VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

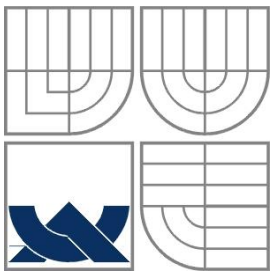
FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

GRAFY VE WEBOVÉM PROHLÍŽEČI POMOCÍ JAVASCRIPTU

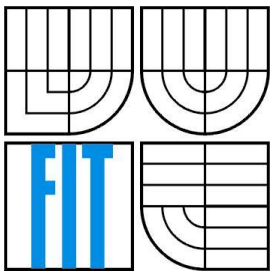
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE JIŘÍ ZAJÍC
AUTHOR

BRNO 2011



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

GRAFY VE WEBOVÉM PROHLÍŽEČI POMOCÍ JAVASCRIPTU

GRAPHS IN WEB BROWSER USING JAVASCRIPT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — JIŘÍ ZAJÍC
AUTHOR

VEDOUCÍ PRÁCE — Ing. LUKÁŠ MÁČEL
SUPERVISOR

BRNO 2011

Abstrakt

Tato bakalářská se práce se zabývá studiem tvorby grafů ve webových prohlížečích. První část obsahuje podrobnou studii současných technologií a návrh aplikace. Praktická část zahrnuje vývoj, popis a dokumentaci aplikace. Výsledkem je funkční přenositelná knihovna pro tvorbu grafů v elementu canvas v jazyce JavaScript a grafické uživatelské rozhraní.

Abstract

This bachelor's thesis deals with displaying graphs in web browsers. The first part introduces in detail current technologies and application design. The practical part includes implementation, description and documentation of the application. The result is a functional portable library for graph creation in JavaScript using canvas element and graphical user interface.

Klíčová slova

graf, JavaScript, Adobe Flash, SVG, HTML5 Canvas, vykreslení grafu

Keywords

graph, JavaScript, Adobe Flash, SVG, HTML5 Canvas, plotter

Citace

Jiří Zajíc: Grafy ve webovém prohlížeči pomocí JavaScriptu, bakalářská práce, Brno, FIT VUT v Brně, 2011.

Grafy ve webovém prohlížeči pomocí JavaScriptu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Lukáše Máčela.
Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

……
Jiří Zajíc
7. 5. 2011

Poděkování

Na tomto místě chci poděkovat všem, kteří mi pomohli s vypracováním bakalářské práce, zejména vedoucímu Ing. Lukáši Máčelovi a externímu konzultantovi Ing. Petru Novotnému.

Obsah

Seznam obrázků

Seznam zdrojových kódů

1 Úvod

Za posledních několik let došlo k masivnímu rozšíření informačních systémů, ke kterým se přistupuje pomocí webového prohlížeče. Uživatel se často setkává s internetovým či intranetovým rozhraním, které pracuje nad databází s nemalým množstvím dat a údajů. Ty je třeba prezentovat rychle a kvalitně ve vhodně zvolené formě. Zobrazení nejen statistických údajů by mělo být nepříliš náročné na vytížení systému, dostatečně dynamické a flexibilní pro případné úpravy a změny a samozřejmě jednoduché z pohledu běžného uživatele, který jen velmi nerad instaluje potřebné doplňky či pluginy.

Mým úkolem bylo nejdříve prostudovat grafy a podrobně zdokumentovat současné technologie pro jejich vytváření ve webovém prohlížeči, což je část spadající pod kapitoly 2 a 3. Ve 4. kapitole se zabývám návrhem a implementací ukázkové knihovny v jazyce JavaScript, která jako součást serverové aplikace umožňuje zadávání vstupních dat několika různými způsoby, nastavení vlastností grafu a následné vykreslení. Závěr práce tvoří částečné porovnání jednotlivých technologií, zhodnocení praktické části a zmíněny jsou možné směry dalšího vývoje.

1.1 Motivace

Během studia na Fakultě informačních technologií v Brně se mi podařilo získat praxi v nadnárodní společnosti Honeywell, Inc., kde pracuji jako vývojář *cloud* systému pro plánování zdrojů.

Stále častěji můžeme v IT slovníku zaslechnout pojem *cloud*. Zjednodušeně lze říci, že se jedná o poskytování služeb či programů, které jsou uloženy na vzdálených serverech a ke kterým uživatelé na internetu přistupují zpravidla pomocí svého webového prohlížeče. Za významný milník v historii *cloud computing*u bývá označován rok 2006, kdy společnost Amazon spustila svůj dlouho připravovaný projekt Amazon Web Service.

Přibližně před rokem se konglomerátní společnost Honeywell, Inc. rozhodla pro nasazení produktu pro řízení podniku SAP R/3 vyvinutém německou firmou SAP AG. Tento finanční software má zjednodušovat plánování a dosahovat tak přehlednějšího výkaznictví, které následně slouží jako podklad pro snazší rozhodování a efektivnější řízení běhu organizace. Během své desetileté existence byl SAP R/3 distribuován statisícům firem.

Není tedy žádným překvapením, že tak masivně rozšířený software jen velmi obtížně splňuje představy každého jednotlivého zákazníka/uživatele. Honeywellu konkrétně chybí funkčnost v oblasti

podrobného plánování zdrojů (angl. *Resource Planning*). Zprvu byl tento nedostatek kompenzován po síti sdílenými soubory tabulkových procesorů. Ty jsou dnes postupně nahrazovány stále se vyvíjející *cloud* aplikací na míru, ke které uživatelé přistupují pomocí standardního webového prohlížeče. Tato práce se mj. zabývá vývojem doplňkové knihovny, která by umožnila generování grafů pomocí JavaScriptu.

2 Grafy

Pro účely mé práce je vhodné vymezit pojem graf. Dále je třeba uvést jeho nejčastější typy, u kterých částečně vycházím z aplikace MS Excel, která je součástí sady MS Office.

2.1 Definice pojmů

Graf lze definovat několika různými způsoby. Vybral jsem následující matematickou definici [1]:

$$Graf\ funkce\ f(x_1, x_2, \dots, x_n)\ je\ množina\ všech\ (n+1)tic\ \left(x_1, x_2, x_{\dots}, f(x_1, x_2, x_{\dots}, x_n)\right).$$

Tu si mohu představit jako grafickou reprezentaci výše zmíněné množiny ve formě křivky nebo plochy v kartézské soustavě souřadnic.

*Diagram*em potom rozumím strukturovanou grafickou reprezentaci číselných, matematických a statistických údajů sloužící k názornému přehledu či objasnění.

2.2 Klasifikace

Na základě charakteristiky dat, určených pro vykreslení, je třeba zvolit správný typ grafu. Nejpoužívanějšími jsou

- sloupcový – vhodný pro zobrazení datových hodnot v závislosti na časovém období,
- spojnicový – zobrazování zpravidla souvislých dat,
- výsečový – oblíbený u široké veřejnosti, z informativního hlediska nevhodný - lidské oko na první pohled špatně odhadne velikost jednotlivých částí; navíc umí zobrazit pouze jednu datovou řadu [7],
- plošný – vychází ze spojnicového typu, ale mnohdy lépe zaujme,
- bodový – velmi rozšířený, nejvíce přizpůsobivý širokému spektru dat.

3 Současné technologie

Dnešní moderní prohlížeče nabízí několik možností, jak přistoupit k vykreslování grafů a ke kreslení obecně. V této kapitole se budu zabývat jednak samotným skriptovacím jazykem JavaScript, a dále pak řešeními, které jsou něm více či méně postavené.

3.1 JavaScript

JavaScript je multiplatformní, objektově orientovaný a prototypově založený skriptovací jazyk. Je dialektem původního jazyka ECMAScript, jehož autorem je Brendan Erich ze společnosti Netscape Communications. V roce 1997 byl standardizován asociací European Computer Manufacturers Association (ECMA). I přes to, že jej lze použít i na straně serveru, mnohem častěji bývá implementován jako součást webového prohlížeče a interpretuje se tedy na straně klienta. Vkládá se přímo do HTML stránek a jeho syntaxe je velmi podobná jazykům C, C++ nebo Java. [9][10]

Kromě dynamického HTML se JavaScript používá k implementaci rozšíření pro další aplikace, jako je například Adobe Acrobat. Lze jej spouštět i v OS Windows pomocí technologie Windows Script Host a nahradit tak klasické dávkové soubory.

3.1.1 SVG

Scalable Vector Graphics (SVG) je deskriptivní značkovací jazyk rozšiřující standard Extensible Markup Language (XML). Slouží pro popis dvoudimenzionální vektorové a smíšené grafiky [4].

V roce 1998 skupina SVG Working Group společenství World Wide Web Consortium (W3C[1]) začala s vývojem, o pět let později byla vydána první verze s označením 1.0 a dnešní verze 2.0 s podporou HTML5, CSS a WOFF je nejčastěji používaným standardem vektorové grafiky na internetu.

Obrázky se definují textovými soubory XML. To umožňuje jejich snadné prohledávání, indexování, skriptování a případnou kompresi. K jejich vytváření a editování postačí libovolný

[1] *„Posláním W3C mezinárodního konsorcia je vést World Wide Web do jeho plného potenciálu vývojem protokolů a směrnic pro zajištění dlouhodobého růstu Webu.“* [2]

textový editor. Kreslit obrázky můžeme i přímo JavaScriptem, ale v praxi jej zpravidla využijeme až ve fázi, kdy chceme objektu přidat dynamičnost.

SVG definuje tři základní typy grafických objektů:

- vektorové grafické tvary (úsečka, křivka, obdélník, kružnice aj.),
- rastrové obrazy,
- text.

Ty se dají seskupovat dohromady, stylovat, zvětšovat či zmenšovat a mohou být vsazeny do jiných již vyrenderovaných objektů. Mezi jejich základní vlastnosti patří vnořené transformace, průhlednost, efekty s filtry a vytváření šablon. SVG kresby mohou být interaktivní a dynamické. Animace je možné definovat a volat buď přímo deklarativně, nebo pomocí skriptů. Protože SVG, stejně jako XHTML (Extensible Hypertext Markup Language), je rozšířením standardu XML, je možné k jeho elementům přistupovat prostřednictvím Document Object Model (DOM) a stylovat je pomocí Cascading Style Sheet (CSS).

Následující zdrojový kód 3-1 demonstruje vytvoření jednoduchého obrazu 3-1 za použití běžného textového editoru. SVG lze však formovat i pomocí grafických editorů, jako je např. open-source projekt Inkscape[2].

```
<?xml version="1.0" standalone="no"?>

<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN"
"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">

<svg width="400px" height="160px" version="1.1"
xmlns="http://www.w3.org/2000/svg">

  <rect x="20" y="20" width="250" height="100"
    fill="red" stroke="black" stroke-width="5" opacity="0.7"/>

  <circle cx="300" cy="100" r="50"
    fill="blue" stroke="green" stroke-width="2" opacity="0.3"/>

  <text x="35" y="100"
    font-family="Verdana"
    font-size="24"
    font-weight="bold"
    fill="green"
    stroke="black">Scalable Vector Graphics</text>

</svg>
```

Zdrojový kód 3-1: Kreslení jazykem SVG

[2] Vektorový grafický editor využívající standardu škálovatelné vektorové grafiky. Více na http://inkscape.org/.

Obrázek 3-1: Výsledek kreslení SVG

Formát SVG začal jako první nativně podporovat Konqueror[3] v roce 2004. Ke konci 2010 už tak činila naprostá většina rozšířených prohlížečů. Jako poslední se přidal Internet Explorer, který bez nutnosti instalace doplňku od Adobe podporuje SVG až od své poslední verze 9 vydané počátkem roku 2011.

Hlavní výhody kreslení lze shrnout do tří následujících bodů:

1. Zabere malé množství dat, ukládá informace o ploše, tvaru a barvě, nikoliv o každém pixelu zvlášť.
2. Umožňuje adresovat jednotlivé části obrázků, skriptovat je a vytvářet tak interaktivní aplikace bez nutnosti instalace doplňků a zcela otevřeně.
3. Každý nakreslený tvar je uchován jako objekt v datové struktuře (*scene graph* nebo DOM). Pokud změním atributy jednoho takového tvaru, prohlížeč je schopný celou scénu s požadovanými změnami snadno rekonstruovat.

3.1.2 HTML5 Canvas

Canvas je novým párovým elementem rozšiřující specifikace HTML5 jazyka HyperText Markup Language (HTML). Umožňuje dynamické skriptovatelné vykreslování 2D tvarů a rastrových obrázků.

Tuto technologii původně představila společnost Apple, Inc. pro použití v Mac OS X WebKit složce, později byl adoptován open-source renderovacím jádrem Gecko a následně standardizován pracovní skupinou Web Hypertext Application Technology Working Group (WHATWG). Odtud už nechybělo mnoho a byl schválen a zařazen W3C mezi standardy [5].

Canvas, neboli plátno, je oblastí na webové stránce s definovanou šířkou a výškou, viz zdrojový kód 3-3. Nejdříve je ale třeba získat na element referenci, což lze standardně použitím DOM, a následně je nutno požádat o kontext voláním kódu 3-2.

[3]Unixový webový prohlížeč a správce souborů implementovaný jako součást K Desktop Environment (KDE).

```
var canvas = document.getElementById('canvas');
var context = canvas.getContext('2d');
```

Zdrojový kód 3-2: Reference a kontext elementu canvas

Proměnná `context` nyní obsahuje rozhraní `CanvasRenderingContext2D`, které disponuje celou sadou kreslících funkcí podobně jako SVG, čímž umožňuje dynamicky vytvářet grafy, animace, hry apod.

```
interface HTMLCanvasElement : HTMLElement {
           attribute unsigned long width;
           attribute unsigned long height;
  DOMString toDataURL(in optional DOMString type, in any... args);
  object getContext(in DOMString contextId, in any... args);
};
```

Zdrojový kód 3-3: Rozhraní elementu canvas

Pro vykreslení shodného obrazu 3-2 s příkladem jako u SVG jsem použil kód 3-4 využívající knihovnu jQuery[4].

```
var canvas = document.getElementById('canvas');
if (canvas.getContext) {
    var context = canvas.getContext('2d');
    // obdelník
    context.globalAlpha = 0.7;
    context.fillStyle = 'red';
    context.fillRect(22.5, 22.5, 245, 95);
    context.strokeStyle = 'black';
    context.lineWidth = '5';
    context.strokeRect(20, 20, 250, 100);
    // kruh
    context.globalAlpha = 0.3;
    context.beginPath();
    context.fillStyle = 'blue';
    context.arc(300, 100, 49, 0, Math.PI * 2);
    context.fill();
    context.arc(300, 100, 50, 0, Math.PI * 2);
    context.strokeStyle = 'green';
    context.lineWidth = '2';
    context.stroke();
    // text
    context.globalAlpha = 1;
    context.font = 'bold 24px Verdana';
    context.fillStyle = "green";
    context.fillText("HTML5 Element Canvas", 35, 100);
    context.lineWidth = "1";
    context.strokeStyle = "black";
    context.strokeText("HTML5 Element Canvas", 35, 100);
}
```

Zdrojový kód 3-4: Kreslení JavaScriptem do elementu canvas

[4] *„Rychlá a stručná JavaScript knihovna pro zjednodušení zpracování událostí, animování a pohybu v HTML dokumentu pro rychlý vývoj webových aplikací.“* [3]

Obrázek 3-2: Výsledek kreslení JavaScriptem do elementu canvas

Canvas je podporován aktuálními verzemi[5] prohlížečů Mozilla Firefox, Google Chrome, Internet Explorer, Safari a Opera. Pro starší verze Exploreru, které jej nativně interpretovat neumí, jsou k dispozici doplňky od Google a Mozilla.

3.1.3 Adobe Flash

Celým názvem Adobe Flash Professional CS5. Jedná se o jeden z nejrozšířenějších grafických vektorových procesorů. Vlastní jej společnost Adobe Systems Incorporated (dříve Macromedia). Používá se mj. pro tvorbu interaktivních animací, her, bannerů a reklam.

Flash má v současné době implementovaný vlastní programovací jazyk ActionScript [6], který vychází ze standardizované verze JavaScriptu zvané ECMAScript. Jeho první tři verze poskytovaly pouze omezené interaktivní vlastnosti; tlačítkům a rámcům bylo možno připojit jednoduchý příkaz `action`. Sada akcí obsahovala základní navigační prvky, jako jsou `play`, `stop`, `getURL` nebo `gotoAndPlay`.

ActionScript je syntaxí podobný JavaScriptu, vyžaduje velmi dobrou znalost objektově orientovaného programování a ve své poslední třetí verzi, vydané v roce 2006, disponuje výbornou kontrolou a znuvupoužitelností kódu, čímž se stává vhodným nástrojem pro vývoj komplexních webových aplikací a prezentací.

[5] Firefox 4.x, Chrome 11.x, Internet Explorer 9.x, Safari 5.x, Opera 11.x

```
// obdelník
var rect:Shape = new Shape();
rect.graphics.beginFill(0xFF0000, 0.7);
rect.graphics.drawRect(22.5, 22.5, 245, 95);
rect.graphics.endFill();
rect.graphics.lineStyle(5, 0x000000, 0.7);
rect.graphics.drawRect(20, 20, 250, 100);
this.addChild(rect);
// kruh
var circ:Shape = new Shape();
circ.graphics.beginFill(0x0000FF, 0.3);
circ.graphics.drawCircle(300, 100, 49);
circ.graphics.endFill();
circ.graphics.lineStyle(2, 0x008000, 0.3);
circ.graphics.drawCircle(300, 100, 50);
this.addChild(circ);
// text
var text:TextField = new TextField();
text.width = 250;
text.x = 35;
text.y = 76;
var myFormat:TextFormat = new TextFormat();
myFormat.size = 24;
myFormat.font = 'Verdana';
myFormat.bold = 'true';
myFormat.color = '0x008000';
text.defaultTextFormat = myFormat;
text.filters = [new GlowFilter(0x000000,1,2,2,2,2)];
text.text = "Action Script 3.0";
this.addChild(text);
```

Zdrojový kód 3-5: Kreslení ActionScriptem

Obrázek 3-3: Výsledek kreslení ActionScriptem

ActionScript sám o sobě je open-source software, jeho specifikace je nabízena zdarma a dostupný je jak překladač, tak i virtuální stroj. Flash Professional je naopak proprietární a Adobe Systems prodává jeho licenci za přibližně dvacet tisíc korun[6].

[6] €845,79 k 11. 5. 2011.

Při vyvíjení interaktivních animací a webových prezentací ve Flashi má programátor doslova volné ruce a může si s aplikací dělat, co jej napadne. To je vykoupeno nezbytnou instalací pluginu do webového prohlížeče, bez kterého uživatel obsah stránky ve formátu ShockWave Flash (SWF) nezobrazí.

3.2 Ostatní technologie

Problematiku zobrazování grafů ve webových prohlížečích je možné řešit nespočtem dalších technologií a metod. Uživatel se často setká např. s Microsoft Silverlight[7], Google Chart Tools[8], Java[9], ASP.NET[10] aj. JavaScriptu se ale týkají jen okrajově nebo vůbec, a proto se jimi nebudu dále zabývat.

3.3 Rozšíření technologií

Bohužel se mi pro účely této práce nepodařilo dohledat věrohodný zdroj statistických údajů. Podpora HTML5 a elementu canvas je dle mého názoru více či méně závislá na rozšíření Internet Exploreru 9. Zatímco weboví vývojáři a nadšenci pro nové technologie už s canvasem pravděpodobně zkušenosti mají, běžný až pokročilejší uživatel PC tento pojem nejspíše slyší poprvé.

Obecně ale rozšíření nového elementu přikládám velkou naději. Je součástí standardu, nevyžaduje instalaci dodatečných pluginů a doplňků, práce s ním je intuitivní a rychlá, je podporován prakticky všemi používanými prohlížeči v jejich nejnovějších verzích. Dle mého názoru se dočká pozitivního ohlasu např. majitelů těch zařízení, které nepodporují Adobe Flash.

[7] Platforma pro tvorbu dynamického obsahu online. Více na http://www.microsoft.com/cze/web/silverlight/.
[8] Zobrazování aktuálních dat pomocí JavaScriptu a XML. Více na http://code.google.com/intl/cs-CZ/apis/chart/.
[9] Samostatná platforma pro tvorbu aplikací s dynamickým obsahem. Více na http://www.java.com/en/.
[10] Součást .NET frameworku s podporou dalších jazyků. Více na http://www.asp.net/.

4 Vývoj knihovny pro tvorbu grafů

Po prostudování technologií potřebných pro zobrazení grafu ve webovém prohlížeči a seznámení se s jazykem JavaScript jsem provedl podrobnou analýzu požadavků zadání práce z pohledu aplikace. Po výběru vhodné technologie jsem začal s implementací.

4.1 Analýza požadavků a návrh aplikace

Základní nároky týkající se funkčnosti JavaScriptové knihovny byly jasně stanovené. Požadavky specifikované příliš obecně nebo s jistou mírou volitelnosti jsem upřesnil a některé z nich v rozumné míře rozšířil.

4.1.1 Formát vstupních dat

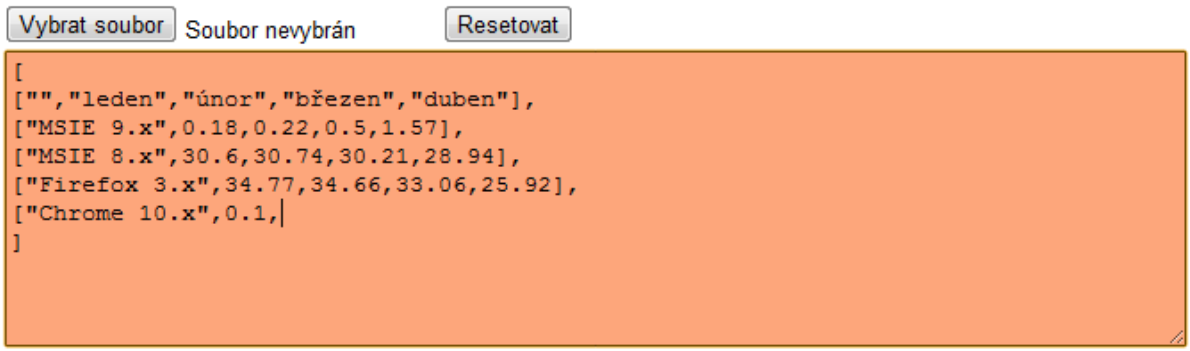
Vstupní data může uživatel definovat dvěma způsoby: interně a externě.

U prvního způsobu jde o zavolání metody `plot(data)`, kde je volitelným parametrem dvoudimenzionální pole definované přímo v kódu JavaScriptu, viz kód 4-1.

```
var data = new Array(
    new Array(           "", "leden",  "únor", "březen", "duben"),
    new Array(   "MSIE 9.x",    0.18,    0.22,      0.5,    1.57),
    new Array(   "MSIE 8.x",    30.6,   30.74,    30.21,   28.94),
    new Array("Firefox 3.x",   34.77,   34.66,    33.06,   25.92),
    new Array("Chrome 10.x",     0.1,    0.13,     6.64,   11.21)
);
```

Zdrojový kód 4-1: Zhuštěný zápis pole v jazyku JavaScript

Parametr `data` je polem polí, které může mimo samotných hodnot pro vykreslení obsahovat i legendu a popisy osy x (resp. názvy jednotlivých řad a sloupců). Do stránky je možné vložit vhodné rozhraní (např. textarea) s patřičným ID, do kterého aplikace data po analýze zobrazí a umožní je upravovat v reálném čase (reaguje na události *onkeyup* a *onchange*). V mé ukázkové aplikaci jsou data zadávána ve formátu JavaScript Object Notation (JSON) a pro syntaktickou kontrolu je použita javascriptová funkce *eval()*.



Obrázek 4-1: Textarea s editací vstupních dat v reálném čase

Data lze definovat i vnějším způsobem – do samostatného souboru. Ten může být buď typu Excel Spreadsheet s příponou XLSX nebo standardní XML. Pro lepší představu uvádím níže obrázek 4-2 a zdrojový kód 4-2. Protože v mém případě dochází ke zpracování JavaScriptu na straně klienta, viz kapitolu 3.1, aplikace neumí/nemůže z bezpečnostních důvodů pracovat se souborovým systémem. Upload souboru přes uživatelské rozhraní tedy řeší skript na straně serveru implementovaný jazykem Hypertext Preprocessor (PHP) a aplikaci předá pole ve formátu JSON.

	A	B	C	D	E	F
1		leden	únor	březen	duben	
2	MSIE 9.x	0,18	0,22	0,50	1,57	
3	MSIE 8.x	30,60	30,74	30,21	28,94	
4	Firefox 3.x	34,77	34,66	33,06	25,92	
5	Chrome 10.x	0,10	0,13	6,64	11,21	

Obrázek 4-2: Příklad vstupních dat ve formátu XLSX

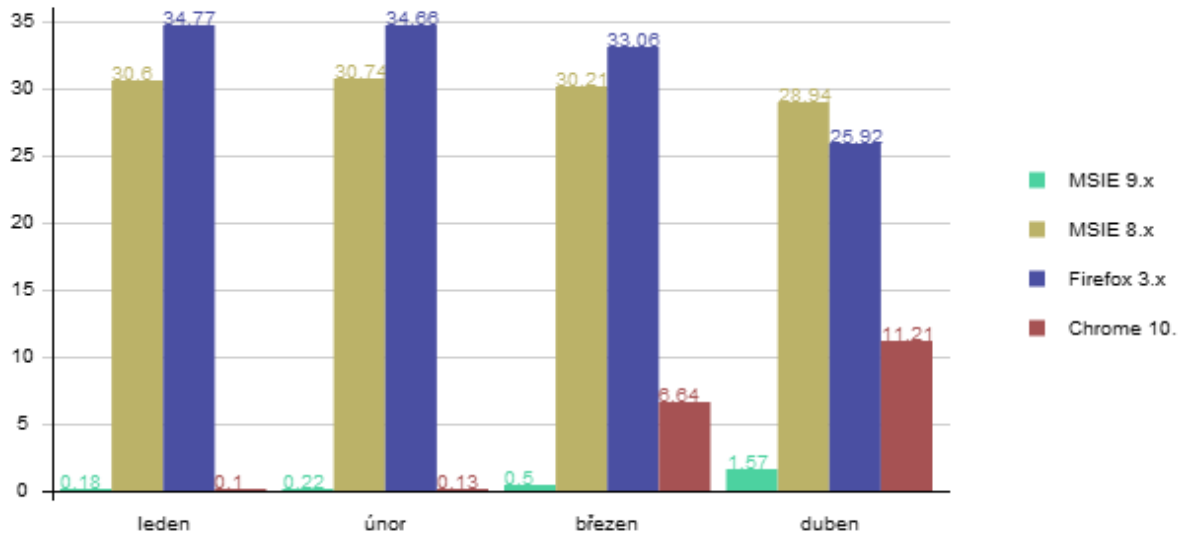
```
<graph>
    <data>
        <MSIE9>
            <leden>0.18</leden>
            <únor>0.22</únor>
            <březen>0.5</březen>
            <duben>1.57</duben>
        </MSIE9>
        <Firefox>
            <leden>30.6</leden>
            <únor>30.74</únor>
            <březen>30.21</březen>
            <duben>28.94</duben>
        </Firefox>
    </data>
</graph>
```

Zdrojový kód 4-2: Příklad vstupních dat ve formátu XML

4.1.2 Typy grafů

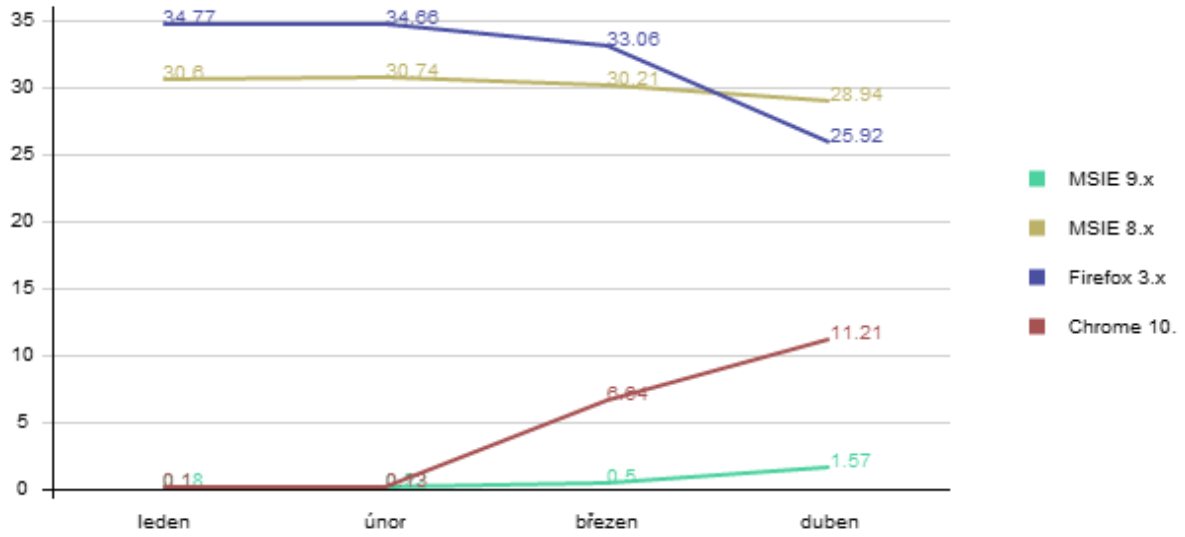
Aplikace umí zobrazit všech 5 typů grafů uvedených v kapitole 2.2 a v zadání práce [7].

Sloupcové grafy se používají pro zobrazení a porovnání dat. Dále je můžeme rozdělit na horizontální a vertikální. Jsou snadno pochopitelné, skládají se z obdélníků o určité výšce (délce) podle jejich hodnoty. Vertikální sloupcový graf 4-3 má na ose x kategorie a na ose y numerické hodnoty, podle kterých se odvíjí výška jednotlivých obdélníků/sloupců. Porovnává celkem 4 datové sady odlišené barvou.



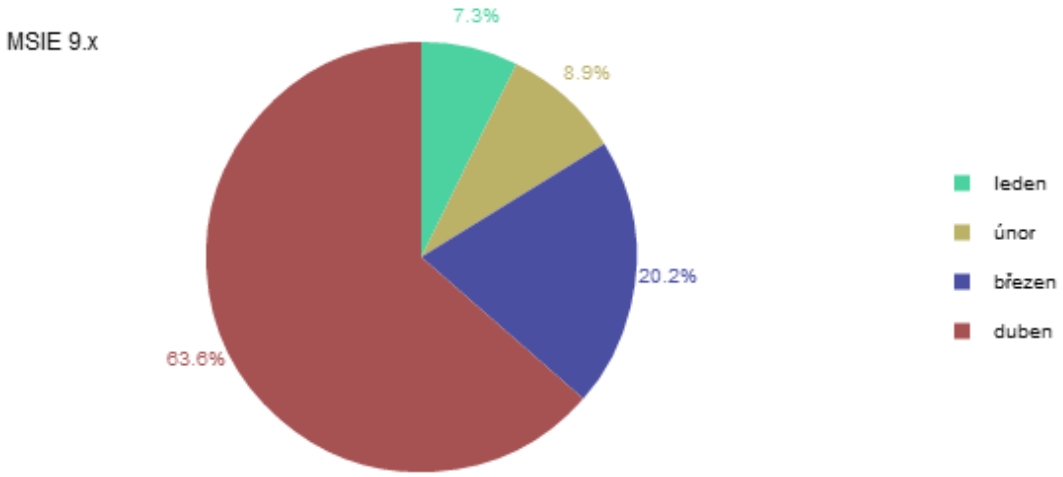
Obrázek 4-3: Sloupcový graf

Dalším typem jsou grafy spojnicové. Jejich implementace je poměrně jednoduchá a z pohledu uživatele jsou velmi snadné pro pochopení. Třídí a představují data jasným způsobem, přičemž zobrazují i souvislosti. Jsou používány pro osobní, výukové i odborné účely. V oblasti vědy a statistiky bývá tento typ grafu také používán pro předpovědi ještě neshromážděných dat. Ačkoliv sdílí spojnicové a sloupcové grafy stejný účel, spojnicové zobrazují změnu směru, zatímco sloupcové změnu rozsahu. Ukázka na obrázku 4-4 představuje průběh času na ose x a hodnoty v čase na ose y.



Obrázek 4-4: Spojnicový graf

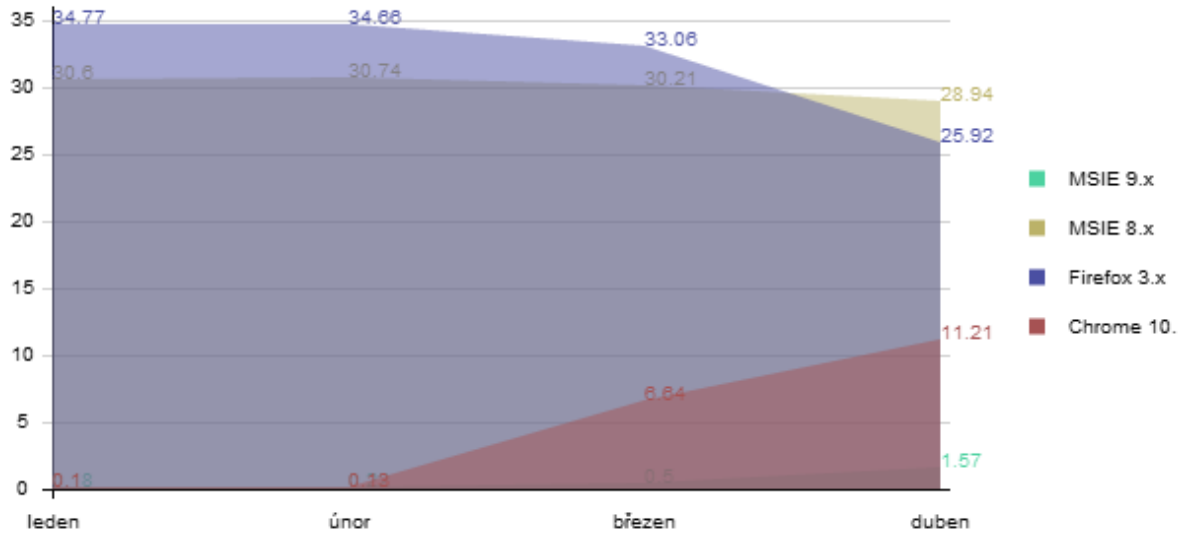
Výsečové grafy se používají pro reprezentaci kategoriálních dat nebo hodnot proměnných. Prakticky jsou to kruhy rozdělené na segmenty, které proporčně odráží svoji hodnotu v poměru k celku. Pro porovnání a popis segmentů se používají procentuální údaje, přičemž celý kruh je roven 100%. Implementačně je vykreslení výsečového grafu náročnější než u ostatních typů. Paradoxně i přes fakt, že výsečový graf umí zobrazit vždy pouze jednu datovou sadu. Použití tohoto typu při více než 5 kategoriích se nedoporučuje z důvodu nepřehlednosti [7].



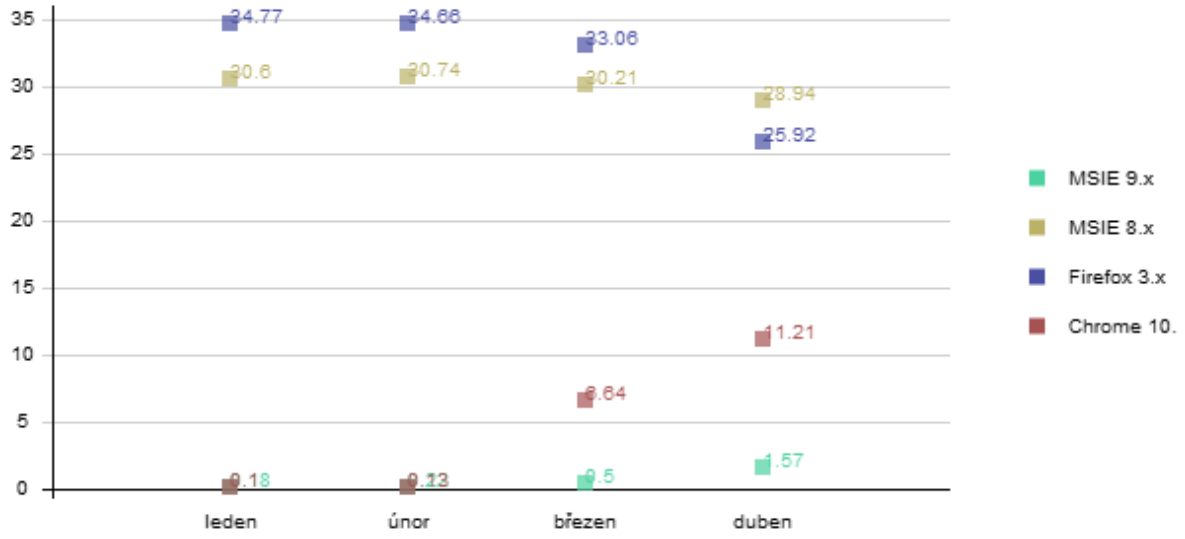
Obrázek 4-5: Výsečový graf

Zbývající 2 typy grafů, plošné a bodové, vychází ze spojnicového typu. Výhodou plošného grafu je, že lépe zaujme; zejména je-li pro jednotlivé plochy použita částečná průhlednost, viz

obrázek 4-6. Bodový graf je považován za nejobecnější a nejpřizpůsobivější. Umožňuje zobrazit jakoukoliv kombinaci dat a závislostí jako zbývající uvedené typy.



Obrázek 4-6: Plošný graf



Obrázek 4-7: Bodový graf

4.1.3 Vlastnosti grafů

Knihovna umožňuje kromě samotného typu grafu specifikovat několik dalších nastavení. Ne všechny možnosti jsou dostupné pro všechny typy grafů (koláčový bude často výjimkou) a současně ne všechny hodnoty jsou v režii uživatele.

Proč? Jak je později zmíněno v kapitole 4.1.3, knihovna se např. při kalkulaci počtu dílků osy y snaží přiblížit číslu 8. Pokud však vstupní data tvoří pouze 2 sloupce o hodnotách [[1, 2], [2, 1]], je 8 zbytečně velké číslo a knihovna raději zvolí optimálnější počet 3.

Právě mezi ty parametry, které uživatel nemůže přímo ovlivnit a které si aplikace sama spočítá na základě vstupních dat, jsou

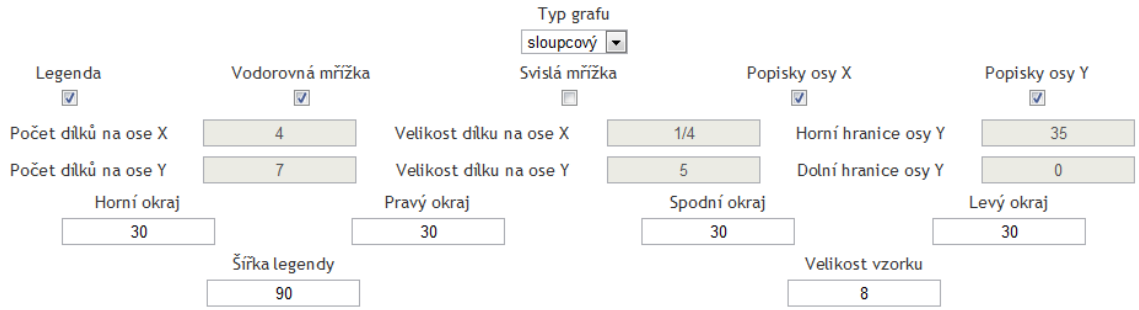
- maximum a minimum osy y (netýká se výsečového grafu),
- počet částí, na které bude osa y rozdělena (zpravidla 8 ± 2; netýká se výsečového grafu) a
- počet částí osy x (= počet sloupců; v případě výsečového grafu počet segmentů).

Pokud aplikace najde v DOM webové stránky vstupy či elementy s ID těchto proměnných (viz kapitolu 4.3.2), uživateli hodnoty zobrazí. Neumožní je ale měnit.

Vlastnosti vykresleného grafu, které měnit lze, jsou typu `boolean` nebo `unsigned integer`. Do první kategorie spadají možnosti zobrazení a skrytí

- legendy (= popis a barevné označení datových sad),
- vodorovné či svislé mřížky,
- stupnici hodnot osy y a
- popisy osy x (= názvy sloupců).

Číselnými hodnotami lze upravovat velikost okrajů (angl. *padding*), šířku legendy a velikost barevného vzorku datových sad. Hodnoty jsou v jednotkách pixelů.



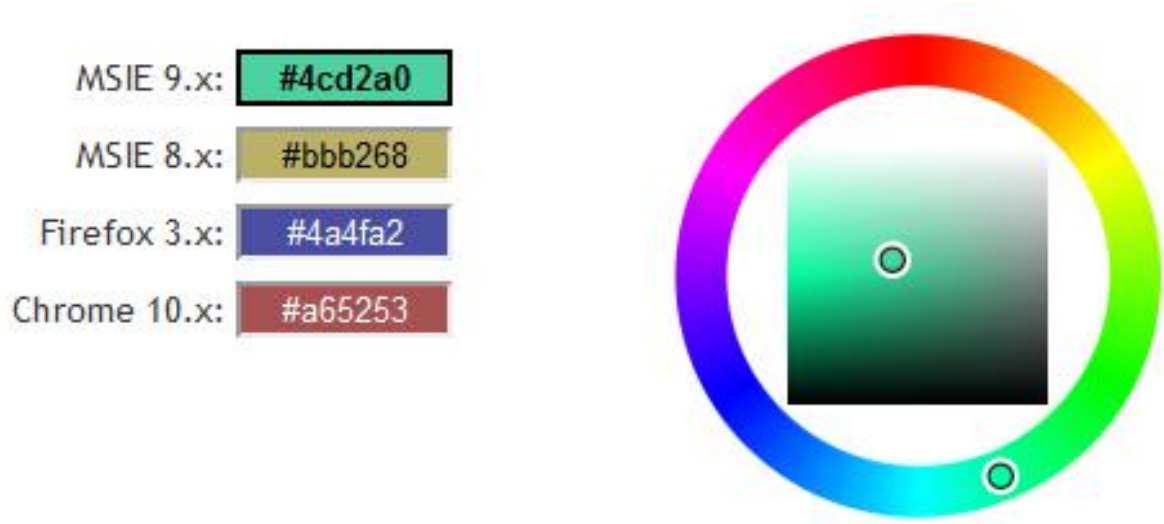
Obrázek 4-8: Možnosti nastavení

4.1.4 Výběr barev pro datové sady

Jedno z volitelných rozšíření zadání této práce. *Color picker*, volně přeloženo jako dialog pro výběr barvy, není přímo součástí knihovny. Použil jsem jQuery modul s názvem Farbtastic, jehož autorem je Steven Wittens a který je dosupný na http://acko.net/dev/farbtastic pod licencí General Public Licence (GPL), a implementoval jej jako samostatnou část aplikace.

Při analýze vstupních dat se kontroluje proměnná *_colors*, zda je k dispozici dostatečný počet barev pro vykreslení grafu. Pokud tomu tak není, knihovna vygeneruje náhodné pastelové barvy sama, viz kapitolu 4.3.6.

Po vykreslení grafu dojde k inicializaci dialogu pro výběr barev, která způsobí zobrazení patřičného počtu vstupů s hodnotami vygenerovaných barev v hexadecimálním kódu. Ty má uživatel následně možnost upravovat pomocí color pickeru.



Obrázek 4-9: Farbtastic *color picker* - dialog pro výběr barvy

4.2 Výběr technologií

Výběr technologií byl z určité části dán zařazením této práce do kategorie webové technologie. Použití XHTML, JavaScriptu/jQuery, PHP a XML bylo prakticky dané. Při volbě nejdůležitější části aplikace, na které bude stát samotná knihovna pro generování grafů ve webovém prohlížeči, jsem vybíral z technologií používaných pro současná řešení v kapitole 3, respektive 3.1.

4.2.1 Canvas

Canvas jsem zvolil proto, že je z výše uvedených technologií nejmladší a vzhledem k jeho podpoře není zatím příliš rozšířený. Současně dle mého názoru nabízí prostředky postačující ke tvorbě grafů a je vhodný ke splnění zadání práce [8]. Významným kladem této volby je, že funguje za podpory HTML5 bez nutnosti instalace jakéhokoliv pluginu či doplňku.

4.3 Implementace

Knihovnu pro vytváření grafů tvoří definice objektu `Graph` a jednotlivých metod. Je psána v čistém JavaScriptovém kódu bez použití jQuery či jiných nástaveb. Očekává právě jeden vstupní parametr typu `string` s ID elementu canvas, obsahuje soukromé proměnné `_canvas` a `_ctx` a metody zmíněné níže v této kapitole.

4.3.1 Inicializace

Po vytvoření instance objektu `Graph` proběhne inicializace jedinou soukromou metodou `_init(elementId)`. Ta nejprve zkontroluje, zda element s ID opravdu existuje, a voláním `getContext('2d')` ověří, jestli se jedná o canvas. Pokud vše proběhne úspěšně, kontextové rozhraní pro kreslení se uloží do `_ctx`. Ve druhé proměnné `_canvas` zůstane reference na canvas, pomocí které se zjišťuje šířka a výška elementu a případně se provádí vymazání celého plátna před novým kreslením, například „změnou" jeho šířky, viz zdrojový kód 4-3. V případě zachycení chyby vyvolá výjimku s patřičnou chybovou hláškou.

```
        this._canvas.width = this._canvas.width;
```

Zdrojový kód 4-3: Vymazání plátna canvas

Knihovna si tedy uklidí plátno a zavolá následující metodu, která se postará o deklaraci a inicializaci potřebných proměnných.

4.3.2 Nastavení výchozích hodnot

Po provedení metody `_setDefault()` je již objekt připraven vykreslit prázdný graf. Všimněme si ve zdrojovém kódu 4-4 proměnné _data, která je inicializována dvoudimenzionálním polem (viz kapitolu 4.1.1) o jediné hodnotě 0.

```
        /** @private */ this._colCount = 0; // number of columns
        /** @private */ this._rowCount = 0; // number of rows
        /** @private */ this._maxYValue = 0; // highest value
        /** @private */ this._minYValue = 0; // lowest value
        /** @private */ this._tickYCount = 8; // wanted number of y ticks
        /** @private */ this._tickYSize = 0; // axis y tick range (value)
        /** @private */ this._tickYSizePx = 0; // horizontal lines distance (pixels)
        /** @private */ this._axisXPos = 0; // axis x position (axis y value)

        /** @private */ this._tickXCount = 0; // wanted number of x ticks
        /** @private */ this._tickXSize = 0; // axis x tick range (value)

        /** @private */ this._data = [[0]]; // graph data
        /** @private */ this._headings = []; // graph headings
        /** @private */ this._legend = []; // graph legend
        /** @private */ this._colors = []; // predefined colors

        /** @public */ this.graphType = 'column'; // column, line, pie, area, scatter
        /** @public */ this.showHorizontalLines = true; // horizontal lines
        /** @public */ this.showVerticalLines = false; // vertical lines
        /** @public */ this.showHorizontalLabels = true; // horizontal line labels
        /** @public */ this.showVerticalLabels = true; // vertical line labels
        /** @public */ this.showLegend = true; // vertical line labels

        /** @public */ this.legendWidth = 90;  // space for legend
        /** @public */ this.patternSquareSize = 8; // legend square size
        /** @public */ this.border = [30, 30, 30, 30]; // canvas borders
```

Zdrojový kód 4-4: Deklarace a inicializace výchozích proměnných

4.3.3 Zpracování vstupních dat

Nejkomplexnější a nejsložitější metodou knihovny je `analyzeData(data)`, která počítá následující stěžejní proměnné pro správné vykreslení os, mřížky, popisků aj. částí grafu na základě vstupních dat předaných parametrem.

- `_colCount` – počet sloupců
- `_rowCount` – počet řádků
- `_legend` – popisy sloupců (pokud je první sloupec numerický, generují se automaticky *Řada 1*, *Řada 2* až *Řada N*)
- `_headings` - popisy řádků (pokud je první řádek numerický, generují se automaticky *0*, *1* až *N*)
- `_data` – pokud je první řádek anebo sloupec nenumerický (obsahuje alespoň v jedné hodnotě nečíselný znak), je celý odstraněn a v `_data` zůstávají pouze čísla, viz kód 4-5
- `_maxYValue`, `_minYValue` – maximální a minimální hodnota pro zobrazení
- `tickYSize`, `tickYCount` – velikost a počet dílků osy y
- `axisXPos` – vertikální posunutí osy x (nenulová v případě záporných hodnot)
- `tickXCount`, `tickXSize` – velikost a počet dílků osy x

```
isNaN(parseFloat(this._data[row][cell])) ?
            0 : parseFloat(this. data[row][cell]);
```

Zdrojový kód 4-5: Ignorování nečíselných hodnot

Právě výpočet `_tickYSize`, tedy absolutní velikosti dílku na ose y, podle kterého se následně upraví hodnoty `_maxYValue` a `_minYValue`, je klíčový a není snadné jej správně zvolit. Sestavil jsem následující funkční algoritmus 4-6, jehož výsledky se při totožných vstupních datech podobají výstupům z aplikace MS Excel.

Za předpokladu, že jsem již zjistil nejvyšší a nejnižší hodnotu vstupních dat, například pomocí `Math.max()` a `Math.max()`, spočítám předběžně `_tickYSize` jako podíl rozdílu `_maxYValue` a `_minYValue` k počtu požadovaných dílků osy y. Následně tuto hodnotu dělím 10 a uchovávám počet dělení, dokud není `_tickYSize` menší než 10. Její řád uložím do proměnné `rank`. Přičtu konstantu 0.5 a číslo zpátky vynásobím mocninou 10^{rank}. Tím docílím toho, že velikost dílku osy y bude končit číslicí 0 nebo 5, tím pádem bude dělitelná pěti. Nyní znovu spočítám `maxYValue` a `minYValue`, tentokrát však na základě `tickYSize` a s použiím zaokrouhlení dolů pomocí `Math.floor()`. Závěrečnou podmínkou pro maximum `maxYValue` i minimum `minYValue` zkontroluji, jestli nově vypočtená nejvyšší/nejnižší hodnota není nižší/vyšší nebo rovna původní hodnotě `_maxYValue`/`_minYValue` a případně přičtu/odečtu jeden `tickYSize` navíc, což může způsobit dříve zmiňovaný rozptyl v očekávaném počtu dílků osy, jak bylo zmíněno v 4.1.3.

```
        tickYSize = (this._maxYValue - this._minYValue) /
        this._tickYCount;
        var rank = 0;
        while(tickYSize > 10)
        {
            tickYSize /= 10;
            rank++;
        }
        tickYSize = Math.round(tickYSize + 0.5);
        tickYSize *= Math.pow(10, rank);
        maxYValue = tickYSize * Math.floor((this._maxYValue + 1) /
        tickYSize);
        minYValue = tickYSize * Math.floor(this._minYValue / tickYSize);
        if(maxYValue <= this._maxYValue
            && this._maxYValue != 0)
        {
            maxYValue += tickYSize;
        }
        if(minYValue >= this._minYValue
            && this._minYValue != 0)
        {
            minYValue -= tickYSize;
        }
```

Zdrojový kód 4-6: Výpočet velikost dílku osy y `_tickYSize`

4.3.4 Vykreslení grafu

Na základě hodnot vypočítaných při analýze dat je knihovna schopna zobrazit 5 výše zmíněných typů grafu. U každého, až na výsečový, který opět tvoří výjimku, vykreslení probíhá ve 3 fázích:

- posazení os a případně i mřížky,
- vykreslení jádra tvořícího podstatu grafu,
- zobrazení popisů sloupců a
- zobrazení legendy.

Metody se jmenují příznačně `plotAxes()`, `plotGraph()`, `plotLabels()` a `plotLegend()`. Protože element canvas nepodporuje vrstvy, v metodě `plotCompleteGraph()`, která sestává z těchto 4 metod, se provádí právě v tomto pořadí, aby nedocházelo k překrývání podstatných informací.

4.3.5 Možnosti

Nastavení možností zmíněných v kapitole 4.1.3 lze provést kromě přímého přístupu k proměnné objektu také metodou `setOptions(options)`. Ta na vstupu očekává jednorozměrné pole, jehož klíče jsou názvy proměnných.

4.3.6 Barvy datových sad

Poslední nepopsanou částí knihovny je sada tří metod `getColor(number)`, `insertColor(number, color)` a `getColors()`, která zajišťuje práci s barvami datových sad. První z nich je využívána metodami `plotGraph()` a `plotLegend()` a na vstupu očekává číslo požadované barvy, kterou vrací ve formátu RGB. Pokud v proměnné `_colors` barva požadovaného čísla neexistuje, metoda vygeneruje náhodný pastelový odstín a zkontroluje, zda není příliš podobný dvěma předchozím.

Jak je vidět na zdrojovém kódu 4-7, složky barev se generují v rozmezí a 150 – 215 a za podobné barvy metoda považuje ty, jejichž vzdálenost v krychli RGB modelu je větší než 120.

```
var red;
var green;
var blue;
var difFromLast = true;
var difFromOneButLast = true;
do
{
    red = parseInt(Math.random() * 150 + 65);
    green = parseInt(Math.random() * 150 + 65);
    blue = parseInt(Math.random() * 150 + 65);
    if(this._colors[number - 1])
    {
        difFromLast = Math.sqrt(
            Math.pow(red - this._colors[number - 1]['red'], 2)
            + Math.pow(green - this._colors[number - 1]['green'], 2)
            + Math.pow(blue - this._colors[number - 1]['blue'], 2)
        ) > 120;
    }
    if(this._colors[number - 2])
    {
        difFromOneButLast = Math.sqrt(
            Math.pow(red - this._colors[number - 2]['red'], 2)
            + Math.pow(green - this._colors[number - 2]['green'], 2)
            + Math.pow(blue - this._colors[number - 2]['blue'], 2)
        ) > 120;
    }
}while(!difFromLast && !difFromOneButLast);
this._colors[number] = new Array();
this._colors[number]['red'] = red;
this._colors[number]['green'] = green;
this._colors[number]['blue'] = blue;
this._colors[number]['rgb'] = ('rgb(' + red + ', ' + green + ', '
    + blue + ')';
```

Zdrojový kód 4-7: Generování náhodných pastelových barev s porovnáváním

4.4 Správa kódu

Celý kód knihovny je komentovaný podle syntaxe JSDoc. K vytvoření dokumentace API jsem použil JsDoc Toolkit, aplikaci napsanou v JavaScriptu pro automatické generování dokumentace.

I přes to, že jsem na projektu pracoval sám a nikoliv v týmu o větším počtu lidí, používal jsem Apache Subversion. Systém napsaný v programovacím jazyce C pro správu verzí zdrojových kódů. Pro integraci do prostředí MS Windows jsem využil TortoiseSVN.

5 Závěr

Práce popisuje novou technologii kreslení ve webovém prohlížeči založenou na elementu canvas, jenž je součástí nejnovější specifikace jazyka pro hypertext HTML5.

5.1 Zhodnocení

Cílem této práce bylo vytvořit aplikaci pro tvorbu grafů ve webovém prohlížeči v jazyce JavaScript. Zadání se mi podařilo splnit, aplikace je plně funkční a nabízí uvedená rozšíření, knihovna je snadno přenositelná a funguje ve všech webových prohlížečích s podporou HTML5.

Aplikace umožňuje dynamicky vytvářet grafy na základě vstupních dat zadaných přímo parametrem nebo soubory typu XML a XLSX. Dovoluje měnit různá nastavení, přizpůsobit rozměry a rozsah vykreslení, měnit barvy datových sad apod.

Práce na aplikaci pro mne byla nesmírným přínosem, neboť jsem si vyzkoušel návrh, implementaci a testování poměrně rozsáhlého samostatného celku kódu s ohledem na přenositelnost a kompatibilitu.

Jádro knihovny pro kreslení, které vzniklo při implementaci praktické části této práce, našlo využití a je používáno ve společnosti Honeywell, Inc. na oddělení Advanced Technology Europe jako součást cloud sysému ATE Tool pro plánování zdrojů.

5.2 Porovnání s ostatními technologiemi

Canvas se ukázal jako velmi dobré 2D plátno pro volné kreslení se stálým výkonem, který je ovlivněn pouze rostoucími rozměry, nikoliv složitostí obrazu. Je vhodné pro tvorbu bitmapové grafiky, do které bych zařadil např. zobrazování fraktálů, práci s fotografiemi, tvorbu jednoduchých her apod. Výsledný obraz lze navíc uložit jako PNG či JPG.

Obrovskou nevýhodou elementu je neznalost vrstev a celkově nemožnost použití DOM pro adresování jednotlivých částí obrázku. Technologie zatím nenabízí mnoho, už vykreslení

i jednoduché animace vzhledem k absenci patřičných API by bylo implementačně zbytečně náročné. Pro tvorbu uživatelských rozhraní a náročnějších interaktivních grafických děl musí programátor sáhnout k jinému, komplexněji vybavenému řešení.

5.3 Další vývoj projektu

Knihovna, která představuje výstup praktické části této práce a splňuje body zadání, stále obsahuje obrovský prostor pro další rozšíření. Vhodná by byla například implementace interaktivnosti s již vykresleným grafem (*tooltips*) nebo odstranění omezení hloubky dat na pouze 2 dimenze (3D grafy). Objektový přístup nabízí snadné a přehledné rozšíření knihovny prakticky libovolným směrem.

Literatura

[1] Graf (funkce). In *Wikipedia : the free encyclopedia* [online]. St. Petersburg (Florida) : Wikipedia Foundation, 30. 10. 2005, last modified on 16. 5. 2011 [cit. 2011-05-16]. URL: http://cs.wikipedia.org/wiki/Graf_(funkce)

[2] W3C Mission [online]. 2009 [cit. 2011-04-02]. W3C Mission. URL: http://www.w3.org/Consortium/mission

[3] *JQuery: The Write Less, Do More, JavaScript Library* [online]. 2010 [cit. 2011-04-02]. JQuery is a new kind of JavaScript Library. URL: http://jquery.com/

[4] *Scalable Vector Graphics (SVG) 1.1 (Second Edition)* [online]. 2010-06-22 [cit. 2011-04-05]. URL: http://www.w3.org/TR/SVG/

[5] *4.8.11 The canvas element — HTML5* [online]. 2010 [cit. 2011-04-05]. HTML5. URL: http://www.w3.org/TR/html5/the-canvas-element.html

[6] *ActionScript - Semantic Web Standards* [online]. 2009-12-04 [cit. 2011-04-07]. ActionScript. URL: http://www.w3.org/2001/sw/wiki/ActionScript

[7] *Types of Graphs: Bar Graph, Scatter Plot, Pie Chart* [online]. 2011 [cit. 2011-04-25]. Types of Graphs. URL: http://www.typesofgraphs.com/

[8] *Canvas tutorial - MDC Docs* [online]. c2011 [cit. 2011-04-07]. Canvas tutorial. URL: https://developer.mozilla.org/En/Canvas_tutorial

[9] RESIG, John. *Pro JavaScript™ Techniques*. [United States of America] : Apress, 2006. 359 s. ISBN 978-1-59059-727-9, ISBN 1-59059-727-3.

[10] ZAKAS, Nicholas C. *Professional JavaScript® for Web Developers, 2nd Edition*. Indianapolis (Indiana) : Wiley Publishing, 2009. 800 s. ISBN 978-0-470-22780-0.

Seznam použitých zkratek

W3C	World Wide Web Consortium
(X)HTML	(Extensible) Hypertext Markup Language
SVG	Scallable Vector Graphics
XML	Extensible Markup Language
CSS	Cascading Style Scheets
WOFF	Web Open Font Format
DOM	Document Object Model
WHATWG	Web Hypertext Application Technology Working Group
SWF	ShockWave Flash
API	Application Programming Interface
ECMA	European Computer Manufacturers Association
JSON	JavaScript Object Notation
PHP	Hypertext Preprocessor
MS	Microsoft Corporation
GPL	General Public Licence

Dodatek A

Obsah CD

Ve složce `/www/` je uložena webová aplikace, ve složce `/www/jquery/` je samotná knihovna `graph.js` pro tvorbu grafů, ve složce `/www/jsdoc/` je dokumentace API a ve složce `/www/source/` se nachází příklady vstupních souborů.

Návod ke zprovoznění aplikace `install.txt` se nachází v kořenovém adresáři.