UNIVERZITA MATEJA BELA V BANSKEJ BYSTRICI
FAKULTA PRÍRODNÝCH VIED

VYUŽITIE TECHNOLÓGIE CSS SPRITE PRI TVORBE WEBOVÝCH APLIKÁCIÍ

DIPLOMOVÁ PRÁCA

af7466da-7324-43bb-8cbf-9912a560781f

Banská Bystrica 2014 **Bc. Roman Mátyus**

UNIVERZITA MATEJA BELA V BANSKEJ BYSTRICI
FAKULTA PRÍRODNÝCH VIED

VYUŽITIE TECHNOLÓGIE CSS SPRITE PRI TVORBE WEBOVÝCH APLIKÁCIÍ

Diplomová práca

af7466da-7324-43bb-8cbf-9912a560781f

Študijný program: 9.2.9 Aplikovaná informatika
Študijný odbor: 9.2.9 Aplikovaná informatika
Pracovisko: Katedra informatiky
Vedúci diplomovej práce: PeadDr. Ivan Brodenec, PhD.

Banská Bystrica 2014 **Roman Mátyus**

UNIVERZITA MATEJA BELA V BANSKEJ BYSTRICI
FAKULTA PRÍRODNÝCH VIED

ZADANIE ZÁVEREČNEJ PRÁCE

Meno, priezvisko a tituly študenta: Roman Mátyus, Bc.

Študijný program: Aplikovaná informatika

Študijný odbor: Aplikovaná informatika

Označenie záverečnej práce: af7466da-7324-43bb-8cbf-9912a560781f

Jazyk práce: slovenský

Meno, priezvisko a tituly vedúceho alebo školiteľa: PeadDr. Ivan Brodenec, PhD.

Názov záverečnej práce: Využitie technológie CSS SPRITE pri tvorbe webových aplikácií

Anotácia:

Školiace pracovisko: Katedra informatiky

Garant študijného programu: doc. RNDr. Bohuslav Sivák, PhD.

Dátum schválenia: .....

podpis

Čestné prehlásenie

Prehlasujem, že som zadanú diplomovú prácu vypracoval sám s pomocou vedúceho práce a použil som iba literatúru uvedenú v práci. Ďalej prehlasujem, že nemám námietky voči požičiavaniu alebo zverejňovaniu mojej záverečnej práce alebo jej časti so súhlasom katedry.

V Banskej Bystrici, dňa 25. 4. 2014 .....

Bc. Roman Mátyus

Poďakovanie

Ďakujem vedúcemu práce PeadDr. Ivanovi Brodencovi, PhD. za pomoc a inšpiráciu pri písaní diplomovej práce.

Abstrakt

Roman Mátyus: Využitie technológie CSS Sprite pritvorbe webových aplikácií [Diplomová práca.] Univerzita Mateja Bela v Banskej Bystrici. Fakulta prírodných vied. Katedra informatiky. Banská Bystrica 2014. 55 strán.

Kľúčové slová: css sprite, css, sprites, php

Táto diplomová práca vytvára ucelený prehľad výhod a nevýhod technológie CSS Sprite v kontexte webových aplikácií. Predstavuje spôsob fungovania samotnej technológie a prínos jej použitia. Na základe jej manuálneho použitia a analýzy dostupných generátorov bol navrhnutý nástroj automatizujúci jej aplikovanie. Výsledkom práce je knižnica MiniSprite v programovacom jazyku PHP.

Abstract

Roman Mátyus:

[Bachelor work.] Matej Bel University in Banská Bystrica, Faculty of Natural Sciences, Department of Informatics. Banská Bystrica 2014. 55 pages.

Keywords: css sprite, css, sprites, php

Obsah

Zoznam ilustrácií

Zoznam tabuliek

Zoznam skratiek a značiek

CSS – Cascading Style Sheets

CSS Sprite – technológia umožňujúca zobrazovať zodpovedajúce fragmenty pozadia zložených obrázkov zmenou pozície pozadia

base64 – formát dát zobrazujúci binárne dáta tlačiteľnými ASCII znakmi

byt/B – 8 bitov

kB – 1 024 bytov

lazy loading – princíp spracúvania/načítavania údajov až v momente keď sú potrebné

fluent interface – podpora čitateľnejšieho a jednoduchšieho zápisu použitia metód nad objektom v OOP

getter – funkcia vracajúca hodnotu premennej ako návratovú hodnotu

GIF – Graphics Interchange Format

HSL – Hue Saturation Lightness – farebný priestor

HSLA – Hue Saturation Lightness Alpha – farebný priestor

HTML – Hypertext Markup Language

HTTP – Hypertext Transfer Protocol, protokol pre prenos hypertexu

IE – Internet Explorer

JPEG/JPG - Joint Photographic Experts Group

konštruktor – metóda volaná pri vytváraní inštancie objektu triedy

LESS – Leaner CSS

metóda – funkcia patriaca triede, resp. objektu

OOP – Object-oriented Programming – objektovo orientované programovanie

OptiPNG – knižnica optimalizujúca PNG obrázok bez straty kvality

PHP – PHP: Hypertext Preprocessor

PNG – Portable Network Graphics

px – pixel – obrazový bod

RGB – Red Green Blue – farebný priestor

RGBA – Red Green Blue Alpha – farebný priestor

setter – funkcia nastavujúca hodnotu premennej podľa prijatého parametra

sprite – obrázok zložený z minimálne dvoch samostatne použiteľných obrázkov

TTD – Test Driven Development – testom riadený vývoj

URL – Uniform Resource Locator, adresa umiestnenia na sieti internet

WebP – obrázkový formát

ZIP – súborový formát pre bezstratovú komprimáciu dát do archívu

Úvod

Súčasná doba masívneho používania internetu na prácu, zábavu komunikáciu a iné oblasti života prináša stále sa zvyšujúce nároky na jeho rýchlosť a dostupnosť. Vďaka obrovskej penetrácií trhu smartfónmy je čoraz častejšie využívaná na prístup k internetu sieť mobilných operátorov. Mobilné siete žiaľ technicky nikdy nebudú rovnocenným partnerom pre pozemné vedenia, napríklad optické siete.

Dôsledkom technických možností je stály rozmach online obsahu a komercie. Vzhľadom na prakticky neobmedzenú konkurenciu je každá konkurenčná výhoda vydavateľa alebo obchodníka nepostrádateľná. Tieto skutočnosti sú podnetom pre vysoký tlak na programové vybavenie sprostredkovateľov obsahu. Keďže je skutočnosťou že dlho sa načítavajúce stránky používatelia opúšťajú pred konverziou, je zrýchlenie prístupu k obsahu jednou z hlavných priorít. Univerzálne riešenie tohto problému neexistuje, sú možné iba čiastkové vylepšenia ktoré spolu dosahujú výsledný efekt.

Jednou z týchto technológií je CSS Sprite. Je určená pre zrýchlenie načítavania obrázkov vo webovej aplikácií. Podstatou prínosu je minimalizovanie počtu HTTP požiadaviek na server a zníženie objemu prenášaných obrázkov. Toto je zabezpečené zlúčením čo najväčšieho počtu obrázkov do jediného zloženého obrázku.

Prvá kapitola opisuje detaily technológie CSS Sprite. Obsahom je vysvetlenie princípu, súradnicového systému a vplyvu typov súborov na výsledok. Zároveň sú v nej detailne vymenované a analyzované výhody a nevýhody použitia. K nevýhodám a možným problémom je navrhnuté aj technicky realizovateľné riešenie. Stručne je popísané aj konkurenčné riešenie z rovnakej oblasti.

V druhej kapitole je vysvetlený výber referenčného dizajnu ktorý je neskôr spracovaný v praktickej časti diplomovej práce. Kapitola obsahuje v tabuľkovej forme analýzu podkladov a ich vhodnosť na aplikáciu technológie.

Obsahom tretej kapitoly je praktická ukážka manuálnej aplikácie techniky na referenčný dizajn z predošlej kapitoly. Na reálnom príklade je ukázaný postup spracovania z čoho zároveň vyplýva aj časová a manuálna náročnosť. Prínos výsledku

práce je následne ohodnotený podľa reálnych úspor prenosu aplikácie k používateľovi.

V štvrtej kapitole je predstavených niekoľko online nástrojov na urýchlenie aplikovania technológie do projektu. Nástroje sú rôznych typov a predstavujú škálu prístupov k riešeniu problému. Predstavené sú ich možnosti použitia na vstupoch z referenčného dizajnu, ich výhody aj nevýhody.

Piata kapitola obsahuje detailný popis implementácie výsledného nástroja – knižnice MiniSprite. Čitateľ je uvedený do prostredia pre ktoré je knižnica vytvorená a sú mu objasnené ciele ktoré má výsledný produkt dosiahnuť oproti nástrojom v štvrtej kapitole. Následne je popísaná vnútorná architektúra knižnice, jej vnútorné väzby a jednotlivé komponenty.

1 Popis technológie CSS Sprite

Pri používaní tejto techniky sú všetky obrázky použité v grafike, ak spĺňajú kritériá, zlúčené tak, aby z nich vznikol jeden obrázok obsahujúci obsah všetkých čiastkových obrázkov.

Následne je pozmenený obsah CSS súbora. Je potrebné identifikovať súradnice a rozmery jednotlivých čiastkových obrázkov. Tieto súradnice sa najčastejšie používajú na vytvorenie nových tried v CSS a sú zapísané v atribútoch `background-position`. Týmito triedami sú následne označené jednotlivé HTML elementy ktorých pozadie má obsahovať daný čiastkový obrázok. Informácie o súradniciach je možné použiť aj priamo na úpravu pôvodného CSS súboru. Takáto úprava je pracnejšia, ale nie je nutné modifikovať pôvodný HTML kód.

1.1 Výstupný súbor PNG

Najčastejším výstupným formátom pri tejto technike je PNG.

Tento súborový formát v sebe zlučuje viacero výhod:

- relatívne malá výsledná veľkosť,
- bezstratová kompresia,
- priehľadnosť pozadia.

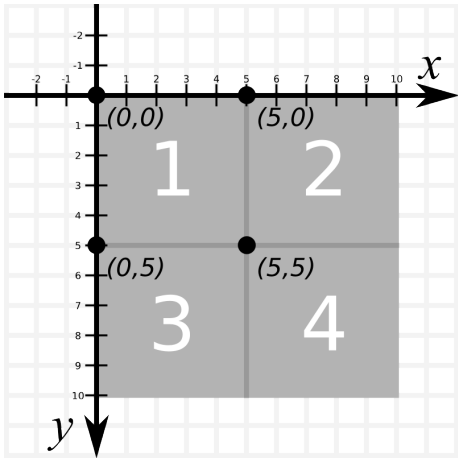
Pri jeho použití treba dbať aj na jeho dve obmedzenia:

- nepodporuje animácie ako GIF,
- Internet Explorer 6 (a staršie) nepodporujú priehľadnosť.

Použitie PNG nemusí byť vyslovene pravidlom, pretože vhodne vytvorený GIF súbor, resp. JPG súbor s kompresiou môžu dosahovať lepšie výsledky. Všetko závisí od vstupných súborov. Ak by bolo na vstupe napríklad viacero väčších obrázkov vo vysokej farebnej hĺbke, tak by malo zmysel uvažovať o JPG výstupe. JPG bude úplne diskvalifikovaný, ak bude niektorý obrázok na vstupe obsahovať priesvitnú časť.

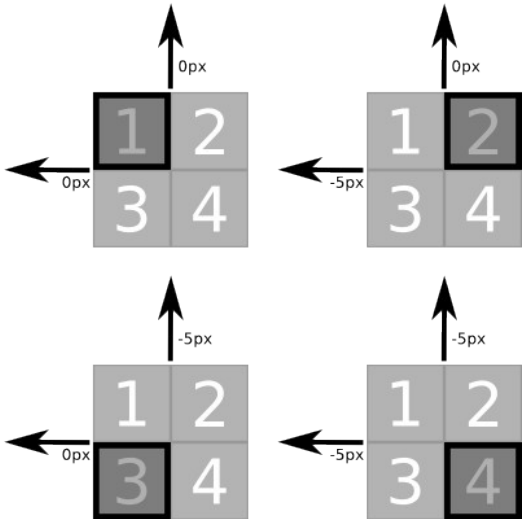
1.2 Súradnicový systém

Pri určovaní pozície zobrazovaného výrezu je použitý dvojrozmerný súradnicový systém s osou *x* a *y*. Koordináty *[0, 0]* označujú ľavý horný roh. Vertikálna os *y* nenadobúda kladné hodnoty smerom hore, ako je to štandardne, ale dolu.



Obrázok 1: Súradnicový systém pozicovania pozadia v CSS

Tiež je dôležité uvedomiť si, že sa nemení pozícia zobrazeného výrezu, ale vo výreze sa posúva pozadie.



Obrázok 2: Ukážka vplyvu súradníc na posun pozadia

Z uvedeného vyplýva, že prakticky každý výrez spritu bude mať počiatočné koordináty $x <= 0$ a $y <= 0$.

Pravidlu sa vymykajú prípady cieleného zobrazenia obrázka iba v časti objektu. Keďže pre tieto extrémne prípady nie je reálne opodstatnenie, budú v práci ignorované.

1.3 Výhody

1.3.1 Úspora počtu HTTP požiadaviek

Namiesto množstva samostatných požiadaviek server vráti jediný obrázok s jednou hlavičkou. Dôsledkom je zníženie počtu volaní a tým pádom aj záťaže servera.

Pri požiadavkách na server sa prenáša relatívne mnoho informácií[1]. Tieto informácie sú teda v bežnom prípade prenášané pre každý obrázok samostatne a teda dochádza k mnohonásobnej duplicite prenosu.

Keďže je veľkosť HTTP hlavičiek veľmi rôznorodá, je vhodné spoľahnúť sa na štatistické údaje. Priemerná veľkosť hlavičky pri HTTP prenose je aktuálne 700 až 800 bytov[2]. Pre účely porovnávania bude braná do úvahy spodná hranica odhadu - 700 bytov.

1.3.2 Úspora prenášaných dát

Každý prenášaný obrázok obsahuje okrem obsahovej časti aj svoju hlavičku. V prípade spojenia obrázkov do jedného, je ušetrený prenos hlavičiek pre každý súbor zvlášť.

Žiaľ úspora nemusí byť vždy pravidlom. Často sa stáva, že obrázky nie je možné vo výslednom obrázku optimálne rozmiestniť a výsledný obrázok má väčšiu plochu (px^2) ako je súčet plôch samostatných obrázkov. Vďaka kompresií obrázka to nemusí nutne znamenať priamo úmerné zväčšenie dátového súboru obrázka, dokonca niekedy je dôsledok opačný.

1 Hlavičky obsahujúce informácie o prehliadači, cookies a iné.

2 http://dev.chromium.org/spdy/spdy-whitepaper, online, 27.3.2014

Sprite je možné po vygenerovaní optimalizovať a tým výrazne znížiť jeho veľkosť.

1.3.3 Okamžitá dostupnosť obrázka

Webový grafici často vyžadujú zmenu grafiky po interakcií kurzora[3] s nejakým elementom na stránke. Toto je zväčša prakticky vyriešené zmenou obrázka pozadia na úrovni CSS. V tomto prípade nastáva problém s oneskoreným načítaním pozadia. Prehliadač zaregistruje potrebu vykreslenia obrázka, ale ten ešte nie je k dispozícií. Preto oň server požiada a stiahne ho až dodatočne. Táto operácia môže bežne trvať približne pol sekundy. Keďže toto prebliknutie kazí používateľský zážitok z používania aplikácie, dizajnéri niekedy vytvárajú čiastkové sprity zložené z pôvodného pozadia a pozadia po interakcií manuálne. Toto je žiaľ pracné a nesystémové riešenie.

1.4 Nevýhody

1.4.1 Pracnosť

Pri manuálnom vytvorení spritu je potrebné spojiť všetky vyhovujúce obrázky do jedného, vyrátať pozície jednotlivých obrázkov a vypočítané pozície zapísať do CSS súboru. Najpracnejším spôsobom je spracovávať požadované súbory v grafickom editore a manuálne dopisovať potrebné pozície do CSS súboru.

Možným zjednodušením je importovanie zdrojových súborov do nástroja generujúceho sprite obrázok použitím vhodného algoritmu. Takto získaný súbor je potrebné manuálne spracovať – aplikovať pozície zdrojových obrázkov do požadovaného CSS súboru.

Niektoré generátory poskytujú ako výstup automaticky triedy CSS štýlov s pozíciami pôvodných obrázkov na základe vygenerovaného obrázka. Tieto triedy sa dajú následne jednoducho aplikovať priamo v HTML prípadne sa dajú použiť CSS preprocesorom SASS, LESS alebo inými.

3 Umiestnenie kurzora nad element, kliknutie naň a iné.

1.4.2 Náročnosť generovania

V prípade použitia generátora za behu aplikácie by mohlo vzniknúť oneskorenie z dôvodu výpočtovej náročnosti generovania obrázka a jeho následné dosadenie do zdrojového súboru. Keďže žiaľ funkčné nástroje generujúce CSS Sprite za behu aplikácie aktuálne neexistujú, nie je to akútny problém. Problém je taktiež jednoducho riešiteľný kešovaním.

1.4.3 Veľkosť obrázka

V ideálnom prípade by sprite obrázok obsahoval množinu obrázkov zhodných rozmerov alebo rozmerov a tvarov ktoré sa jednoducho bez strát usporiadavajú. Výsledný obrázok by teda mal plochu v px^2 zhodnú so súčtom plôch jednotlivých obrázkov. V reálnom nasadení sa toto nestáva a teda pri tvorbe obrázka dochádza k vzniku prázdnych miest. Tento efekt je síce nežiadúci, ale nie kritický, pretože prázdne miesta zaberajú relatívne málo bytov.

Dátová veľkosť výsledného obrázka je často dokonca menšia ako súčet veľkostí vstupných obrázkov. Ak sú vstupné obrázky vhodných rozmerov nevzniká nevyplnený priestor a ušetrí sa prenos n-1 hlavičiek súborov. Úspora vzniká aj vyššou mierou kompresie väčšieho obrázka oproti kompresii malých vstupných obrázkov.

Problém prázdneho miesta v sprite je čiastočne riešiteľný:

- použitím vhodného algoritmu rozvrhnutia elementov,
- výberom najvhodnejšieho výstupného súboru prípadne
- použitia kompresie pri ktorej nedochádza k evidentnému znehodnoteniu kvality výstupu.

Tieto riešenia je možné neobmedzene kombinovať.

Vhodné podmienky výstupu je možné pri niektorých aktuálne použiteľných generátoroch manuálne nastaviť. Jednoduchým riešením je aj zmenšiť farebnú hĺbku výstupu.

Zvoliť vhodný algoritmus volieb výstupného súboru nemusí byť vždy ľahké. Hlavne v prípade ak sú na vstupe výborne optimalizované obrázky, môže byť výstup

zo štandardných generátorov bez extra nastavenia, prípadne ďalšej úpravy v externom editore, výrazne dátovo väčší ako súčet veľkosti dát obrázkov na vstupe. V takomto prípade treba previesť merania a zistiť či sa oplatí takto pregenerovaný súbor nasadiť.

Vždy to záleží od konkrétneho nasadenia, je však veľmi pravdepodobné, že sa aplikácia techniky napriek väčšiemu objemu oplatí vďaka úspore HTTP požiadaviek.

1.4.4 Zväčšenie obsahu CSS

Použitím CSS Sprite je nutné preniesť dodatočné informácie o pozícií pozadí k používateľom a teda zväčšiť množstvo prenášaných dát. Generátory zväčša vytvárajú vlastné sekcie do CSS súborov ktoré sa majú aplikovať priamo v HTML kóde. Dopad tejto nevýhody je možné minimalizovať napríklad tak, že namiesto dopísania nových CSS štýlov sú dodané štýly pozmenené a tak následne distribuované. Toto sa v praxi používa v závislosti od schopností generátora prípadne kódera, ktorý spracúva sprite obrázok manuálne. Väčšina generátorov touto funkcionalitou nedisponuje vzhľadom na relatívne zložitú implementáciu, keďže je mnoho spôsobov ako môže dôjsť ku sporom v definícií štýlov a následnému znehodnoteniu dizajnu.

1.4.5 Úprava HTML

Veľkou nevýhodou je pri použití štandardných generátorov nutnosť meniť CSS triedy súborov priamo v HTML kóde webu. Tieto úpravy môžu byť niekedy pracné a neprehľadné. Podstatne vhodnejšie by bolo iba upraviť kód v triedach metódou spomenutou v predošlom odstavci. Tým by bola zachovaná spätná kompatibilita.

1.4.6 Obmedzenia CSS

Samotný návrh štandardu CSS neobsahuje dokonalú podporu pre CSS Sprite. Táto technika je preto použiteľná iba na určitú množinu obrázkov, nie na všetky.

1.4.6.1 Vzorka

Štandardne nie je možné túto techniku využiť na obrázky ktoré sú vzorkou - opakovane vypĺňajú pozadie v nejakom smere.

V prípade ak sa to oplatí, je možné toto obmedzenie čiastočne eliminovať a vygenerovať viac typov obrázkov pre každý typ opakovania. Jeden *sprite* pre obrázky ktoré sa neopakujú a ďalšie zvlášť pre horizontálne a vertikálne opakované pozadia. Dôsledkom môžu byť až tri výsledné súbory. Ak je na stránke definovaných väčšie množstvo horizontálne/vertikálne sa opakujúcich pozadí môžu sa tým získať výrazné úspory.

Je možné kombinovať niektorý z orientovaných spritov so spritom bez orientácie. Napríklad v prípade ak chceme kombinovať neorientovaný a horizontálne sa opakujúci sprit do jedného, je potrebné horizontálne sa opakujúcu časť rozšíriť na plnú šírku pôvodného spritu a vložiť pod alebo nad pôvodný obrazec. V tomto prípade je vhodné umiestňovať čiastkové obrázky pod seba, aby bola šírka obrazca čím menšia a o to menší bol aj rozmer nútene rozšíreného horizontálne sa opakujúcej časti. Obdobne to funguje aj pre vertikálne sa opakujúce pozadia. Žiaľ v súčasnosti nie je podľa špecifikácie CSS možné zlúčiť a následne použiť horizontálne aj vertikálne sa opakujúce pozadie v rámci jedného obrazca.

Z technológie vyplýva, že v prípade horizontálneho opakovania by musel byť každý obrázok najprv upravený na šírku toho najširšieho. Ďalším riešením je zúženie širokých opakujúcich sa pozadí na minimálnu možnú šírku a až následné rozšírene na šírku najširšieho. Vo väčšine prípadov ide iba o jednoduché prechody a je ich teda možné zúžiť na 1 px.

Obdobne to platí pre vertikálne opakované obrázky. Obrázky ktoré sa podľa definície majú opakovať v horizontálnom aj vertikálnom smere nemôžu byť v CSS Sprite obrázkoch vôbec použité.

1.4.6.2 Pozicovanie pozadia

Problém nastáva aj v prípade, ak je pozadie elementu pozicované relatívne[4]. Výnimkou je kombinácia relatívnej pozície `left top`, pretože je ekvivalentom súradníc *[0, 0]*. Pozície musia byť buď nulové, určené v pixeloch alebo kľúčovými slovami `left top`.

4 Percentuálne alebo top, bottom, left, right a center.

1.4.7 Formát vstupných súborov

Z dôvodu chýbajúcej podpory animácie pri PNG súboroch[5], nemôže byť štandardne do spritu zaradený žiadny animovaný GIF. Toto obmedzenie sa dá teoreticky obísť vďaka simulácií animácie použitím techník CSS3 ale je to veľmi zložité.

1.4.8 Prenášanie nepotrebných dát

V prípade použitia spritov je prenášaný komplexný obrazec obsahujúci všetky čiastkové obrázky. Z toho dôvodu môžu nastať prípady, keď sa používateľovi odošlú aj obrazové body, ktoré nie sú pre aktuálne zobrazenie potrebné. Dôsledkom je o niečo menšia efektivita použitia technológie.

Problém je čiastočne minimalizovaný menšou výslednou veľkosťou spritu ako je súčet veľkostí vložených obrázkov. Preto nie sú dôsledky kritické.

Pre zmenšenie dopadu tohoto problému je potrebné udržovať v CSS súboroch poriadok. Obsahom majú byť iba definície, ktoré sú pre daný dizajn potrebné. Ak je táto podmienka splnená a obrazec sa na strane klienta uchováva v dočasnej pamäti, prakticky je problém eliminovaný. Aj keď sa pri prvom spojení prenesú nejaké byty s nepotrebnými fragmentami obrázkov, tie budú neskôr použité a vo výsledku ušetria väčšie množstvo komunikácie so serverom.

Jednotlivé sprity je vhodné vytvárať pre každý CSS súbor zvlášť a nie až po zlúčení odosielaných CSS súborov. Ak bude množstvo CSS súborov veľmi variabilné, boli by generované sprity pre rôzne pohľady rôzne. Toto by spôsobovalo problémy s efektným využívaním dočasnej pamäte.

1.5 Konkurenčné riešenie Data URL

Pre úplnosť je vhodné spomenúť konkurenčnú techniku urýchlenia načítania webu pomocou Data URL[6]. Pri tejto technike je v CSS odkaz na súbor (obrázok) nahradený za hlavičku a textovú reprezentáciu obrázka kódovaním base64. Hlavnou nevýhodou je zväčšenie prenášaného množstva dát o 33% oproti originálu. Preto býva

5 Formát PNG je štandardne používaný ako výstupný formát sprite obrázka.

6 http://tools.ietf.org/html/rfc2397

technika používaná iba na veľmi malé obrázky, kde je absolútny nárast veľkosti iba v bytoch.

Zaujímavou možnosťou sa javí kombinácia CSS Sprite a Data URL tak, že sa do Data URL prevedú obrázky, ktoré nemohli byť z nejakého dôvodu zaradené do spritu.

Možným obmedzením je problematická podpora pseudoprotokolu data v Internet Exploreri 8 a starších. IE 8 podporuje Data URL s maximálnou veľkosťou 32 kB a staršie nepodporujú túto funkciu vôbec.

	Štandardnévolanie	**Data URL**	**CSSSprite**
Počet HTTP komunikácií pre zobrazenie n obrázkov	`n`	0	1 až `n`
Veľkosť dát oproti originálu	100%	133 %	± 100%

Tabuľka 1: Porovnanie štandardného volania obrázkov, Data URI a CSS Sprite

Štruktúra takéhoto reťazca je podľa špecifikácie nasledovná:

```
data:[<mediatype>][;base64],<data>
```

1.5.1 Príklad

Namiesto

```
background: white url('dot.png')
```

je cesta k súboru v CSS nahradená za textovú reprezentáciu obrázka a štýl vyzerá nasledovne:

```
background: white url('
AAANSUhEUgAAAAUAAAAFCAYAAACNbyblAAAAHElEQVQI12P4//8/w38GI
AXDIBKE0DHxgljNBAAO9TXL0Y4OHwAAAABJRU5ErkJggg==')
```

2 Referenčný dizajn

Ako referenčný dizajn pre testovanie postupov, rôzne merania a analýzu som pri práci zvolil open source šablónu pre CMS Joomla s názvom Eclipse[7]. Šablóna bola vybratá náhodne. Podmienkou výberu bol moderný dizajn a otvorený zdrojový kód pre licenčne bezproblémové použitie v práci. Šablónu je možné stiahnuť z webstránky joomlatemplates.me.

Obrázok 3: Šablóna Eclipse

2.1 Analýza podkladov

Šablóna obsahuje spolu 6 CSS súborov. Súbory *bootstrap.min.css*, *font-awesome.min.css*, *font-awesome-ie7.min.css* neobsahujú žiadne obrázky, takže nie sú relevantné pre spracovanie. Súbor *flexslider.css* obsahuje iba jediný obrázok, takže sprite taktiež nemá zmysel.

V súbore *styles.css* sa nachádza definícia vzhľadu samotnej šablóny.

Obsahuje 17 obrázkov s nasledujúcimi definíciami:

7 http://joomlathemes.co/demo3x/eclipse

Názov	Použiteľný pre sprite	Veľkosť	Rozmer	Plocha
bg.png	horizontálne	19,0 kB	50 x 600 px	30 000 px^2
header.png	nie	18,8 kB	980 x 600 px	588 000 px^2
facebook.png	áno	1,9 kB	32 x 32 px	1 024 px^2
twitter.png	áno	2,1 kB	32 x 32 px	1 024 px^2
google.png	áno	2,0 kB	32 x 32 px	1 024 px^2
youtube.png	áno	2,0 kB	32 x 32 px	1 024 px^2
dribbble.png	áno	2,8 kB	32 x 32 px	1 024 px^2
flickr.png	áno	2,0 kB	32 x 32 px	1 024 px^2
pinterest.png	áno	2,2 kB	32 x 32 px	1 024 px^2
picasa.png	áno	2,6 kB	32 x 32 px	1 024 px^2
linkedin.png	áno	1,9 kB	32 x 32 px	1 024 px^2
reddit.png	áno	1,9 kB	32 x 32 px	1 024 px^2
more.png	horizontálne	1,4 kB	1 x 36 px	36 px2
sidebar-li2.png	nie	1,5 kB	7 x 7 px	49 px2
sidebar-li1.png	nie	1,5 kB	7 x 7 px	49 px2
arrow_up.png	nie	1,3 kB	50 x 50 px	2 500 px2
error.png	áno	1,9 kB	560 x 72 px	40 320 px^2
SPOLU		**66,8 kB**		**671 194 px^2**

Tabuľka 2: Detailný rozbor obrázkov

2.1.1 Výpočet náročnosti prenosov

Pre určenie vhodnosti použitia techniky je potrebné určiť metriky podľa ktorých sa určí vhodnosť použitia. Sledovanými veličinami bude počet volaní servera a úspora množstva prenášaných dát.

Pri výpočtoch sa do úvahy berie iba súbor `styles.css` a relevantné obrázky z neho odkazované.

2.1.1.1 Počet HTTP požiadaviek

Prioritným dôvodom zavedenia tejto techniky je ušetrenie počtu volaní servra.

Pri použití pôvodného súboru `styles.css` je prenášaných 17 samostatných PNG obrázkov.

2.1.1.2 Objem prenášaných dát

Objem prenášaných dát zahŕňa dve množiny prenášaných údajov. Na jednej strane sledujeme množstvo bytov HTTP hlavičiek a na druhej samotnú veľkosť súborov.

Vzhľadom na to, že je potrebné preniesť 17 samostatných obrázkov a približne 700 bytov hlavičiek na jeden súbor, je výsledná veľkosť relevantných HTTP hlavičiek 11,6 kB.

Súčet veľkostí obrázkov je 66,8 kB. Veľkosť CSS súboru nie je relevantná.

3 Manuálne vytvorenie CSS Spritu

3.1 Spracovanie obrazových podkladov

V programe Gimp[8] som čo najoptimálnejšie manuálne vyskladal obrázky. Výsledkom sú dva obrazce. Pre minimalizovanie vonkajších vplyvov pri porovnávaní rôznych riešení nebola použitá žiadna extra optimalizácia výstupu, iba integrovaná kompresia vrámci štandardu PNG.

Prvý *sprite* s názvom *manual-sprite-horizontal.png* obsahuje súbory *bg.png* a *more.png* pod sebou. Súbor *bg.png* mal pôvodne šírku 50 px. Vzhľadom na nutnosť rovnakej šírky všetkých použitých horizontálnych obrázkov som manuálne zúžil obrázok na 1 px. Bolo to možné, pretože súbor obsahuje iba prechod a nie zložitejšiu vzorku.

Druhý zostavený súbor, *manual-sprite.png*, obsahuje zvyšných 11 použiteľných obrázkov. Najprv som do ľavého horného rohu umiestnil najväčší obrázok - *error.png*. Následne som vedľa neho vpravo vyskladal zvyšné použiteľné obrázky.

V rámci manuálneho skladania som vyskúšal umiestniť logá sociálnych sietí do dvoch radov, ale z dôvodu spôsobu fungovnia kompresie PNG to spôsobilo nárast dátovej veľkosti obrázka o 439 B. Od toho som ustúpil, pretože je prioritnejšia výsledná dátová veľkosť súboru a nie percentuálne využitie plochy obrázka.

Z dôvodu snahy o minimalizáciu počtu obrázkov, som sa pokúsil vložiť do obrázku `manual-sprite.png` aj horizontálne opakované obrázky `bg.png` a `more.png`. Pri tomto postupe je nutné roztiahnuť horizontálne sa opakujúce obrázky na celú šírku obrázka. Pre zúženie obrázka na minimum som vykladal ikony sociálnych sietí pod najširší obrázok `error.png`. Po spracovaní vznikol obrazec s rozmermi 560 x 740 px a veľkosťou 44,4 kB. V tomto prípade je teda dôsledkom úspora prenosov o jeden obrázok, ale veľkosť obrázku je až 44,4 kB oproti súčtu obrázkov dvoch samostatných spritov - 13,8 kB. Z dôvodu viacnásobného zväčšenia

8 http://www.gimp.org

spritu som v tomto konkrétnom prípade postup zavrhol. V niektorých vhodnejších prípadoch môže poskytovať zaujímavé výsledky.

Obrázok 4: Manuálne zostavený obrázok manual-sprite.png.

Ukážku súboru manual-sprite-horizontal.png nebudem prezentovať vzhľadom na nevhodné rozmery na umiestnenie na stranu.

Názov súboru	**Veľkosť**	**Dátová úspora**[9]	**Rozmer**	**Využitosť plochy**[10]
manual-sprite-horizontal.png	5,8 kB	3460 %	1 x 636 px	100 %
manual-sprite.png	13,2 kB	177 %	880 x 72 px	79,8 %

Tabuľka 3: Porovnanie spritov a pôvodných obrázkov.

Názov súboru	**Súradnice**
bg.png	0,0
more.png	0,600

Tabuľka 4: Zoznam obrázkov v manual-sprite-horizontal.png.

9 Pomer súčtu dátovej veľkosti jednotlivých čiastkových obrázkov s výsledným obrázkom.

10 Pomer súčtu plôch jednotlivých čiastkových obrázkov (v sprite) s rozmerom výsledného obrázka.

Názov súboru	Súradnice
dribbble.png	560,0
facebook.png	592,0
flickr.png	624,0
google.png	656,0
linkedin.png	688,0
picasa.png	720,0
pinterest.png	752,0
reddit.png	784,0
twitter.png	816,0
youtube.png	848,0
error.png	0,0

Tabuľka 5: Zoznam obrázkov v manual-sprite.png.

3.2 Úprava CSS súboru

Pre aplikáciu výsledných obrázkov je potrebné previesť zodpovedajúce zmeny v súbore `styles.css`. Zmeny sa týkajú výhradne názvu odkazovaného obrázka a súradníc pozadia.

Nasledujúce fragmenty kódu obsahujú všetky reálne potrebné zmeny. Posledný riadok v oboch ukážkach sa v súbore nachádza dva krát - v dvoch rôznych definíciách.

3.2.1 Pôvodný súbor

```
background: #3e3e3e url(../images/bg.png) 0 0 repeat-x;
background:url(../images/social/facebook.png) 0 0 no-repeat;
background:url(../images/social/twitter.png) 0 0 no-repeat;
background:url(../images/social/google.png) 0 0 no-repeat;
background:url(../images/social/youtube.png) 0 0 no-repeat;
background:url(../images/social/dribbble.png) 0 0 no-repeat;
background:url(../images/social/flickr.png) 0 0 no-repeat;
```

```
    background:url(../images/social/pinterest.png) 0 0 no-repeat;
    background:url(../images/social/picasa.png) 0 0 no-repeat;
    background:url(../images/social/linkedin.png) 0 0 no-repeat;
    background:url(../images/social/reddit.png) 0 0 no-repeat;
    background:#f26b04 url(../images/more.png) 0 0 repeat-x;
```

3.2.2 Zmenený súbor

```
    background: #3e3e3e url(../images/manual-sprite-horizontal.png)
0 0 repeat-x;
    background:url(../images/manual-sprite.png) -592px 0 no-repeat;
    background:url(../images/manual-sprite.png) -816px 0 no-repeat;
    background:url(../images/manual-sprite.png) -656px 0 no-repeat;
    background:url(../images/manual-sprite.png) -848px 0 no-repeat;
    background:url(../images/manual-sprite.png) -560px 0 no-repeat;
    background:url(../images/manual-sprite.png) -624px 0 no-repeat;
    background:url(../images/manual-sprite.png) -752px 0 no-repeat;
    background:url(../images/manual-sprite.png) -720px 0 no-repeat;
    background:url(../images/manual-sprite.png) -688px 0 no-repeat;
    background:url(../images/manual-sprite.png) -784px 0 no-repeat;
    background:#f26b04 url(../images/manual-sprite-horizontal.png) 0
-600px repeat-x;
```

3.3 Výpočet užitočnosti

Pre určenie vhodnosti použitia techniky je potrebné určiť metriky podľa ktorých sa určí vhodnosť použitia. Sledovanými veličinami bude počet volaní servera a úspora množstva prenášaných dát.

Vplyv na zmenu veľkosti zdrojového CSS súboru nebudeme brať do úvahy, pretože je zanedbateľný a väčšinový vplyv má naň názov vytvoreného obrázka. Ten bol zvolený samopopisne a nie dátovo úsporne, takže by to prípadnú štatistiku výrazne skreslilo. Prírastok veľkosti súboru spôsobený zmenou pozicovania je iba 60 bytov.

3.3.1 Počet HTTP požiadaviek

Prioritným dôvodom zavedenia tejto techniky je ušetrenie počtu volaní servra.

Pri použití pôvodného súboru `styles.css` bolo prenášaných 17 PNG obrázkov. Po aplikovaní zmien je potrebné preniesť iba x obrázkov - 2 *sprite* obrázky a 4 obrázky ktoré nemohli byť pre obmedzenia technológie použité.

Výsledkom je úspora 11 spojení na server, čo prestavuje úsporu 64,7 % počtu požiadaviek. Vo výpočte sú brané do úvahy iba obrázky zo súboru `styles.css`.

3.3.2 Objem prenášaných obrázkov

Po zmene je potrebné preniesť HTTP hlavičky pre šesť rôznych obrázkov. To znamená, že veľkosť HTTP hlavičiek bude približne 4,1 kB.

Súčet veľkostí šiestich prenášaných obrázkov je 42,1 kB, čo predstavuje úsporu 28,1 % prenášaných obrázkových dát. Vo výpočte sú brané do úvahy iba obrázky zo súboru `styles.css`.

4 Existujúce nástroje

Pre zjednodušenie aplikovania technológie CSS Sprite existujú rôzne pomocné služby. V tejto kapitole zanalyzujeme niektoré online generátory ktoré sa snažia manuálnu prácu urýchliť.

4.1 CSS Sprite Generator (Project Fondue)

Web: http://spritegen.website-performance.org

Služba poskytuje generovanie sprite obrázku zo vstupného ZIP súboru s obrázkami. Výstupom je sprite obrázok a definície nových css tried pre následné použitie v HTML kóde.

Medzi nastavenia patrí:

- Voliteľné ignorovanie duplicitných obrázkov.
- Zmena veľkosti obrázkov.
- Horizontálne/vertikálne usporiadanie obrázkov do spritu.
- Rozostupy medzi obrázkami.
- Formát výstupu.
- Voliteľná kompresia PNG výstupu OptiPNG.
- Základné nastavenia generovaného CSS.
- Voliteľné definovanie výšky a šírky CSS elementov.

Nastavenia sú o niečo detailnejšie, ale pre utvorenie prehľadu o službe výpis postačuje.

CSS definícia dodaná službou:

```
.sprite-dribbble{ background-position: 0 0;}
.sprite-facebook{ background-position: -33px 0;}
.sprite-flickr{ background-position: -66px 0;}
.sprite-google{ background-position: -99px 0;}
.sprite-linkedin{ background-position: -132px 0;}
.sprite-picasa{ background-position: -165px 0;}
.sprite-pinterest{ background-position: -198px 0;}
.sprite-reddit{ background-position: -231px 0;}
.sprite-twitter{ background-position: -264px 0;}
.sprite-youtube{ background-position: -297px 0;}
```

Obrázok 5: Výstup horizontálneho usporiadania generátorom CSS Sprite Generator (Project Fondue)

Pomocou tohto nástroje nie je možné automatizovane spracovať horizontálne alebo vertikálne sa opakujúce pozadia. Bolo by to možné upravením pozadí na rovnakú šírku v grafickom editore a následným nastavením usporiadania podľa orientácie. Keďže na to služba nie je stavaná, ide skutočne o zbytočne nepohodlné riešenie. Cieľ je rýchlejšie dosiahnuteľný manuálne.

Nástroj poskytuje široké možnosti nastavenia. Na výber sú dva jednoduché algoritmy zloženia spritu – vyskladanie obrázkov vedľa seba alebo pod seba v abecednom poradí. Použitím služby ušetrí grafik mnoho času, ale hlavnou nevýhodou tejto kategórie nástrojov je nutnosť upravovať nie iba kaskádové štýly, ale aj HTML kód stránky. V prípade úpravy dizajnu je potrebné manuálnu prácu s nástrojom zopakovať, nahradiť príslušné CSS a znovu upravovať HTML.

4.2 CSS Sprites Generator

Web: http://csssprites.com

Ide o veľmi jednoduchý generátor sprite obrazcov. Na hlavnej stránke je formulár so štandardnými formulárovými prvkami na nahrávanie. Spôsob nahrávania obrázkov je v súčasnej dobe zastaralý a zbytočne zložitý.

Možnosti nastavenia sú štandardne skryté. Nastaviteľný je priestor ponechaný okolo obrázka, jeho ohraničenie čiarou, spôsob usporiadania obrázkov a farba pozadia. Predvolené hodnoty sú nastavené pre bežné použitie a nie je ich vyslovene nutné meniť. Pre získanie čo najpodobnejšieho výsledku k manuálne vytvorenému spritu som zmenil priestor okolo obrázka z 10px na 0px a usporiadanie obrázkov hore. Usporiadanie hore znamená, že sa obrázky vyskladajú vedľa seba prisunuté k hornému okraju obrazca. Ohraničenie obrázkov čiarou som nenastavoval. Pozadie som tiež ponechal priehľadné.

Obrázok 6: Výstup generátora CSS Sprites Generator.

Výstupom generátora je sprite obrazec takmer zhodný s výstupom predošlého nástroja. Poskytnuté sú aj súradnice jednotlivých obrázkov v sprite a ukážka ich použitia. Tieto údaje je potrebné následne manuálne doplniť do CSS definície.

4.3 CSS Sprite Generator

Web: http://csssprite generator.net

Na prvý pohľad je tento nástroj moderne poňatý. Je postavený na moderných webových technológiách a celkovo pôsobí sviežo, takže je práca s ním príjemná. Kliknutím na odkaz z hlavnej stránky sa spustí editor. Doň je možné nahrať obrázky presunom myšou. Tie sú následne automaticky optimálne usporiadané.

Globálne je možné nastaviť pre všetky obrázky farbu pozadia, ohraničenie prázdnym priestorom a spôsob opakovania. Kliknutím je možné tieto vlastnosti nastaviť zvlášť pre každý obrázok. Zaujímavou možnosťou je nastavenie príznaku opakovania. podľa globálneho nastavenia povolenej orientácia je možné vybrané obrázky nastaviť ako vzorku – vyplnia celú šírku (resp. výšku) obrazca. Po ukončení nastavovania je pre každý obrázok vytvorená samostatná CSS trieda podľa názvu súboru, ktorej názov je možné interaktívnym formulárom zmeniť. Takto nastavený výstup je možné stiahnuť ako ZIP archív.

Obrázok 7: Výstup generátora CSS Sprite Generator.

V zbalenom archíve sa nachádza samotný sprite obrazec, CSS súbor s triedami na okamžité použitie aj s minifikovanou verziou a HTML ukážka použitia. V samostatnej zložke sú nahrané originálne súbory. Ak je tento nástroj použitý v čase tvorby šablóny, môže byť veľmi užitočným pomocníkom, ale v prípade potreby dopracovania spritu do funkčného dizajnu zostáva dosť práce na kóderovi.

4.4 Spritegen

Web: http://css.spritegen.com

Podobne ako predošlé generátory vytvára sprity z nahraných obrázkov. Disponuje hromadným nahrávaním obrázkov. Nastaviteľné parametre sú:

- veľkosť obtekania,
- súborový formát spritu,
- prefix CSS triedy.

Obrázok 8: Výstup generátora CSS Sprite Generator.

Po spracovaní je zobrazená stránka so zobrazeným spritom, CSS triedami a ukážkou použitia v HTML. Nástroj je extra jednoduchý a účelný. Rovnako ako pri predošlých predstavených nástrojoch je potrebné doplniť alebo zmeniť pôvodné CSS definície, prípadne upraviť HTML kód šablóny.

4.5 Stitches

Web: http://draeton.github.com/stitches

Ide o projekt demonštrujúci možnosti moderných jednostránkových aplikácií. Obrázky je možné do aplikácie vložiť natiahnutím myšou, nahratím alebo importom v minulosti exportovaného projektu.

Nastaviteľný je spôsob ukladania obrázkov, prefix a formát štýlopisu s prefixom. Voliteľne je možné vložiť sprite ako Data URL obsah priamo do CSS. Na výber sú tri druhy ukladania: kompaktný, vertikálny a horizontálny. Podporovanými štýlopismi sú CSS a LESS.

Obrázok 9: Výstup generátora CSS Sprite Generator.

Po kliknutí na *Download* sa zobrazí okno s ukážkou spritu, štýlopisom, ukážkovým HTML kódom a jeho praktickou ukážkou.

4.6 CSS Sprite Creator

Web: http://www.web2generators.com/graphism/css_sprite_creator

CSS Sprite Creator slúži k inému účelu ako predošlé. Jeho úlohou nie je generovať sprity, ale k existujúcim (napríklad manuálne vytvoreným) tvoriť CSS triedy so súradnicami.

Svoju prácu zvláda na dobrej úrovni vďaka zvládnutému technickému prevedeniu, ale pre jeho použitie nevidím väčší zmysel keďže existuje mnoho nástrojov ktoré toto zvládnu ako doplnok ku generovaniu spritu.

4.7 Sprite Me

Web: http://spriteme.org

Táto služba je vďaka plnej autonómnosti na použitie najjednoduchšia, ale jej výsledky sú použiteľné iba na ďalšie manuálne spracovanie alebo študijné účely. Jej samostatnosť vyplýva z analýzy HTML šablóny a CSS súborov namiesto štandardného importu obrázkov. Koncept používania je navrhnutý tak, že sa samotný

javascriptový kód generátora uloží do záložiek prehliadača. Následne sa po kliknutí na záložku spracuje práve otvorená stránka. Najprv sa zobrazí okno so zoznamom použiteľných a nepoužiteľných obrázkov. Po kliknutí na *Make* sa sprite vygeneruje a automaticky sa pregeneruje aj HTML kód načítaného webu a sprite sa použije.

Služba automaticky použije všetky obrázky použité v HTML ktoré majú parameter `background-repeat: no-repeat` bez ohľadu na iné parametre a zdrojový CSS súbor. To má za následok, že niektoré obrázky môžu mať nesprávne odhadnuté súradnice. Pri spracovaní referenčného dizajnu problém nenastal, ale je tam potenciál na jeho vznik. Spôsob prehľadávania HTML kódu namiesto CSS pre získanie zoznamu obrázkov je na ukážku výborný, ale pre produkčné nasadenie nie je príliš vhodný. Plne automatizovaný nástroj musí mať takto k dospozícií dva vstupy namiesto jedného a výsledný sprite by bol pri každej šablóne iný a bol by teda zle kešovateľný.

Výstupom je sprite obrazec a prehľadne zobrazené CSS definície ktoré boli upravené.

4.8 Záver

Predstavené služby sa líšia svojim technickým spracovaním a mierne sa líši aj ich poslanie. Vo všeobecnosti by sa dali zadeliť do troch kategórií:

- Generátor sprite obrazca a CSS tried zo vstupných obrázkov.
- Generátor CSS tried zo spritu.
- Generátor spritu a rekompilátor CSS z kódu webaplikácie.

V súčasnosti je najbežnejšou cestou k vytvoreniu spritu použitie niektorého nástroja z prvej kategórie. Vyžaduje to náročné manuálne spracovanie a následné manuálne aplikovanie zmien v CSS kóde, obrázkoch a nezriedka aj HTML šablóne.

V druhej kategórií bol spomenutý iba jeden generátor. Jeho využitie je veľmi limitované. Jeho použitie si viem predstaviť iba pri doplnení obrázku do existujúceho spritu. Vtedy by pomohol určiť súradnice, ale pri ručnom doplnení obrazcu by to bolo aj tak jednoduchšie priamo v grafickom editore.

Do tretej kategórie patrí iba posledný testovaný nástroj. Jeho špecialitou je generovanie spritu na mieru pre aktuálne načítanú stránku. Toto riešenie je veľmi zaujímavé, ale jeho praktické použitie je obmedzené, resp. je potrebné pamätať na jeho vlastnosti a výsledok prekontrolovať. Jeho bezpracné použitie je veľmi pohodlné a ak by ho bolo možné zaradiť ako kompilátor do reálneho produktu spĺňal by takmer všetky podmienky ideálneho nástroja.

5 Vlastné riešenie

Praktickou časťou diplomovej práce je vytvorenie ideálneho nástroja na využitie technológie CSS Sprite. Nástroj by mal poskytovať pohodlnejšie a pokročilejšie spracovanie vstupov ako doteraz existujúce riešenia.

Výsledným nástrojom je knižnica v jazyku PHP s podporou minimálnej verzie PHP 5.3.

Knižnica bude programovaná vo vlastnom mennom priestore s názvom *MiniSprite*.

5.1 Ciele

Na základe predchádzajúcej analýzy vlastného manuálneho spracovania a polo-automatizovaného spracovania použitím dostupných generátorov je možné stanoviť ciele a postupy ideálneho generátora.

5.1.1 Plne automatické spracovanie

Na rozdiel od väčšiny súčasných generátorov vstupom nebude sada obrázkov, ale samotná CSS definícia. Modulárny systém s inteligentným hľadaním optimálnych riešení zabezpečí bezpracné a plne automatizované spracovanie.

5.1.2 Bezpečnosť

Predvolená konfigurácia knižnice musí vylučovať akékoľvek defekty grafiky spôsobené použitím generátora. Môžu nastať situácie, keď by benevolentnejšie nastavenie mohlo spôsobiť zvýšenie efektivity. V prípade ak by v určitých prípadoch mohlo toto nastavenie spôsobovať problémy, predvolene bude vypnuté.

5.1.3 Konfigurovateľnosť

Všetky dôležité parametre spracovania budú jednoducho nastaviteľné bez zásahu do zdrojového kódu aplikácie.

Kontrola správnych typov a použiteľnosti nastavení bude prevádzaná pri samotnom nastavovaní a nie až pri použití nezmyselného nastavenia. Predíde sa tým mnohým problémom spôsobeným neskorím odhalením chýb. Tieto chyby nebudú špeciálne vykresľované ani nič podobné, ale korektne sa vytvoria výnimky.

5.1.4 Online generátor

Schopnosti generátora majú byť demonštrovateľné online, bez nutnosti inštalácie špeciálneho softvéru, alebo začlenenia zdrojového kódu generátora do projektu.

5.2 Prostredie

Pre zvýšenie stability a pohodlnejší vývoj bude knižnica programovaná testami riadeným vývojom, tzv. TTD[11]. Testy sa budú nachádzať v zložke tests a budú programované v testovacom frameworku Nette/Tester.

5.3 Popis funkčnosti

Pre čo najväčšiu modulárnosť je dôležité knižnicu rozdeliť do viacerých vrstiev. Každá vrstva bude mať zodpovednosť za konkrétnu časť spracovania a bude nahraditeľná výmenou triedy implementujúcej zvolené rozhranie. Riadenie a správu modulov bude mať na starosti minimalistické jadro.

Dôvodom rozhodnutia pre takúto architektúru systému je hlavne možnosť vkladania nových, zdokonalených, algoritmov na rozvrhnutie obrázkov v sprite. Negatívnym dôsledkom architektúry je väčšie množstvo súborov a tým aj zložitosť riešenia oproti monolitickej architektúre v ktorej by boli všetky súčasti napevno.

5.3.1 Jadro

Jadro softvéru bude v triede MiniSprite s rovnomenným menným priestorom.

Hlavné úlohy:

- konfigurácia,

11 Test Driven Development

- prevzatie vstupných dát,
- získanie všetkých obrázkových vstupov,
- volanie rôznych algoritmov na skladanie spritu,
- vyhodnotenie najlepšieho algoritmu podľa priorít,
- zloženie obrázku,
- zmena css súboru.

Knižnicu bude možné prevádzkovať v MVC frameworkoch ako službu. To znamená, že po prvotnom inicializovaní a korektnom nastavení môže viacnásobne spracúvať rôzne vstupy.

5.3.1.1 Konfigurácia

Nastavenie prostredia a obmedzení pre beh knižnice bude možné vykonať viacerými spôsobmi.

1. Dodanie asociatívneho poľa kľúčov a hodnôt ako prvého parametru pri vytváraní objektu.
2. Dodanie asociatívneho poľa kľúčov a hodnôt metódou `setConfig()`.
3. Nastavením konkrétnych parametrov špecializovanými metódami. Kvôli rozsahu budú tieto metódy existovať iba pre najdôležitejšie parametre.

Názov	Typ	Metóda	Popis
`basePath`	reťazec	`setBasePath()`	Nastaví základnú cestu od ktorej sa tvoria relatívne cesty v CSS.

`outputNormal`
`outputHorizontal`
`outputVertical`
	reťazec
	reťazec
	reťazec
		`setOutput()`	Nastaví cestu k spritu:
			- bez orientácie
			- s horizontálnou o.
			- s vertikálnou o.
			(tri parametre)

`folders`	pole `IFolders`		Obsahuje pole tried s algoritmami skladania obrazcov.
	`IFolders`	`addFolder()`	Pridá triedu s algoritmom na skladanie obrazcov.
`analyzer`	`IAnalyzer`	`setAnalyzer()`	Nastaví triedu na vyhodnotenie najvhodnejšieho vyskladania.
`composer`	`IComposer`	`setComposer()`	Nastaví triedu na vygenerovanie sprite obrázka.
`cssWriter`	`ICssWriter`	`setCssWriter()`	Nastaví triedu pregenerovanie CSS súboru.

Tabuľka 6: Detail konfiguračných možností.

5.3.1.2 Prevzatie vstupných dát

Rozhraním pre vstup bude verejná metóda `compile()` s dvomi parametrami:

1. samotný CSS kód určený na spracovanie,
2. základná cesta k priečinku odkiaľ sa majú určovať relatívne cesty k zdrojom.

Návratovou hodnotou bude prekompilovaný obsah CSS vstupu.

5.3.1.3 Získanie obrázkov

Zo vstupného CSS kódu je potrebné získať definíce všetkých obrázkov. Na to je možné použiť regulárny výraz `~\bbackground(-image)?\s*:(.*?)url\s*\(\s*(\'|")?(?<image>.*?)\3?\s*\)~i`.

Pre unifikáciu práce so vstupnými obrázkami bude použitý *prototyp* `Image`.

5.3.1.4 Volanie algoritmov skladania spritu

Po naplnení vstupných dát knižnica iteruje nad zaregistrovanými objektami s algoritmami skladania spritov. Algoritmy obsahujú triedy rozhrania `IFolder`.

Jednotlivé objekty generujú ako výstup unifikovaný objekt triedy - prototyp `Fold`.

5.3.1.5 Vyhodnotenie najlepšieho algoritmu

Na základe výstupov algoritmov skladania je potrebné zvoliť jeden ktorý bude následne použitý. Objekt typu `IAnalyzer`, zvolený ako ideálny podľa preferencií používateľa, ohodnotí pole prototypov `Fold` získané v predchádzajúcom kroku a vyberie víťaza.

Analyzované môžu byť rôzne hľadiská, ako napríklad rozmer, veľkosť, počet ušetrených požiadaviek a podobne.

5.3.1.6 Vygenerovanie obrázku

Samotné generovanie obrázku má na starosti trieda s rozhraním `IComposer`. Vstupom triedy je víťazný prototyp `Fold`. Na základe obsiahnutých informácií o obrázkoch a ich súradniciach sa obrázky zapíšu do výsledného spritu, prípadne viacerých podľa orientácií.

5.3.1.7 Prepísanie obsahu CSS

Rovnaký vstup ako je potrebný na vykreslenie obrázku je použitý na pregenerovanie CSS vstupu. Aktualizuje sa názov obrázku pozadia a jeho súradnice. Zvyšok zostane pôvodný. Túto operáciu prevádza trieda rozhrania `IcssWriter`.

5.3.2 Prototyp CssBlock

Objekty tejto triedy obaľujú dátovú a logickú funkcionalitu jednotlivých blokov CSS definícií. Ich základ je univerzálne použiteľný pre parsovanie, modifikáciu a následné použitie blokov definícií.

Konštruktor prijíma jediný parameter - textový reťazec zdrojového kódu jazyka CSS. Následne je tento kód regulárnymi výrazmi analyzovaný a po častiach uložený do lokálnych premenných `$selector` a `$parameters`. Obsahom premennej $selector je popis elementu ku ktorému sa definícia viaže. Do asociatívneho pola $parameters sa ukladajú jednotlivé páry *parameter – hodnota*.

Využitím magických metód `__set()`, `__get` a `__unset` je možné priamo volaním názvu parametra meniť alebo získať jeho vlastnosti. Magická metóda `__toString()` zabezpečí pri pokuse o vypísanie objektu vrátenie korektne formátovaného textového reťazca s CSS definíciou zohľadňujúcou vykonané zmeny. Pri konverzií sú kvôli úspore priestoru parametre s prefixom `background-` zlúčené do jedného parametru a zoznamu hodnôt.

Parametre s prefixom `background-`[12], ktoré sú pre sprite relevantné, môžu byť zapísané dvomi rôznymi spôsobmi. Buď samostatne, pričom je upresnenie názvu parametra za prefixom `background-`, alebo vrámci jedinej definície parametru `background` s vymenovanými parametrami v presne definovanom poradí. Nevýhodou združených parametrov je zložitejšie strojové spracovanie, keďže je podľa špecifikácie možné niektoré parametre ignorovať. Tým vzniká problém s identifikáciou hodnôt jednotlivých parametrov.

12 background-color, background-image, background-position, background-repeat, background-attachment

Možnosťou dvojakého zapisovania hodnôt disponujú viaceré skupiny CSS parametrov, ale pre účel použitia tejto triedy by bolo ich rozoznávanie zbytočné. Z toho dôvodu nebolo v ich prípade použité žiadne protiopatrenie.

Objekt rieši spomenutý problém automatickým parsovaním hodnôt parametra `background` pri vytváraní objektu – v konštruktore. Ak je niektorý z podriadených parametrov už definovaný, predošlá nastavená hodnota sa prepíše. Toto správanie je kompatibilné s prehliadačmi. Pre validný zápis stačí aby bola v parametri background minimálne jedna hodnota. Viac ich nie je povinných, ale môžu byť nastavené.

Parameter `background-color` môže nadobúdať štyri typy hodnôt:

- Slovný názov farby - Anglické kľúčové slovo. Napríklad `Black`, `Red` a iné.
- Hexadecimálny zápis - mriežka a tri alebo šesť znakov v šestnástkovej sústave. Jeden alebo dva znaky reprezentujú množstvo farby z farebného priestoru RGB v danej výslednej farbe. Napríklad `#000`/`#000000` pre čiernu, `#F00`/`#FF0000` pre červenú a podobne.
- Desiatkový zápis množstva farebných zložiek farebného priestoru RGB vo výslednej farbe. Napríklad `rgb (0, 0, 0)` pre čiernu alebo `rgb (255, 0, 0)` pre červenú a podobne. Je možné definovať aj úroveň alfa kanálu štvrtým parametrom, napríklad `rgba(255, 0, 0, 0.5)` pre červenú s 50 % priehľadnosťou.
- Percentuálny zápis vo farebnom priestore HSL a HSLA. Napríklad `hsl (120, 100%, 50%)` pre zelenú alebo `hsla(120, 100%, 50%, 0.5)` pre zelenú s 50 % krytím.
- Samostatnou hodnotou je `transparent` pre objekty bez farby a krytia pozadia.

Parameter `background-image` má iba dve možnosti zápisu:

- Reťazec `url()` s cestou k obrázku v zátvorkách.
- Reťazec `none` pre element bez obrázku.

Parameter `background-position` je možné zapísať nasledovnými spôsobmi:

- Pre prvý parameter určujúci horizontálnu orientáciu je možné zapísať kľúčovými slovami `left`, `center`, `right` a druhý pre vertikálnu osu slovami `top`, `center` a `bottom`.
- Percentuálny zápis vyjadrujúci posun pozadia, napríklad `33 %`.
- Zápis v pixeloch, napríklad `10 px`. Pre hodnotu `0` netreba udávať jednotky.

Parameter `background-repeat` môže nadobudnúť iba štyri hodnoty. Sú to kľúčové slová `repeat`, `no-repeat`, `repeat-x` a `repeat-y`.

Posledný možný parameter `background-attachment` nadobúda, obdobne ako predošlí, iba hodnoty kľúčových slov: `scroll`, `fixed` a `local`.

Nakoniec je potrebné, aby sa parametre viažuce sa k pozadiu automaticky obojsmerne synchronizovali. To znamená, že ak sa zmení parameter s prefixom `background-` aktualizuje sa parameter `background`. V opačnom prípade, ak je aktualizovaný jeden z podriadených parametrov, synchronizuje sa aj hlavný parameter. S týmto opatrením súvisí aj úprava metódy `__toString()` v ktorej sú automaticky ignorované hodnoty ktoré môžu byť zahrnuté v parametri `background`. Kvôli jednoduchšej neskoršej manipulácií sú hodnoty `top`, `left` a `0px` parametra `background-position` automaticky menené na `0`.

5.3.3 Prototyp Image

Táto trieda je unifikovanou prepravkou pre distribúciu obrázkov naprieč knižnicou. V prvom rade zabezpečuje základné ukladanie informácií o obrázkoch. Okrem toho obsahuje metódy na získanie týchto ako aj doplnkových informácií. Dôležitou úlohou je udržiavanie konzistentného stavu informácií pre prípad ak by časť knižnice obrázok nejakým spôsobom upravila.

Vzhľadom na nevhodnosť voľného používania textových reťazcov v kóde som definoval konštanty `GIF`, `JPG`, `PNG`, `NORMAL`, `HORIZONTAL`, `VERTICAL` a

`WITHOUT` pre zjednotenie definícií obrázkov. Tieto konštanty sú využiteľné naprieč knižnicou.

Vstupom pri vytváraní objektu je konštruktor s URL adresou obrázka a CSS definíciou prototypom `CssBlock` z ktorej je obrázok odkazovaný.

Pre komunikáciu s okolím je navrhnuté rozhranie niekoľkých metód:

- getUrl() pre získanie adresy obrázka v súborovom systéme.
- getType() je možné získať typ obrázku – GIF, JPG alebo PNG.
- getContent() pre získanie binárneho obsahu obrázka.
- getCssBlock() s návratovou hodnotou prototypu CssBlock ku ktorému sa obrázok vzťahuje.
- getSize() vracia veľkosť obrázku v bytoch.
- getRepeating() je aliasom pre parameter background-repeat z CssBlock-u a vracia orientáciu opakovania .
- getWidth() návratovou hodnotou je šírka obrázku v pixeloch.
- getHeight() návratovou hodnotou je výška obrázku v pixeloch.

Pôvodne som pre zisťovanie reálnych rozmerov obrázka uvažoval nad úspornejším lazy loadingom informácií o obrázkoch. Vzhľadom na nutný duplicitný prístup k súborovému systému kvôli metódam na získanie oboch rozmerov obrázkov som sa rozhodol prístup zavolať iba raz. Počas toho jedného prístupu je možné odkontrolovať aj formát súboru. Keďže je vhodné kontrolovať formát už pri vytváraní objektu v konštruktore, zároveň sú zistené aj rozmery a tým pádom odpadá lazy loading.

Trieda obsahuje aj statickú metódu `parseBackgroundPosition()` parsujúcu parameter `background-position`. Vracia dvojprvkové pole so zvlášť horizontálnou a vertikálnou pozíciou. Keďže je k nim v knižnici potrebné pristupovať na rôznych miestach samostatne, je to iba univerzálny pomocník z dôvodu deduplikácie kódu.

5.3.4 Prototyp Fold

Objekty tejto triedy predstavujú jeden sprite. Skladacie algoritmy môžu vrátiť jeden až tri predpisy na skladanie spritov – v základe pre každú orientáciu jeden.

Jediným vstupným parametrom konštruktoru je pole položiek prototypu `Image`. Tie sú uložené do chránenej premennej a prístup k nim je zabezpečený výhradne getterom `getImages()`. Rozhranie objektu uzatvárajú metódy `getWidth()` a `getHeight()` s návratovými hodnotami rozmerov sprite obrázku. Rozmery sú prepočítavané podla pozícií obrázkov v pridruženej CSS definícií a reálnych rozmerov obrázkov.

Názov metódy	**Popis**
`addImage()`	Pridáva obrázok do kolekcie. Má parametre orientácia, Image a x/y - súradnica.
`getWidth()`	Vráti šírku obrazca.
`getHeight()`	Vráti výšku obrazca.
`getImages()`	Vráti pole obrázkov v kolekcií. Voliteľným parametrom je orientácia - filter vracaných obrázkov.

Tabuľka 7: Metódy prototypu Fold

5.3.5 Rozhranie IFolder

Rozhranie IFolder zabezpečuje jednotnosť objektov s algoritmami generujúcimi sprity. Obsahujú iba jedinú verejnú metódu `generate()` s jediným vstupným parametrom – polom prototypov `Image`.

Návratovou hodnotou metódy generate je pole prototypov `Fold`. Vo vnútri tried budú odfiltrované obrázky ktoré nie sú pre sprite použiteľné. Použiteľné obrázky sú následne rozdelené podla kategórií a rozmiestnené na virtuálnu plochu budúcich spritov priamou úpravou ich `CssBlock`-u. Z výsledných zoznamov sa vytvorí pole `Fold`-ov a vrátia sa na ďalšie spracovanie.

Presná vnútorná implementácia bude záležať od princípu funkčnosti vnútorného algoritmu.

5.3.6 Implementácia FolderHorizontal

Objekt implementuje rozhranie IFolder a slúži na návrh skladby spritu. Ako vyplýva už z názvu, obrázky jednoducho ukladá vedľa seba. Poradie obrázkov nie je menené, spracúvané sú poradí v akom sú uvedené v CSS.

V prvom kroku sú dodané obrázky roztriedené podla orientácie opakovania pozadia. Zároveň sú odfiltrované obrázky ktoré majú definovanú pozíciu inú ako v pixeloch. Následne sa nad obrázkami s opakovaním `no-repeat` iteruje. V cykle je pre každý obrázok zistená pôvodná pozícia pozadia, zvlášť horizontálna aj vertikálna. Algoritmus pre svoju prácu potrebuje pracovať s číselnými údajmi pozícií, takže sú odstránené jednotky – `px`.

Keďže majú byť spracovateľné aj už existujúce sprity, je potrebné detegovať zhodné zdroje obrázkov. Počítanie nových pozícií sa líši podľa toho, či obrázok s rovnakým zdrojom už bol spracovaný alebo nie. Aktuálna horizontálna pozícia sa zaznamenáva do samostatnej premennej. Na začiatku vykonávania je vynulovaná.

Ak tento obrázok ešte nebol zaradený do spritu, nastaví sa jeho pozícia horizontálna pozícia rovná aktuálnej horizontálnej pozícií. Vertikálna pozícia obrázku je v tomto algoritme vždy nulová. Výpočet horizontálnej pozície posunu pozadia pozostáva zo súčtu pozície obrázku vynásobenej `-1` a pôvodného posunu. Vertikálny posun zostáva zhodný s pôvodným vertikálnym posunom. Následne je počítadlo posunu posunuté o reálnu šírku obrázku, aby sa čiastkové obrázky neprekrývali.

Ak sa spracúvaný obrázok v sprite už nachádza nastavia sa jeho súradnice zhodne ako ich má nastavené rovnaký obrázok s inou CSS definíciou. Pozícia posunu pozadia sa počíta rovnako ako v prípade ak je obrázok v sprite nový, ale namiesto súradnice aktuálneho posunu sa použijú súradnice rovnakého obrázku zo spritu. Tým sa zabráni duplicite obrázkov.

Zo spracovaných obrázkov sa vytvorí prototyp `Fold` a ten a vráti v jednoprvkovom poli.

5.3.7 Implementácia FolderVertical

Tento algoritmus priamo vychádza z predošlého – horizontálneho. Detaily kódu netreba rozvádzať keďže kód je tiež veľmi podobný. Rozdiel je iba posune po vertikálnej osi namiesto horizontálnej.

5.3.8 Rozhranie IAnalyzer

5.3.9 Implementácia AnalyzerMinArea

5.3.10 Implementácia AnalyzerMinTransport

5.3.11 Rozhranie IComposer

5.3.12 Implementácia SimpleComposer

5.4 Spätná väzba

Záver

Zoznam použitej literatúry

VRÁNA, J. 2010. 1001 tipů a triků pro PHP. Brno : Computer Press, a. s., 2010. 456 s. ISBN 978-80-251-2940-1.

SPDY: An experimental protocol for a faster web. [online]. [cit. 2014.03.27.] Dostupné na internete: <http://dev.chromium.org/spdy/spdy-whitepaper>.

NETWORK WORKING GROUP. The "data" URL scheme. [online]. [cit. 2014.03.27.] Dostupné na internete: <http://tools.ietf.org/html/rfc2397>.

Eclipse. [online]. [cit. 2014.03.27.] Dostupné na internete: <http://joomlathemes.co/demo3x/eclipse>.

THE PHP GROUP. PHP: Hypertext Preprocesor. [online]. [cit. 2014.03.28.] Dostupné na internete: <http://php.net>.

PROJECT FONDUE. CSS Sprite Generator. [online]. [cit. 2014.03.27.] Dostupné na internete: <http://spritegen.website-performance.org>.

MIROSLAV PECKA. Regulární výrazy. [online]. [cit. 2014.04.03.] Dostupné na internete: <http://regularnivyrazy.info>.

TANTEK ÇELIK, CHRIS LILLEY, L. DAVID BARON, STEVEN PEMBERTON, BRAD PETTIT. CSS Color Module Level 3. [online]. [cit. 2014.04.12.] Dostupné na internete: <http://www.w3.org/TR/css3-color>.

W3SCHOOLS. CSS Color Names. [online]. [cit. 2014.04.12.] Dostupné na internete: <http://www.w3schools.com/cssref/css_colornames.asp>.

W3SCHOOLS. CSS Legal Color Values. [online]. [cit. 2014.04.16.] Dostupné na internete: <http://www.w3schools.com/cssref/css_colors_legal.asp>.

Obsah priloženého CD

Na priloženom CD sa nachádza samotný text diplomovej práce v súbore *diplomova-praca-matyus-2014.pdf*.

Priečinok *MiniSprite* obsahuje výslednú knižnicu vytvorenú v rámci tejto diplomovej práce. V priečinku je licencia v súbore *license*, finálne implementácie všetkých častí knižnice v samopopisných PHP súboroch. Priečinok *API* obsahuje dokumentáciu knižnice vygenerovanú nástrojom ApiGen. Ukážky použitia knižnice sú v priečinku *example*.