České vysoké učení technické v Praze
Fakulta stavební

SVG a XSL transformace využitelné v kartografii

BAKALÁŘSKÁ PRÁCE

Praha, 2011 Autor: Filip Zavadil

České vysoké učení technické v Praze
Fakulta stavební

BAKALÁŘSKÁ PRÁCE

SVG a XSL transformace využitelné v kartografii

SVG and XSL transformations used in cartography

Vedoucí práce: Ing. Jiří Cajthaml, Ph.D.

Praha, 2011

Autor: Filip Zavadil

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
Fakulta stavební
Thákurova 7, 166 29 Praha 6

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

studijní program: Geodézie a kartografie
studijní obor: Geoinformatika
akademický rok: 2010/2011

Jméno a příjmení studenta: Filip Zavadil
Zadávající katedra: Katedra mapování a kartografie
Vedoucí bakalářské práce: Ing. Jiří Cajthaml, Ph.D.
Název bakalářské práce: SVG a XSL transformace využitelné v kartografii
Název bakalářské práce v anglickém jazyce: SVG and XSL transformations used in cartography

Rámcový obsah bakalářské práce: Popište základní principy počítačové grafiky včetně jejího vývoje. Zaměřte se na srovnání vektorové a rastrové formy. Dále se zaměřte na jazyk XML a jeho podpůrné technologie. Stěžejní část práce bude věnována popisu jazyka SVG a XSL transformacím, které jsou využitelné v kartografii. Vše dokumentujte vhodnými příklady.

Datum zadání bakalářské práce: 16.2.2011 Termín odevzdání: **13.5.2011**
(vyplňte poslední den výuky příslušného semestru)

Pokud student neodevzdal bakalářskou práci v určeném termínu, tuto skutečnost předem písemně zdůvodnil a omluva byla děkanem uznána, stanoví děkan studentovi náhradní termín odevzdání bakalářské práce. Pokud se však student řádně neomluvil nebo omluva nebyla děkanem uznána, může si student zapsat bakalářskou práci podruhé. Studentovi, který při opakovaném zápisu bakalářskou práci neodevzdal v určeném termínu a tuto skutečnost řádně neomluvil nebo omluva nebyla děkanem uznána, se ukončuje studium podle § 56 zákona o VŠ č. 111/1998. (SZŘ ČVUT čl. 21, odst. 4)

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

..... vedoucí bakalářské práce

..... vedoucí katedry

Zadání bakalářské práce převzal dne: .....

..... student

Formulář nutno vyhotovit ve 3 výtiscích – 1x katedra, 1x student, 1x studijní odd. (zašle katedra)

Nejpozději do konce 2. týdne výuky v semestru odešle katedra 1 kopii zadání BP na studijní oddělení a provede zápis údajů týkajících se BP do databáze KOS.
BP zadává katedra nejpozději 1. týden semestru, v němž má student BP zapsanou.
(Směrnice děkana pro realizaci studijních programů a SZZ na FSv ČVUT čl. 5, odst. 7)

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci na téma „SVG a XSL transformace využitelné v kartografii“ vypracoval samostatně a použil jsem pouze podklady uvedené v přiloženém seznamu.

V Praze dne _____ _____

podpis

Abstrakt

V této práci se pojednává o základech zobrazování počítačové grafiky. Jsou zde shrnuty základní vlastnosti rastrové a vektorové reprezentace obrazových dat. Dále se práce podrobně věnuje škálovatelnému vektorovému grafickému formátu SVG. Jsou zde popsány základní prvky a vlastnosti tvořící SVG dokument. Dále je pojednáno o „stylových“ jazycích (CSS, XSL), které je vhodné spolu s SVG využívat.

Praktická ukázka je celistvým spojením poznatků o SVG. Je zde demonstrováno použití rozlišných prvků tohoto jazyka, který je v této práci konfrontován s kartografií.

Klíčová slova

grafika, vektor, rastr, SVG, XML, CSS, XSL, XPath, JavaScript

Abstract

This thesis shows the basics of computer graphics. There is a summary of key properties of raster and vector image data. The main emphasis is on Scalable Vector Graphics which is explained in detail. In addition, styling languages (like CSS, XSL) are also covered thanks to a close relationship with SVG.

A sample study demonstrates the application of various SVG elements into practice and its confrontation with cartography.

Key Words

graphics, vector, raster, SVG, XML, CSS, XSL, XPath, JavaScript

Obsah

1 Úvod **1**

2 Vývoj před SVG **3**
2.1 Počítačová grafika obecně . . . 3
2.1.1 Geometrické základy . . . 3
2.1.2 Vývoj technologií . . . 3
2.2 Vektorová a rastrová grafika . . . 4
2.2.1 Vektorová reprezentace . . . 4
2.2.2 Rastrová reprezentace . . . 5
2.2.3 Výhody a nevýhody . . . 5
2.3 Vznik a vývoj grafických formátů . . . 5
2.3.1 Formáty vektorové grafiky . . . 6
2.3.2 Formáty rastrové grafiky . . . 7
2.4 Úvod do XML . . . 8
2.4.1 Syntaxe XML . . . 9
2.4.2 XML dialekty . . . 10
2.4.3 XML přidružené technologie . . . 11

3 Formát SVG **12**
3.1 Verze . . . 12
3.2 Kompatibilita . . . 12
3.3 Hlavička SVG dokumentu . . . 13
3.3.1 Element `defs` . . . 13
3.3.2 Barevné přechody . . . 15
3.4 SVG elementy základních geometrických prvků . . . 16
3.4.1 Element `line` (úsečka) . . . 17
3.4.2 Element `rectangle` (obdélník) . . . 17
3.4.3 Element `circle` (kružnice) . . . 17
3.4.4 Element `ellipse` (elipsa) . . . 17
3.4.5 Element `polyline` (lomená čára) . . . 18
3.4.6 Element `polygon` (mnohoúhelník) . . . 18
3.5 Element `path` (křivka) . . . 18
3.6 Textové řetězce . . . 20
3.6.1 Element `text` . . . 20
3.6.2 Element `tspan` . . . 21
3.6.3 Element `textPath` . . . 22
3.7 Instrukce pro vykreslování . . . 22

3.7.1 Vlastnosti atributu `stroke` . . . 22
3.7.2 Vlastnosti atributu `fill` . . . 24
3.7.3 Atributy `opacity,display,visibility` . . . 24
3.7.4 Barevné modely . . . 24
3.8 Souřadnicový systém a jeho transformace . . . 26
3.8.1 Atribut `transform` . . . 26
3.9 Rastr v `SVG` . . . 28
3.9.1 Element `image` . . . 28
3.9.2 Element `filter` . . . 28
3.10 Atributová data . . . 31
3.11 Metadata . . . 31

4 Stylové jazyky **33**
4.1 CSS . . . 33
4.2 XSL . . . 34
4.2.1 Hlavička XSL . . . 35
4.2.2 Element `xsl:value-of` . . . 36
4.2.3 `XPath` výrazy . . . 36
4.2.4 Podmínečné zpracování . . . 38
4.2.5 Cykly . . . 39
4.2.6 Proměnné . . . 39
4.2.7 Další použité prvky . . . 40

5 Aplikace do praxe **41**
5.1 Zdroj dat a způsob exportu do `XML/SVG` . . . 41
5.1.1 Program `shp2svg` . . . 41
5.2 Popis aplikace . . . 42
5.2.1 XML datová část . . . 43
5.2.2 Stylová část . . . 43
5.3 Popis jednotlivých aplikací . . . 43
5.3.1 Evropa . . . 44
5.3.2 Česká republika . . . 44
5.3.3 Plán kampusu ČVUT v Dejvicích . . . 47
5.3.4 Plán města Pelhřimov . . . 48
5.4 Kartografické prvky v `SVG` . . . 48
5.4.1 Vizualizace liniových značek . . . 48
5.4.2 Vizualizace areálových značek . . . 51
5.4.3 Vizualizace bodových prvků . . . 51
5.5 SVG v HTML . . . 52
5.6 JavaScript . . . 52

6 Závěr **54**

Literatura **55**

Rejstřík **56**

A Evropa **I**

B Česká republika **II**

C Plán kampusu ČVUT Praha **III**

D Plán města Pelhřimov **IV**

Seznam obrázků

2.1 Časová osa vybraných klíčových momentů 4
2.2 Popis bézierovy křivky 4
2.3 Znázornění odlišného způsobu záznamu obrazových informací 5
2.4 Ikony vybraných grafických formátů a editorů 8
2.5 Ukázka struktury dokumentu IBM GML 8
2.6 Popis nejjednoduššího XML dokumentu 9
2.7 XML dokument popisující geobod 10
2.8 Stromová struktura XML elementu 10

3.1 Některá z oficiálních log formátu SVG 12
3.2 Podíl používaných internetových prohlížečů ve světě v lednu 2011 13
3.3 Element `path` a příkazy `M, L` pro vykreslování úseček 18
3.4 Element `path` a příkazy `H, V` pro vodorovné a svislé úsečky 19
3.5 Element `path` a příkazy `C,S` pro kubickou Bézierovu křivku 19
3.6 Element `path` a příkaz `arc` pro eliptickou výseč 20
3.7 Přehled základních atributů pro práci s textem 21
3.8 Přehled souřadnicového systému a délkových jednotek 26
3.9 Ukázka použití rastrových obrazových dat 28
3.10 Praktické využití filtrů pro symbol informačních center 31

4.1 Princip XSL transformací 35
4.2 Přehled vstupních a výstupních souboru XSL transformací 36

5.1 Kartogram Evropa, HDP na obyvatele 45
5.2 Ukázka kartografických prvků na mapě ČR 47
5.3 Ukázka některých symbolů vytvořených v SVG 52
5.4 Ukázka autorské funkce v jazyce Javascript 53
5.5 Struktura adresářů na doméně `http://svg.filipzava.cz` 53

Kapitola 1

Úvod

V dnešní době se drtivá většina kartografických prací provádí v digitální formě na počítačích a tato skutečnost s sebou nese problém, jak grafická data ukládat. Rastrový model jako nosič není vhodný, jelikož nezachovává geometrické vyjádření objektů. Proto je využíván vektorový formát, kterých je dnes velké množství. Většina vektorových grafických a kartografických editorů využívá vlastní formát, který je více či méně otevřený. Tato práce popisuje nejznámější a nejpoužívanější rastrové i vektorové formáty s detailním zaměřením na formát SVG *(Scalable Vector Graphics)*. Hlavním důvodem pro výběr SVG je jeho otevřenost a tedy nezávislost na proprietárních aplikacích nebo platformách. Mezi další klady SVG patří jeho jednoznačná kompatibilita s internetovým prostředím, způsobená použitím stejných technologií. V průběhu práce, kdy jsou popisovány jednotlivé prvky SVG jazyka, je uváděn jejich vztah a případné využití pro kartografické účely.

Jak již bylo řečeno v úvodní kapitole, práce se zabývá obecnou problematikou počítačové grafiky. Jsou zde také uvedeny hlavní softwarové firmy a jejich vliv na vývoj grafických formátů. Jelikož formát SVG je dialektem značkovacího jazyka XML, je v práci uveden také úvod do XML popisující jeho vývoj, základní charakteristiky a způsoby použití. Důležité je také seznámení s často využívanými pojmy.

Druhá kapitola je věnována jazyku SVG. Popis je doprovázen autorskými názornými obrázky s uvedeným zdrojovým kódem a vizualizací. Krátce je zmíněno o vývojových verzích a jejich kompatibilitě s dnešními internetovými prohlížeči. Následuje popis všech geometrických prvků, jejich způsobu zápisu a základních vlastností. Vždy je u každého uveden také význam pro kartografické účely. Důležitou součástí je styl vykreslování prvků (barva a tloušťka tahu, styl výplně atd.). Neopomíjeným prvkem je zápis textového řetězce a nastavení jeho vlastností. I když je SVG vektorový formát je schopen ukládat rastrová data a provádět s nimi různé efekty pomocí definovaných filtrů. Díky XML základům je zde nabídnuta možnost ukládat i jiné než obrazové informace. Tato skutečnost nahrává použití SVG jako jednoduchý GIS formát. Možný je také zápis metadat.

Pro určení způsobu vykreslení je vhodné použít takzvané stylové soubory. Jedná se o dokumenty, kde jsou nastaveny pravidla zobrazení SVG geometrických prvků. Mezi dva nejvyužívanější patří CSS a XSL, kdy každý se hodí k trochu jinému účelu. V práci jsou uvedeny základy těchto stylových jazyků, doplněné o příklady použití.

Nejlepší cestou jak porozumět dané problematice je praktické vytvoření SVG aplikace. Protože je SVG navrženo pro web, může být aplikace interaktivní. Použitím dalších technologií (JavaScript) je dodána i dynamičnost. V poslední kapitole jsou mimo jiné uvedeny způsoby získání či exportu dat do SVG formátu.

Jednotlivé poznatky zjištěné v průběhu práce a celkové shrnutí je uvedeno na samotný závěr práce.

Kapitola 2

Vývoj před SVG

2.1 Počítačová grafika obecně

2.1.1 Geometrické základy

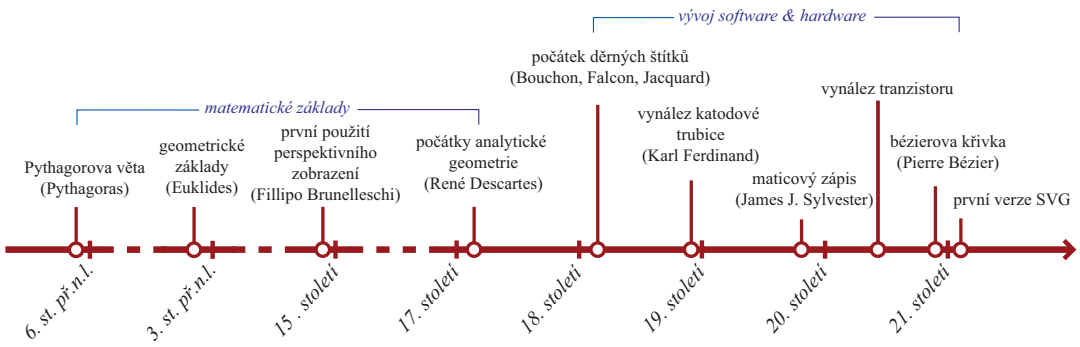
Geometrické základy užívané počítačovým zpracováním grafických dat se datují už 300-250 př.n.l., kdy se jim věnoval řecký matematik a geometr Euklides ve svých spisech. Filippo Brunelleschi (1377-1446) objevil lineární perspektivu, tedy první vizualizaci $3\mathcal{D}$ prostoru. René Descartes (1596-1650) je jeden ze zakladatelů analytické geometrie, zavedl také kartézský systém souřadnic a tím základy pro popis objektů v $2\mathcal{D}$ prostoru. James Joseph Sylvester (1814-1897) popsal maticovou notaci, využívanou v informatice pro geometrické transformace. Pierre Bézier zavedl popis *bézierovy křivky* využívaný ve vektorových grafických formátech.

2.1.2 Vývoj technologií

Od doby, kdy se začaly výpočetní technologie rozšiřovat, zaznamenala oblast počítačového zpracování grafických dat velký pokrok. V roce 1897 byla německým fyzikem Karlem Ferdinandem vynalezena katodová trubice, která dala vznik televizím a CRT monitorům. Tato skutečnost spolu s vynálezem tranzistoru v roce 1947 v Bellových laboratořích dala základ pro rozvoj počítačové grafiky. Mezi další významné milníky patří zavedení pojmu *počítačová grafika* (William Fetter, 1964). O rok později přichází na svět tzv. *Bresenhamův algoritmus*, který popisuje vykreslování čar do rastrového pole (použití v nynějších rastrových monitorech) [Sho00]. Firma Apple Computer v roce 1977 představuje první barevný grafický mikropočítač Apple II. V 80. letech se představuje řada počítačů Apple Macintosh s grafickým uživatelským rozhraním (GUI) a počítačovou myší. Od 90. let, kdy přichází Bill Gates s operačním systémem Windows 3.0, se osobní počítače masivně rozšiřují do pracovišť a domácností.

Dalším mezníkem pro počítačovou grafiku byl vývoj internetu. V roce 1969 byla sestavena první experimentální síť *ARPANET* a o čtyři roky později vyvstaly úvahy vedoucí k nynějšímu protokolu TCP/IP. V roce 1987 je do sítě připojeno 27 000 stanic a vzniká pojem *internet*. Roku 1992 je oficiálně připojena Česká republika a ČVUT v Praze. Zhruba o 2 roky později se internet komercializuje a tím dochází k masovému nárůstu

uživatelů. Po roce 2005 je zaznamenána miliarda uživatelů. Stále více se prosazuje *on-line* obsah i v oblasti kartografie. Vznikají nové grafické formáty zaměřené přímo pro internet.

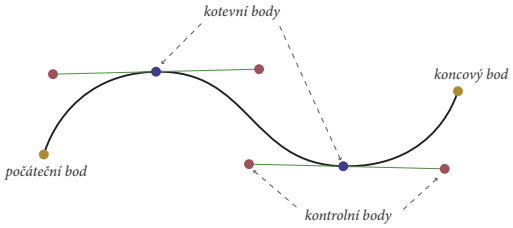


Obrázek 2.1: Časová osa vybraných klíčových momentů

2.2 Vektorová a rastrová grafika

2.2.1 Vektorová reprezentace

Vektorová grafika má základy v analytické geometrii. Ukládají se parametry geometrických těles, jejich poloha, popř. výplň. Francouzský matematik a konstruktér Pierre Bézier popsal křivku pomocí čtyř bodů. Jedná se o *krajní* body a tzv. body *kotevní* a *kontrolní* viz. obr.č. 2.2. Krajními a kotevními body křivka prochází a pomocí kontrolních bodů *(tečny křivky)* lze určovat její zakřivení. Skládáním těchto křivek lze popsat libovolný zakřivený tvar. Složitější výkres je složen z primitivních tvarů tzv. **objektů**. Například jednoduchá kresba sněhuláka je popsána pomocí tří kružnic, kde jsou uloženy jen hodnoty poloměru a souřadnice středů kružnic. Je zjevné, že takto popsaná kresba nebude náročná na paměť počítače a datová velikost souboru bude minimální. Jelikož jsou data popsána matematicky, nedochází při zvětšení výkresu k poklesu kvality. Protože se jednotlivé kresby ukládají jako objekty (bod, úsečka, polygon, křivka atd.), mohou se jednoduše provádět **modifikace** (např. zvětšení poloměru kružnice, translace, atd.), aniž by to mělo vliv na ostatní objekty. Tato struktura je vhodná k uložení dat méně tvarově rozmanitých. Využívá se při tvorbě firemních log, schématických obrázků a diagramů, technických výkresů (CAD) a animací. Díky své malé datově objemové náročnosti je vektorová grafika vhodná pro internet, kde se stále více prosazuje.



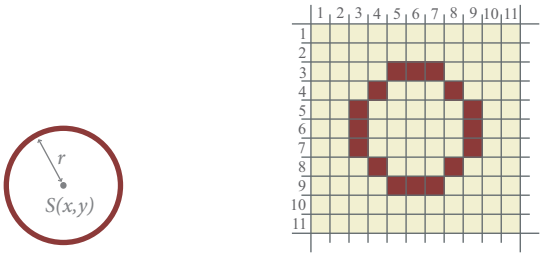
Obrázek 2.2: Popis bézierovy křivky

2.2.2 Rastrová reprezentace

Rastrový (často též *bitmapový*) způsob popisu obrazových informací ukládá informace o jednotlivých prvcích umístěných na **mřížce**. Mřížka (rastr) může být složena z různých tvarů (trojúhelník, obdélník, šestiúhelník), avšak nejčastěji je tvořena čtverci tzv. **pixely**. Každý prvek mřížky nese určitou hodnotu (nejčastěji se jedná o hodnotu barvy). Paměťová zátěž bude logicky silně závislá na rozměru výsledného obrázku. Rastrový popis výše uvedené kresby sněhuláka bude tedy náročnější. Kružnice bude reprezentována vyplněnými pixely a jejich počet bude záležet na obrazovém rozlišení. Aby se kružnice jevila realisticky a nebyla „kostrbatá“, je nutné volit větší rozlišení, čímž ale dojde i ke zvětšení datové velikosti souboru na paměťovém médiu. Fotografie jsou typickým představitelem rastrové grafiky, protože obsahují velký počet rozmanitých a tudíž těžko matematicky popsatelných tvarů. V tomto případě je vektorový model nevhodný. Také metoda sběru dat (osvícení CMOS čipu fotoaparátu) zaznamenává skutečnost do mřížky a nepopisuje ji matematicky. Nad bitmapovými obrázky lze s výhodou aplikovat spoustu filtrů a efektů.

2.2.3 Výhody a nevýhody

Nelze obecně říci, jaký způsob je výhodnější. Vždy záleží na rozsahu a rozmanitosti kresby, zadanému účelu a způsobu sběru dat. Pro znázornění technických výkresů je vhodný vektorový model. Pro výkresy, které se mají blížit realitě je zase vhodnější rastrový model. S nadsázkou si lze přestavit, že Pablo Picasso by pro své kubistické malby zvolil vektorový formát a Claude Monet v období impresionismu by použil vhodnější rastrový formát. V oboru geodézie a kartografie jsou využívány oba typy dat. Např. produktem fotogrammetrie a dálkového průzkumu Země je rastrový obsah. Naopak grafické zpracování geodetického měření je vektorový výkres. Vizuální rozdíly obou způsobů jsou zobrazeny na obr. 2.3



Obrázek 2.3: Znázornění odlišného způsobu záznamu obrazových informací

2.3 Vznik a vývoj grafických formátů

Informace jsou na paměťových mediích ukládána v bitech a tvoří posloupnost jedniček a nul. Formát souboru sděluje procesoru, jak danou posloupnost rozkódovat (čtení) nebo zakódovat (zápis). Pro reprezentaci grafických dat bylo vytvořeno mnoho formátů pro vektorová i rastrová data.

2.3.1 Formáty vektorové grafiky

Každý významnější výrobce softwaru pro zpracování vektorové grafiky si vytváří vlastní výměnný formát, který průběžně inovuje a doplňuje. Vznikají tak problémy se zpětnou kompatibilitou a nezřídka je tato skutečnost řešena vydáním nového formátu. Méně častým jevem je vzájemná **kompatibilita** mezi aplikacemi. Hlavně tvůrci proprietárního softwaru vyvíjejí a bedlivě si střeží vlastní uzavřený formát (tato práce pojednává o otevřeném formátu). Vektorové grafické formáty lze dle [Tiš07] rozdělit na tyto skupiny:

Významnější souborové formáty určené pro $2\mathcal{D}$ grafiku a CAD

- **CDR** je složitější proprietární formát vyvíjený společností Corel Corporation, je využívaný v softwaru CorelDRAW. Mezi jeho odvozeniny patří: CDX, CDT, CMX. Formát podporuje vícestránkový dokument.
- **AI** nativní formát aplikace Adobe Illustrator je velmi podobný formátu PDF. Podporuje vrstvy, ale nepodporuje vícestránkový dokument ani animace.
- **DXF** (Drawing eXchange Format) je textový formát vyvinutý společností AutoDesk. Tento formát podporuje většina CAD systémů. Jeho binární podoba má příponu DXB.
- **DWG** (DraWinG) je dalším CAD/CAM formátem společnosti AutoDesk. Umožňuje ukládat $2\mathcal{D}$ i $3\mathcal{D}$ data.
- **DGN** (DesiGN) je nativní formát CAD programu MicroStation od Bentley Systems.
- **SHP** (SHaPefile) je původní binární formát softwaru ESRI ArcMap. Popisuje geometrické prvky (point, line, polygon) a se soubory typu *SHX* a *DBF* tvoří geodatabázi. Jedná se o nejrozšířenější formát pro **GIS** aplikace.
- **OCD** je interní formát kartografického softwaru OCAD.

Formáty určené pro tisk a metaformáty

- **PS** (PostScript) je jeden z výměnných tiskových formátů vyvinutý společností Adobe Systems (1985). Jde o popis dokumentu pomocí programovacího jazyka.
- **EPS** (Encapsulated PostScript) je odvozenina *PS* formátu určená pro popis obrázku.
- **PDF** (Portable Document Format), navržený společností Adobe je na platformě nezávislý výměnný formát založený na jazyce *PostScript* a v současné době velmi rozšířen. Formát je schopen nést i rastrová obrazová data a metadata aplikací, ve kterých byl vytvořen. Zejména produkty firmy Adobe jsou schopné si v tomto formátu uchovat i nastavení, vrstvy nebo objekty využitelné pro pozdější editaci.
- **WMF, EMF** (Windows MetaFile, Enhanced Windows MetaFile) je vyvíjený společností Microsoft pro operační systém MS Windows. Ukládá geometrické objekty i vybrané příkazy.

- **SWF** (Shockwave Flash) je proprietární multimediální formát obsahující nejen vektorové prvky. Byl vyvíjen společností Macromedia (nyní Adobe Labs), která také vydává editor Adobe Flash. I přes svou nesvobodu je silně rozšířen na webovém prostředí a je konkurentem otevřeného formátu SVG.

Grafické vektorové formáty založené na XML technologii

- **SVG** (Scalable Vector Language) je předmětem této práce. Bližší seznámení od kapitoly 3. *W3C* standardem se stal v roce 1999.
- **VML** (Vector Markup Language) je formát starší než SVG, který se ale příliš neprosadil. Jeho přípony jsou `.htm` nebo `.html`. W3C Standard v roce 1998.
- **PGML** (Precision Graphics Markup Language) je další předchůdce SVG v dnešní době již nevyužívaný. W3C Standardem se stal v roce 1998.

2.3.2 Formáty rastrové grafiky

Rastrové formáty se od sebe rozlišují použitými kompresními algoritmy (LZW, JPEG, RLE, Huffmanovo kódování).

Nejběžnější rastrové formáty

- **PNG** (Portable Network Graphics) je nástupce *GIFu*. Používá bezztrátovou kompresi a jeho první verze vyšla v roce 1996 (Standard W3C). Jelikož podporuje transparentní pozadí je hojně používaný v internetovém prostředí.
- **JPEG** (Joint Photographic Experts Group) je ztrátový komprimační algoritmus a také formát souboru. Je vhodný k použití u obrázků, kde se nevyskytují ostré hrany (narozdíl od formátu PNG). Je klasickým výstupem digitálních fotoaparátů.
- **BMP** (Microsoft Windows Bitmap), jeho první verze byla představena v roce 1988. Obrázky jsou ukládány po jednotlivých pixelech a proto je souborová velikost značně větší než u předchozích formátů.
- **TIFF** (Tag Image File Format) používá libovolnou dostupnou kompresi a je schopen ukládat vícestránkové dokumenty. Vznikl jako vstupní formát pro scannery a faxové přístroje. Jeho varianta GeoTIFF umožňuje nosit také prostorová data.
- **GIF** (Graphics Interchange Format) využívá bezztrátovou kompresi *LZW84*. Kvůli tomuto komprimačnímu algoritmu, který byl patentově chráněn, je nahrazen formátem PNG. Je ovšem schopen animace, proto byl dříve hojně využíván na internetu v reklamních bannerech.

Formáty rastrových editorů

- **PSD** je nativní formát aplikace Adobe Photoshop. Je schopen ukládat metadata jako vrstvy, kanály, cesty, nastavení průhlednosti atd.
- **XCF** je původní formát freewarového programu GIMP (GNU Image Manipulation Program). Uchovává podobné informace jako předchozí PSD.

Obrázek 2.4: Ikony vybraných grafických formátů a editorů

2.4 Úvod do XML

Při vývoji výpočetní technologie vyvstal problém jak stroji sdělit význam jednotlivých částí textu a jak je v textu označovat. Za analogovou formu lze považovat ruční vpisování poznámek mezi autorův text třeba pro potřeby formátování obsahu pro tisk. Právě zde byl inspirován vývoj tzv. **značkovacích jazyků**. Obsah dokumentu tím získal **hierarchickou strukturu** a mohl být použit k více účelům. Také mohl být jednodušeji transformován pro jiná zařízení a stával se tedy kompatibilním prostředkem mezi systémy. Jelikož je předmětem této práce formát SVG, který je odvozen z jazyka XML s pevným *DTD*, budou v této kapitole popsány základní principy a syntaxe jazyka XML.

IBM GML lze považovat za první značkovací jazyk. Vytvořen byl v 60-tých letech minulého století v IBM laboratoří. Za jeho vývojem stáli pánové Charles Goldfarb, Edward Mosher a Raymond Lorie[1]. Výsledný dokument byl `ASCII` text s hierarchickým popisem obsahu. Příklad jeho zápisu je na ukázce č. 2.5.

```
:h1.Kapitola 1: Uvod
:p.Znacka uvozujici odstavec
:oil
:li.prvni polozka seznamu
:li.druha polozka seznamu
:eol.
:p.Nazvy tagu jsou dodnes vyuzivany.
```

Obrázek 2.5: Ukázka struktury dokumentu IBM GML

Dalším produktem založeným na GML je značkovací jazyk **SGML** (Standard Generalized Markup Language) vyvinutý jako univerzální jazyk pro výměnu informací mezi počítači, který umožňoval definici vlastních značek. Byl přijat jako ISO norma (8879) v roce 1986. Instrukce pro označení obsahu jsou vloženy mezi tzv. **řídící znaky** (ostré závorky „`<,>`“). Poprvé byl použit *DTD* (Document Type Definition), kde jsou uvedeny definice dostupných *elementů* (značek). Jelikož se jedná o bezkontextový jazyk[2], jednotlivé elementy se nemohou překrývat, ale mohou se vnořovat. Rozšíření tohoto jazyka pomohlo americké ministerstvo obrany, které požadovalo od dodavatelů dokumentaci právě v tomto formátu [Kos00]. Jazyk SGML byl příliš komplexní (umožňoval nejen definici vlastních značek, ale i určení oddělovačů těchto značek) a to ho inhibovalo v praktickém využití.

[1] V souvislosti s GML je zajímavá zkratka utvořená z počátečních písmen příjmení autorů.

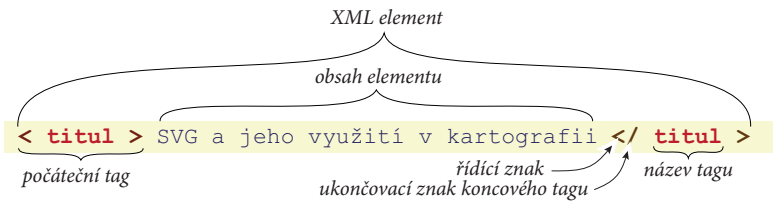
[2] Formální jazyk využívaný především pro programování. Umožňuje sjednocení, substituci, iteraci, morfismus, ale ne průnik ani rozdíl.

Jako jedna z nejznámějších odvozenin je značkovací jazyk *HTML* (HyperText Markup Language) dodnes využívaný jako nositel obsahu pro webové prostředí. HTML má pevně definované DTD, které bylo s každou novou verzí mírně odlišné. Tato skutečnost vedla k tomu, že se potenciál HTML vyčerpal.

XML *(Extensible Markup Language)* je rozšiřitelný značkovací jazyk, který si jako podmnožina jazyka SGML zachovává definici vlastních DTD, ale oddělovače znaků a speciální znaky jsou pevně určené. Také syntaxe jazyka musí dodržovat přísnější pravidla než jazyk SGML. Verze XML 1.0 byla dokončena v roce 1998 a nyní je vydána její 5. revize. Verze XML 1.1 byla představena v roce 2006 a nyní je ve druhé revizi. Obě verze se mezi sebou mírně liší v požadavcích na použité znaky v názvech elementů a atributů. Verze 1.0 dovoluje použití znaků obsažených v Unicode 2.0. Formát XML by měl být nositelem „čistých“ informací, tedy oddělených od prvků určujících vzhled výsledného dokumentu. K tomu účelu jsou určeny *stylové jazyky* viz. kapitola 4. Asi neznámější z nich je **CSS** (Cascade Style Sheet, kaskádové styly), které definují vzhled webových stránek a jsou také využívány u formátu SVG. Dalším, avšak složitějším jazykem, je **XSL** (eXtensible Style Language), který se využívá při takzvaných XSL transformacích (XSLT) pro úpravu a transformace dokumentu. XSLT bude také předmětem této práce v podkapitole 4.2. Jak již bylo řečeno výše, formát XML striktně dodržuje pravidla zápisu. Pro kontrolu slouží program nazývaný *parser* . Tento program kontroluje správnost dokumentu a odhalí chyby, které by mohly být příčinou problémů. Implicitně ho obsahují dnešní internetové prohlížeče.

2.4.1 Syntaxe XML

Pro lepší orientaci v kódu SVG je vhodné znát XML syntaxi a pravidla jeho zápisu, která musí být narozdíl od HTML striktně dodržována. XML dokument je sestaven z objektů nazývaných **elementy** . Element je nositelem samotné informace i jejího označení. Každý element se skládá z *tagů* , které jsou uzavřeny do ostrých závorek (tj. mezi znaky ‘<’ a ‘>’). Celý element se skládá z počátečního tagu, obsahu elementu a koncového tagu, který je uvozen znakem ‘/’ (pro odlišení od počátečního). Jednotlivé části nejjednoduššího XML dokumentu, který obsahuje pouze jeden element, jsou popsány na obr. 2.6. Často



Obrázek 2.6: Popis nejjednoduššího XML dokumentu

nastane situace, kdy element nenese žádný obsah, ale jeho výskyt může mít vliv třeba na formátování. Způsoby, jak takový element napsat jsou tyto:

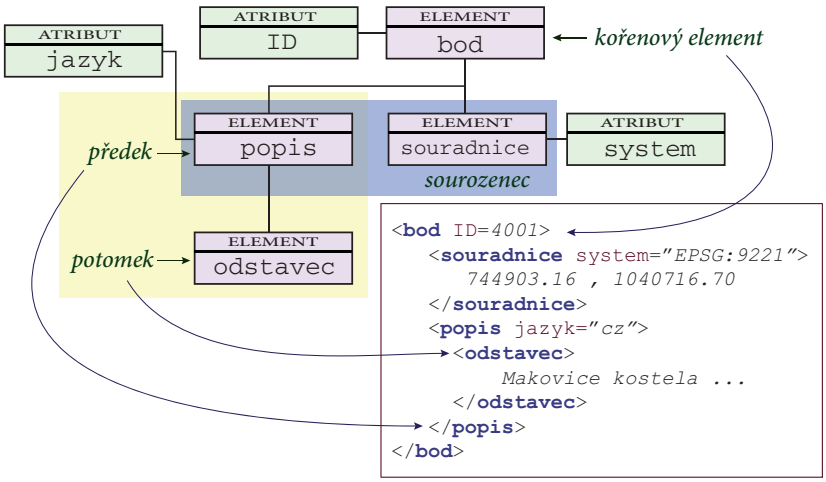
`<titul> </titul>` - prosté vynechání obsahu je v souladu s pravidly
`<br/>` - používá se v XHTML pro ukončení řádky (break line)

`<text kodovani="utf-8"/>` - prázdný element, obsahující pouze atribut, se také často využívá

V poslední ukázce se vyskytuje atribut, který se zapisuje do počátečního tagu. Je to nositel informace o daném tagu. Každý atribut se skládá z názvu a hodnoty, která je uzavřena do uvozovek. V ukázce 2.7 jsou hodnoty souřadnic bodu v souřadnicovém systému S-JTSK. Zvolený souřadnicový systém i číslo bodu jsou ukládány jako atributy. Způsob zápisu komentáře: `<!-- text komentáře je parserem ignorován -->`.

```
<bod cislo="4001">
        <souradnice system="EPSG:5221">
                744903.16 , 1040716.70
        </souradnice>
</bod>
```

Obrázek 2.7: XML dokument popisující geobod



Obrázek 2.8: Stromová struktura XML elementu

2.4.2 XML dialekty

Jazyků odvozených z XML je opravdu mnoho[3], zde budou uvedeny ty nejvyužívanější z nich a ty, které mají něco společného s oborem geodézie a kartografie.

- **XHTML** (XML HyperText Markup Language)
  Značkovací jazyk nahrazující HTML (SGML) je postaven na XML základech. Je přísnější na syntaxi a umožňuje definici vlastních sad značek. Jeho použití se doporučuje společně s CSS (Cascade Style Sheet).

- **DocBook**
  Odvozenina XML určená pro psaní nejen technické dokumentace. Používá se také

[3] `http://en.wikipedia.org/wiki/List_of_XML_markup_languages`

pro psaní knih, prezentací, článků atd. Obsah lze snadno (podle šablon) transformovat pro různé účely použití.

- **MathML** (Mathematical Markup Language)
  Matematický značkovací jazyk popisuje matematické vztahy a vzorečky. Je také uložen logický smysl výrazu. MathML podporují kancelářské balíky MS Office, OpenOffice, matematický software Mathematica a webové prohlížeče.

- **SVG** (Scalable Vector Graphics)
  Škálovatelný vektorový formát určený nejen pro internet. O jeho výhodách, způsobech použití a praktických ukázkách pojednává kapitola 3.

- **KML** (Keyhole Markup Language)
  Formát KML[4] je standardem Open Geospatial Consortium (OGC[5]) a nositelem geografických dat. Využívají ji Google aplikace (Earth, Maps, SketchUp). Díky standardizaci se využívá i v GIS aplikacích. Souřadnice jsou v systému WGS-84 a volitelná výška je v systému EGM96.

- **GML** (Geographic Markup Language)
  Formát ukládá geografické prvky jako geometrii (Point, LineString, Polygon), použitý referenční systém, topologii, čas, použité jednotky měření, observace atd. Formát je také vyvíjen a udržován konsorciem OGC.

- **GeoRSS**
  Geografický formát určený pro *RSS* čtečky, tzv. „Web feed“ služby. Nové zprávy mohou tak nést i geografickou lokalizaci. GeoRSS je také projektem organizace OGC.

- **GPX** (GPS eXchange Format)
  Do tohoto formátu jsou ukládány data některých GPS přijímačů (Garmin) a popisují trasy.

2.4.3 XML přidružené technologie

XML dokumenty mohou být mezi sebou provázány použitím technologie odkazů. Jazyky XPath, XPointer a XLink nabízejí větší možnosti adresování než příkaz `<a href="http://svg.filipzava.cz" > SVG </a>` využívaný v HTML. Tyto jazyky také definuje a standardizuje konzorcium W3C. Aplikace, která bude shrnujícím výsledkem této práce, bude využívat i tyto moduly jazyka XML. Více o jazyku XPath v 4.2.3.

- **XLink**

- **XPointer**

- **XPath**
  XML Path Language se používá pro prohledávání XML dokumentu, který je reprezentován stromovou hierarchií. Jednotlivé elementy tvoří uzly, po kterých je snadné se s XPath pohybovat. Používají se pojmy jako předek a potomek.

[4] `http://code.google.com/intl/cs/apis/kml/documentation/kmlreference.html`
[5] `http://www.opengeospatial.org/`

Kapitola 3

Formát SVG

3.1 Verze

Vektorový formát SVG je vyvíjen od roku 1998. Už od prvopočátku byl koncipován, aby se stal internetovým standardem. Ve vývojovém týmu zasedali zástupci významných firem (Adobe, Corel, Macromedia, Sun, Microsoft, Hewlett-Packard, Apple, IBM). Jeden z rozhodných okamžiků nastal v září 2001, kdy verze `SVG 1.0` získala doporučení od konsorcia W3 (předstupeň standardizace). Standardizován byl formát `SVG 1.0` v dubnu 2000. O několik měsíců později (leden 2001), vyšla verze `1.1`, která bere v potaz expanzi internetu na rozvíjející se mobilní zařízení. Pro vzájemnou kompatibilitu mezi těmito zařízeními bylo `SVG` rozvětveno na formát pro mobilní zařízení `SVGB (SVG Basic)` a odvozeninu `SVGT (SVG Tiny)` pro mobilní telefony (nezahrnuje průhlednost, CSS styly nebo gradient). V době psaní této publikace se verze nacházela ve stádiu `SVG 1.2` a je očekáváno vypuštění `SVG 2.0`. V dnešní době nabírají na trhu svůj podíl tablety, které svým dotykovým displayem umožňují přirozenější ovládání aplikací, tedy i využití interaktivity SVG. V těchto zařízení je většinou zabudována podpora formátu `SVG Full` [Eis02].

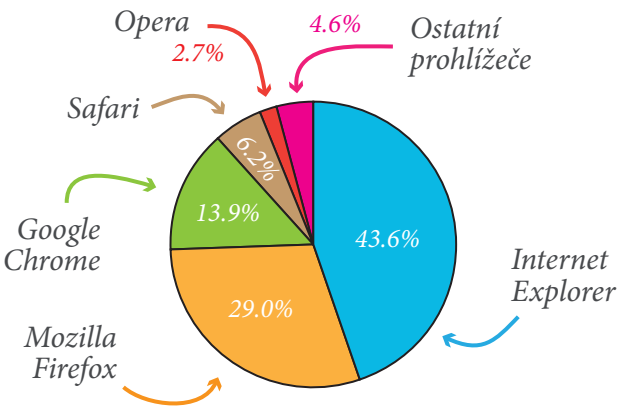
Obrázek 3.1: Některá z oficiálních log formátu SVG

3.2 Kompatibilita

V počátečních letech byl `SVG` formát podporován významnou společností Adobe, která se nejspíš díky tomuto formátu chtěla stát konkurentem flash formátu od Macromedia. Pro zobrazování SVG v prohlížečích, které v té době neměly nativní podporu tohoto formátu, byl využíván plugin Adobe SVG Viewer. Jeho vývoj byl však ukončen v roce 2009 (vzrůstající nativní podpora prohlížečů).

Mozilla Firefox zavedla plnou podporu `SVG 1.1` ve svém jádru *Gecko* v roce 2005. **Opera** má plnou podporu `SVG Tiny 1.1` od verze 8. Prohlížeče založené na WebKit

(**Google Chrome, Apple Safari**) částečně podporují `SVG 1.1` od roku 2006. Zatím celosvětově nejvíce rozšířený prohlížeč **Internet Explorer** od Microsoftu zavedl nativní podporu teprve v roce 2011 ve verzi IE9.



Obrázek 3.2: Podíl používaných internetových prohlížečů ve světě v lednu 2011

3.3 Hlavička SVG dokumentu

Tak jako správně formovaný XML dokument by i SVG obrázek měl obsahovat hlavičku. V této části jsou uvedeny povinné prvky jako *(procesní instrukce, definice typu dokumentu)* a kořenový element `svg`.

procesní instrukce společné pro všechny xml dokumenty

```
<?xml version="1.0" encoding="utf-8" ?>
```

definice typu dokumentu 1.interní, 2. externí zdroj

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

kořenový element, uvedená verze SVG a zdroj jmenného prostoru, všechny ostatní elementy musí být mezi těmito tagy

```
<svg version="1.1" xmlns="http://www.w3.org/2000/svg">

        ... obsah ...

</svg>
```

Je vhodné použít i nepovinné elementy pro označení obsahu, tzv. *metadat* (viz. 3.11) informujících o autorovi, formátu a zdroji dat. O zápisu dalších elementů ve vztahu ke kartografii je pojednáno v 5.4.

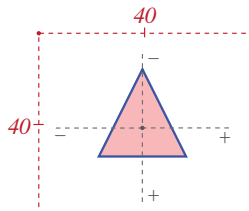
3.3.1 Element `defs`

Do elementu `defs` lze zapsat části kódu, které mohou být v dokumentu několikrát použity. Jedná se zejména o definování barev, barevných přechodů, vzorů atd. Tímto způsobem lze také předávat část kódu. Tato skutečnost se hodí také u rozmísťování kartografických značek, kdy samotný kód je uveden pouze jednou a následně se jen umísťuje na zadané souřadnice. To vede k úspoře datového objemu dokumentu a přispívá k přehlednosti kódu.

- **atribut** `id`
  Pro označení objektu slouží atribut `id`, jehož hodnota by v dokumentu měla být unikátní, aby nedošlo ke kolizi. Podle tohoto označení lze prvek kdykoliv zavolat a inicializovat jeho vykreslení. Identifikátor lze zapsat ke každému elementu geometrických tvarů, textové složce nebo rastrovému obrázku.

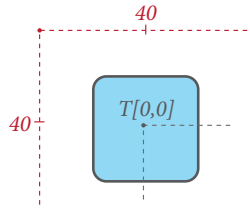
- **element** `use`
  Pro vykreslení symbolu v určité části dokumentu je použit element `use` s atributy v podobě souřadnic (`x,y`) a odkazu k symbolu `xlink:href`, jehož hodnota je `id` symbolu uvozené znakem '`#`'.

```svg
<defs>
   <path id="trojuhelnik" stroke="blue" fill="red"
         d="M-10,10 L0,-10 L10,10 L-10,10" />
</defs>
...
<use x="40" y="40" xlink:href="#trojuhelnik" />
```



- **element** `symbol`
  Mezi tagy s názvem `symbol` se zapisuje kód objektů určených pro vícenásobné použití. Lze zapsat libovolný element geometrických tvarů a textu. Ke každému symbolu je nutné také přidat atribut `id`, který vede k jednoznačné identifikaci symbolu. Atribut `overflow` (nastavený na `visible`) zajišťuje vykreslení celého prvku. Objekt je zapsán relativními souřadnicemi, proto je vhodné těžiště objektu umístit na počátek soustavy souřadnic. Tímto způsobem lze vytvářet a používat kartografické značky. Podobného výsledku lze dosáhnout použitím identifikátoru pro skupinu : `<g id="znacka" >` ... *obsah značky* ... `</g>`

```svg
<defs>
   <symbol id="znacka" overflow="visible">
      <rect x="-20" y="-20" width="40" height="40"
            r="20" fill="#A2DFF4" stroke="black"
            stroke-width="2" rx="5"/>
   </symbol>
</defs>
...
<use x="40" y="40" xlink:href="#znacka" />
```



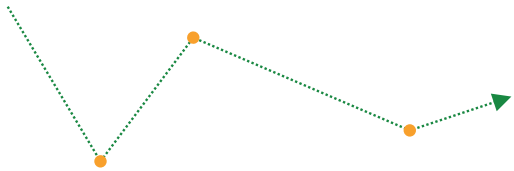
- **element** `marker`
  Pro tvary elementů `line, polyline, polygon` a `path` je k dispozici nastavení počáteční, střední nebo koncové značky pomocí elementu `marker`. Typické použití je znázornění směru šipkou. Atributy, které lze využít pro formátování jsou:

  - `id` - identifikátor, povinný parametr
  - `refX, refY` - souřadnice referenčního bodu
  - `markerUnits` - definice souřadnicového systému hodnoty: *strokeWidth, userSpaceOnUse*
  - `markerWidth, markerHeight` - velikost značky v závislosti na SS

- `orient` - nastavení orientace značek, hodnota ‘auto’ natočí značku automaticky dle tečny ke zvolenému bodu, lze také zadat úhel pootočení

```
<defs>
    <marker id="sipka" refX="0" refY="0"
            markerUnits="strokeWidth"
            orient="auto" overflow="visible">
         <polygon points="-5,-5 -5,5 5,0 -5,-5" fill="green" />
    </marker>

    <marker id="bod" refX="0"  refY="0"
            markerUnits="strokeWidth"
            overflow="visible"/>
        <circle cx="0" cy="0"  r="2" />
    </marker>
</defs>
          ...
<path d="M20,70 L50,20 80,60 150,30 180,40" stroke="green"
     marker-end="url(#sipka)" marker-mid="url(#bod)"  />
```



Obsah tohoto elementu tvoří samotný objekt určený k vykreslení. Pro použití značky slouží atributy:

- `marker-start` počáteční značka
- `marker-mid` značka lomových bodů
- `marker-end` značka koncového bodu

Hodnota je odkaz na danou značku ve tvaru `"url(#adresa)"`.

3.3.2 Barevné přechody

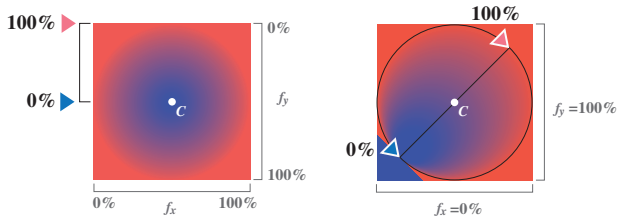
Plynulý přestup jedné barvy do druhé se nazývá „*barevný přechod*“ **(gradient)**. SVG nemá žádný gradient nastavený a uživatel si musí definovat vlastní. Samotná definice je umístěna do hlavičky dokumentu v elementu `defs`. Existují dva druhy barevných přechodů: *lineární* a *radiální* přechod.

- **element** `linearGradient`
  Přechod je nutné pojmenovat atributem `id`. Následují atributy `x1,y1,x2,y2` udávají (v procentech) směr přímky, podle které je barevný přechod veden. Obsahem elementu `linearGradient` jsou barevné body `<stop>`, které mají nastaven odstup (`offset`), barvu v atributu (`style`) hodnotou (`stop-color`) nebo neprůhlednost (`stop-opacity`).

```
<defs>
     <linearGradient id="modro_cervena"
                     x1="0%" y1="0%" x2="100%" y2="0%">
      <stop offset="0%" style="stop-color:blue; stop-opacity:1"/>
      <stop offset="100%" style="stop-color:red; stop-opacity:1"/>
     </linearGradient>
</defs>
          ...
<rect width="400" height="50" r="20" fill="url(#modro_cervena)"/>
```

- **element** `radialGradient`
  Podobné atributy jako lineární přechod má i element `radialGradient`. Je ale nastaven atributy `cx,cy` (souřadnice středu) a `fx,fy`, které po osách posunují barevné body.

```
<defs>
     <radialGradient id="modro_cervena" cx="50%" cy="50%"
                     r="50%" fx="50%" fy="50%">
      <stop offset="0%" style="stop-color:blue; stop-opacity:1"/>
      <stop offset="100%" style="stop-color:red; stop-opacity:1"/>
     </radialGradient>
</defs>
          ...
<rect width="200" height="200" fill="url(#modro_cervena)"/>
```



Barevné přechody lze použít jako barvu výplně (`fill`), barvu tahu (`stroke`) nebo barvu textu. Na definovaný gradient je odkázáno použitím jeho identifikátoru:
`fill="url(#prechod)"`;

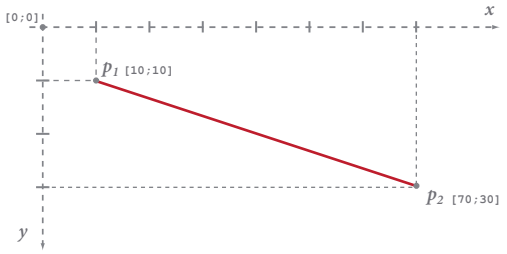
Mezi další prvky, které se zapisují do hlavičky `defs` patří typografické definice. Tyto prvky nejsou v kartografii příliš používané, proto také nebudou podrobněji zmiňovány.

3.4 SVG elementy základních geometrických prvků

Jako u většiny formátů a editorů vektorové grafiky je výsledný obrázek složen ze základních prvků jako jsou obdélníky, kružnice, křivky nebo úsečky. Tyto prvky jsou matematicky popsány dle parametrů nutných k jejich sestrojení. V následujících podkapitolách jsou popsány tzv. „SVG primitiva“. Je také uvedena jejich kartografická hodnota (způsob jejich využití pro zákres map). Stylová složka (barva a tloušťka čáry, barva výplně atd.) je řešena v kapitole 4.

3.4.1 Element `line` (úsečka)

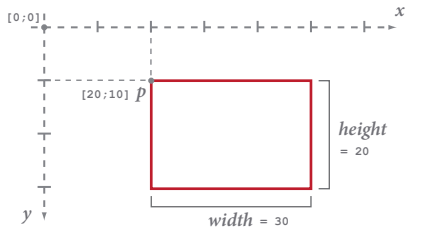
```
<line x1="10" y1="10" x2="70" y2 ="30"/>
```



Jeden z nejjednodušších tvarů je úsečka. Je popsána pouze počátečním `(x1,y1)` a koncovým bodem `(x2,y2)`. V kartografii tento tvar sám o sobě nenachází velkého uplatnění, je ale využíván jako základní prvek křivek, kde tvoří spojnici mezi lomovými body. Prvek je často zastupován elementem `path`.

3.4.2 Element `rectangle` (obdélník)

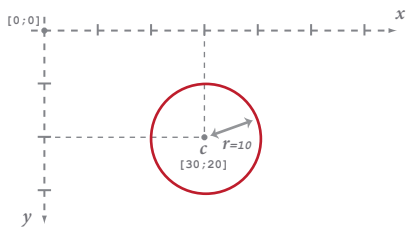
```
<rectangle x="20" y="10"
           width="30" height="20"/>
```



Obdélník je tvořen elementem `rectangle`. Pro sestrojení jsou udány souřadnice severozápadního rohu `(x,y)` a šířka s výškou tělesa `(width,height)`. V mapě může být tento prvek použit u zákresu budov obdélníkového tvaru. Ve většině případech by však musel být pootočen. V aplikaci tvoří tabulky.

3.4.3 Element `circle` (kružnice)

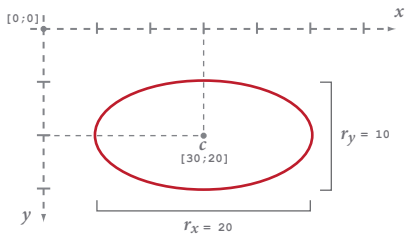
```
<circle cx="30" cy="20" r="10" />
```



Další lehce popsatelný tvar je kružnice. Mezi její parametry patří souřadnice středu `(cx,cy)` a poloměr `(r)`. V kartografii se využívá nejspíše jako kruh (s vyplněním) při značení bodových prvků (např. bodová vrstva obcí). K tomuto účelu je také element použit v aplikaci.

3.4.4 Element `ellipse` (elipsa)

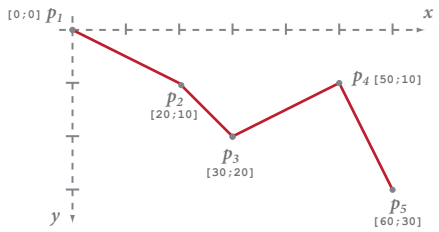
```
<ellipse cx="30" cy="20"
         rx="20" ry="10" />
```



Podobný element jako předcházející kružnice je `ellipse`. Je zřejmé, že kružnice je zvláštní případ elipsy (stejné poloosy). V mapě může být použit k podobným účelům jako kruh. Oproti `circle` je přidán attribut `(rx,ry)`, což jsou délky poloos. Tento element se v aplikaci nenalézá.

3.4.5 Element `polyline` (lomená čára)

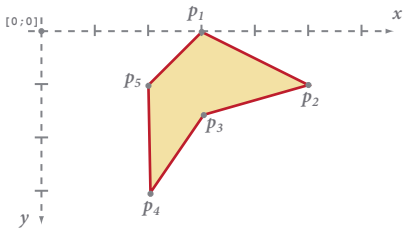
```
<polyline points=" 0,0  20,10 30,20
                   50,10 60,30" />
```



Lomená čára je složena z úseček a je popsána pomocí lomových bodů v atributu `points`. Formát zápisu obsahuje vždy páry souřadnic (x, y) oddělených čárkou. V kartografii může být využit ke znázornění linií jako jsou komunikace, vodní toky, železnice atd. V aplikaci je tento prvek použit u liniových prvků.

3.4.6 Element `polygon` (mnohoúhelník)

```
<polygon points="30,0  50,10 30,15
                 20,30 20,10" />
```



Jedním z prvků pro vykreslování mnohoúhelníků je element `polygon`. Je reprezentován posloupností párů souřadnic, stejně jako element `polyline` v atributu `points`. Tento objekt je vhodný pro areálové (plošné) značky, jako lesy, vodní plochy, budovy atd.

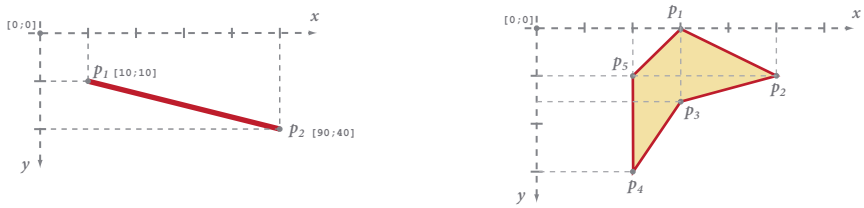
3.5 Element `path` (křivka)

Trochu složitějším a zároveň obecnějším popisem geometrického prvku je křivka (`path`). Pomocí tohoto elementu je možné popsat výše uvedené tvary, ovšem ve složitější a datově objemnější formě. Tento prvek využívá tvz. *virtuálního pera*, které se řídí příkazy uvedenými v atributu s názvem `d` (data). Za příkazy následují páry souřadnic pro udání pozice. Pokud je příkaz velkým písmenem, udává absolutní souřadnice, pokud je zapsán malým písmenem, souřadnice jsou chápány jako relativní (přírůstky k předchozímu bodu). Dostupné příkazy pro element `path`:

- `M` **(move)** - virtuální pero je přesunuto na danou pozici, aniž by byla kreslena čára
- `L` **(line)** - z předchozího bodu je vedena úsečka na souřadnice uvedené za tímto příkazem

```
<path d="M10,10 L90,40" />
```

```
<path d="M30,0  l20,10 L30,15
         l-10,15  L20,10 Z" />
```



Obrázek 3.3: Element `path` a příkazy `M`, `L` pro vykreslování úseček

Jak lze odvodit z obrázku, tímto způsobem lze nahradit elementy `line, polyline, rectangle, polygon, circle` a `ellipse`, (kresba mapy by šla provést za použití pouze toho typu elementu, ovšem datová objemová náročnost a přehlednost by značně utrpěla). Na obrázku vpravo jsou použity absolutní i relativní souřadnice.

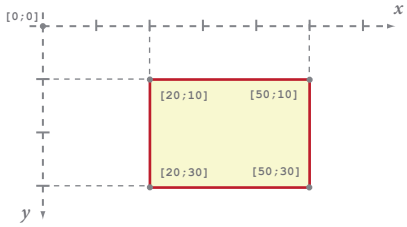
- `H` **(horizontal line)** - vodorovná čára, není zadávána *y-ová* souřadnice
- `V` **(vertical line)** - svislá čára, není zadávána *x-ová* souřadnice, zápis tvaru obdélníku či čtverce je tímto způsobem sice kratší, ale méně přehledný pro editaci

Zápis absolutními souřadnicemi:

```
<path d="M20,10 H50 V30 H20 V10" />
```

Analogický zápis s relativními souřadnicemi:

```
<path d="M20,10 h30 v20 h-30 v-20" />
```

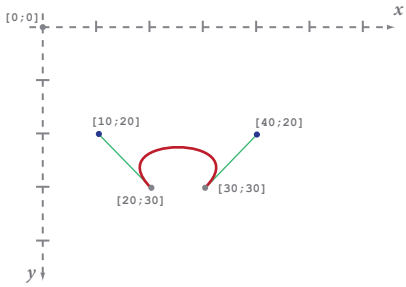


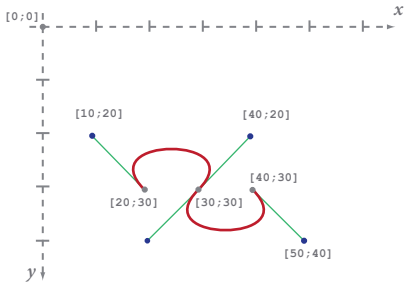
Obrázek 3.4: Element `path` a příkazy `H, V` pro vodorovné a svislé úsečky

- `C` **(curve)** - nástroj pro vykreslování kubické Bézierovy křivky
- `S` **(smooth curve)** - nástroj pro vykreslování kubické Bézierovy křivky s hladkým napojením, tento typ elementu `path` je vhodný pro vykreslení zaoblených spojitých

```
<path d="M20,30
         C10,20 40,20 30,30" />
```

```
<path d="M2 0,30
         C10,20 40,20 30,30
         S50,40 40,30" />
```



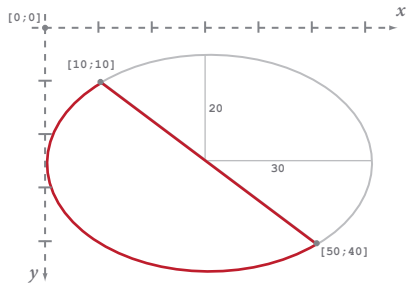


Obrázek 3.5: Element `path` a příkazy `C,S` pro kubickou Bézierovu křivku

křivek, parabol a hyperbol, využívá kotevní body a určení tečen řídícími body, složitější tvary jsou utvořeny spojením více kubických Bézierových křivek formou hladkého napojování

- `Q` **(quadratic Béziere curve)** - sestrojení kvadratické Bézierovy křivky
- `T` **(smooth quadratic Béziere curve)** - sestrojení kvadratické Bézierovy křivky s hladkým napojením, kvadratické Bézierovy křivky nejsou ve vektorových editorech běžně využívány, pro uspokojivou aproximaci vyhovují kubické Béz. křivky

```svg
<path d="M10,10 A30,20 0 0,0 50,40 Z" />
```



A	
30,	*hlavní polosa*
20	*vedlejší polosa*
0	*rotace (stupně)*
0,	*kratší/delší výseč (0/1)*
0	*orientace výseče (0/1)*
50,	*x - koncový bod*
40	*y - koncový bod*

Obrázek 3.6: Element `path` a příkaz `arc` pro eliptickou výseč

- `A` **(elliptical arc)** - eliptický oblouk, prvek má sice složitější zápis, ale mohl by být využíván pro tvorbu diagramů
- `Z` **(close path)** - příkaz pro označení konce křivky

3.6 Textové řetězce

Jedním ze stěžejních prvků vektorových grafických formátů je možnost implementace textových řetězců. Ani formát SVG nezůstává v tomto ohledu pozadu a umožňuje široké možnosti nastavení jeho vykreslení. Také v oboru kartografie je tento prvek hojně používán zejména v popiscích ulic, místních názvů atd. Zde nachází uplatnění vedení textu po křivce.

3.6.1 Element `text`

- `x,y` - souřadnice bodu, na který je umístěné počáteční účaří textu, přidáním atributů `dx, dy` lze tato pozice ještě „doladit“
- `font-size` - velikost písma dle jednotek uvedených v 3.8
- `font-family` - rodina písem *(serif, sans-serif, cursive, monospace atd.)*
- `font-style` - styl písma *(normal, italic, oblique)*
- `font-variant` - kapitálky *(normal, small-caps)*
- `font-weight` - nastavení tučnosti písma *(normal, bold, [100-900])*
- `font-stretch` - vodorovné roztažení písma
- `letter-spacing, kerning` - nastavení mezery mezi písmeny
- `word-spacing` - mezery mezi slovy
- `text-decoration` - dekorace textu *(none, underline, overline, line-through, blink)*

- `rotate` - otočení každého písmene o zadaný úhel ve stupních
- `textLength` - šířka, na kterou má text roztáhnout
- `lengthAdjust` - způsob jakým má text vyplnit zadanou šířku *(spacing, spacingAndGlyphs)*

Obrázek 3.7: Přehled základních atributů pro práci s textem

3.6.2 Element `tspan`

Uvnitř elementu `text` lze použít prvek `tspan` pro „jemnější“ přizpůsobení částí textu. S výhodou lze také využít relativních souřadnic. K dispozici jsou stejné atributy jako u elementu `text`.

```
<text font-weight="bold" font-size="30" fill="black"> Lorem ipsum no laoreet nostrum eum.
    <tspan dy="15" fill="red"> Etiam euismod efficiantur id sed. </tspan>
    <tspan dy="40" dx="-800" font-style="italic">Ius ea intellegat mnesarchum.</tspan>
    <tspan dy="30" rotate="10" fill="none" stroke="blue">
        Ea dicta legimus pri, mea fabellas assentior curo.</tspan>
</text>
```

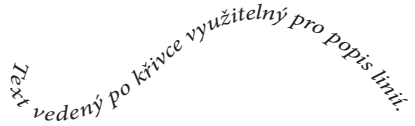
3.6.3 Element `textPath`

V kartografii užívaný způsob psaní textu podél křivky lze v SVG zapsat elementem `textPath`. Křivka, která nahrazuje účaří písma, je definována v hlavičce dokumentu v části `defs`. Zmíněná křivka musí také obsahovat identifikátor, dle kterého je jednoznačně určena. Část `textPath` musí být umístěna mezi tagy `<text>,</text>`, kde bývá nastaveno formátování textu. Element `textPath` může obsahovat atributy:

- `xlink:href(#id)` - povinný atribut, adresa křivky
- `startOffset` - odsazení začátku textu od začátku křivky

```
<defs>
    <path id="tah" d="M40,70 C0,110 40,150 70,80 S250,20 300,80"/>
</defs>

<text fill="black" font-family="cursive">
       <textPath xlink:href="#tah">
           Text vedený po křivce využitelný pro popis linií.
       </textPath>
</text>
```



3.7 Instrukce pro vykreslování

Pro správné vykreslení nestačí pouze uvést geometrickou podstatu objektů, ale sdělit parseru, jakým způsobem se má prvek vykreslit (viditelnost čar polygonů). Lze také nastavovat pravidla napojování hran.

3.7.1 Vlastnosti atributu `stroke`

Vlastnosti tahu se vztahují ke všem elementům uvedených v 3.4. Tedy jak k plošným objektům, tak i k liniím a textům.

- `stroke`
  Samotný atribut `stroke` slouží k určení barvy tahu. Jeho hodnoty mohou nabývat barev dle 3.7.4.

- `stroke-width`
  Tento atribut nastavuje tloušťku čáry. Hodnotou je číslo s jednotkou uvedenou na obr. 3.8. Pokud je tloušťka čáry neudána, je použita implicitní hodnota (`1px`). Je to často využívaný atribut. Může být *dynamicky* nastavován v závislosti k měřítku. V kartografickém zobrazení je takto nastavena tloušťka linií (znázorňující jejich kvalitu).

- `stroke-linecap`
  Tento typ určuje jakým způsobem má být tah ukončen. Hodnoty nabývají tvarů: `butt, round, square`.

- `stroke-linejoin`
  Pro určení tvaru napojení jsou k dispozici tyto hodnoty: `miter, round, bevel`. Při zobrazování v mapě lze použít různé hodnoty, v závislosti na zobrazovaném prvku. Například pro vykreslení vodních toků je vhodné použít hodnotu `round`, ale pro vyobrazení průběhu hranic hodnoty `miter`.

- `stroke-miterlimit`
  Nastavení „špičatosti“ lze provést jen pro objekty s `stroke-linejoin="miter"`. Hodnota je zadána číslem. V kartografii není zobrazování „ostrého“ napojování žádoucí a u prvků s vysokou fraktální dimenzí by mělo rušivý efekt.

- `stroke-dasharray`
  Pro vykreslení přerušované čáry je využíván atribut `stroke-dasharray` určující vzdálenost jednotlivých čar a mezer. Do hodnot lze zapsat vektor libovolné velikosti, podle kterého se kreslí čáry nebo prázdné mezery. Tento způsob lze využít v kartografii při zákresu osy železnice.

- `stroke-dashoffset`
  Tato část určuje, v jakém bodu se má začít s vykreslováním.

- `stroke-opacity`
  Neprůhlednost tahu je zadána číslem v intervalu $< 0, 1 >$.

3.7.2 Vlastnosti atributu `fill`

- `fill`
  Barva výplně je nastavena v tomto atributu.

- `fill-rule`
  Pravidlo výplně nabývá hodnot: ***nonzero***, *evenodd*

- `fill-opacity`
  Nastavení neprůhlednosti je nastaveno hodnotami v intervalu $< 0, 1 >$.

3.7.3 Atributy `opacity`,`display`,`visibility`

- `opacity`
  Nastavení neprůhlednosti celého objektu nebo skupiny objektů včetně tahu, výplně a textu je možné nastavit atributem `stroke`, jehož hodnoty jsou z intervalu $< 0, 1 >$. Pro odlišení neprůhlednosti tahu nebo výplně lze samostatně nastavit `stroke-opacity` či `fill-opacity`.

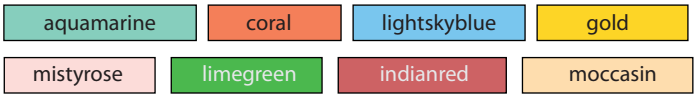
- `display`
  inline, *block, list-item, run-in, compact, marker, table, inline-table, table-row-group, table-header-group, table-footer-group, table-row, table-column-group, tabel-column, table-cell, table-caption, none*
  Objekty mohou být také vyjmuty ze zobrazovaní (hodnota *„none“*), i když se v dokumentu vyskytují. Této skutečnosti lze využít při vypínání vrstev. Stejného efektu lze dosáhnout nastavením atributu `opacity` na hodnotu 1, avšak pokud je pro tento objekt nastavena interaktivita, zůstává na rozdíl od atributu `display="none"` zachována.

- `visibility`
  visible,*hidden*,*collaspse*

3.7.4 Barevné modely

Způsobů jak do atributů `fill`, `stroke` a podobných zapsat určitou barvu je několik. Liší se použitým barevným modelem a zápisem. Kromě běžně známých barev obsahuje i ty méně známé.

- **Color names** jsou barvy pojmenovány anglickými názvy. Je to nejsnadnější a člověku nejsrozumitelnější forma zápisu. Příklad použití pro výplň „lososovou“ barvou: `fill="salmon"`.



- **RBG** je nejpoužívanější barevný model, kde výsledná barva vznikne aditivním míšením tří barevných složek (červená, zelená, modrá). Tento model je používaný v monitorech a projektorech. Hodnota `R,G,B` kanálů je zapsaná 8-bitovými celými čísly v intervalu $< 0, 255 >$. Hodnoty lze odečíst na barevné krychli. Zápis je proveden jako `RGB()` funkce se třemi číselnými hodnotami: `fill="RGB(240,32,12)"`. Možný je také hexadecimální zápis *(HEX)*, sestavený ze tří dvojic alfanumerických znaků uvozených znakem ‘`#`’ (`fill="#F0200C"`).

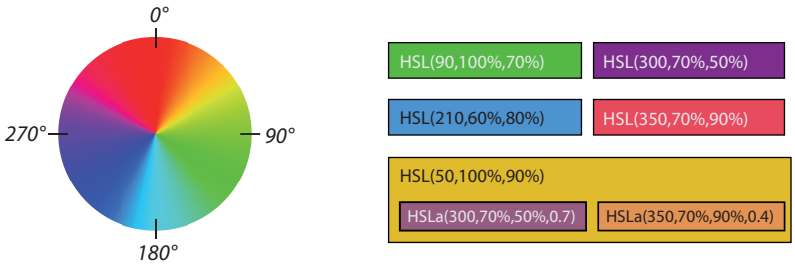
- **RGBa** má obdobný zápis jako předchozí *RGB* model, kde čtvrtý parametr nastavuje *alpha* kanál (neprůhlednost). Příklad použití: `fill="RGBa(255,0,255,0.4)"`, kde α-kanál je reprezentován hodnotou typu *float* v intervalu $< 0, 1 >$.

- **HSL** (Hue Saturation Lightness) určuje barvu dle barevného kruhu, sytosti a odstínu.

  - **Hue** je úhel (ve stupních) na barevném kruhu. $< 0, 360 >$ *[ve stupních]*
  - **Saturation** (sytost) barev je nastavena procenty. (100% - nejsytější barvy)
  - **Lightness** (světlost) je udána také procenty. (100% odpovídá bílé barvě, 0% černé barvě)

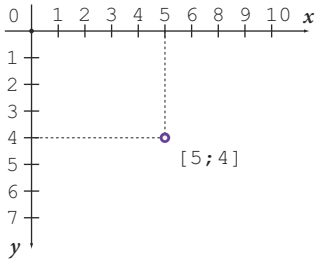
  Tohoto modelu lze využít při tvorbě kartogramu, kdy světlost barvy (lightness) území je nastavena dle hodnot z tabulky. V době psaní práce tento model bohužel ještě nepodporovaly všechny prohlížeče.
  Příklad zápisu: `fill="HSL(130,80%,30%)"`.

- **HSLa** je obdoba modelu HSL, kde je jako čtvrtý parametr hodnota α-kanálu.



3.8 Souřadnicový systém a jeho transformace

Scalable Vector Graphics česky znamená „škálovatelná vektorová grafika“ a proto také formát `SVG` nabízí různé typy transformací. K jejich popisu se bude vhodné nejprve seznámit se souřadnicovým systémem, jak ho chápe `SVG`. V závislosti se zažitými konvencemi v počítačové grafice a zobrazovacích zařízeních byl zvolen souřadnicový systém s počátkem v severozápadním rohu.



používané délkové jednotky:

px	*(pixels)*	pixely (implicitní hodnota)
pt	*(points)*	body
cm	*(centimeters)*	centimetry
mm	*(milimetres)*	milimetry
in	*(inches)*	palce
em	*(current font size)*	velikost aktuálního fontu
ex	*(x-height)*	velikost písmene x
pc	*(pica)*	[: pajka :], 1 pica = 12 points
%	*(percent)*	procenta vůči zobrazované oblasti

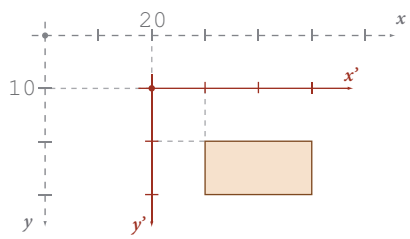
Obrázek 3.8: Přehled souřadnicového systému a délkových jednotek

3.8.1 Atribut `transform`

Do většiny elementů, počínaje tím kořenovým `<svg>`, lze zapsat atribut `transform`, se kterým je možno provádět transformace souřadnic. Tento fakt nahrává použití `SVG` ke kartografickým účelům, kde se transformace využívají velmi často. K dispozici je několik typů: translace, rotace nebo zkosení. Všechny tyto typy lze zahrnout do jednoho prvku pomocí transformační matice.

- **translate** (translace)

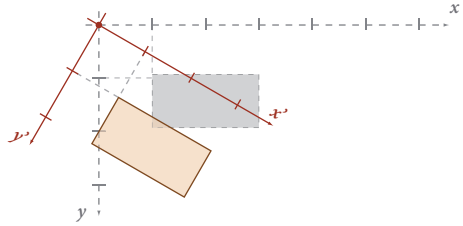
```
transform="translate(20,10)"
```



Pro posun objektů lze použít funkci `translate`. Její argumenty jsou souřadnice bodu, na který má být počátek transformované soustavy umístěn. Lze využít pro pozicování symbolů, které jsou zapsány v místní soustavě.

- **rotate** (rotace)

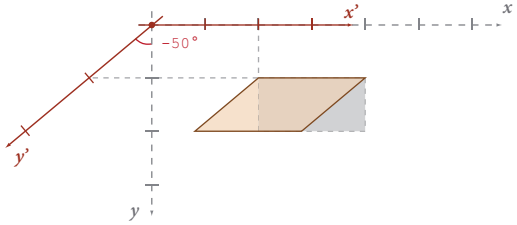
```
transform="rotate(30)"
```



Pootočení počátku SS se provádí funkcí `rotate()`. Úhel je zadáván ve stupních a může nabývat záporných hodnot (protisměr hodinových ručiček). Jedná se také o často používanou transformaci, např. pootočení textu v názvech ulic.

- **skew** (zešikmení)

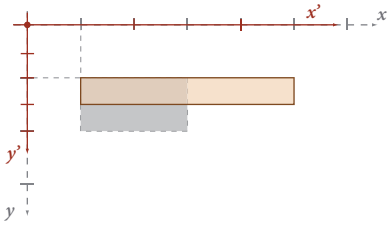
```
transform="skewX(-50)"
```



Funkce má dva tvary zápisu, `skewX()` pro zkreslení na ose X a `skewY()` pro zkreslení na ose y. V obou případech jako argument vstupují stupně vychýlení zvolené osy od osy původní. V kartografii nenachází přímé uplatnění.

- **scale** (měřítko)

```
transform="scale(2,0.5)"
```



Vhodná a užitečná funkce `scale()` určuje zvětšení nebo zmenšení objektů. Proměnná je bezrozměrná jednotka a určuje násobek, kterým se souřadnicový systém násobí. Funkce je *přetížená*, lze tedy zadávat jak jeden argument pro změnu měřítka obou os, nebo dva argumenty pro každou osu zvlášť. Změna měřítka je vztažena nejen na geometrii, ale i na styl vykreslování. V aplikaci je tato funkce využívána mimo jiné i k „zoomování“.

- **matrix** (transformační matice)
  Pro obecný popis je vhodný zápis pomocí transformační matice. Pro každý výše zmíněný prvek existuje i varianta zapsaná pomocí funkce `matrix()`. Násobením matic lze dosáhnout kombinaci jednotlivých transformačních činitelů (transformace probíhá ve stejném pořadí jako násobení matic). Aby se matice mohly násobit a byly čtvercového typu, je jako poslední řádek přidán vektor `(0,0,1)`. V kartografii se transformační matice využívají zejména pro *georeferencování*.

```
transform="matrix(a,b,c,d,e,f)"
```

$$\begin{bmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{bmatrix}$$ obecná tranformační matice

$$\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$ **translace**
`matrix(1,0,0,1,`t_x`,`t_y`)`
`translate(`t_x`,`t_y`)`

$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ **měřítko**
`matrix(`s_x`,0,0,`s_y`,0,0)`
`scale(`s_x`,`s_y`)`

$$\begin{bmatrix} 1 & \tan(\alpha) & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ **zešikmení dle osy ‚x‘**
`matrix(1,0,tan(`α`),1,0,0)`
`skewX(`α`)`

$$\begin{bmatrix} 1 & 0 & 0 \\ \tan(\alpha) & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ **zešikmení dle osy ‚y‘**
`matrix(1,tan(`α`),0,1,0,0)`
`skewY(`α`)`

$$\begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$ **rotace**
`matrix(cos(`α`),sin(`α`),-sin(`α`),cos(`α`),0,0)`
`rotate(`α`)`

3.9 Rastr v SVG

Jak již bylo zmíněno dříve, SVG umožňuje nejen vkládat a ukládat rastrovou grafiku, ale nabízí i možnosti editace typu transformace nebo aplikaci předdefinovaných filtrů. Jelikož se i v moderních digitálních kartografických pracích počítá s rastrovým vyjádřením (v podobě ortofotomap nebo stínovaného reliéfu), nalezne i rastrový formát své uplatnění.

3.9.1 Element image

Vložení obrázku elementem image je provedeno obdobným způsobem jako v HTML. Lze využít tyto základní atributy:

- x,y - souřadnice, na které bude umístěn pravý horní roh obrázku
- width,height - velikost vkládaného obrázku
- xlink:href - absolutní/relativní cesta k obrázku

Pro vložení obrázku přímo do dokumentu lze využít kódování Base64 (převod binárních dat na znaky ASCII). Rastrový obrázek v binární formě lze po překódování na ACSII vložit do dokumentu tímto způsobem: xlink:href=" ...", kde místo teček je dosazen samotný kód. Rastrový objekt lze transformovat atributem transform, kde jsou k dispozici všechny typy transformací. Tohoto faktu lze využít třeba při georeferencování rastrového podkladu mapy.

```
<image x="40" y="50" width="512" height="512"
       xlink:href="Lenna.bmp" />

<circle cy="160" cx="160" r="130"
        fill="red" opacity="0.3"/>

<circle cy="160" cx="440" r="130"
        fill="green" opacity="0.3"/>

<circle cy="160" cx="305" r="130"
        fill="blue" opacity="0.3"/>
```

Obrázek 3.9: Ukázka použití rastrových obrazových dat

3.9.2 Element filter

Na SVG objekty lze aplikovat rastrové filtry známé z editorů jako jsou *Adobe Photoshop, Gimp* atd. Důvod proč na vektorovou grafiku používat rastrové filtry je jednoduchý. Rastrová reprezentace složitých, rozmanitých a členitých obrazových dat je výhodnější na výpočet i na vykreslení. Je důležité zmínit, že *škálování* (změna měřítka) nemá na výsledné vyobrazení vliv (v podobě ztráty kvality), jelikož dochází k novému výpočtu prvků pro každou hladinu. I když přílišné zdobení prvků v mapě má rušivý ráz, s drobnými a rozvážnými efekty lze přispět ke zdokonalení vizuální i informační hodnoty výsledné práce. Filtry se dají aplikovat nejen na geometrické tvary a texty, ale i na rastrové obrázky.

Element `filter` (zapsaný v hlavičce v části `defs`) slouží jako kontejner pro zápis posloupnosti efektů. Obsahuje jediný atribut `id` sloužící jako identifikátor. K použití existuje mnoho elementárních filtrů, ze kterých lze dosáhnout kýženého výsledku. Přehled základních filtrů[1]:

- `feBlend`
  Nastavení způsobu prolínání obrazů. K dispozici jsou tyto prolínací režimy: *Normal, Multiply, Screen, Darken, Lighten.* Jedná se o výpočet nové barevné hodnoty částí obrázků, které se překrývají. Pro kartografické účely není tento filtr využíván.

- `feColorMatrix`
  Jednou z možností jak ovlivnit barevnost celého výsledného obrazu je použití tohoto filtru. Barevnou maticí lze upravovat jas, transparentnost nebo posouvat barevné spektrum. Je možno použít několik typů nastavení:

  - **matrix** (obecná barevná matice),
  - **saturate** (úprava jasu, kdy hodnoty leží v intervalu $< 0, 1 >$),
  - **hueRotate** (barevné spektrum, hodnoty stupně na barevném kruhu z modelů HSL nebo HSV),
  - **luminanceToAlpha** (transparentnost je nastavena podle jasu bodu, bez hodnoty)

- `feComponentTransfer`
  V grafických editorech je využíván *histogram jasu*, který umožňuje dle křivky nastavit poměr jasu a kontrastu. Histogram je možné upravovat pro kanály *RGBA* zapisované `feFuncR`, `feFuncG`, `feFuncB` a `feFuncA`. Zápis vztahů do různých kanálů atributem `type` s parametry:

  - **identity** - je použita původní kompozitní barva
  - **table** - podle hodnot v atributu `tableValues` je provedena lineární interpolace křivky
  - **discrete** - křivka sestavena z diskrétních hodnot z parametrů atributu `tableValues`
  - **linear** - křivka nahrazena přímkou, která je určena atributy `slope` (sklon ve stupních) a `intercept` (posun na svislé ose)
  - **gamma** - křivka určena hodnotou amplitudy (atribut `apmlitude`), exponentu (`exponent`) a svislým posunem (`offset`)

- `feComposite, feMerge`
  Různé režimy slučování rastrů dle algoritmu *Porter-Duff.*

- `feConvolveMatrix`
  Výpočet hodnoty pixelu v závislosti k sousedním prvkům a parametry matice. Matice (kernel) postupně prochází obrazem a přepočítává hodnoty. Matice obsahuje $m \times n$ prvků. Výsledek je ovlivněn velikostí kernelu a jeho parametry. Tímto filtrem je možné provádět úlohy typu rozostření, zostření, detekce hran atd. Známá je aplikace na data získaná dálkovým průzkumem Země. Na stejném principu je stavěna řada dalších bitmapových operací. Matice je určena základními atributy:

[1] úplný přehled filtrů na `http://www.w3.org/TR/SVG11/filters.html`

- `order` - rozměr matice
- `kernelMatrix` - po řádcích zapsány prvky matice
- `bias` - konstanta, která je přičtena k výsledné hodnotě pixelu
- `edgeMode` - pravidla pro počítání krajních pixelů, kdy sousední prvky neexistují, parametry: `duplicate` (duplikace okrajových bodů), `wrap` (jsou použity hodnoty z protilehlého konce bitmapy), `none` (použity nulové hodnoty)
- `kernelUnitLength` - velikost buňky matice, implicitně je rovna jednomu pixelu
- `preserveAlpha` - zachování transparentnosti nastavením atributu na „`true`“

- `feGaussianBlur`
  Velmi využívaným filtrem je *gaussovské rozostření*. Je to forma aplikace kernelu o určitých parametrech. Atributem `stdDeviation` je nastavena standardní odchylka tedy „velikost“ rozostření. Možné je nastavení odchylky pro každou osu. Zajímavé je použití tohoto filtru pro liniové prvky (např. odlišení cyklotras od turistických), kdy vzrůstá čitelnost mapy.

- `feTile`
  Vyplnění objektu vzorkem (rastr/vektor). Obsahuje atributy `x, y, width, height` (umístění a velikost vzorku) a pro externí vzorky atribut `xlink:href`.

- `feFlood`
  Vyplnění vzorku barvou. Analogie atributu `fill` i pro rastrová data. Nabízí i možnost nastavení průhlednosti. Obsahuje atributy `flood-color` (barva výplně) a `flood-opacity` (neprůhlednost v intervalu $< 0, 1 >$).

- `feTurbulence`
  Přidáním šumu takzvanou *Perlin* funkcí lze dosáhnout vzniku textury jako jsou oblaka či mramor. Zvolený typ lze určit atributy:
  - `type` - hodnotou `fractalNoise` je generován šum a hodnotou `turbulence` víření
  - `baseFrequency` - frekvence šumové funkce může být pro každou osu odlišná
  - `numOctaves` - počet period
  - `seed` - startovní číslo generátoru náhodného šumu

- `feDisplacementMap`
  Filtr zajištující deformace obrazu.

- `feImage`
  Nástroj pro vložení externího grafického souboru. Obdobný zápis jako `SVG` element `image` (cesta k souboru atributem `xlink:href`). Nabízí i možnost vložit další `SVG` obrázek.

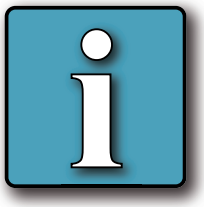
- `feMorphology`
  Provádí zesílení (ztučnění) nebo ztenčení prvku. Nastavení je provedeno atributy: `operator` (parametr `erode` pro zúžení a `diletate` pro ztučnění) a `radius` určuje jejich rozsah (velikost).

- `feOffset`
  Posun prvku v ose `x, y`. Atributy `dx,dy` pro zadání posunu. Filtr nahraditelný atributem `transform="translate(dx,dy)"`.

- `feSpecularLighting feDiffuseLighting`
  Filtry využívající *Phongův osvětlovací model* zejména pro reálné vykreslení 3$\mathcal{D}$ grafiky.

- `feDistantLight, fePointLight, feSpotLight`
  Druhy osvětlení (směrové, bodové, reflektor).

```svg
<defs>
   <filter id="stin">
      <feGaussianBlur stdDeviation="5" />
      <feOffset dx="7" dy="7"/>
   </filter>
</defs>
...
<rect filter="url(#stin)"  x="10" y="330" rx="10"
      width="200" height="200" fill="black" />
<rect x="10" y="330" rx="10" width="200" height="200"
      fill="#4FA0BA" stroke="black" stroke-width="2"/>
<text filter="url(#stin)" font-style="serif" x="75" y="515" font-weight="bold"
      font-size="250" opacity="0.8" fill="black">  i  </text>
<text font-style="serif" x="75" y="515" font-weight="bold" font-size="250"
      fill="white">  i  </text>
```



Obrázek 3.10: Praktické využití filtrů pro symbol informačních center

3.10 Atributová data

V geografických informačních systémech *(GIS)* jsou obsahové informace geograficky určeny. Většinou se jedná o soubor popisující geometrickou složku a další popisující atributová data. Ta jsou většinou v určitém databázovém systému. Jelikož byl formát `XML` původně určen jako nositel informací, je více než vhodné použít tento formát k uložení atributových dat. Ke každému SVG elementu lze zapsat libovolný atribut. Příklad zápisu názvu komunikace: `<path typ="dálnice" d="..."/>`. Přidané atributy, které se nezúčastňují vykreslování, je vhodné definovat v DTD, čímž bude zaručena bezchybovost dokumentu.

3.11 Metadata

Metadata lze interpretovat jako „data o datech“. Popisují informace obsažené v dokumentu. Je to běžný prvek, který je používán pro popsání obsahu. Jedná se zejména o původ obsahu, rok jeho vzniku, údaje o autorovi nebo platnosti dat. V *GIS* systémech jsou popisovány geometrické i atributové prvky. Jedná se o popis a určení souřadného systému, způsobu pořízení dat, informace o zhotoviteli atd. Jedním z nejvhodnějším a zároveň nejpoužívanějším formátem pro zápis metadat je právě `XML` se svou hierarchickou

strukturou. Proto data obsažená v `SVG` formátu nemusí být pouhá obrazová data, ale mohou nést i informaci o referenčním systému. Tyto informace jsou zapisovány do elementu `metadata`. Následující kód je převzat ze stránek W3C.org[2] a popisuje použití definovaného geografického systému souřadnic podle kódu EPSG. V ukázce je využit model pro zápis metadat **RDF** (Resource Description Framework).

```
<metadata>
   <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
            xmlns:crs="http://www.ogc.org/crs"
            xmlns:svg="http://www.w3.org/2000/svg">
     <rdf:Description rdf:about="">
       <!-- In case of a well-known Coordinate Reference System
            an 'Identifier' is enough to describe the CRS -->
       <crs:CoordinateReferenceSystem svg:transform="rotate(-90) scale(2,2)">
         <crs:Identifier>
           <crs:code>4326</crs:code>
           <crs:codeSpace>EPSG</crs:codeSpace>
           <crs:edition>5.2</crs:edition>
         </crs:Identifier>
       </crs:CoordinateReferenceSystem>
     </rdf:Description>
   </rdf:RDF>
</metadata>
```

[2] `http://www.w3.org/TR/SVG/coords.html#GeographicCoordinates`

Kapitola 4

Stylové jazyky

XML jako formát je nositelem obsahu, který by měl být oddělen od formy, aby v něm obsažené informace mohly být prezentovány více způsoby. Například jiný výstup bude určen pro tisk, jiný pro internet nebo grafickou vizualizaci. Metodu jak toho dosáhnout poskytují stylové jazyky CSS (Cascade Style Sheets) a XSL (XML Style Language). Tyto jazyky jsou samozřejmě dostupné i pro XML dialekty SVG nevyjímaje.

4.1 CSS

Kaskádové styly jsou známy již delší dobu ve spojení s HTML, kde se trvale usídlily. Nyní se o vývoj stará W3C a v době psaní této práce se pracuje na verzi CSS3[1]. Vlastnosti a chování elementů se definují ve stylovém souboru, který může být interní (součást dokumentu) nebo externí (samostatný soubor). Jednoduchá ukázka CSS jazyka bude předvedena na příkladu 2.7.

```
bod   font-size: 20pt;
      text-align: center;
      font-weight: bold;
      color: red;
```

Názvům elementů `bod` a `souradnice` se v CSS říká selektory. Ty určují k jakým elementům se budou uvedené vlastnosti vztahovat. Přehled nejdůležitějších selektorů je uveden v tabulce 4.1. Vztahu rodič — potomek, lze s výhodou použít i při stylování `SVG`. Zdrojový soubor s koncovkou *'.svg'* může nést pouze geometrické vyjádření, přičemž instrukce pro vykreslování mohou být uvedeny v jiné části dokumentu nebo i v jiném souboru. Dále lze ty samé prvky rozlišovat pomocí tříd *(class)* nebo identifikátoru *(id)*. Tato skutečnost je velice výhodná pro kartografické účely. Kdy dochází k zohlednění kvalitativních prvků (šířka vodního toku). Definováním CSS stylového souboru lze utvořit jazyk mapy, který může být jednoduše upraven nebo nahrazen, aniž by to mělo vliv na polohovou správnost. Dokonce lze tento styl i *dynamicky* měnit (použitím *JavaScriptu*). Mezi prvky, které by mohly být využity pro kartografické účely patří:

[1] `http://www.w3.org/Style/CSS/`

- barva
  Umožňuje nastavení barvy do atributů `stroke` a `fill`. K dispozici jsou barevné modely uvedené v 3.7.4.
- definice tahu
  Vykreslení tahu lze v CSS upravit použitím atributů jako `stroke-width, stroke-linejoin, stroke-dasharray`
- styl popisu
  Velikost písma, jeho typ a styl lze CSS nastavit použitím selektorů `font-weight, font-family, font-style`

Způsobů jak použít CSS stylování je několik. Jedním z nich je použití v dokumentu interně. Pokud je CSS definován externě, je nutné v dokumentu určit `url` CSS souboru tímto způsobem:

```
<?xml-stylesheet href="soubor.css" type="text/css"?>
```

selektor	*výběr*
*****	*všechny elementy*
bod	*všechny elementy* ***bod***
bod souradnice	*elementy* ***souradnice****, které jsou potomkem el.* ***bod***
bod > souradnice	*analogický zápis:* ***bod souradnice***
souradnice[system]	*elementy* ***souradnice****, která mají atribut system*
souradnice[systém="EPSG:9221"]	*elementy* ***souradnice****, s atributem systém: S-JTSK*
bod#4001	*element* ***bod****, který má* ***ID*** *4001*
bod.II	*element* ***bod****, který má atribut class nastaven na "II"*

Tabulka 4.1: Příklady použití selektorů v CSS

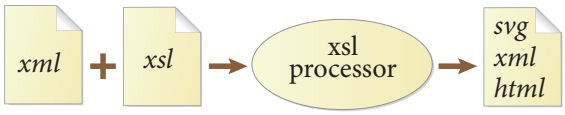
4.2 XSL

Dalším ze stylových jazyků je `XSL` (*eXtensible Stylesheet Language*), často též ve tvaru `XSLT` (*eXtensible Stylesheet Language Transformation*). Přináší trochu odlišný způsob než předchozí *CSS*, kdy se „stylování" provádí podle předem definovaných šablon. Výsledný dokument je sestaven programem, kterému se říká *XLST Processor*. Mezi nejznámější patří *Saxon*[2] od Michaela Kaye a *Xalan*[3] (Apache project). Oba jsou napsané v jazyce *Java*(*Xalan* je implementován i v `C++`). Dnes je již standardem, že internetové prohlížeče mají takový XSLT procesor implementován. `XSL` nabízí možnost použití `XPath` výrazů, které umožňují nejen navigaci ve zdrojovém dokumentu, ale i logické řízení [Her10]. Vývoj `XSL` je přibližně stejně dlouhý jako `SVG`. `XSL` získalo W3C doporučení v listopadu 1999. Jmenný prostor je deklarován tímto způsobem:

```
<xsl:stylesheet version="1.2"
                xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
```

[2] `http://sourceforge.net/projects/saxon`
[3] `http://xalan.apache.org/`

Způsob jak probíhá transformace je zřejmý na obrázku 4.1. Jeden či více `XML` souborů vstupují společně se stylovým souborem (koncovka *.xsl*) do XSLT procesoru na jehož výstupu je dokument libovolného typu. Může to být další XML, HTML, TXT nebo SVG dokument. Výstupních dokumentů může být také více, pro různé typy zařízení (obrazovka, tiskárna atd.)



Obrázek 4.1: Princip XSL transformací

Externí stylový soubor je definován obdobným způsobem jako v *CSS*. Daný kód odkazuje na XSL soubor `sablona.xsl`:

```
<?xml-stylesheet type="text/xsl" href="sablona.xsl"?>
```

Jak už bylo řečeno výše, transformace je prováděna podle šablon, které se zapisují mezi tagy `<xsl:template match="/" > </ xsl:teplate >`. Znak '/' označuje kořenový (root) element, na který bude šablona aplikována. Místo toho znaku může být jakýkoliv platný `Xpath` výraz, který označuje určitý uzel v dokumentu. Více o možnostech adresování v 4.2.3.

4.2.1 Hlavička XSL

Kromě povinného uvedení jmenného prostoru a verze v elementu `xsl:stylesheet` je vhodné také specifikovat výstupní formát pomocí elementu `xsl:output`.

```
<xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
                xmlns:svg="http://www.w3.org/2000/svg" xmlns="http://www.w3.org/2000/svg">


   <xsl:output method="xml" version="1.2" encoding="windows-1250"
               doctype-public="-//W3C//DTD SVG 1.1//EN"
               doctype-system="http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd"
               indent="yes" omit-xml-declaration="yes" media-type="image/xml+svg"/>

   ... obsah stylového dokumentu ...

</xsl:stylesheet>
```

Atribut `method` určuje typ výstupního formátu, užité kódování je specifikováno v atributu `encoding`. Následují atributy `doctype-public` a `doctype-system` popisující definice typu výstupního dokumentu. Zarovnání kódu dle XML pravidel je nastaveno atributem `indent` s parametrem `yes`. Atribut `omit-xml-declaration="yes"` zaručuje vytisknutí klasické XML deklarace. Jako poslední je v ukázce uveden atribut `media-type` popisující typ *(MIME, Content-type)* výstupního dokumentu.

4.2.2 Element `xsl:value-of`

Způsob jak získat hodnotu elementu nebo atributu, je použití tagu `value-of` ve jmenném prostoru `xsl`. Povinný atribut `select`, zajišťuje uvedení přesné polohy a názvu uzlu.

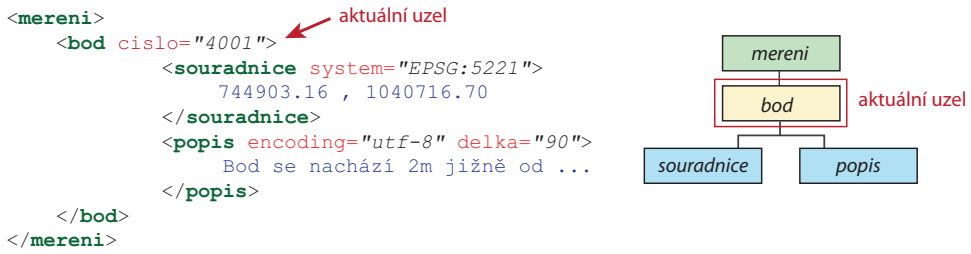
```
<xsl:value-of select="jmeno"/>
```

Jedná se o nejvíce využívaný element, jeho použití je ukázáno na obrázku 4.2.

Obrázek 4.2: Přehled vstupních a výstupních souboru XSL transformací

4.2.3 `XPath` výrazy

Hierarchická struktura XML dokumentu je zprostředkována pomocí tzv. uzlů *(nodes)*. Pro „cestování“ po takových uzlech jsou využívány `XPath` identifikátory. Díky dědičným vztahům (rodič — potomek) se lze jednoduše odkazovat na vnořené elementy. Základní osy, kam je možno adresovat jsou tyto: (příklady se vztahují k ukázkovému *xml* dokumetu, kdy je aktuální uzel element `bod`)[Wil09]:



- **absolutní/relativní cesta** (aktuální element `bod`)
  - `/bod/popis/` ⇒ absolutní cesta k elementu `popis`

  - `popis/` ⇒ relativní cesta k elementu `popis`

- **kořenový element**
  - `/` ⇒ kořenový element `mereni`

- **předek**
  - identifikátor: `parent, ancestor, ancestor-or-self`
  - `../mereni, parent::bod, /, ancenstor::bod` ⇒ vybere element `mereni`

- **potomek**
  - identifikátor: `child, descendant, descendant-or-self`
  - `/bod/souradnice/, souradnice/` ⇒ vybere element `souradnice`
  - `descendant::bod, child::bod` ⇒ vybere elementy `souradnice, popis`
  - `descendant-or-self::bod` ⇒ vybere elementy `bod, souradnice, popis`
  - `//popis` ⇒ vyhledá elementy `popis` v libovolné hloubce

- **sourozenec**
  - identifikátor: `preceding-sibling, preceding, following`
  - `/mereni/bod/popis/, popis/, following::souradnice` ⇒ vybere element `popis`
  - `/mereni/bod/souradnice/, souradnice/, preceding::popis` ⇒ vybere element `souradnice`

- **sám na sebe**
  - identifikátor: `self`
  - `., self::node()` ⇒ vybere element `bod`

- **atribut**
  - identifikátor: `attribute, @`
  - `popis/attribute::encoding, popis/@encoding` ⇒ zjistí hodnotu atributu `encoding` *(utf-8)*
  - `popis[@encoding]` ⇒ vybere všechny elementy `popis`, které mají atribut `encoding`

- **index**
  Pokud by element `bod` obsahoval více elementů `popis`, je možné k nim přistupovat pomocí indexů:
  - `/bod/popis[1]` ⇒ vrátí první element `popis`
  - `/bod/popis[last()]` ⇒ vrátí poslední element `popis`

- **logická podmínka**

- `/bod/popis[@delka > 80]` ⇒ vybere elementy popis, které mají atribut `delka` nastaven více než 80 znaků
- `/bod/popis[@delka > 80][@delka < 120]` ⇒ vybere elementy popis, které mají atribut `delka` nastaven více než 80 znaků, ale méně než 120

- **libovolná hodnota**
  Libovolný element lze zastoupit znakem '*'.
  - `/bod/*, child::*` ⇒ vybere všechny libovolné elementy, které jsou potomky elementu `bod`
  - `attribute::*` ⇒ vybere všechny atributy aktuálního uzlu`bod`
- **Xpath operátory**
  Na uzly lze aplikovat XPath operátory, příklad použití:
  `/souradnice | /souradnice/@system`.
  - `|`- sjednocení (vrátí hodnoty obou uzlů)
  - `+, -, *, div` - matematické operátory (číselné hodnoty), součet, rozdíl, součin, podíl
  - `=, !=` - rovnost, nerovnost
  - `<, >, <=, =>` - méně než, více než
  - `or` - logické „nebo“
  - `and` - logické „a zároveň“
  - `mod` - zbytek po celočíselném dělení

4.2.4 Podmínečné zpracování

V XSLT lze provádět různé logické úkony známé z ostatních programovacích jazyků. Jedná se o podmínky nebo cykly. Tato skutečnost činí rozdíl mezi CSS. Lze tímto způsobem roztřídit (seřadit) nebo klasifikovat data.

- `xsl:if` - jedna ze dvou možností jak použít podmíněný výraz je příkaz `xsl:if`

  ```
  <xsl:if test="podmínka"> ... </xsl:if>
  ```

  Podmínka nemá `else` část, tato nevýhoda je odstraněna použitím příkazu `xsl:choose` viz. níže.

- `xsl:choose` - prvek obsahující plnou podmínku je `xsl:choose` společně s instrukcí `xsl:when` a `xsl:otherwise`

  ```
  <xsl:choose>
          <xsl:when test=" podmínka ">
                  ... tělo podmínky ...
          </xsl:when>
          <xsl:otherwise test=" 2. podmínka ">
                  ... tělo 2. podmínky ...
          </xsl:otherwise>
  </xsl:choose>
  ```

  první podmínka, obdoba příkazu **`if`**

  druhá podmínka, obdoba příkazu **`else`**
  libovolně násobné použití

  Tohoto příkazu lze využít při klasifikaci prvků, např. velikost značky pro obce dle počtu obyvatel nebo šířka vodního toku podle jeho délky.

4.2.5 Cykly

I „stylovací“ jazyk XSLT má k dispozici metodu pro iterativní zpracování, tedy obdobu k programovacímu příkazu `for`. Jedná se o část šablony, která je opakovaně použita. Základní syntaxe je tato :

```
<xsl:for-each select="cesta"> ... </xsl:for-each>,
```

kde instrukce `xsl:for-each` značí počátek cyklu a atribut `select` nastavuje cestu k uzlům, na které chceme část této šablony aplikovat. Přitom lze využít všech možností zápisu cesty podle 4.2.3.

vstupní dokument [.xml]	*transformační šablona* [.xsl]	*výstup* [.html]
`<mereni>` `<bod cislo="4001"> ... </bod>` `<bod cislo="4002"> ... </bod>` `<bod cislo="4003"> ... </bod>` `<bod cislo="4004"> ... </bod>` `<bod cislo="4005"> ... </bod>` `<bod cislo="4006"> ... </bod>` `</mereni>`	`<xsl:for-each select="mereni/bod">` `bod č.` `<xsl:vaule-of select="@cislo"/>` ` ` `</xsl:for-each>`	`bod č. 4001` `bod č. 4002` `bod č. 4003` `bod č. 4004` `bod č. 4005` `bod č. 4006`

Tabulka 4.2: Užití instrukce `xsl:for-each`

4.2.6 Proměnné

Pojem *proměnné* nelze u XSLT chápat jako u ostatních programovacích jazyků. Jedná se spíše o definici konstanty, jelikož hodnota již definované proměnné už nelze změnit. I přes tuto nevýhodu je použití tohoto prvku přínosné. Do proměnné lze uložit například cestu k určitému uzlu, číselnou nebo textovou hodnotu. Zápis hodnoty probíhá elementem `xsl:variable`, který obsahuje dva atributy. První (`name`) nastavuje název a druhý (`select`) hodnotu proměnné. Zavolání hodnoty je zprostředkováno přes element `xsl:value-of`, kdy v atributu `select` je určen název proměnné uvozené znakem „$“.

```
<xsl:variable name=" foo1 " select=" 320 " />
<xsl:variable name=" foo2 " select=" 425 " />

Počet vyder je:
<xsl:value-of select=" $foo1 + $foo2 "/>
jednotek !
```

Počet vyder je: 745 jednotek !

XSLT se nemusí uvádět datové typy proměnných, lze ale zapsat „boolean“ hodnoty `true()` a `false()` využitelné v podmínečném zpracování. Často je také do proměnné ukládána cesta k určitému uzlu, kdy dojde (při vhodném pojmenování) ke zvýšení čitelnosti a přehlednosti kódu.

4.2.7 Další použité prvky

- `document()`
  V případě použití další externích dokumentů lze využít funkci `document()`. Ta je volána z elementu `xsl:variable`. Načtení externího dokumentu `dokument2.xml`:

  ```
  <xsl:variable name="etapa2" select="document('C:\ dokument2.xml')"/>
  ```

  Následné použití uzlu `cas` v externím dokumentu

  ```
  <xsl:value-of select="$etapa2/cas"/>
  ```

Kapitola 5

Aplikace do praxe

Praktická ukázka využití `SVG` a přidružených technologií jako (CSS, XSL, JavaScript) v kartografii je vystavena na `http://svg.filipzava.cz`. Důvod proč se nejedná o samotný soubor `SVG` je jednoduchý. Použitím stylových souborů jako CSS nebo XSL je dosaženo jisté jednotnosti a přehlednosti. Je také jasně oddělena geometrická složka od té vizuální. Navíc body, které tvoří hranice územních jednotek mohou být jednoduše exportovány z ostatních programů (*ArcMap, AutoCad, Kokeš atd.*) a ihned použity k vykreslení. Další výhodou „stylování“ je *dynamická* změna způsobu vykreslení (v aplikaci změna klíče mapy u plánu Dejvic). *JavaScript* zajišťuje ovládání dokumentu (*zoom,posun*) a dynamičnost dokumentu (vyplňování atributové tabulky daty po kliknutí na příslušnou obec u mapy ČR). Aplikace je trvale vystavena na vlastním serveru, kde jsou k dispozici také zdrojové kódy.

5.1 Zdroj dat a způsob exportu do `XML/SVG`

Ve většině případů byla data získána z formátu *Shapefile* společnosti ESRI. Data pro Českou republiku a Dejvice byly staženy jako *shp* balíček ze serveru cloudmade.com[1] a pochází z dílny *OpenStreet Map*. Práva na tyto data jsou ošetřena licencí CC BY-SA 2.0[2]. Další krok byla jejich úprava v GIS aplikaci *ArcMap10* s univerzitní licencí. Upravována byla zejména atributová data a místy i polohopis jednotlivých objektů.

5.1.1 Program `shp2svg`

Export do formátu `xml/svg` byl zajištěn aplikací `shp2svg`. Je to projekt serveru *carto.net*[3] uvolňován pod licencí LGPL. Projekt také využívá program `shp2pgsl`, který je součástí projektu PostGis[4]. Dále je využit program `ogis2svg.pl`, který data převede na SVG formát. Výsledný dokument má tyto vlastnosti:

- Bodová vrstva je převedena na elementy `point` nebo `symbol` (program se dotazuje).
- Polygonová vrstva je reprezentována elementy `polygon` a `path`.

[1] `http://downloads.cloudmade.com/`
[2] Creative Commons — Attribution-ShareAlike 2.0 Generic
[3] `http://www.carto.net/papers/svg/utils/shp2svg/`
[4] `http://postgis.refractions.net/`

- Liniové prvky jsou převedeny na element `path` nebo `multiline`.
- Výstup obsahuje relativní souřadnice (redukce velikosti dokumentu).
- Souřadnice mohou být také zaokrouhleny.
- Vybrané prvky z atributové tabulky(volí uživatel) jsou vloženy k příslušným elementům jako atributy. Z tohoto důvodu je také vytvořena *definice typu dokumentu (DTD)*, která umožňuje přidání negrafických dat.
- Možnost vybrání atributu, který má sloužit jako unikátní identifikátor *(id)*.
- K elementům je možno přiřadit také „event-handlers“ *(sloty pro akce)* při použití JavaScriptu.

Program napsaný v jazyce `C++` je ke stažení v binárním souboru zkompilovaným pro 3 platformy (MS Windows,GNU Linux a MacOSX). V prostředí Windows se program spouští z příkazového řádku příkazem:

```
ogis2svg.exe --input yourshapename --output yourshapename.svg --roundval 0.1
```

Parametry `--input` a `--output` nastavuje jméno vstupního/výstupního souboru, parametr `--roundval` určuje desetinné místo pro zaokrouhlení. Seznam nepovinných parametrů:

`--inputunits, --outputunits` - definice vstupních, výstupních jednotek

`--viewBox` - určení obzoru, hodnoty: `'480000 -300000 350000 220000'`

`--scale` - nastavení měřítka, hodnota `'25000'`, pro měřítko 1: 25 000

`--referenceframe` - souřadnice referenčního rámu (xmin, xmax, ymin, ymax)

K výslednému SVG dokumentu stačí připojit stylový soubor, kde jsou definovány značky mapy. Lze ho také otevřít ve vektorových editorech a přidat prvky mapy jako grafické měřítko a tiráž.

5.2 Popis aplikace

V aplikaci je rozlišena datová složka *(body polygonů, linií nebo symbolů)* a vizualizační složka *(barvy, značky atd.)* . Pomocí XSL transformací jsou tyto složky převedeny na výsledný SVG dokument. Datová část je v „surovém“ fomátu XML, zatímco vizualizační *(stylová)* je zapsána v jazycích CSS a XSL.

5.2.1 XML datová část

Po převodu z formátu *Shapefile* do SVG byl dokument dodatečně upraven v textovém editoru jEdit, kde byly vymazány zbytečné (pro požadovaný účel) atributy a elementy. Následně byl dokument uložen ve formátu XML, protože transformace na SVG byla zajištěna použitím XSL. Byl kladen důraz, aby XML data byla co nejjednodušší a měla co nejmenší datový objem. Většina XML souborů obsahuje kořenový element s názvem `features` označující geometrický obsah. Další elementy a atributy se liší v závislosti na typu dat.

- **Polygonová vrstva**
  Každý polygon (budova, les, vodní plocha atd.) obsahuje souřadnice definičních bodů a data z atributových tabulek. Toto vše je uzavřeno do elementu, který má ve většině případů název `path` (kvůli přehlednosti je stejný jako v SVG).

- **Liniová vrstva**
  Liniové prvky (komunikace, vodní toky, železnice atd.) jsou určeny souřadnicemi lomových bodů. Obsahují také další atributy jako název nebo typ linie.

- **Bodová vrstva**
  Souřadnice pro umístění symbolů jsou uloženy v XML dokumentu popisující bodovou vrstvu. Mohou to být adresní body, označení obcí u map větších měřítek nebo body zájmu.

Protože data byla získána ze souborů *.shp* a také kvůli přehlednosti jsou tyto typy (polygony, linie a body) každý zvlášť ve svém XML souboru. Detailní pohled a rozbor zdrojových XML dokumentů je uveden pro každou mapu zvlášť.

5.2.2 Stylová část

Pomocí XSLT jsou načteny jednotlivé XML soubory, které se zúčastňují vykreslování. XSL šablona pomocí cyklu zapíše do výstupního dokumentu SVG elementy, které mají definován styl dle určitých pravidel (podmínky, klasifikace, atd.). Soubor je také opatřen SVG deklarací, cestou k JavaScipt knihovně zajišťující ovládaní mapy a dalšími důležitými prvky. Jednotlivé XML soubory jsou načítány XSL funkcí `document`. Ve stejném pořadí jako na výsledné mapě je na vrstvy aplikován příkaz `xsl:for-each`, který v cyklu provede zápis do SVG elementů. Probíhá zde také logické zpracování (podmínky, klasifikace apod.).

5.3 Popis jednotlivých aplikací

Každá z aplikací se snaží využít trochu jinou část z dovednosti SVG jazyka, aby bylo jeho využití co nejvíce a co nejnázorněji popsáno. U každého oddílu je uveden zdroj dat, způsoby jeho editace se seznamem prvků a atributů.

5.3.1 Evropa

Kartogram „Evropa“ znázorňuje podíl hrubého domácího produktu na hlavu ve vybraných státech Evropy. Odstín barvy je nastaven podle přepočtených hodnot na procenta a zobrazen pomocí barevného modelu *HSL*, kde poslední parametr je jas (lightness). U každého státu se také vyskytuje odkaz na stránku státu na serveru *cs.wikipedia.org*. Ze stejného serveru jsou také získávány miniatury vlajek států ve formátu `png`. Kartogram Evropy je v Lambertově azimutální projekci s (*LAEA Europe, EPSG:3035)*[5] a skládá označením z těchto vrstev:

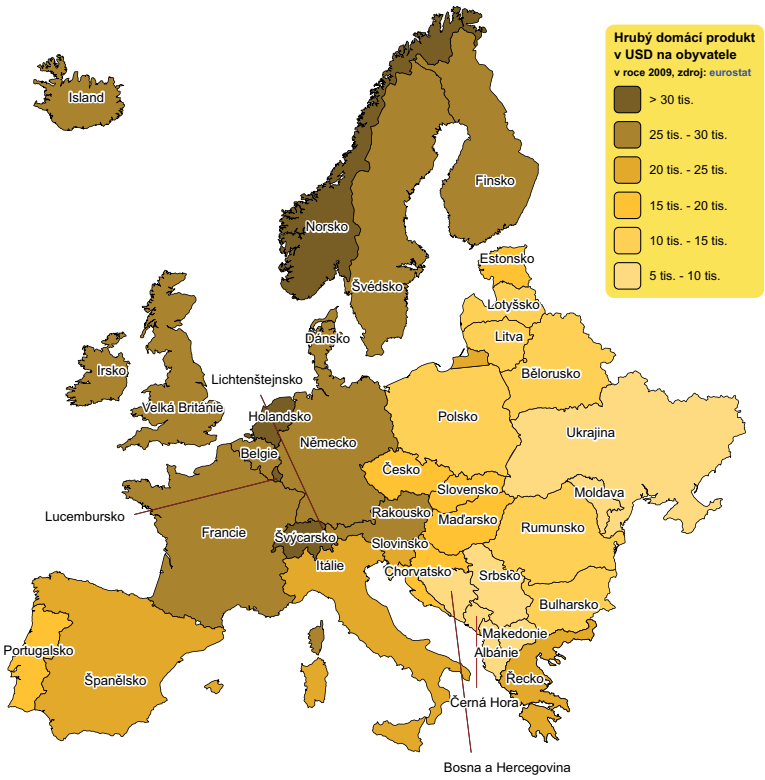
- `europe.xml`
  Každý stát je uveden v elementu `polygon`, který obsahuje tyto atributy:
  - `id` - identifikátor, zkratka státu dle normy *ISO 3166-1 (country codes)*[6], podle kterého dochází ke spojení hodnot
  - `d` - souřadnice lomových bodů hranic států
- `eurinfo.xml`
  V kořenovém elementu `info` je pro každá stát vyčleněn element `country` s těmito atributy:
  - `id` -identifikátor, kód země
  - `name` - název státu v angličtině
  - `cz` - název státu v češtině
  - `x,y` - souřadnice referenčního bodu, pro umístění názvu textu
- `euroGDP.xml`
  Data získaná ze serveru *Eurostat*[7] převedená do XML.
  - `id` -identifikátor, kód země
  - `value` - hodnota hrubého domácího produktu na obyvatele

5.3.2 Česká republika

Mapa České republiky zobrazuje územně správní jednotky kraje (NUTS 3), obce s rozšířenou působností, dále obsahuje silniční a železniční síť, vodní toky a plochy a bodovou vrstvu obcí. Data byla získána převodem z formátu *shapefile*.

- `CZECHrelief.png`
  Rastrový podklad znázorňující reliéf získaný ze serveru
  `http://www.shaded-relief.com`.
- `orp.xml`
  Souřadnice lomových bodů hranic rozšířené působnosti obcí.

[5] `http://spatialreference.org/ref/epsg/3035/`
[6] `http://www.iso.org/iso/english_country_names_and_code_elements`
[7] `http://epp.eurostat.ec.europa.eu/`



Obrázek 5.1: Kartogram Evropa, HDP na obyvatele

- `waterWays.xml`
  Liniová vrstva vodních toků obsahuje atributy:
  - `id` - název řeky
  - `length` - délka řeky, podle které se pomocí XSL určuje tloušťka vykreslení toku
  - `d` - posloupnost souřadnic lomových bodů linie
- `waterArea.xml`
  Zde jsou popsány vodní plochy (přehrady, jezera a větší rybníky).
  - `name` - název vodní plochy
  - `area_m2` - plošná výměra v m^2
  - `d` - souřadnice lomových bodů polygonu
- `roads.xml`
  Liniová vsrtva komunikací je vizualizována podle typu silnice. Zde jsou zobrazeny pouze dálnice a silnice I. třídy.
  - `type` - značí typ silnice (motorway,primary)
  - `ref` - označení komunikace
  - `maxspeed` - informace o maximální povolené rychlosti
  - `d` - souřadnice lomových bodů linií

- `railways.xml`
  Tento soubor obsahuje liniové prvky (osa železnice) a bodové prvky (železniční uzlové zastávky)

  - `way` - označení železničních tratí
  - `length` - délka tratě
  - `ele` - typ tratě (elektrifikovaná/neelektrifikovaná)
  - `kolej` - číselné označení jedno či dvojkolejné tratě
  - `d` - souřadnice lomových prvků osy železnice

  Železniční stanice uzlových tratí jsou v elementech `circle` s atributy:

  - `name` - název železniční stanice
  - `way` - název tratí
  - `cx, cy` - souřadnice (S-JTSK) železničních stanic

- `region.xml`
  V tomto souboru jsou uvedené souřadnice lomových bodů hranic krajů.

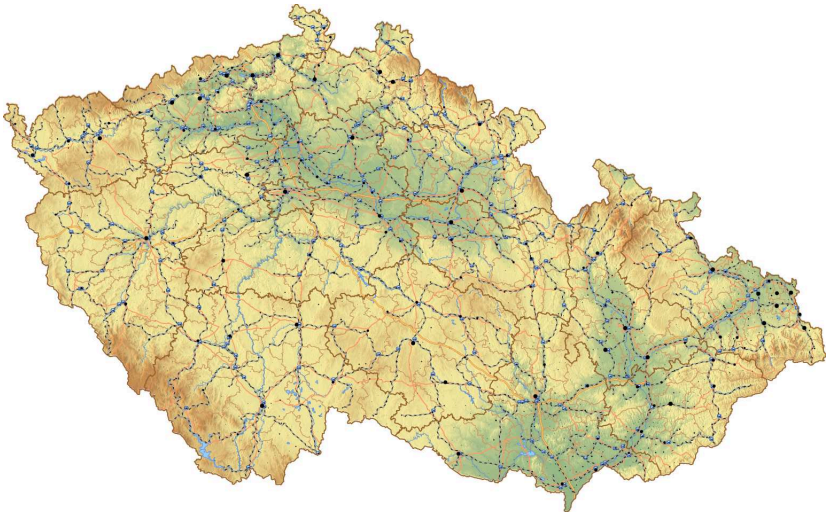
- `places.xml`
  Bodová vrstva obcí s více než 500 obyvateli.

  - `nazob` - název obce
  - `nazorp` - název příslušné obce s rozšířenou působností
  - `nk` - název kraje
  - `psc` - poštovní směrovací číslo
  - `ob91` - počet obyvatel obcí k roku 1991
  - `cx, cy` - souřadnice (S-JTSK) bodů

- `volby.xml`
  Zde je předvedena ukázka užití statistických údajů. Tento XML dokument obsahuje výsledky krajských voleb v roce 2008. Data byla převzata ze ČSÚ a převedena na XML. Vizualizace se účastní pouze nynější parlamentní strany. Kořenový element `doc` obsahuje elementy `region`, kde každý zastupuje kraj, ve kterém se volilo.

  - `id` - identifikátor, dvojmístná zkratka názvu kraje
  - `ODS,VV,TOP09,CSSD,KSCM` - zkratky politických stran, hodnoty jsou absolutní počty hlasů
  - `cx,cy` - souřadnice referenčního bodu pro umístnění grafu

  Byl zvolen sloupcový graf, jelikož jeho implementace byla nejjednodušší. Diagram by sice vyhovoval více, ale pro zpracování by byly nutné goniometrické funkce, které nejsou XSLT 1.0 definovány. Použití dalších technologií *(JavaScript, php)* by bylo v XSL nevhodné. Graf je zkonstruován z SVG elementů `rect, line, text`.



Obrázek 5.2: Ukázka kartografických prvků na mapě ČR

5.3.3 Plán kampusu ČVUT v Dejvicích

Data byla stažena ve formátu *shapefile* ze serveru *cloudmade.com*. Poté byla v programu ESRI ArcMap upravena atributová tabulka, protože některé prvky byly nekompletní. V tomto programu byla také provedeno georeferencování a transformace obou rastrových ortofotomap a jejich export do formátu JPEG. Výčet jednotlivých vrstev je proveden ve stejném pořadí jako v SVG aplikaci (malířův algoritmus).

- `DejviceBackgorund.xml`
  Polygonová vrstva obsahující podkladové prvky jako zastavěná, zatravněná či průmyslová plocha. Vrstva jde vypnout, aby mohly být využity podkladové ortofotomapy.

- `ortofoto.jpg`
  K dispozici je barevná ortofotomapa z roku 2008 od firmy GEODIS Brno přístupná na serverech `maps.google.cz` a `mapy.cz`.

- `ortofotoHistorical.jpg`
  Druhá historická ortofotomapa z roku 1953 je ze serveru `kontaminace.cenia.cz` poskytnutá VGHMÚř v Dobrušce. Obě fotomapy byly georeferencovány a transformovány na data v systému JTSK.

- `DejvicePolygon.xml`
  Plošné symboly jsou v tomto plánu použity pro vizualizaci budov, hřišť, parků nebo parkovišť. V kořenovém elementu `features` jsou polygony uvedeny jako pod tagem `path` a s atributy:

  - `name` - názvy veřejných budov nebo parků, v případě žádné hodnoty je uvedeno '`NULL`'
  - `class` - atribut udává typ zobrazovaného mnohoúhelníku, hodnoty jsou: `houses` *(rodinné domy)*, `apartment` *(byty)*, `university` *(vysoká škola)*, `park`, `parking`*(parkoviště)*, `station` *(zastávka)*, `playground` *(hřiště)*

  - `d` - souřadnice lomových bodů polygonů

- `DejviceLines.xml`
  Pod kořenovým elementem `features` jsou obsaženy tagy `path` s atributy:
  - `name` - udává název ulice či komunikace, pokud je neuvedena je atribut nastaven na '`NULL`'
  - `type` - podle tohoto atributu jsou v CSS specifikovány značky mapy
  - `d` - souřadnice lomových bodů linií

- `DejvicePoints.xml`

5.3.4 Plán města Pelhřimov

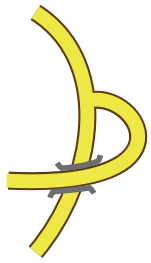
Tato část není stylována XSL transformací ani kaskádovými styly. Je to příklad exportu do formátu SVG z kartografického programu OCAD. Po exportu byly v textovém editoru jEdit odstraněny zbytečné prvky, které zvětšovaly datový objem souboru. I přes snahu o úsporu velikosti souboru je dokument pro internetové prostředí se svými přibližně 8Mb stále trochu objemnější a načítání dokumentu může trvat delší dobu. Při použití CSS by soubor mohl být menší, ale program OCAD možnost vytvoření stylových souborů nepodporuje a manuální vytvoření je obtížnější. Plán města byl vytvořen v rámci předmětu 153DIK. Součástí SVG dokumentu je mimo plánu také název, znak města a legenda.

5.4 Kartografické prvky v SVG

V této podkapitole jsou rozebrány možnosti grafického jazyka SVG ke kartografickým účelům. Postupně boudou uvedeny příklady kartografických značek a způsoby jejich zákresu pomocí SVG. U každého typu značky je také uveden obrázek a algoritmus zápisu značky v pořadí v jakém by se měl vyskytovat v SVG dokumentu.

5.4.1 Vizualizace liniových značek

Kartografické liniové značky lze v SVG vykreslit opakovaným použitím elementu `path` s odlišnými stylovými atributy, ale s identickým atributem se souřadnicemi lomových bodů. Dále zde najdou využití atributy `stroke-width, stroke-linecap, stroke-dasharray, stroke-dashoffset`.

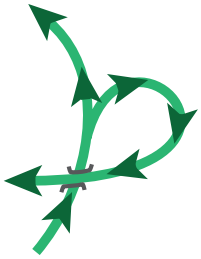


Tato značka je využívána pro vizualizaci silnic vyšších tříd. Osa komunikace je tvořena dvěma totožnými elementy `path`, s odlišnou barvou a tloušťkou tahu.
Algoritmus zápisu:

1. `<path stroke="braun" stroke-width="24px" d="..."/>`
2. `<path stroke="yellow" stroke-width="20px" d="..."/>`

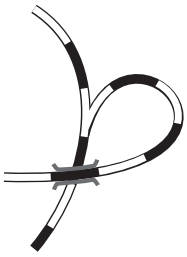
Další možnost vizualizace pozemní komunikace. Zde je přidán další element `path`, který vykresluje úzkou černou křivku.

1. `<path stroke="braun" stroke-width="24px" d="..."/>`
2. `<path stroke="yellow" stroke-width="20px" d="..."/>`
3. `<path stroke="black" stroke-width="4px" d="..."/>`



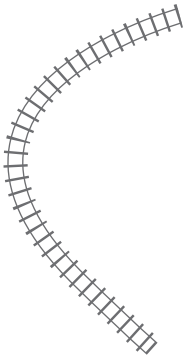
Kartografická liniová značka se šipkami může být využita u silnic nebo stezek s přikázaným směrem pohybu. Značka je vytvořena elementem `path`, který je rozdělen na úseky. V části `defs` je zapsán element `marker`, kde je definována šipka *viz. 3.3.1*

1. `<marker id="sipka" > <polygon points="..."/> </marker>`
2. Segmenty `<path marker-end="url(#sipka)" d="..."/>`



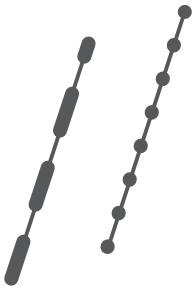
Jedním ze způsobů značení železniční tratě je využití čárkované křivky. Je tvořena dvěma elementy, kde první je vyplněn černou barvou a druhý je bílý, přerušovaný.

1. `<path stroke="black" stroke-width="24px" d="..."/>`
2. `<path stroke="white" stroke-width="20px" stroke-dasharray="20px" d="..."/>`



Další možnost vizualizace dráhy je tvořena třemi elementy. Dva elementy `path` tvoří koleje dráhy a další (mezi ně vložený) vykresluje železniční pražce.

1. `<path stroke="black" stroke-width="24px" d="..."/>`
2. `<path stroke="black" stroke-width="35px" d="..." stroke-dasharray="3 10" stroke-linecap="butt"/>`
3. `<path stroke="white" stroke-width=" 20px" d="..."/>`

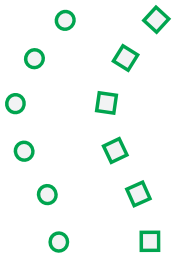


Lanovka nebo vlek je vykreslen dvěma elementy. První je plná čára a druhý je křivka čárkovaná s nastaveným atributem `stroke-linecap` na hodnotu `round`.

1. `<path stroke="black" stroke-width="5px" d="..."/>`
2. `<path stroke="black" stroke-width="25px" d="..." stroke-dasharray="40" stroke-linecap="round"/>`

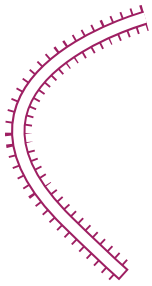
Chráněná území či oblasti jsou vyznačeny složitějším způsobem. Prvek je tvořen dvěma elementy, přičemž první je široký a čárkovaný a druhý (užší) je zvětšen a mírně posunut.

1. `<path stroke="green" stroke-width="25px" d="..." stroke-dasharray="40"/>`
2. `<path stroke="green" stroke-width="5px" d="..." transform="scale(1.1) translate(-5,-5)"/>`



Přírodní prvky jako alej nebo stromořadí může být tvořeno dvěma čárkovanými nebo tečkovanými křivkami. První křivka vykresluje zelené kruhy a druhá bílé kruhy s menším průměrem. Tím je dosaženo vykreslení mezikruží.

1. `<path stroke="green" stroke-width="25px" d="..." stroke-dasharray="1 40" stroke-linecap="round"/>`
2. `<path stroke="white" stroke-width="30px" d="..." stroke-dasharray="1 40" stroke-linecap="round"/>`



Liniová značka informující o ostrém terénním zlomu (násep, skalní linie) je tvořena třemi elementy v tomto pořadí: plná fialová čára, široká fialová čárkovaná čára a plná bílá čára.

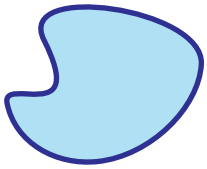
1. `<path stroke="violet" stroke-width="25px" d="..."/>`
2. `<path stroke="violet" stroke-width="35px" d="..." stroke-dasharray="1 40" stroke-linecap="butt"/>`
3. `<path stroke="white" stroke-width="20px" d="..."/>`

Státní hranici lze vykreslit pomocí tří elementů. První (šedý, široký), druhý a třetí je čárkovaný s odlišným nastavením atributu `stroke-linecap`.

1. `<path stroke="grey" stroke-width="25px" d="..." stroke-dasharray="40"/>`
2. `<path stroke="black" stroke-width="5px" d="..." stroke-dasharray="30 50" stroke-linecap="square"/>`
3. `<path stroke="black" stroke-width="5px" d="..." stroke-dasharray="1 70" stroke-linecap="round"/>`

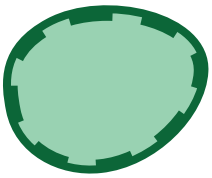
5.4.2 Vizualizace areálových značek

Kartografické plošné značky lze vytvářet dvěma způsoby. Elementem `polygon` nebo `path`. Mnohoúhelník je definován lomovými body, kdežto tvar plochy u elementu `path` lze i Beziérovými křivkami. Tento typ značek se využívá pro zákres vodních, lesních, zatravněních nebo zastavěných ploch a budov.



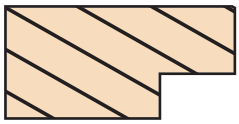
Barvou vyplněné areálové značky jsou nejjednodušší pro implementaci. Je možné využití elementu `stroke` pro zobrazení okraje.

1. `<polygon stroke="darkblue" fill="blue" points="..."/>`



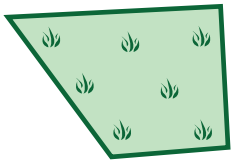
I u plošných značek lze využít formátování okraje plochy jako u liniových prvků.

1. `<polygon stroke="green" fill="lightgreen" points="..."/>`
2. `<polygon stroke="green" stroke-width="25px" stroke-dasharray="40" fill="none"/>`



Šrafování lze definovat v hlavičkové části `<defs>` elementem `pattern`, kde je specifikována linka

1. `<pattern id="srafovani" > <line x1="0" y1="0" x2="10" y2="10"/></pattern>`
2. `<polygon stroke="braun" stroke-width="25px" fill="url(#srafovani)"/>`

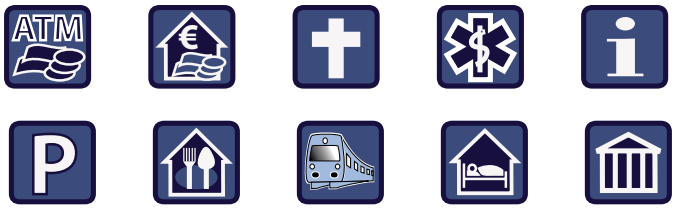


K výplni ploch lze také využít předem definované vzory. Typické využití se nalezne u vinic, lesů, parků, hřbitovů atd. V hlavičce SVG dokumentu je definován symbol a u elementu `polygon` odkaz.

1. `<symbol id="zatravneni" overflow="visible" > <path fill="darkgreen" d="..."/> <symbol/>`
2. `<polygon stroke="darkgreen" fill="url(#zatravneni)"/>`

5.4.3 Vizualizace bodových prvků

Definování a opětovné použití symbolů je SVG jednoduché. Stačí v části `defs` pod elementem `symbol` zapsat kód bodové značky a následně v dokumentu použít element `use` s udanými souřadnicemi. Celý kód je napsán v části popisu SVG elementů viz. 3.3.1. Na značky symbolů lze aplikovat všechny vykreslovací možnosti SVG. Symboly by ale měly být jednoduché, aby nebyla ohrožena jejich čitelnost. Ukázka některých autorských SVG symbolů je na obr. č. 5.3.



Obrázek 5.3: Ukázka některých symbolů vytvořených v SVG

5.5 SVG v HTML

Už bylo zmíněno, že SVG formát je určený pro internet. Existuje více způsobů začlenění SVG dokumentu do webové stránky:

- `embed` - tento element není definován ve specifikaci HTML, proto jeho použití není doporučováno
- `iframe` - vkládání interaktivního obsahu do rámu `iframe` není také doporučovanou formou (problémy s kompatibilitou)
- `object` - používaný a v této době doporučovaný způsob, příklad použití:

```
<object data="rect.svg" width="300pt" height="100pt"
        type="image/svg+xml" codebase="http://..." />
```

- `svg:namespace` - v nové specifikaci jazyka HTML 5 se počítá se zápisem SVG využitím jmenného prostoru `svg`

5.6 JavaScript

Aby aplikace byla dynamická, je třeba využít i programovací jazyk *JavaScript*. Ovládací prvky mapy *(přibližování a posun)* využívají knihovnu *svgpan*[8] od autora Andrea Leofreddi. Projekt je vystavován pod *BSD licencí*[9]. Dále se v aplikaci vyskytují autorské funkce, které pomocí *Javascript DOM* technologie zajišťují přepínání viditelnosti vrstev, doplňování atributů do tabulek a změnu načteného CSS souboru. Příklad funkce zprostředkující zápis do atributových tabulek je uveden na obrázku č. 5.4.

[8] `http://code.google.com/p/svgpan/`
[9] `http://www.opensource.org/licenses/bsd-license.php`

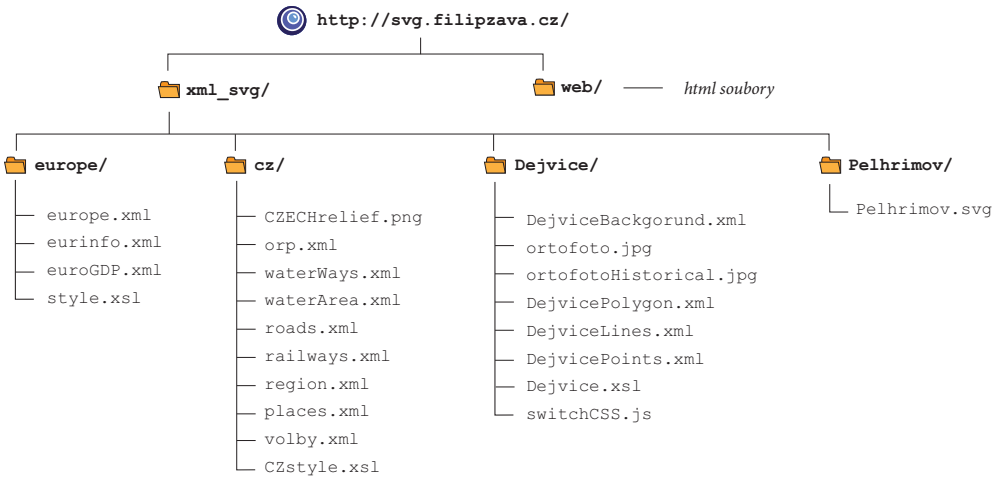
```
function setTable(evt)
{
svgDocument = evt.target.ownerDocument;
var target = evt.target;
if (target.getAttribute("Y")>10)
{
e1=svgDocument.getElementById("region");
e1.textContent=target.getAttribute("nk");
}
```

definice funkce `setTable(evt)`

uložení instance dokuemntu do proměnné `svgDocument`
uložení instance události do porměnné `target`
funkce `getAttribute()` vrátí hodnotu atributu

funkce `getElementByID()` vrátí uzel `e1` s hodnotou `id="region"`
funkce zapíše do elementu `e1` hodnotu z příslušného atributu `nk`

Obrázek 5.4: Ukázka autorské funkce v jazyce Javascript



Obrázek 5.5: Struktura adresářů na doméně `http://svg.filipzava.cz`

Kapitola 6

Závěr

Formát SVG se stal doporučením W3C už v roce 2001, za tu dobu se postupně zaváděla jeho podpora v internetových prohlížečích. Ale teprve v době psaní této práce zavedl nativní podporu světově nejpoužívanější prohlížeč MS Internet Explorer ve verzi 9. Používání SVG na svých stránkách doporučuje i wikipedia.org, kde jsou zatím převáděny do `png` formátu s odkazem k SVG obrázku. Také server Google Maps umožňuje použití SVG vrstvy nad mapou. Tyto a mnoho dalších skutečností přispívají ke stále větší oblibě tohoto formátu. Programy jako OCAD nebo ArcMap jsou dnes schopny ukládat mapy či plány do různých verzí formátu SVG. Vhodné je také postupné nahrazování proprietárního flash obsahu formátem SVG.

Co se týká kartografických prací, mohu říci, že formát SVG by mohl být nositelem dat, ovšem na trhu neexistuje propracovanější nástroj na zákres SVG map. Jeho výhoda oproti ostatním je zřejmá zejména při použití na internetu. Tam se ovšem také skrývá jedna slabina SVG. Tím, že je to formát otevřený, naskýtá pro ostatní možnost stáhnout mapu z internetu ve vektorové podobě a „vykrást“ obsah. Další nevýhodou je datová velikost SVG dokumentu. To je způsobeno textovým formátem. Soubor lze ale zkomprimovat ZIP kompresí a vytvořit tak binární soubor s koncovkou *.svgz*.

Výsledky poznávání SVG formátu jsou zpracovány v internetové aplikaci na adrese `http://svg.filipzava.cz` zobrazující mapy velkých měřítek (Evropa) , středních měřítek (Česká republika) a malých (plán města Pelhřimov, plán kampusu ČVUT v Dejvicích). K výslednému efektu musely být použity další technologie, kromě SVG také JavaScript(změna zoom, posouvání a tisk atributových tabulek), XSL a CSS (stylové jazyky). Větší rozdíly ani problémy s kompatibilitou v různých prohlížečích nebyly při vykreslování SVG grafiky pozorovány (při použití aktuálních verzí).

Literatura

[Eis02] J. David Eisenberg. *SVG Essentials*. O´Reilly Media, 2002.

[Her10] Pavel Herout. *XSLT 2.0 a SVG prakticky*. Kopp, 2010.

[Kos00] Jiří Kosek. *XML pro každého, podrobný průvodce*. Grada Publishing, 2000.

[Sho00] William Shoaff. A Short History of Computer Graphics. `http://www.cs.fit.edu/~wds/classes/graphics/History/history`, June 2000.

[Tiš07] Pavel Tišnovský. Vektorové grafické formáty a metaformáty. `http://www.root.cz/clanky/vektorove-graficke-formaty-a-metaformaty/`, March 2007.

[Wil09] Ian Williams. *Beginning XSLT and Xpath*. Wrox, 2009.

Rejstřík

bézierova křivka, 4

HSL, 25

metadata, 31

RGB, 25

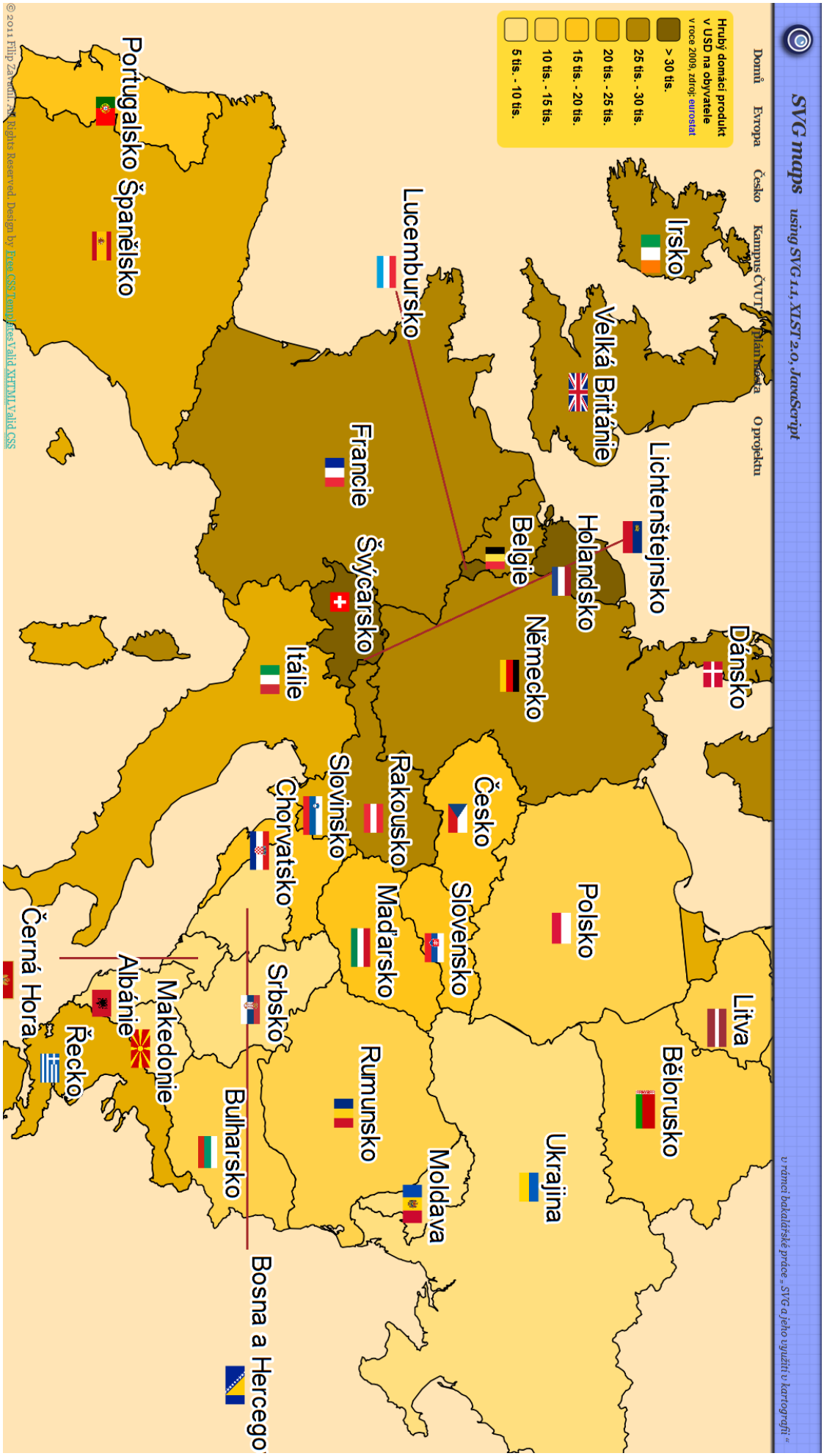
SVG, 12
- circle, 17
- colors, 24
- defs, 13
- display, 24
- ellipse, 17
- fill, 24
- fill-opacity, 24
- fill-rule, 24
- filter, 28
- id, 14
- line, 17
- linearGradient, 15
- marker, 14
- metadata, 32
- opacity, 24
- path, 18
- polygon, 18
- polyline, 18
- radialGradient, 16
- rectangle, 17
- souřadnicový systém, 26
- stroke, 22
- stroke-dasharray, 23
- stroke-dashoffset, 23
- stroke-linecap, 23
- stroke-linejoin, 23
- stroke-miterlimit, 23
- stroke-opacity, 23
- stroke-width, 22
- symbol, 14
- text, 20
- textPath, 22
- transform, 26
- tspan, 21
- use, 14
- visibility, 24

XML
- atribut, 10
- CSS, 9, 33
- element, 9
- parser, 9
- tag, 9
- XSL, 9

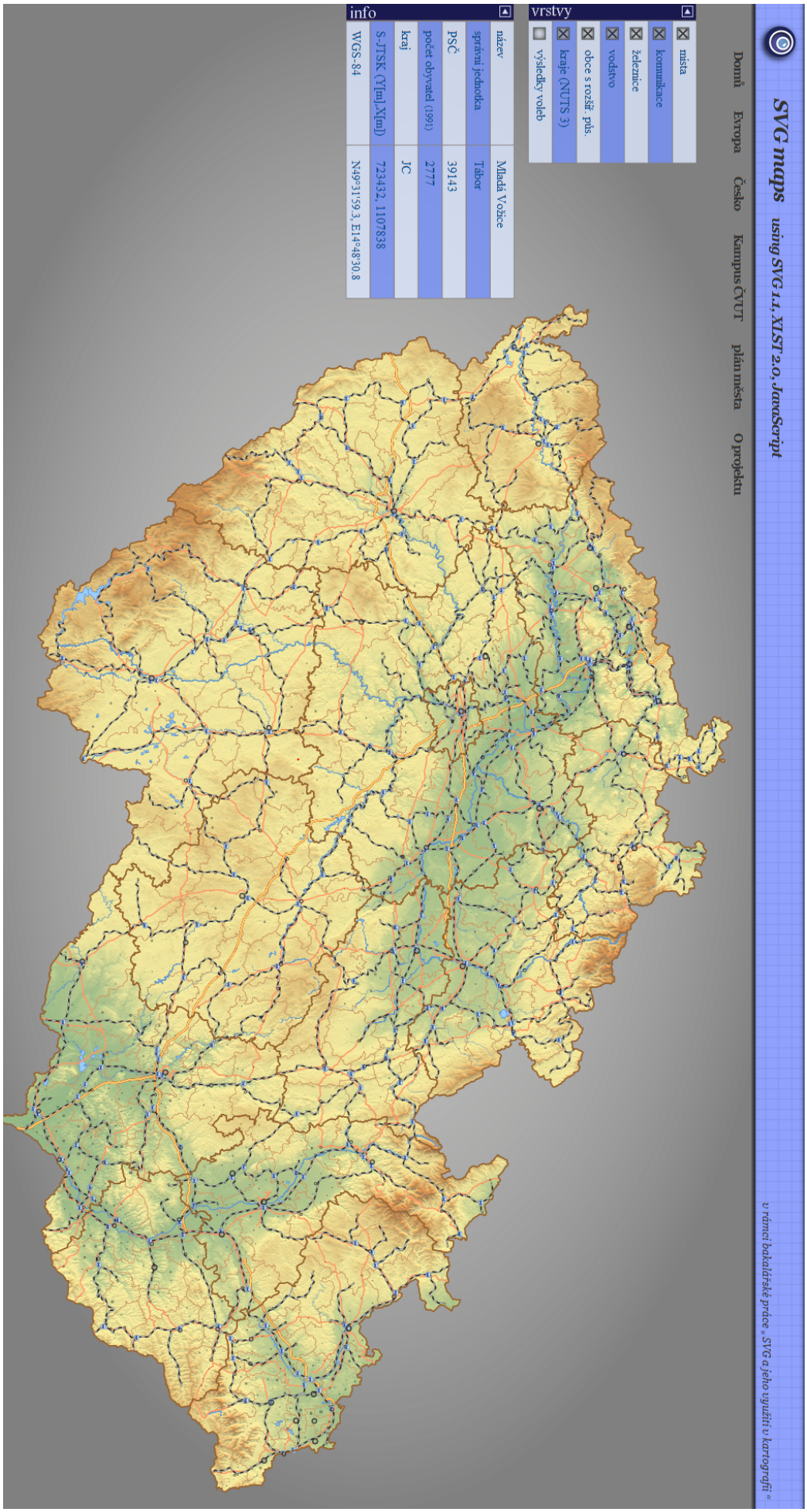
Xpath, 36

XSL, 34
- choose, 38
- for-each, 39
- header, 35
- if, 38
- value-of, 36
- variable, 39

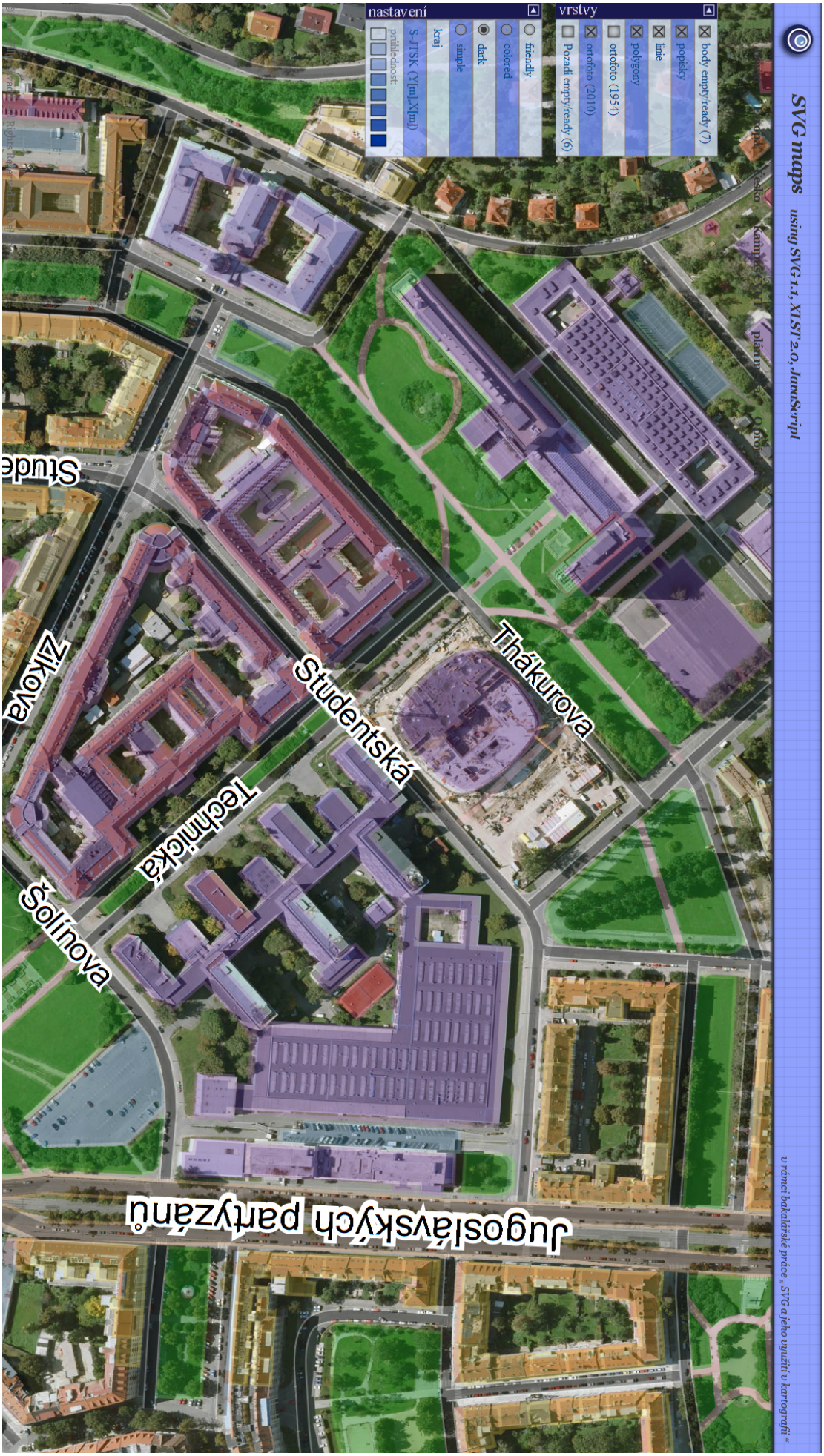
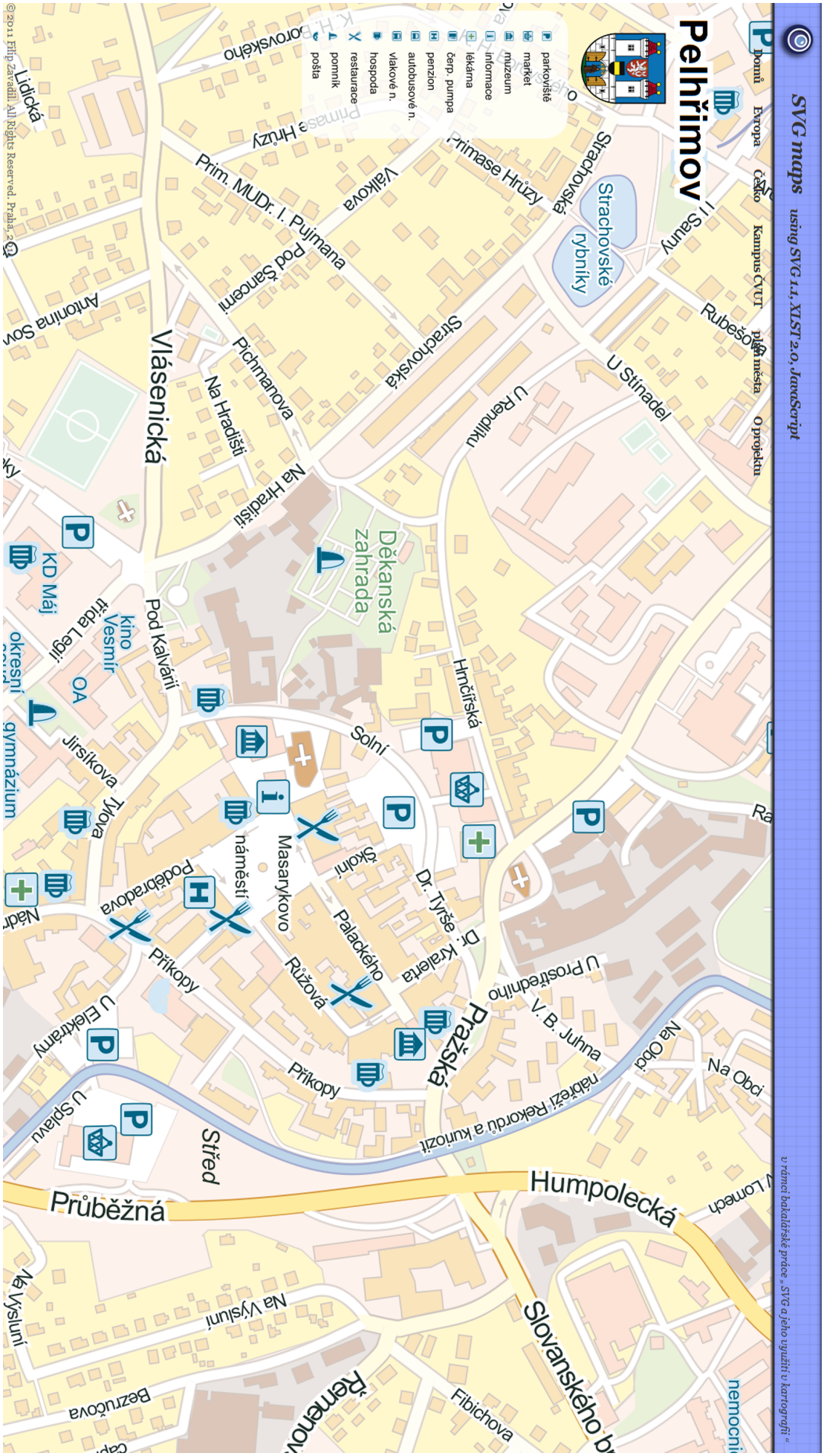
Příloha A - GUI aplikace, Kartogram Evropa



Příloha B - GUI aplikace, mapa České republiky



Příloha C - GUI aplikace, plán kampusu ČVUT v Praze



Příloha D - GUI aplikace, Plán města Pelhřimov