy. Ekonomický přínos zde tedy záleží spíše na politice jednotlivých firem. V práci je ovšem, na rozdíl od doporučení některých autorů vysloven předpoklad, že může být vhodné použít CSS framework i pro produkční prostředí.

Co se týče budoucího vývoje těchto technologií, je otázkou, nakolik se v budoucnu prosadí spojování jednotlivých odvětví webového vývoje – html,css, javascript, serverové části. Koncept HTML 5, který je v současnosti pomalu zaváděn do prohlížečů na stolních počíačích a ještě mnohem víc do mobilních zařízení by tomuto vývoji mohl napovídat. Je možné, ža začnou vznikat frameworky, které budou řešit všechny tyto technologie najednou a frameworky, o kterých pojednává tato práce se stanou zbytečnými.

Závěr

Hlavním cílem práce bylo analyzovat vhodnost nasazení css a javascript frameworku pro tvorbu bankovní aplikace. Bylo prokázáno, že co se týče javascriptových frameworků, dle měření významných společností zabývajících se internetem (W3C, Akamai), přesáhlo jejich nasazení již 50% webových stránek a aplikací. JavaScriptové frameworky jsou již dnes, v roce 2012, běžně nasazovány a přináší velkou úsporu vývojářského času a možnost lepší spolupráce v rámci týmu skrze definované standardy a kvalitní dokumentace. Díky těmto faktorům se použití JavaScriptového frameworku (nejčastěji jQuery) firmám ekonomicky vyplatí, protože je snazší proškolit nové lidi, kteří již zpravidla nějaký framework ve své dřívější praxi využili a nevynakládají se prostředky na vývoj nízkoúrovňových částí řešení (bylo popsáno například v části o feature detection), jejich dokumentaci a testování.

Dále byla vybrána kritéria pro vhodnost použití na modelové aplikaci a použitím metody aspiračních úrovní byl jako nejvhodnější javascriptový framework zvolen jQuery, který jako jediný vyhověl všem kritériím.

Vhodnost použití, nebo nepoužití CSS frameworku a její ekonomické přínosy oproti tomu nejsou zdaleka tak jasné. Jejich nasazení je neměřitelné současnými technologickými prostředky, protože se často používají jen části. Byly ukázány vrstvy, které CSS frameworky většinou pokrývají a také sumarizovány výhody a nevýhody jejich použití. V souvislosti s výhodami a nevýhodami bylo uvedeno, že někteří autoři vystupují výrazně proti použití CSS frameworku, zatímco jiní jej zejména v případě prototypování berou na milost.

Pro výběr vhodného CSS řešení byla stanovena dvě kritéria a na jejich základě vybrán Blueprint, ze kterého je vhodné, na základě provedených analýz, použít resetovací a typografickou vrstvu a dále mřížku pro rozmístění jednotlivých prvků.

Tato práce by mohla být dále rošířena o měření výkonnosti a efektivity zdrojových kódu jednotlivých frameworků a provedení analýzy používání jednotlivých technologií z hlediska firem a vývojářů.

Seznam použitých zdrojů

1. **JOVANOVIC, Janko.** Web Form Validation: Best Practices and Tutorials. *Smashing magazine.* [Online] 7.. 7 2009. [Citace: 15. 1 2012.] http://www.smashingmagazine.com/2009/07/07/web-form-validation-best-practices-and-tutorials/.

2. **Akamai.** State of the internet. *akamai.com.* [Online] 20. 2 2012 (automaticky generováno). [Citace: 20. 2 2012.] http://www.akamai.com/stateoftheinternet/.

3. **Network Working Group.** Hypertext Transfer Protocol -- HTTP/1.1. *W3.* [Online] 1. 9 2004. [Citace: 18. 2 2012.] http://www.w3.org/Protocols/rfc2616/rfc2616.html. RFC 2616.

4. **HICKSON, Ian.** The WebSocket API. *W3C editor's draft.* [Online] W3C, 12. 3 2012. [Citace: 20. 3 2012.] http://dev.w3.org/html5/websockets/.

5. **Google.** Přidání měřicího kódu (tradiční). *Nápověda Analytics.* [Online] Google inc. [Citace: 16. 02 2012.] http://support.google.com/googleanalytics/bin/answer.py?hl=cs&answer=55488.

6. **RAGETT, Dave, LE HORS, Arnaud a JACOBS, Ian.** HTML 4.01 Specification. *W3C reccomendation.* [Online] W3C, 24. 12 1999. [Citace: 1. 3 2012.] http://www.w3.org/TR/REC-html40/.

7. **SWEETING, Mark.** Base64 Encoded Images Embedded in HTML. *MARK SWEETING'S BLOG.* [Online] 12. 5 2005. [Citace: 24. 1 2012.] http://www.sweeting.org/mark/blog/2005/07/12/base64-encoded-images-embedded-in-html.

8. **Exceptional Performance team.** Best Practices for Speeding Up Your Web Site. *Yahoo! developer network.* [Online] Yahoo! [Citace: 16. 01 2012.] http://developer.yahoo.com/performance/rules.html#iframes.

9. jQuery.ajax(). *jQueryAPI.* [Online] [Citace: 25. 1 2012.] http://api.jquery.com/jQuery.ajax/.

10. JavaScript Libraries and functions distribution. *JavaScript Usage statistics.* [Online] Builtwith. [Citace: 4. 2 2012.] http://trends.builtwith.com/javascript.

11. **LE HÉGARET, Philippe, WHITMER, Ray a WOOD, Lauren.** Document Object Model (DOM). *W3C architecture domain.* [Online] 19. 1 2005. [Citace: 20. 1 2012.] http://www.w3.org/DOM/.

12. **The jQuery Project.** jQuery. [Online] 2010. [Citace: 14. 1 2012.] http://jquery.com/.

13. **RUTTER, Jake.** *Smashing jQuery.* 1. místo neznámé : Wiley, 2011. 978-0470977231.

14. **The jQuery Project.** Plugins/Authoring. *Documentation.* [Online] [Citace: 14. 1 2012.] http://docs.jquery.com/Plugins/Authoring.

15. **PROIETTI, Valerio.** *MooTools.* [Online] 2009. http://mootools.net/.

16. **PEHLIVANIAN, Ara, a další, a další.** *Professional JavaScript Frameworks: Prototype,YUI, ExtJS, Dojo and MooTools.* místo neznámé : Wrox, 2009. 978-0470384596.

17. "mootools" package in Ubuntu. *Ubuntu.* [Online] 2012. [Citace: 23. 2 2012.] https://launchpad.net/ubuntu/+source/mootools.

18. **Prototype Core Team.** *Prototype.* [Online] 2007. [Citace: 23. 2 2012.] http://prototypejs.org/.

19. **The Prototype Core Team.** Defining classes and inheritance. *Prototype.* [Online] 16. 10 2007. [Citace: 27. 2 2012.] http://www.prototypejs.org/learn/class-inheritance.

20. **FUCHS, Thomas.** *script.aculo.us.* [Online] [Citace: 27. 2 2012.] http://script.aculo.us/.

21. **Yahoo! Inc.** *YUI.* [Online] 2012. [Citace: 27. 2 2012.] http://yuilibrary.com/.

22. **CARRER, Vladimir.** *Carrer web log.* [Online] [Citace: 28. 2 2012.] http://www.vcarrer.com/.

23. **MICHÁLEK, Martin.** CSS frameworky pro masy, díl první. *Zdroják.cz.* [Online] 19. 8 2009. [Citace: 28. 2 2012.] http://zdrojak.root.cz/clanky/css-frameworky-pro-masy-dil-prvni/.

24. **KOZEL, Ewout.** *gridsystemgenerator.com.* [Online] 2012. [Citace: 28. 2 2012.] http://www.gridsystemgenerator.com/.

25. **BRADLEY, Steven.** Earn More By Becoming An Efficient Web Designer. *vanseodesign.* [Online] [Citace: 10. 3 2012.] http://www.vanseodesign.com/wordpress/efficient-web-design/.

26. **RUTTER, Richard.** Compose to a Vertical Rhythm. *24 Ways to impress your friends.* [Online] 12. 12 2006. [Citace: 10. 3 2012.] http://24ways.org/2006/compose-to-a-vertical-rhythm.

27. **MINER, Wilson.** Setting Type on the Web to a Baseline Grid. *A list apart.* [Online] 9. 4 2007. [Citace: 10. 3 2012.] http://www.alistapart.com/articles/settingtypeontheweb.

28. **MICHÁLEK, Martin.** *Vzhůru dolů.* [Online] [Citace: 10. 3 2012.] http://www.vzhurudolu.cz/css-frameworks/.

29. **SMITH, Nathan.** *960 Grid System.* [Online] [Citace: 12. 3 2012.] http://960.gs/.

30. **CURZI, Stéphane.** *Baseline.* [Online] 2011. [Citace: 12. 3 2012.] http://baselinecss.com/.

31. **Blueprint community.** *Blueprint.* [Online] 2011. [Citace: 10. 3 2012.] http://www.blueprintcss.org/.

32. **CRITTENDEN, Mike.** *Bluetrip CSS Framework.* [Online] [Citace: 10. 3 2012.] http://bluetrip.org/.

33. **JESSE, Dirk.** *YAML 4 CSS Framework.* [Online] 2012. [Citace: 12. 3 2012.] http://www.yaml.de.

34. **YAHOO! Inc.** *YUI.* [Online] [Citace: 12. 3 2012.] http://yuilibrary.com/.

35. **CEDERHOLM, Dan a MARCOTTE, Ethan.** *Handcrafted CSS: More Bulletproof Web Design.* místo neznámé : New Riders Press, 2009. 978-0321643384.

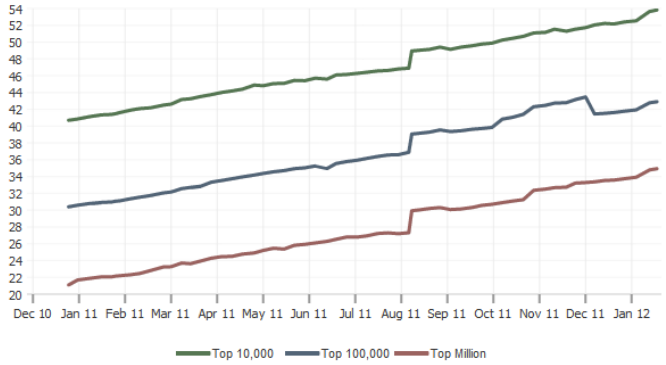
36. **The jQuery Project.** .show(). *jQuery API.* [Online] 2010. [Citace: 10. 3 2010.] http://api.jquery.com/show/.

37. —. .hide(). *jQuery API.* [Online] 2010. [Citace: 10. 3 2012.] http://api.jquery.com/hide/.

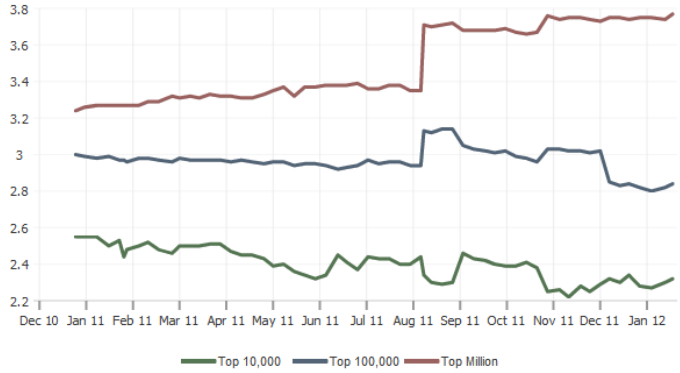
38. **PLAVEB Corporation.** How To Decrease The Loading Time Of A Website. *plaweb.* [Online] 14. 12 2010. [Citace: 13. 3 2012.] http://www.plaveb.com/blog/how-to-decrease-the-loading-time-of-a-website.

39. **BERÁNEK, Ondřej.** Podnikáte v oblasti IT? Můžete pro svou firmu získat dotace z EU. *podnikatel.cz.* [Online] 6. 8 2010. [Citace: 13. 3 2012.] http://www.podnikatel.cz/clanky/it-podnikani-dotace-eu-finance/.

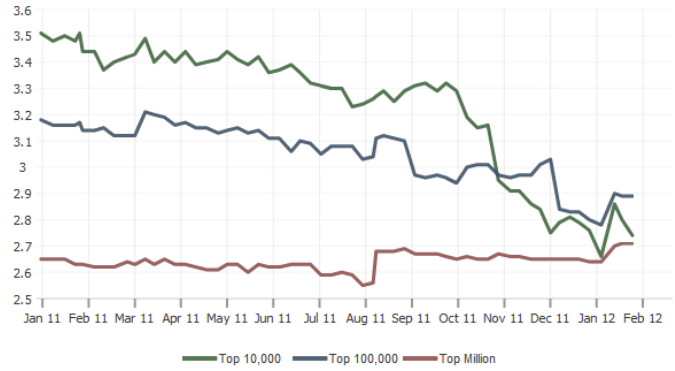
Přílohy - obrázky



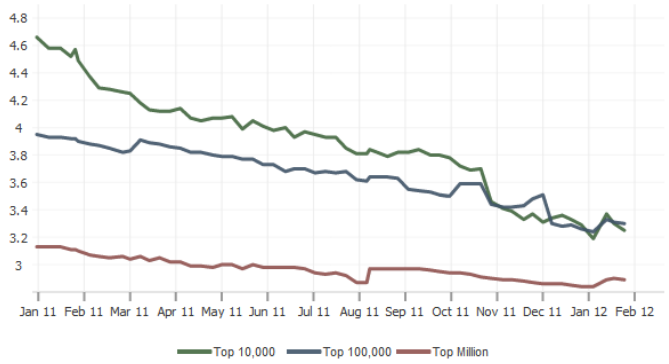
Obrázek 7 Vývoj nasazování jQuery na nejnavštěvovanější stránky



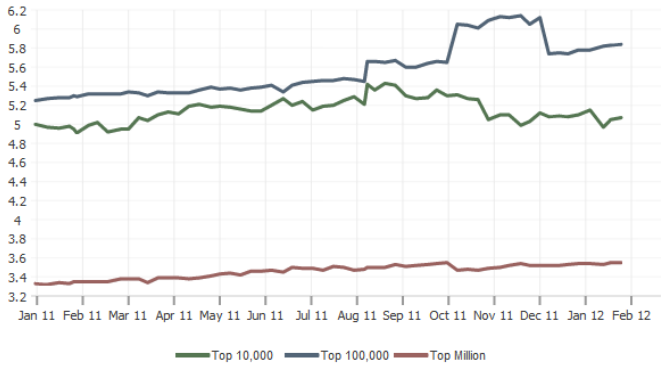
Obrázek 8 Vývoj nasazování MooTools na nejnavštěvovanější stránky



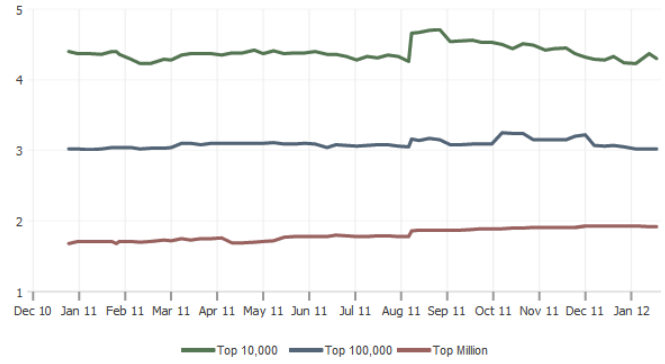
Obrázek 9 Vývoj nasazování script.aculo.us na nejnavštěvovanější stránky



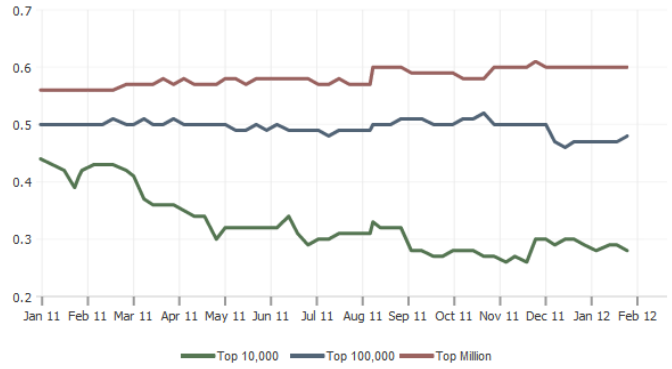
Obrázek 10 Vývoj nasazování Prototype na nejnavštěvovanější stránky



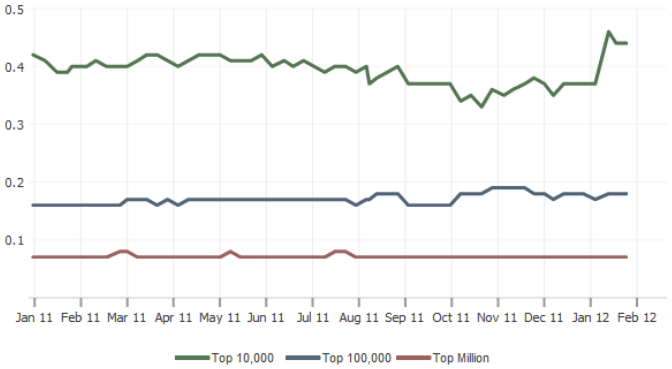
Obrázek 11 Vývoj nasazování ASP.NET Ajax na nejnavštěvovanější stránky



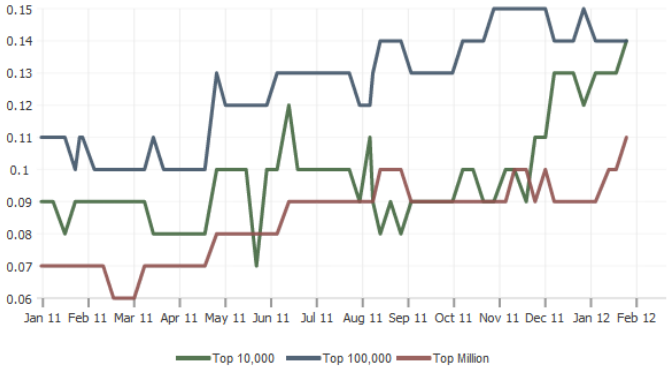
Obrázek 12 Vývoj nasazování YUI na nejnavštěvovanější stránky



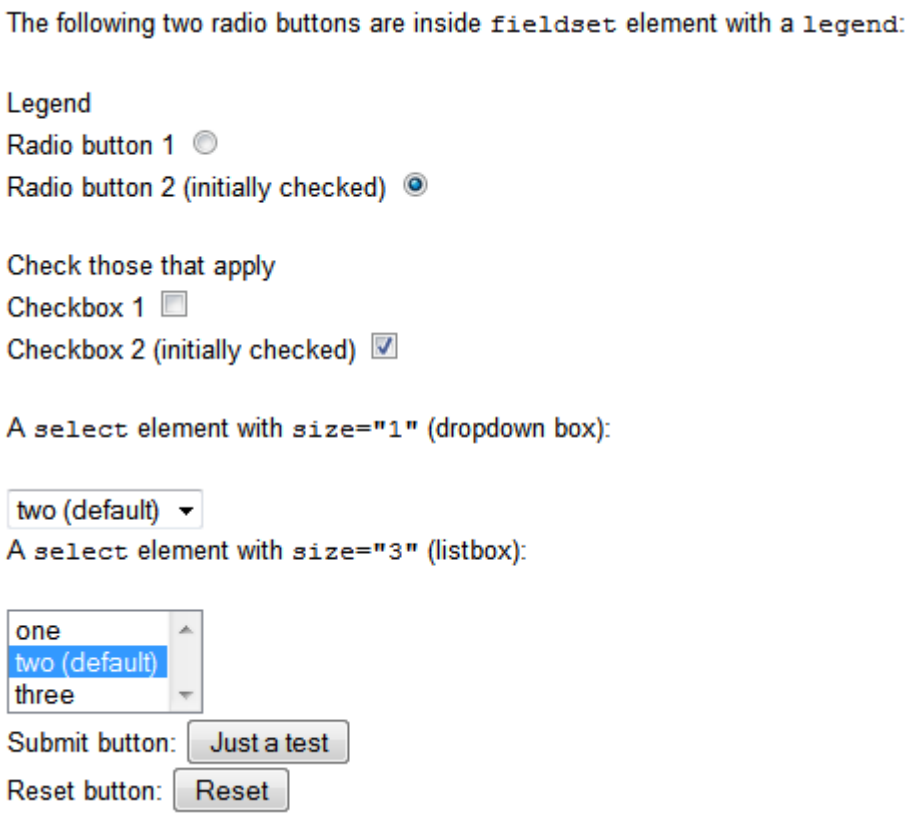
Obrázek 13 Vývoj nasazování Spry na nejnavštěvovanější stránky



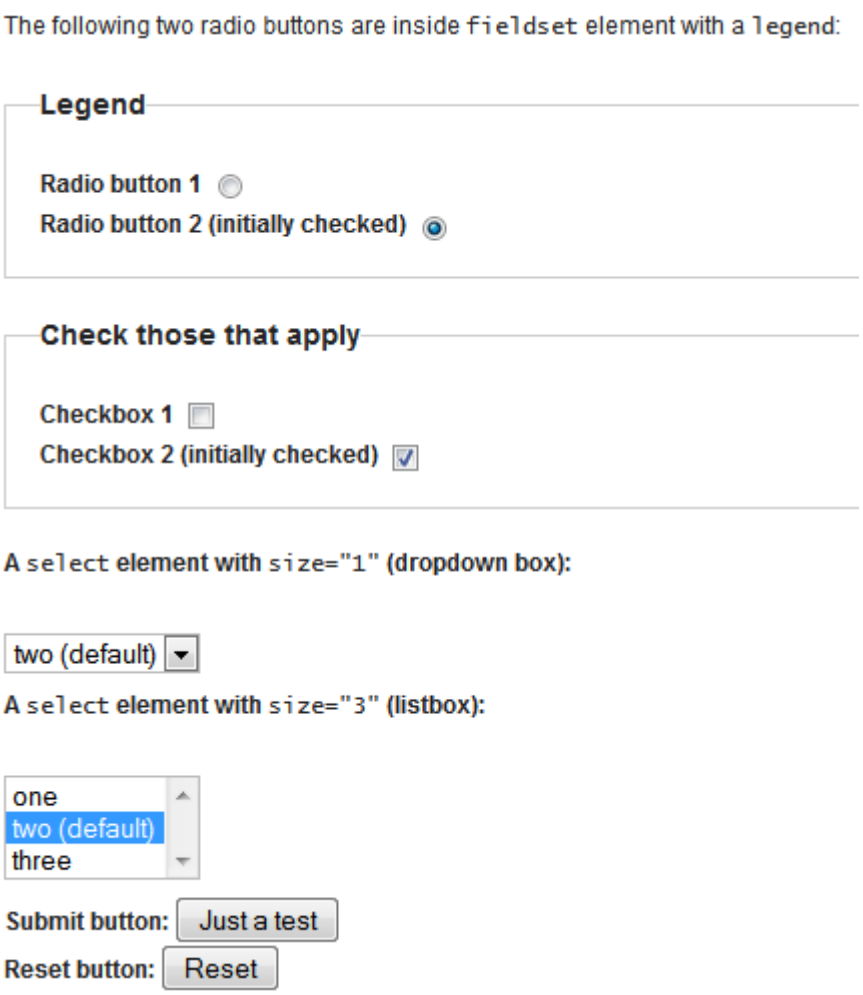
Obrázek 14 Vývoj nasazování Dojo na nejnavštěvovanější stránky



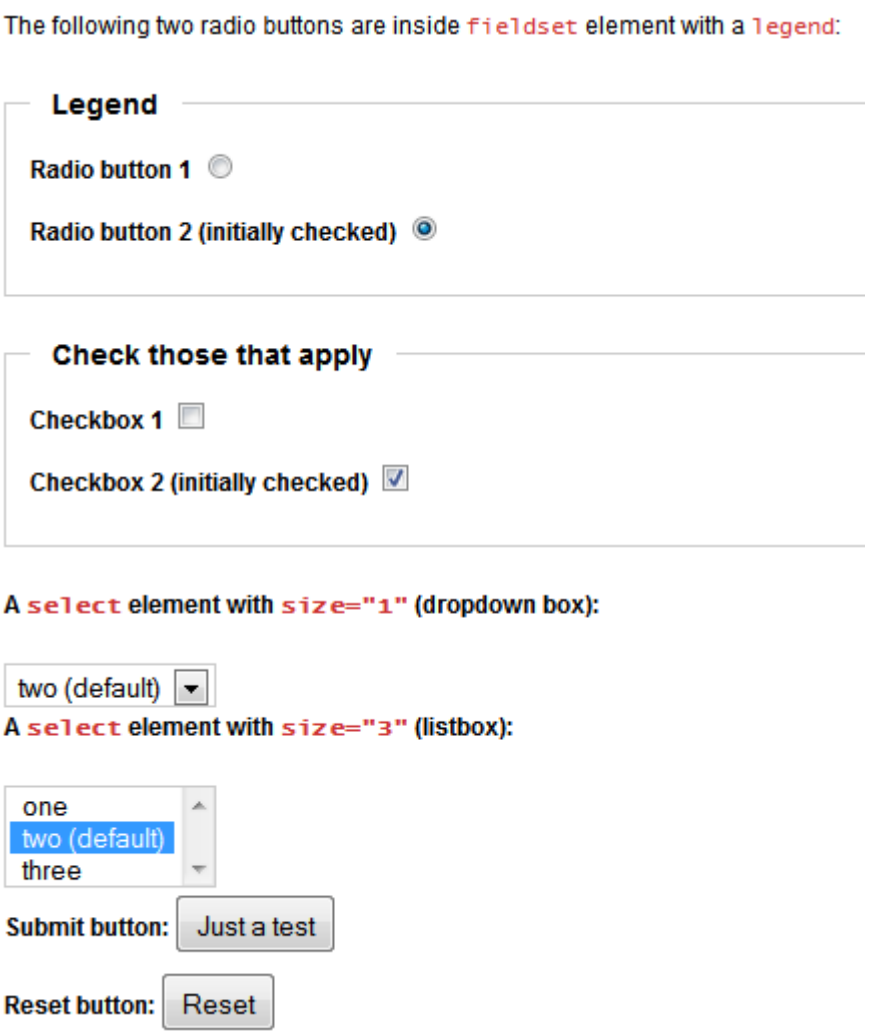
Obrázek 15 Vývoj nasazování Ext JS na nejnavštěvovanější stránky



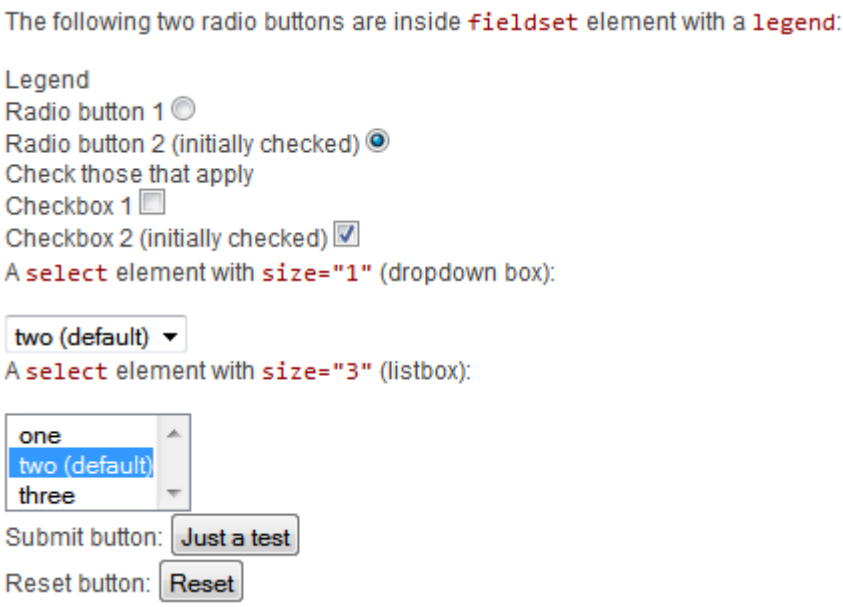
Obrázek 16 Ukázka formuláře stylovaného pomocí 960.gs



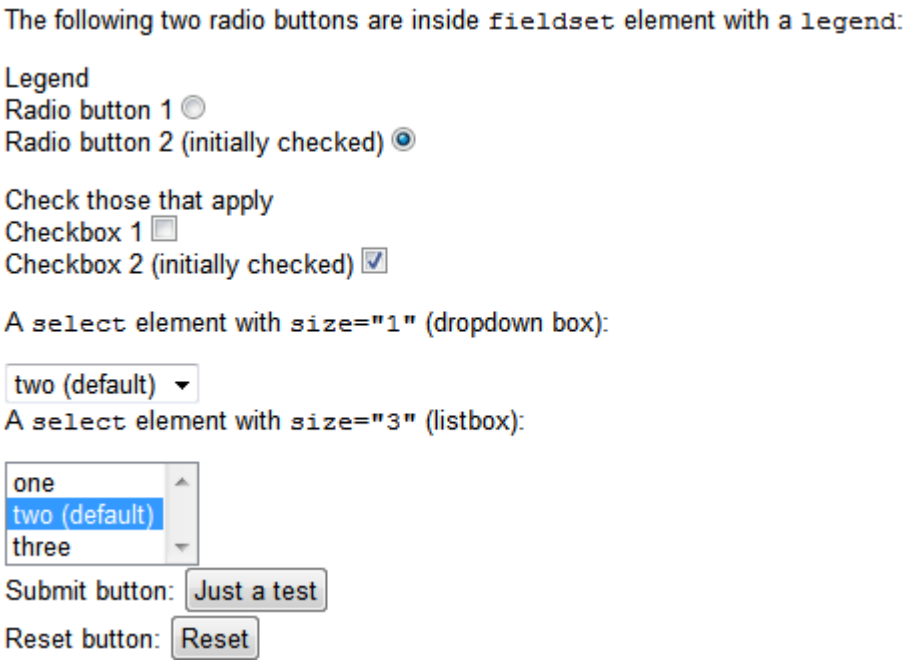
Obrázek 17 Ukázka formuláře stylovaného pomocí Blueprint



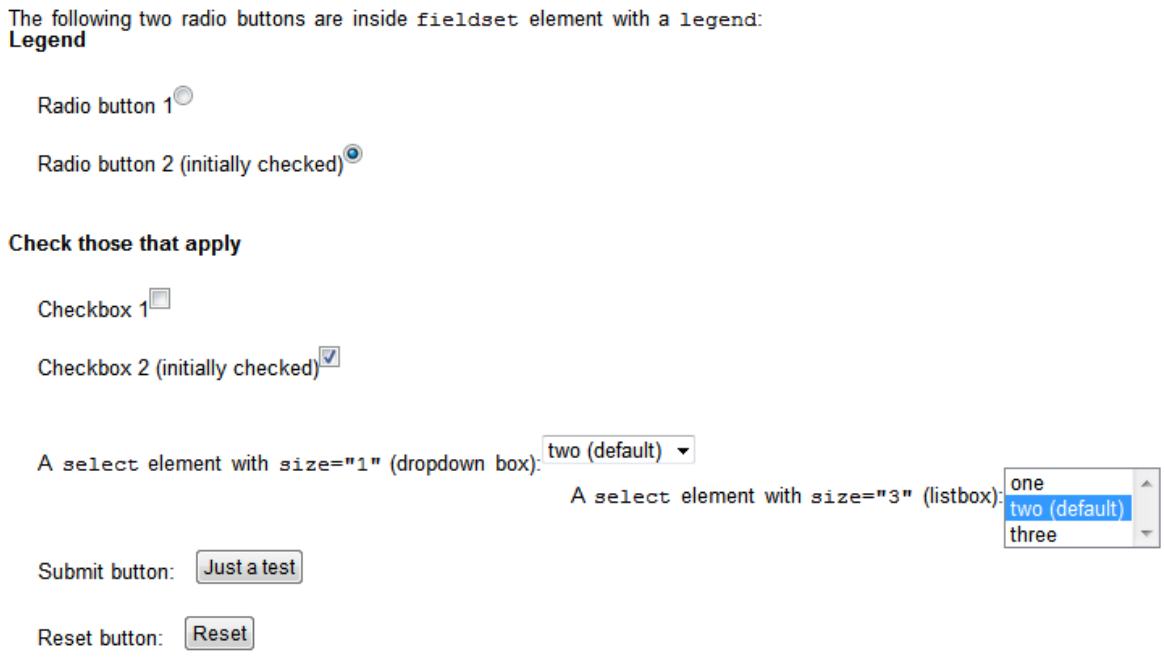
Obrázek 18 Ukázka formuláře stylovaného pomocí Bluetrip CSS



Obrázek 19 Ukázka formuláře stylovaného pomocí YAML



Obrázek 20 Ukázka formuláře stylovaného pomocí YUI



Obrázek 21 Ukázka formuláře stylovaného pomocí Baseline CSS