Pedagogická fakulta
Faculty of Education

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v Českých Budějovicích
Pedagogická fakulta
Katedra informatiky

Bakalářská práce

Mapový server pro potřeby malé obce

Vypracoval: Jakub Macillis
Vedoucí práce: Ing. Tomáš Dolanský, Ph.D.

České Budějovice 2014

Anotace

Tato bakalářská práce se zabývá možnostmi zobrazení mapových podkladů pro webové stránky, například pro weby obcí a měst, a porovnává jednotlivé možnosti publikace. Důraz je kladen na finanční náročnost konečného řešení a práce se tedy zaměřuje na opensource řešení, které je dostupné za minimální a většinou nulové náklady. Práce dále obsahuje porovnání opensource řešení s dalšími vybranými technologiemi pro zobrazení mapových dat. Obsahuje popis webové aplikace, kterou uživatel nalezne na webové stránce, a umožňuje tak snadné porozumění jednotlivým částem aplikace. V další části se zabývá publikačními aplikacemi, které dodávají aplikaci mapové podklady. Věnuje se také technické stránce, obsahuje porovnání rychlosti mapového serveru a lokálně uložených souborů o velikosti dat malé obce. Součástí práce je ukázkové řešení aplikace zakomponované do webové stránky obce s popsaným kódem využívajícím výhodnější technologie, která vzešla z porovnání.

Klíčová slova

mapa, webová stránka, publikace mapových dat, Openlayers, mapový server

Abstract

This thesis focuses on displaying of map basis on the websites of towns and cities and it compares several possibilities of how the map bases are published. The emphasis is placed on the costs of the final solution, so the thesis is directed to find an open source solution. The thesis includes the comparison of JavaScript libraries, comparing their pros and cons and describing the source code of the application created using the library chosen in the comparison. The second part of the thesis is focused on the map publishing applications and the hardware, which is going to be running the map server. It includes the comparison of the speed of map server and locally stored data of the small town. Practical part of the thesis includes the sample solution of the web application made for a small town website with faster technology chosen in the comparison.

Keywords

map, website, map data publication, Openlayers, map server

Prohlášení

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

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. V platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě fakultou elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích

dne: ....

Podpis

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH
Fakulta pedagogická
Akademický rok: **2012/2013**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jakub MACILLIS**
Osobní číslo:	**P11053**
Studijní program:	**B7507 Specializace v pedagogice**
Studijní obor:	**Informační technologie a e-learning**
Název tématu:	**Mapový server pro potřeby malé obce**
Zadávající katedra:	**Katedra informatiky**

Zásady pro vypracování:

Při zpracování bakalářské práce bude student postupovat následujícím způsobem:

1. Shromáždí informace o možných řešeních mapových serverů a klinetů pro webové prohlížeče. Prostuduje základní literaturu o OpenLayers, MapGuide a ArcGIS Server.
2. Student provede porovnání technologií OpenLayers, Google Maps API případně i Mapy API z hlediska prohlížení mapových dat malé obce. Pro vybranou technologii vytvoří vzorový kód pro jednoduché využití obcemi.
3. Student porovná a vyzkouší alespoň dva mapové servery pro publikování mapových dat malé obce. Zároveň stanoví hw a sw požadavky na server se zaměřením na hostované servery.
4. Student po analýzách mapového serveru, prohlížecí technologie a hostovaného serveru navrhne a zrealizuje běh mapového serveru a publikování dat konkrétní malé obce.

Cíle práce:
Analýza technologií pro zobrazení mapového obsahu na webových stránkách.
Analýza mapových serverů a instalace vybraného řešení na hostovaný server.
Návrh komplexního řešení mapového serveru a publikace mapových dat pro malou obec s důrazem na dostupnost a cenu služeb.
Realizace a ověření navrženého řešení pro vybranou malou obec ve formě mapového portálu.
Publikování výsledků práce v časopise nebo na konferenci.

Rozsah grafických prací: **CD ROM**

Rozsah pracovní zprávy: **40**

Forma zpracování bakalářské práce: **tištěná**

Seznam odborné literatury:

1. **FU, Pinde, SUN, Jiulin. Web GIS: Principles and Applications. ESRI Press, Redlands, 2011. ISBN 978-1-58948-245-6**
2. **FLANAGAN, David. JavaScript: kapesní příručka. Gliwice: Helion, 2004. ISBN 83-7361-466-4.**
3. **MITCHELL, Tyler. Web mapping illustrated: [using open source GIS toolkits]. 1st ed. Beijing [u.a.]: O'Reilly, 2005. ISBN 978-059-6008-659.**
4. **GOOGLE. Google Maps API: Google Developers [online]. 2012 [cit. 2013-03-06]. Dostupné z: https://developers.google.com/maps/**
5. **SEZNAM.CZ. API Mapy.cz [online]. 2013 [cit. 2013-03-06]. Dostupné z: http://api4.mapy.cz/**
6. **CHAMPEON, Steve. JavaScript: How Did We Get Here?. In: O'Reilly Media [online]. 2001 [cit. 2013-03-06]. Dostupné z: http://www.oreillynet.com/pub/a/javascript/2001/04/06/js_history.html**
7. **OPENLAYERS. What is OpenLayers? [online]. 2008 [cit. 2013-03-06]. Dostupné z: http://docs.openlayers.org/index.html#**

Vedoucí bakalářské práce: **Ing. Tomáš Dolanský, Ph.D.**
Katedra informatiky

Datum zadání bakalářské práce: **16. dubna 2013**

Termín odevzdání bakalářské práce: **30. dubna 2014**

Mgr. Michal Vančura, Ph.D.
děkan

L.S.

doc. PaedDr. Jiří Vaníček, Ph.D.
vedoucí katedry

V Českých Budějovicích dne 16. dubna 2013

Poděkování

Rád bych poděkoval mé rodině, která mi pomáhala v celém průběhu mého studia a také při tvorbě této bakalářské práce, za její trpělivost, ochotu, rady a v neposlední řadě také finanční zajištění studia.

Dále bych rád poděkoval vedoucímu práce, panu Ing. Tomáši Dolanskému, Ph.D. za vedení během tvorby práce, čas, který mi věnoval během konzultací i mimo ně a mnoho cenných rad, které mi během tvorby práce poskytl.

Jako poslední bych rád poděkoval všem, kteří mi během tvorby práce pomáhali a poskytovali své rady.

Obsah

1 Úvod

Publikování mapových podkladů je v posledních letech převážně zaměřeno na mapové portály, jakými jsou například Mapy.cz společnosti Seznam.cz a Google Mapy amerického webového gigantu Google. Tyto portály umožňují vyhledávat nejrůznější objekty a adresy, zobrazovat satelitní snímky, dopravu a dokonce i fotografické zobrazení ulic. Často také na tyto portály narazíte na webových stránkách obcí i ostatních institucí, ať již jako pouhý odkaz, nebo v podobě vloženého objektu samotné mapy.

Implementace těchto portálů je relativně jednoduchá a nabízí pro návštěvníky plný komfort výše zmíněných webů. Někdy ale standardní mapy nestačí a je potřeba využít vlastních mapových podkladů. Nejčastěji se s tímto setkáváme u obcí, které potřebují svým občanům nabídnout například katastrální mapy, mapy infrastruktury, bezpečnostní mapy, a jiné podobné speciální mapové podklady.

Vývojem takovýchto webových aplikací se zabývá mnoho komerčních firem, ale i když si větší města takovéto nabídky mohou dovolit, menší obce, které často ve svých rozpočtech nemají dostatek finančních prostředků na okrajové položky, jakými jsou například webové stránky a mapové servery, si takovouto zakázku dovolit často nemohou. Pro tyto potřeby ale existují knihovny, které umožní publikaci vlastních mapových podkladů a jsou dostupné zdarma, jako opensource, nebo za opravdu minimální náklady. Překážkou k implementaci takovýchto knihoven však může být obava z neznalosti složitých technických náležitostí, ovšem opak může být pravdou.

Základní syntaxe těchto knihoven je většinou v jazyce JavaScript, který na první pohled může vypadat složitě, ale ve skutečnosti se díky pevné struktuře knihoven jedná o opakující se a snadno pochopitelné části kódu. Personalizace je většinou ještě jednodušší, ve většině případů se jedná o úpravu CSS souboru známých ze stylování webových stránek. Další možností, jak webovou aplikaci upravit je přes existující upravené třídy, nebo celé nadstavbové knihovny, jakou je například GeoExt postavená nad OpenLayers, která umožní vytvoření aplikace profesionálního vzhledu.

Publikace mapových podkladů na webových stránkách však není jen otázkou rozhraní, které je zobrazuje. Data, která mají být zobrazena, jí je nutné nějak předat. Pro tento způsob je možné využít mapového serveru, který bude data zasílat ze svého disku, či je možné data uložit společně s aplikací do webového prostoru ve formátu KML.

1.1 Cíle práce

Prvním cílem této práce je porovnat knihovny pro publikaci mapových dat na internetu, jakými jsou vybrané Google Maps API, Mapy API, nebo Openlayers. Toto porovnání poté zpracovat ve formě popisu jednotlivých kritérií a stanovit nejvhodnější knihovnu pro použití ve formě mapového portálu malé obce.

Druhým cílem práce je analyzovat rozdíly v rychlosti zpracování dat přes mapový server a za pomoci KML souborů. Toto porovnání poté popsat za pomoci měření na několika sestavách a vrstvách dat.

Dva předchozí cíle poté budou využity při vytvoření praktické části práce, a tím i jejího třetího cíle, ukázkové webové aplikace. V rámci tohoto cíle vytvořit aplikaci, která bude mít požadované funkční náležitosti a následně tuto aplikaci popsat tak, aby bylo možné z popisu aplikaci upravit pro použití jinou obcí.

1.2 Metoda práce

Pro vypracování prvního cíle práce byla stanovena kritéria s ohledem na požadované funkce aplikace. Tato kritéria byla následně obodována a rozdělena do následujících kategorií.

- Možnosti implementace vlastních mapových podkladů
- Nabídka ovládacích prvků a jejich implementace
- Možnosti editace a úprava vzhledu
- Licenční podmínky a použití
- Rozsah poskytované dokumentace a podpory

Tyto kategorie byly následně zohledněny za pomoci vah. Jednotlivá kritéria byla následně za pomoci oficiální dokumentace a materiálů vyhodnocena a výsledně obodována.

Při zpracování druhého cíle jsem se zaměřil na dostupná opensource řešení pro realizaci mapového serveru. Tato řešení jsem otestoval na připraveném hostovaném serveru, který byl pro tento účel zřízen. Po instalaci bylo následně publikováno několik vrstev přes mapový server a porovnány jejich načítací časy s řešením, kdy jsou data uložena v souborech KML.

V poslední části byla spojena vybraná technologie publikace s aplikací a sestaven ukázkový mapový server pro vybranou obec. Programování aplikace probíhalo postupným přidáváním jednotlivých funkcí, které odpovídají požadavkům obce. Například implementace upraveného přepínače vrstev, který přepíná aktivní vrstvu, zobrazení legendy k jednotlivým vrstvám, nebo výpis údajů o vrstvě, po kliknutí do ní. Po vytvoření aplikace byl celek publikován na webový prostor a důkladně otestován. Výslednou aplikaci byla následně popsána tak, aby bylo pochopitelné, co která část kódu dělá a jak ji upravit pro odlišné implementace.

2 Porovnání API pro zobrazení map na internetu

Pod zkratkou API (Application Programming Interface) se nachází označení pro soubor nástrojů, metod a funkcí pro vytvoření aplikace. V rámci aplikací pro zobrazení mapových dat jsou tato API, mimo jiné, napsána také v jazyce Javascript. Výhodou tohoto jazyka je kvalitní podpora ve všech aktuálně nejpoužívanějších prohlížečích. Javascriptové aplikace bez problémů podporuje Mozilla Firefox, Google Chrome, ale také Internet Explorer, či Safari od společnosti Apple. Podporu Javascript nachází také v segmentu pokročilých mobilních prohlížečů pro mobilní zařízení na platformách Android, Windows Phone a také iOS. Podpora Javascriptu v rámci prohlížečů je tedy na velmi dobré úrovni.

Vytvoření aplikace však není jen otázkou, jak je daná technologie podporována. Pro kvalitní aplikaci je také nutné vybrat správné API, které nabídne ty správné funkce pro potřeby malé obce. Tedy minimální náklady na zobrazení a požadavek několika vlastností k zobrazení dat. Pro tyto potřeby byly vybrány tři API, které každé reprezentují jiný segment. Google Maps API, společnosti Google, které zastupuje velké a profesionální API, Mapy API od společnosti Seznam.cz, které ukazuje, jaké může mapové API vzniknout v České republice, a Openlayers, populární open source řešení.

2.1 Příprava hodnocení

Pro samotné porovnání bylo následně stanoveno pět kategorií, do kterých byla rozdělena jednotlivá kritéria.

- Mapové podklady (k_1)
- Ovládací prvky (k_2)
- Přizpůsobení vzhledu (k_3)
- Licence (k_4)
- Dokumentace (k_5)

Kritéria byla následně obodována dle důležitosti a důležitost jednotlivých kategorií byla poté vyzdvižena za pomoci vah. Tyto váhy byly vypočítány použitím metody párového porovnání a výsledné bodové ohodnocení v každé kategorii bylo následně touto váhou vynásobeno.

Kategorie	k_1	k_2	k_3	k_4	k_5	Počet preferencí	Pořadí kritéria	Váha kritéria
k_1		1	1	1	1	4	1	5
k_2			2	2	2	3	2	4
k_3				4	5	0	5	1
k_4					5	1	4	2
k_5						2	3	3

Tabulka 1 Kategorie použité v rámci porovnání společně s jejich preferencemi a váhami

2.2 Kategorie porovnání

Jak již bylo zmíněno, v rámci porovnání je použito 5 kategorii, kdy každá obsahuje jednotlivá kritéria. První z nich je kategorie s názvem Mapové podklady, která ukazuje možnost zobrazení vlastních mapových podkladů v rámci API. Výsledné bodové hodnocení této kategorie dává najevo, jak API zvládá zobrazit externí mapové podklady za pomoci tří zásadních technologií. Tyto jednotlivé technologie tvoří tři kritéria této kategorie.

Kritérium	Bodové ohodnocení
API umí zobrazit mapové podklady přes WMS vrstvu	0 bodů / 4 body
API umí zobrazit vektorové vrstvy	0 bodů / 4 body
API umí zobrazit rastrové vrstvy	0 bodů / 4 body
Maximální bodový zisk	**12 bodů**

Tabulka 2 Bodové ohodnocení kritérii kategorie Mapové podklady

Druhá kategorie Ovládací prvky hodnotí možnosti a počet možných ovládacích prvků, které dané API nabízí. V rámci této kategorie jsou použita kritéria, která značí potřebu, které ovládací prvky by měla aplikace pro malou obec mít. Výsledné bodové ohodnocení této kategorie jednoduše ukazuje, zda API daný ovládací prvek nabízí, či nikoliv.

Kritérium	Bodové ohodnocení
API umí přidat ovládání pro posun mapou	0 bodů / 8 bodů
API umí přidat ovládání pro přibližování	0 bodů / 8 bodů
API umí přepínat viditelnost jednotlivých vrstev	0 bodů / 2 body / 8 bodů
API umí měřit vzdálenost a vzdálenost zobrazit v rámci obrazovky	0 bodů / 2 body
API umí měřit plochu a plochu zobrazit v rámci obrazovky	0 bodů / 2 body
API umí vybrat prvek z mapy a zobrazit o něm informace	0 bodů / 3 body
API umí strukturovat vrstvy pro zobrazení například v podobě stromu	0 bodů / 4 body
API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko	0 bodů / 7 bodů
API umí exportovat mapu a informace, které obsahuje	0 bodů / 4 body
Maximální bodový zisk	**46 bodů**

Tabulka 3 Bodové ohodnocení kritérii kategorie Ovládací prvky

Další kategorií je kategorie Přizpůsobení vzhledu. Kritéria v této kategorii hodnotí, jaké možnosti nabízí API pro své vlastní stylování a změnu vzhledu. Bodové ohodnocení z této kategorie ukazuje, nakolik je API přívětivé ke změně svého vzhledu.

Kritérium	Bodové ohodnocení
API umí změnit své barevné schéma	0 bodů / 4 body
API umí editovat své prvky za pomoci CSS	0 bodů / 8 bodů
Maximální bodový zisk	**12 bodů**

Tabulka 4 Bodové ohodnocení kritérii kategorie Přizpůsobení vzhledu

V předposlední kategorii jsou kritéria zaměřena na licenční podmínky. Kritéria hodnotí, jak finančně náročné je použití daného API a zda má toto použití nějaké omezení. Bodové hodnocení této kategorie je vyhodnoceno vždy v rámci každého kritéria, což znamená, že API spadá vždy pod jediné kritérium, a jeho bodový zisk z této kategorie je hodnota daného kritéria.

Kritérium	Bodové ohodnocení
API je dostupné zdarma ve formě Open Source	0 bodů / 8 bodů
API je dostupné zdarma s omezením	0 bodů / 4 body
API je dostupné komerčně	0 bodů
Maximální bodový zisk	**8 bodů**

Tabulka 5 Bodové ohodnocení kritérii kategorie Licence

Poslední kategorií v rámci hodnocení je Dokumentace. Kritéria v této kategorii hodnotí, zda dokumentace API popisuje veškeré důležité a základní prvky API. Hodnocení dokumentace probíhá kontrolou dokumentace hlavních prvků a funkcí, doplněných o kontrolu dokumentace částí API potřebných pro vytvoření aplikace dle předchozích kritérií. Dalším kritériem v rámci dokumentace je přítomnost doplňkových materiálů, jakými mohou být například tutoriály, ukázkové aplikace, apod. Bonusové body získá API za nabídnutí těchto materiálů v českém jazyce.

Kritérium	Bodové ohodnocení
Dokumentace pokrývá všechny části API	0 bodů / 2 body / 4 body / 6 bodů
V rámci dokumentace jsou dostupné i další materiály	0 bodů / 6 bodů
Materiály jsou dostupné v češtině	2 body
Maximální bodový zisk	**14 bodů**

Tabulka 6 Bodové ohodnocení kategorie Dokumentace

2.3 Popis jednotlivých API

Popis jednotlivých API je zaměřen na krátké textové odůvodnění každého kritéria v rámci hodnocení.

2.3.1 Google Maps API

2.3.1.1 API umí zobrazit externí mapové podklady

API Google Map je primárně určeno pro využívání mapových podkladů společnosti Google a jejích dodavatelů. V základní nabídce jsou samozřejmostí zobrazení

v podobě silniční mapy, satelitních snímků, hybridního zobrazení a zobrazení terénu a další doplňkové služby a vrstvy, které jsou dobře známé ze standardních Google Map.

Pokud se jedná o externí mapové podklady, tak API Google Map zvládne zobrazit rastrová data v podobě obrázkové překryvné vrstvy. Data mohou mít formát jednoho obrázkového souboru, který se zobrazí nad základní mapou, či sady předem připravených obdélníkových dlaždic, které jsou rozděleny dle svých souřadnic a obsáhnou i jednotlivé úrovně přiblížení.

Dalším způsobem, jak v Google Maps API zobrazit externí data, jsou vektorové vrstvy. Google Maps API podporuje KML formát, který lze do mapy načíst přes třídu, která je pro tento formát přímo určena. I přes to, že KML je prakticky formát společnosti Google, existuje v API několik omezení v jeho používání, které se týká například maximální velikosti souborů, počtu prvků, nebo počtu vrstev, které KML využívají. Dle dokumentace[1] jsou tyto limity dočasné a mohou se měnit.

Zpracování WMS služby formou vlastní knihovny Google Maps API neumožňuje a je nutné využít stejného způsobu zobrazení, jako v případě rastrových dat.

2.3.1.1.1 Hodnocení

API umí zobrazit mapové podklady přes WMS vrstvu	0 bodů
API umí zobrazit vektorové vrstvy	4 body
API umí zobrazit rastrové vrstvy	4 body
Celkem	**8 bodů**

Tabulka 7 Bodové ohodnocení možností zobrazení vlastních mapových podkladů

2.3.1.2 Ovládací prvky

2.3.1.2.1 API umí přidat ovládání pro posun mapou

Ovládání posunu mapy by mělo být pro každé API samozřejmostí a Google Maps API jej samozřejmě nabízí. Nastavení ovládání umožňuje prosté posouvání mapy za pomoci myši nastavením parametru **draggable** na hodnoty true (mapa může být

[1] KML Support in Google Maps. *Google Developers* [online]. 2013 [cit. 2014-04-01]. Dostupné z: https://developers.google.com/kml/documentation/mapsSupport?hl=cs&csw=1

posouvána za pomoci myši), či false (mapu nelze posouvat za pomoci myši), či zobrazení ovládacího kříže příkazem **panControl**, který je použit při vytváření mapy v jejím nastavení a nabývá hodnotu true (ovládání zobrazeno), či false (ovládání nezobrazeno). Pro ovládací kříž lze poté nastavit jeho pozici do několika pozic mapového panelu.

2.3.1.2.2 API umí přidat ovládání pro přibližování

Ovládání přiblížení zvládá Google Maps API podobně dobře jako posun mapy. Pomoci příkazu **scrollwheel** umožňuje zakázat či povolit přiblížení pomoci kolečka na myši. Dalšími možnostmi jsou například příkazy pro nastavení minimálního a maximálního přiblížení, či přidání ovládacího prvku na obrazovku. Ten může být umístěn v předpřipravených pozicích a nabývat dvou různých podob, a to tlačítek pro přiblížení a oddálení a přiblížení formou slideru.

2.3.1.2.3 API umí přepínat viditelnost jednotlivých vrstev

Přepínání vrstev řeší Google Maps API za pomoci třídy **mapTypeControl**, která umožní přepínat standardní typy map, které API nabízí, tedy silniční mapu, satelitní zobrazení atd. Pokud však chceme implementovat vypínání a zapínání jednotlivých vrstev i pro externí mapová data, se kterými API nepočítá, je také nutné vytvořit vlastní mapový typ, nebo vytvořit metodu, která bude vrstvy přepínat. API tedy v základu neumožňuje přepínání vrstev, které se sestávají z externích dat a je nutné doplnit mapové typy, či metody, které toto umožní.

2.3.1.2.4 API umí měřit vzdálenost a plochu a zobrazit je v rámci obrazovky

Google Maps API neobsahuje nástroje, které zajišťují přímé měření postupným klikáním do mapy. Tato skutečnost se týká měření vzdálenosti, i měřením plochy.

2.3.1.2.5 API umí vybrat prvek z mapy a zobrazit o něm informace

Vybrání a zobrazení informací z prvku na mapě je možné v rámci KML vrstvy, a to zpracováním popisu, který KML soubor obsahuje a jeho následným zobrazením v rámci "bubliny" na mapě, či zobrazit do některého z HTML elementů.

V případě potřeby získání informací ze služby WMS je nutné využít vlastní způsob, jelikož Google Maps API WMS vrstvu, a tudíž ani výběr z této vrstvy neumí.

2.3.1.2.6 API umí strukturovat vrstvy pro zobrazení například v podobě stromu

Google Maps API obsahuje pouze jednoduchý přepínač základních mapových typů, který zobrazuje jednotlivé vrstvy v řadě, či, v případě výběrového menu, rolovací seznam. V rámci tohoto přepínače API zobrazí také obsažené překryvné vrstvy, které daný typ nabízí, ale ty však nijak nestrukturuje. Strukturovaný seznam vrstev, či jejich členění do stromu není tedy v Google Maps API k dispozici.

2.3.1.2.7 API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko

Vzhledem k absenci pokročilého přepínače vrstev zpracování aktivní vrstvy pro výběr prvků Google Maps API neumožňuje. Tuto funkčnost je tedy nutné v případě potřeby do aplikace implementovat externě.

2.3.1.2.8 API umí exportovat mapu a informace, které obsahuje

Export současného stavu mapy není v Google Maps API možný. Google nabízí Static Maps API V2, které umožní vygenerovat statický obraz mapy, ale toto API je striktně omezeno a dle jeho licenčních podmínek musí být snímek na stránku umístěn za pomoci speciálního odkazu a nesmí být dále ukládán a používán.

2.3.1.2.9 Hodnocení

API umí přidat ovládání pro posun mapou	8 bodů
API umí přidat ovládání pro přibližování	8 bodů
API umí přepínat viditelnost jednotlivých vrstev	2 body
API umí měřit vzdálenost a vzdálenost zobrazit v rámci obrazovky	0 bodů
API umí měřit plochu a plochu zobrazit v rámci obrazovky	0 bodů
API umí vybrat prvek z mapy a zobrazit o něm informace	3 body
API umí strukturovat vrstvy pro zobrazení například v podobě stromu	0 bodů
API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko	0 bodů
API umí exportovat mapu a informace, které obsahuje	0 bodů
Celkem	**21 bodů**

Tabulka 8 Bodové ohodnocení ovládacích prvků

2.3.1.3 Přizpůsobení vzhledu

2.3.1.3.1 API umí změnit své barevné schéma

Barevné schéma Google Maps API lze měnit pouze, pokud jej chceme aplikovat na mapu samotnou. To znamená, že lze měnit barevné odstíny ulic, vodních ploch, apod. Tyto styly však lze aplikovat pouze na defaultní mapové typy. Pro stylování ovládacích prvků je nejdříve nutné vytvořit své vlastní kopie ovládání a odstranit ty původní.

2.3.1.3.2 API umí editovat své prvky za pomoci CSS

V rámci Google Maps API lze stylovat za pomoci CSS pouze element div, ve kterém se nachází mapa, a to standardním způsobem, jakým se element div v rámci CSS styluje. Pro ostatní prvky není v API dostupná žádná CSS třída a je nutné vytvořit vlastní reprezentaci daného prvku, kterou lze poté stylovat.

2.3.1.3.3 Hodnocení

API umí změnit své barevné schéma	0 bodů
API umí editovat své prvky za pomoci CSS	0 bodů
Celkem	**0 bodů**

Tabulka 9 Bodové ohodnocení úpravy vzhledu

2.3.1.4 Licence

Google Maps API je dostupné ve dvou licenčních verzích. První, která je dostupná zdarma, nabízí veškeré funkce API, ale v jeho užívání se od druhé, placené verze (Google Maps API for Business), liší v nesčetných limitech na počet požadavků, které může aplikace vykonat, například 25 tisíc načtení v jeden den, či 2500 požadavků na geokódování (převod adresy na zeměpisné souřadnice). Další omezení se týká použití Google Streetview a statických map, které lze zobrazit v nižším rozlišení (640 x 640 oproti 2048 x 2048). Omezení se také dotklo podpory, která je u bezplatné verze omezena pouze na dokumentaci API dostupnou na webových stránkách API. Placená verze má k dispozici technickou podporu se speciálním portálem. Pro využití obou verzí API je nutné, aby výsledná aplikace byla koncovému uživateli nabízena zdarma a bez omezení přístupu. Plné a aktuální znění licenčních podmínek lze nalézt na webových stránkách API[2].

2.3.1.4.1 Hodnocení

API je dostupné zdarma ve formě Open Source	0 bodů
API je dostupné zdarma s omezením	4 body
API je dostupné komerčně	0 bodů
Celkem	**4 body**

Tabulka 10 Bodové ohodnocení licenčních podmínek

[2] Google Maps/Google Earth APIs Terms of Service. *Google Developers* [online]. 2013 [cit. 2014-04-01]. Dostupné z: https://developers.google.com/maps/terms

2.3.1.5 Dokumentace

2.3.1.5.1 Dokumentace pokrývá všechny části API

Dokumentace Google Maps API je popsána v rámci jedné webové stránky a nemá tedy složitější strukturovanou formu. Popis jednotlivých částí je většinou řešen za pomoci tabulky, která obsahuje název metody, či hodnoty a její popis, případně argumenty a návratové hodnoty. Popis je však řešen velice stručně a v některých případech chybí úplně, a to nejen u těch nejméně používaných funkcí, ale popis se nenachází ani u funkcí, které by se daly považovat za středně používané, jakou je například značka na mapě, či dokonce chybějící popis u mapové třídy samotné. U jednotlivých částí API také ve většině případů chybí, byť krátký, příklad, jak danou funkci použít, či jakých hodnot může parametr nabývat a dokumentace pouze nabízí informaci o jeho datovém typu. Dokumentace Google Maps API je obsáhlá, ale její chybějící části a převážně stručnost sráží použitelnost celé dokumentace a vývojář začátečník by mohl mít potíže, pokud se již v API více neorientuje.

2.3.1.5.2 V rámci dokumentace jsou dostupné i další materiály

Google Maps API nabízí v rámci dokumentace také příklady na využití nejčastějších funkcí. Ty jsou děleny do kategorií dle jejich zaměření na určitou část API. Tyto příklady jsou však ve většině případů pouze hotové úseky kódu, které postrádají jakékoliv vysvětlení, co se v daném kódu odehrává. API také nabízí galerii, která obsahuje odkazy na hotové aplikace nejrůznějších zaměření. Většina příkladů je funkčních, ale některé jsou odkazovány na již neexistující webové stránky, či stránky jiného zaměření.

2.3.1.5.3 Materiály jsou dostupné v češtině

Webová stránka dokumentace, příklady a odkazy v galerii řešení v rámci Google Maps API jsou dostupné pouze v anglickém jazyce.

2.3.1.5.4 Hodnocení

Dokumentace pokrývá všechny části API	2 body
V rámci dokumentace jsou dostupné i další materiály	6 bodů
Materiály jsou dostupné v češtině	0 bodů
Celkem	**8 bodů**

Tabulka 11 Bodové ohodnocení dokumentace

2.3.2 Mapy API verze 4.8

2.3.2.1 API umí zobrazit externí mapové podklady

Stejně jako společnost Google, tak také Seznam, který je vlastníkem mapového portálu Mapy.cz, nabízí v API vlastní mapová data. Lze zobrazit silniční mapu, satelitní zobrazení, turistickou mapu a také historické mapy, které nabízí právě mapový portál Mapy.cz.

Pro zobrazení rastrových dat jsou v API dostupné podobné metody, jako v případě Google Map. Pomoci třídy SMap.Layer.Image můžeme zobrazit obrázek, který se vykreslí do mapy jako její překryv, či můžeme využít třídy SMap.Layer.Tile, která umožní nastavit vrstvu sestavenou z obrázkových dlaždic, které poté podle pozice a přiblížení načítá.

Mapy API také dokáže zpracovat vektorový formát KML, pro který je připravena třída SMap.Layer.KML zajišťující jeho zobrazení. API taktéž zvládne pracovat s WMS službou pomoci třídy SMap.Layer.WMS.

2.3.2.1.1 Hodnocení

API umí zobrazit mapové podklady přes WMS vrstvu	4 body
API umí zobrazit vektorové vrstvy	4 body
API umí zobrazit rastrové vrstvy	4 body
Celkem	**12 bodů**

Tabulka 12 Bodové ohodnocení kritérii kategorie Mapové podklady

2.3.2.2 Ovládací prvky

2.3.2.2.1 API umí přidat ovládání pro posun mapou

Mapy API obsahuje možnosti pro přidání standardních způsobů pro pohyb mapou, a to pohyb za pomoci myši a za pomoci ovládacího prvku v rozhraní API, za pomoci třídy SMap.Control.Compass. V rámci této třídy umožňuje API také nastavení velikosti posunu po mapě, či její délku. Posun mapy za pomoci myši lze vypnout a využívat pouze kompasu.

2.3.2.2.2 API umí přidat ovládání pro přibližování

Ovládání pro přibližování Mapy API zvládá bez potíží ve dvou formách, přibližování kolečkem myši, které lze aktivovat za pomoci metody **addDefaultControls**, která jej přidá do mapy, a poté také samostatným ovládacím prvkem s tlačítky pro přiblížení a oddálení mapy, které nabízí třída SMap.Control.Zoom.

2.3.2.2.3 API umí přepínat viditelnost jednotlivých vrstev

Přepínání vrstev v rámci Mapy API je realizováno podobně jako u Google Maps API. V knihovně API existuje třída SMap.Control.Layer, která nabízí jednoduchý přepínač vrstev pro vestavěné vrstvy. Pokud je nutné pro aplikaci implementovat přepínač vrstev, je nutné to udělat mimo API.

2.3.2.2.4 API umí měřit vzdálenost a plochu a zobrazit je v rámci obrazovky

Měření vzdálenosti Mapy API umožňují za pomoci třídy SMap.Coords, která ve své instanci obsahuje zeměpisné souřadnice a umožňuje využít metodu pro měření vzdálenosti mezi dvěma jednotlivými body. Tuto hodnotu je poté možné dále využít

a nechat zobrazit v rámci aplikace. Metodu pro měření plochy Mapy API nenabízí a v případě, že je v aplikaci zapotřebí, je nutné ji naprogramovat externě.

2.3.2.2.5 API umí vybrat prvek z mapy a zobrazit o něm informace

V oblasti výběru prvků z mapy je Mapy API značně omezené. Nabízí pouze značky, které lze umístit na mapu a jim přiřadit krátký text, který poté budou zobrazovat. Pro KML a WMS vrstvy Mapy API také žádné metody na vybírání prvků a zobrazování informací nenabízí.

2.3.2.2.6 API umí strukturovat vrstvy pro zobrazení například v podobě stromu

Mapy API je v oblasti strukturování vrstev velice podobné Google Maps API a neobsahuje žádný pokročilý přepínač vrstev. Vrstvy tedy není možné strukturovat a následně zobrazovat, například ve formě stromu.

2.3.2.2.7 API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko

Vzhledem k omezené funkčnosti, co se týče výběru prvků z mapy, neobsahuje Mapy API ani možnost volby aktivní vrstvy.

2.3.2.2.8 API umí exportovat mapu a informace, které obsahuje

Mapy API v současné verzi neobsahuje nástroj, který by umožnil export mapy, jako obrázku, či jako statického obrazu mapy.

2.3.2.2.9 Hodnocení

API umí přidat ovládání pro posun mapou	8 bodů
API umí přidat ovládání pro přibližování	8 bodů
API umí přepínat viditelnost jednotlivých vrstev	2 body
API umí měřit vzdálenost a vzdálenost zobrazit v rámci obrazovky	2 body
API umí měřit plochu a plochu zobrazit v rámci obrazovky	0 bodů
API umí vybrat prvek z mapy a zobrazit o něm informace	0 bodů
API umí strukturovat vrstvy pro zobrazení například v podobě stromu	0 bodů
API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko	0 bodů
API umí exportovat mapu a informace, které obsahuje	0 bodů
Celkem	**20 bodů**

Tabulka 13 Bodové ohodnocení ovládacích prvků

2.3.2.3 Přizpůsobení vzhledu

2.3.2.3.1 API umí změnit své barevné schéma

V rámci Mapy API není změna barevného schématu základních ovládacích prvků možná. Na rozdíl od Google Maps API neumožňuje Mapy API ani změnu barevného schématu defaultních mapových typů.

2.3.2.3.2 API umí editovat své prvky za pomoci CSS

Podobně jako Google Maps API, tak také Mapy API umožňuje stylovat element div, ve kterém se nachází mapa standardním způsobem. Další stylování ovládacích prvků pomoci CSS však API nenabízí a je tedy nutná implementace vlastních ovládacích prvků, které lze poté v rámci CSS stylovat.

2.3.2.3.3 Hodnocení

API umí změnit své barevné schéma	0 bodů
API umí editovat své prvky za pomoci CSS	0 bodů
Celkem	**0 bodů**

Tabulka 14 Bodové ohodnocení úpravy vzhledu

2.3.2.4 Licence

Mapy API ve verzi v4.8 je dostupné zdarma, a to včetně komerčního využití, za předpokladů, že aplikace splní podmínky užití, mezi které patří například přítomnost log společnosti Seznam a také Mapy.cz, či nebude po koncových uživatelích požadovat placený přístup k webové stránce, na které se nachází mapa. Důležitou podmínkou je také část o zákazu finančního, i jiného obohacování na API, či aplikacích na jeho základě vytvořených. Za pomoci Mapy API není také možné stahovat a dále používat mapové podklady, které API obsahuje. Plné a aktuální znění licenčních podmínek lze nalézt na webové stránce Mapy API[3].

2.3.2.4.1 Hodnocení

API je dostupné zdarma ve formě Open Source	0 bodů
API je dostupné zdarma s omezením	4 body
API je dostupné komerčně	0 bodů
Celkem	**4 body**

Tabulka 15 Bodové ohodnocení licenčních podmínek

2.3.2.5 Dokumentace

2.3.2.5.1 Dokumentace pokrývá všechny části API

Dokumentace Mapy API je řešena za pomoci webové stránky, která ji třídí dle jednotlivých tříd, kdy každá třída obsahuje vlastní podstránku, kde se nachází jednotlivé parametry a metody. Dokumentace nabízí jednoduchý popis daných metod a parametrů, který je však ve většině případů stručný a v některých případech chybí popis parametrů, a to i u zásadních tříd, jakou je například třída mapy. Dokumentace

[3] Mapy API verze 4.8 – Hanzelka a Zikmund. *Mapy API verze 4.8 – Hanzelka a Zikmund* [online]. 2014 [cit. 2014-04-01]. Dostupné z: http://api4.mapy.cz/#pact

Mapy API taktéž ve většině případů nenabízí žádné informace, jak metody použít, či jakých možných hodnot může daný parametr nabývat. Tato skutečnost, společně se stručností dalších částí dokumentace, může být pro neznalého API zmatečné a dotyčný je odkázán pouze na podpůrné materiály, které společnost Seznam nabízí.

2.3.2.5.2 V rámci dokumentace jsou dostupné i další materiály

Webová stránka Mapy API nabízí v první řadě část nazvanou "Návod k použití", ten je však pouhou částí kódů, který zajistí zobrazení nejjednodušší mapy a ovládacích prvků. Dále portál nabízí ukázková řešení některých funkcí, které nabízí nejen ukázku kódu, ale také výsledné řešení v provozu. Bohužel podobně, jako u dokumentace Google Maps API, tak také ukázková řešení Mapy API nenabízejí komentáře, které by vysvětlovaly, proč je daná část kódu použita tak, jak je a v kombinaci s minimalistickou dokumentací API je tedy vývojář v mnoha případech, jakými je například implementace KML vrstvy, odkázán na zkoušení metodou pokus-omyl, či nucen kontaktovat tvůrce API. V tomto případě nabízí webový portál jednoduché diskusní fórum, na kterém může probíhat komunikace s tvůrci API, ale také s ostatními vývojáři.

2.3.2.5.3 Materiály jsou dostupné v češtině

Jednoznačnou výhodou Mapy API je jeho český původ. Tudíž veškerá dokumentace, ale také příklady a další části webového portálu, je v českém jazyce. Taktéž diskusní fórum je provozované v českém jazyce.

2.3.2.5.4 Hodnocení

Dokumentace pokrývá všechny části API	2 body
V rámci dokumentace jsou dostupné i další materiály	6 bodů
Materiály jsou dostupné v češtině	2 body
Celkem	**10 bodů**

Tabulka 16 Bodové ohodnocení dokumentace

2.3.3 Openlayers 2.13.1

2.3.3.1 API umí zobrazit externí mapové podklad

Openlayers, na rozdíl od předchozích dvou API nemá vlastní mapové podklady a spoléhá na data od třetích stran, které umí implementovat. Mezi podporovanými poskytovateli nalezneme Open Street Maps, Google Mapy, nebo například Bing mapy. Tyto podklady mají v knihovně připraveny třídy pro jejich implementaci v rámci Openlayers.

Zobrazení rastrových dat v rámci OpenLayers je standardně možné v rámci jediného obrázkového souboru pomoci třídy OpenLayers.Layer.Image. Sady obrazových dlaždic lze v Openlayers využít za pomoci několika tříd, jakými jsou například OpenLayers.Layer.TMS, či OpenLayers.Layer.Grid, u kterých je nutné doplnit metodu, která určí, podle jakých pravidel se bude dlaždice z daného umístění zpracovávat a získá soubor reprezentující danou dlaždici. Po doplnění této funkce je možné data načítat standardním způsobem.
Pro vektorová data nabízí OpenLayers třídu OpenLayers.Layer.Vector, která umožní zpracování, a zároveň zajistí jejich vykreslení na mapě. Mezi podporovanými formáty nechybí KML, či GML, se kterými umí API nadále pracovat.

Pro podporu WMS služby má API připravenou také samostatnou třídu OpenLayers.Layer.WMS, která po přidání do mapy zajistí propojení.

2.3.3.1.1 Hodnocení

API umí zobrazit mapové podklady přes WMS vrstvu	4 body
API umí zobrazit vektorové vrstvy	4 body
API umí zobrazit rastrové vrstvy	4 body
Celkem	**12 bodů**

Tabulka 17 Bodové ohodnocení kategorie Mapové podklady

2.3.3.2 Ovládací prvky

2.3.3.2.1 API umí přidat ovládání pro posun mapou

V knihovně Openlayers je samozřejmě také možné nastavit posun mapy v obou režimech, posun myší i ovládací kříž, který posun vykoná po kliknutí na příslušný

směr. Ovládání za pomoci myši je v Openlayers zajišťováno za pomoci třídy **OpenLayers.Control.Navigation**, která sdružuje všechny základní události, a mezi nimi i zmíněné tažení mapou. Pro pohybový kříž obsahuje Openlayers třídu **OpenLayers.Control.PanPanel**. Ta vytvoří prvek, který se sestává ze čtyř tlačítek reprezentující jednotlivé světové strany. Další možností, jak do aplikace přidat pohybový kříž je za pomoci třídy **OpenLayers.Control.PanZoomBar**, která vytvoří kříž a zároveň posuvník pro přiblížení mapy.

2.3.3.2.2 API umí přidat ovládání pro přibližování

Přiblížení lze v Openlayers implementovat za pomoci třídy **OpenLayers.Control.Navigation**, která umožní přibližovat mapu za pomoci kolečka myši. Další je možnost implementace tlačítek pro přiblížení a oddálení, či slideru pro přesné nastavení přiblížení. Openlayers také umožňuje implementaci přiblížení za pomoci výběru požadované oblasti.

2.3.3.2.3 API umí přepínat viditelnost jednotlivých vrstev

Pro přepínání vrstev nabízí Openlayers třídu **OpenLayers.Control.LayerSwitcher**, která načte vrstvy, které daná mapa obsahuje a zobrazí je v rámci přepínače. Ten vrstvy rozdělí podle toho, zda jsou základní (může být aktivní vždy jedna a přepíná se radiobuttonem), či překryvné (lze jich zviditelnit více za pomoci checkboxu).

2.3.3.2.4 API umí měřit vzdálenost a plochu a zobrazit je v rámci obrazovky

Pro měření vzdálenosti a plochy nabízí Openlayers třídu **OpenLayers.Control.Measure**, která dle použitého handleru určí, zda bude ovládací prvek měřit vzdálenost, nebo plochu. Nástroj pro měření umožňuje nastavit, zda se bude jednat o okamžité měření, kdy se naměřená hodnota přepočítává a zobrazuje s pohybem myši, či zda se má hodnota vypočítat a zobrazit až po ukončení dvojklikem.

2.3.3.2.5 API umí vybrat prvek z mapy a zobrazit o něm informace

Knihovna Openlayers obsahuje nástroj pro výběr prvků z vektorových vrstev, ale také získání informací pomoci WMS služby. Výběrový nástroj pro vektorovou vrstvu je obsažen ve třídě **OpenLayers.Control.SelectFeature**, který zpracuje kliknutí na

daný prvek a následně uloží informace, které lze zobrazit ve formě vyskakovacího okna, tzv. popupu, či jinak zpracovat v rámci aplikace.

Pro zpracování informací o WMS prvcích je v Openlayers umístěn nástroj ve třídě **OpenLayers.Control.WMSGetFeatureInfo**, který zpracuje kliknutí na WMS vrstvu a vrátí informace, které lze následně zpracovat opět ve formě popupu, či dále zpracovat v rámci aplikace.

2.3.3.2.6 API umí strukturovat vrstvy pro zobrazení například v podobě stromu

Přepínač vrstev v Openlayers nabízí základní dělení na základní a překryvné vrstvy. Tyto vrstvy se poté zobrazí pod názvy těchto dvou kategorií. Další větvení vrstev v rámci přepínače API neumožňuje.

2.3.3.2.7 API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko

Openlayers v současné době v základu neumožňuje zpracování aktivní vrstvy, ani zobrazení tlačítka, které by tuto funkčnost ovládalo. Pro potřeby aktivní vrstvy je nutné funkčnost doplnit, či využít některé nadstavby, která by ji umožňovala.

2.3.3.2.8 API umí exportovat mapu a informace, které obsahuje

Export mapy není v Openlayers možný a je nutné jej implementovat externě. V budoucí verzi API je však plánována podpora této funkčnosti, která bude využívat HTML5 elementu canvas.

2.3.3.2.9 Hodnocení

API umí přidat ovládání pro posun mapou	8 bodů
API umí přidat ovládání pro přibližování	8 bodů
API umí přepínat viditelnost jednotlivých vrstev	8 bodů
API umí měřit vzdálenost a vzdálenost zobrazit v rámci obrazovky	2 body
API umí měřit plochu a plochu zobrazit v rámci obrazovky	2 body
API umí vybrat prvek z mapy a zobrazit o něm informace	3 body
API umí strukturovat vrstvy pro zobrazení například v podobě stromu	0 bodů
API umí zpracovat aktivní vrstvu pro výběr a zobrazit potřebné tlačítko	0 bodů
API umí exportovat mapu a informace, které obsahuje	0 bodů
Celkem	**31 bodů**

Tabulka 18 Bodové ohodnocení ovládacích prvků

2.3.3.3 Přizpůsobení vzhledu

2.3.3.3.1 API umí změnit své barevné schéma

Změna barevného schématu je v Openlayers možná, jelikož ikony ovládacích prvků jsou uloženy ve formátu png v jedné složce API a je možné je upravovat dle potřeby a vzhledu výsledné aplikace.

2.3.3.3.2 API umí editovat své prvky za pomoci CSS

K možnosti přetvářet ovládací prvky za pomoci obrázků přidává Openlayers také editaci prvků pomoci CSS. API obsahuje pro ovládací prvky předpřipravené třídy, které stylování umožní. Většinu prvků lze stylovat za pomoci tříd s názvem .olControl, za který se připojí název daného ovládacího prvku, například **.olControlLayerSwitcher**. U některých prvků, převážně tlačítek, je nutné rozlišit, zda se jedná o styl aktivní, či neaktivní. Dále Openlayers nabízí přesun některých ovládacích prvků, například přepínače vrstev, do div elementu, což umožní lepší zapojení prvku do vzhledu aplikace.

2.3.3.3.3 Hodnocení

API umí změnit své barevné schéma	4 body
API umí editovat své prvky za pomoci CSS	4 bodů
Celkem	**8 bodů**

Tabulka 19 Bodové ohodnocení úpravy vzhledu

2.3.3.4 Licence

Openlayers jsou dostupné pod Opensource licencí BSD, konkrétně dvoučlánkovou BSD licencí, která je také známá jako FreeBSD, která umožňuje použití zdarma. Jedinou podmínkou této volné licence je obsažení jejího textu a připojení souboru s autory Openlayers v rámci výsledné aplikace. Text licence lze nalézt například na webových stránkách Openlayers[4].

2.3.3.4.1 Hodnocení

API je dostupné zdarma ve formě Open Source	8 bodů
API je dostupné zdarma s omezením	0 bodů
API je dostupné komerčně	0 bodů
Celkem	**8 bodů**

Tabulka 20 Bodové ohodnocení licenčních podmínek

2.3.3.5 Dokumentace

2.3.3.5.1 Dokumentace pokrývá všechny části API

Dokumentace Openlayers je dostupná ve formátu webového portálu, který nabízí provázané odkazy mezi jednotlivými třídami a metodami. Ačkoliv je dokumentace rozsáhlá, tak i zde se vyskytují části, kde chybí popis jednotlivých metod, či parametrů a v dokumentaci je tedy napsán pouze název daného prvku. Tato skutečnost se ve větší míře týká okrajových částí API a v nejpoužívanějších třídách je tento nedostatek minimální. V rámci dokumentace se také ve vybraných nejpoužívanějších třídách dočká čtenář ukázky, jak danou funkci použít.

[4] Openlayers/openlayers · GitHub. *GitHub · Build software better, together.* [online]. 2014 [cit. 2014-04-01]. Dostupné z: https://raw.github.com/openlayers/openlayers/master/license.txt

Dokumentace má tedy i v případě Openlayers nedostatky, ale čte a používá se lépe, než v případě Google Maps API a Mapy API.

2.3.3.5.2 V rámci dokumentace jsou dostupné i další materiály

Po dokumentaci nabízí Openlayers také knihovnu příkladů, které nabízejí pohled na to, jak mohou některé vybrané funkce fungovat. Příklady jsou však, stejně jako u Google Maps API a Mapy API, pouze části kódu, kde není vysvětleno, proč jsou dané parametry a metody použity tak, jak jsou. Dalším dodatečným materiálem je několikastránková příručka, která nabízí úvod do problematiky Openlayers a umožňuje pochopení základů syntaxe API a zorientovat se v nejpoužívanějších funkcích.

2.3.3.5.3 Materiály jsou dostupné v češtině

Webový portál dokumentace Openlayers, veškeré příklady, i příručka jsou dostupné pouze v anglickém jazyce.

2.3.3.5.4 Hodnocení

Dokumentace pokrývá všechny části API	4 body
V rámci dokumentace jsou dostupné i další materiály	6 bodů
Materiály jsou dostupné v češtině	0 bodů
Celkem	**10 bodů**

Tabulka 21 Bodové ohodnocení dokumentace

2.4 Vyhodnocení porovnání

Porovnání bylo nakonec vyhodnoceno sečtením bodů a započtením vah. Výsledky jsou zahrnuty v následující tabulce, která znázorňuje jednotlivé součty bodů za kategorii, výsledný součet a na konec výsledný součet se započtenými kritérii.

	Google Maps API	Mapy API	Openlayers
Mapové podklady	8 bodů	12 bodů	12 bodů
Ovládací prvky	21 bodů	20 bodů	31 bodů
Přizpůsobení vzhledu	0 bodů	0 bodů	8 bodů
Licence	4 body	4 body	8 bodů
Dokumentace	8 bodů	10 bodů	10 bodů
Celkem	**41 bodů**	**46 bodů**	**69 bodů**
Celkem po započtení vah	**144 bodů**	**162 bodů**	**234 bodů**

Tabulka 22 Celkové bodové hodnocení

Porovnání API ukázalo, že Google Maps API společně s Mapy API jsou spíše zaměřeny na využití služeb svých mateřských společností a nejsou primárně určeny pro použití jiných dat jednodušším způsobem, který by dosáhl podobného výsledku, jako v případě API, které dosáhlo nejvíce bodů, Openlayers. Google Maps API a Mapy API skončily kvůli tomuto zaměření v porovnání velice podobně. Praktický rozdíl mezi nimi nastal pouze v české lokalizaci dokumentace a podporou měření a výběru z mapy.

Oproti Openlayers zaostávají také v nabídce funkcí, či schopnosti zobrazit vlastní mapová data. V některých případech daná funkce chybí úplně, nebo se vyskytuje pouze jako nepojmenovaný záznam v dokumentaci a není možné z ní vyčíst, jak jej použít. V případě Openlayers také části dokumentace chybí, ačkoliv se tento nedostatek objeví až u hlubších funkcí, které jsou menšího významu.

Pokud Google Maps API a Mapy API danou funkci nabízí a je zdokumentovaná, ve většině případů se jedná o nástroje vytvořené na míru potřeb tvůrců a je obtížné je měnit, API také nabízí velké omezení, co se úpravy vzhledu týče.

Pro vývoj ukázkové aplikace byla tedy na základě porovnání vybrána knihovna Openlayers, která nabízí největší množství funkcí, jež odpovídají požadavkům, a dobře zpracovanou oficiální dokumentaci. Výhodou Openlayers je také možnost použití nadstaveb a dalších knihoven pro vylepšení obou částí API, vzhledu i funkčnosti. To vše za nulové, či minimální finanční nároky.

3 Publikace mapových dat

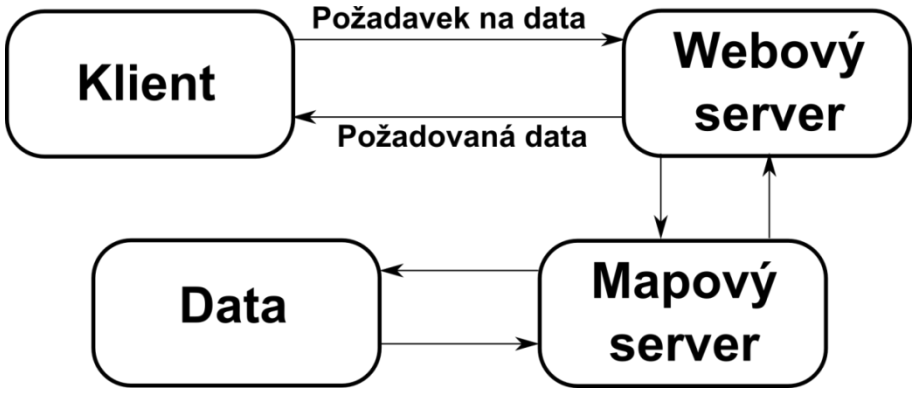
Publikaci mapových dat pro potřeby jejich webové prezentace lze vyřešit více způsoby. Nejjednodušším způsobem je tzv. statická mapa. V tomto způsobu mapu umístíme na webové stránky, jako statický obrázek. Veškeré znalosti, které jsou k tomuto způsobu publikace potřeba, jsou základní znalosti tvorby webových stránek a zpracování obrazových materiálů. Pokud je ale požadavkem interaktivnost, tento způsob naprosto selhává a je nutné použít jiný.

Interaktivní aplikace umožní uživatelů prohlížet mapy, přibližovat je, vybírat vrstvy, či získávat informace o prvcích v mapě. K použití této aplikace je však nutné připravit data a v rámci této přípravy je možné použít také několik způsobů. Data můžeme připravit ve formě souborů, které aplikace dokáže zpracovat, například ve formátu KML, a tyto soubory poté uložíme do webového prostoru společně se soubory stránky. Druhou možností je použití mapového serveru, který má mapová data uložena na svém disku.

3.1 Mapový server

3.1.1 Popis technologie

Mapový server je soubor nástrojů a dat obsahující webovou aplikaci, webový server, mapové soubory a software mapového serveru, který umožní publikaci mapových dat pro zpracování na webových stránkách. Ve většině případů fungují na principu klient-server, kdy server zpracuje požadavky klienta.



Obr. 1 Nákres struktury mapového serveru

Uživatel zadá požadavek za pomoci klienta, ten může být libovolným webovým prohlížečem, který je kompatibilní s klientskou aplikací, například podporuje zobrazení aplikací v jazyce JavaScript. Požadavek, jenž může obsahovat posun mapy, či informace o prvku, je zpracován za pomoci webového serveru, například Apache, který je zdarma dostupný pod tzv. Apache licencí, či Internet Information Services (IIS) od společnosti Microsoft, dostupného jako součást operačních systémů Windows a Windows Server. Webový server slouží jako spojnice mezi webovými stránkami a samotnou aplikací mapového serveru, která spravuje data a vytváří požadované mapy, ta poté, dle původních požadavků, navrátí data, které webový server předá klientské aplikaci, a ta je následně zobrazí. Komunikaci mezi jednotlivými částmi serveru bylo nutné unifikovat, a právě pro tuto potřebu vznikl protokol WMS, čili Web Map Service.

3.1.2 Web Map Service

Protokol WMS byl poprvé uveden v roce 1999 organizací OGC (Open Geospatial Consortium). V současnosti je nejpoužívanější verze 1.1.1 z roku 2002, poslední verzí je verze 1.3.0 z roku 2004.Od roku 2005 se jedná o standard organizace ISO pod označením ISO 19128:2005 . WMS slouží právě k propojení aplikace, ať již webové nebo desktopové, s daty uloženými na jiném serveru.

WMS protokol využívá několika příkazů, například GetCapabilities, který získá souhrnné informace o daném WMS serveru ve formátu xml. V tomto souboru lze nalézt například seznam vrstev, jejich parametry či informace o formátu mapy. Dalším používaným příkazem je GetMap, který navrací požadovaná obrazová data reprezentující mapu, a to buď celkově, nebo omezený čtverec, ze kterých se mapa skládá.

Při použití webové aplikace však potřeba znalosti těchto příkazů odpadá. Většina technologií, které jsou pro komunikaci s WMS serverem způsobilé, obsahuje například WMS vrstvy, které příkazy sestaví samy a zašlou jej na server, aniž by uživatel věděl, že komunikace probíhá v rámci WMS. Příklad takového vygenerovaného dotazu, konkrétně dotaz GetMap, který vrací určitý čtverec mapy, z aplikace vytvořené za použití knihovny Openlayers je vidět níže.

```
37.157.197.46:8080/cgi-
bin/mapserv.exe?map=C:/Data/mapa2.map&SERVICE=WMS&VERSION
=1.0.0&LAYERS=001&REQUEST=GetMap&FORMAT=image%2Fpng&SRS=E
PSG%3A900913&BBOX=1586221.2107532,6243376.4694641,1587444
.2032056,6244599.4619165&WIDTH=256&HEIGHT=256
```

3.1.3 Aplikace mapového serveru

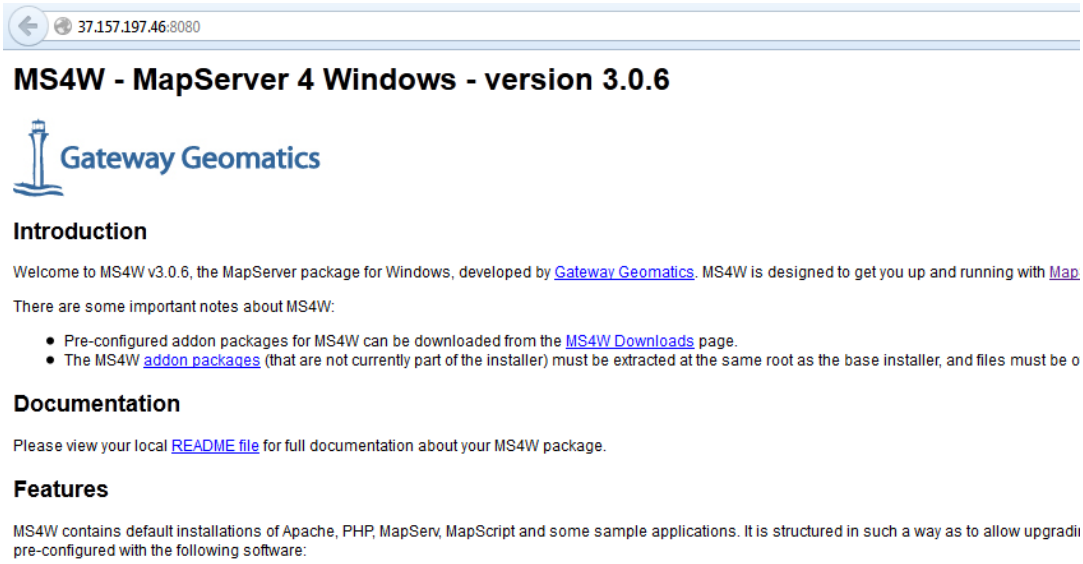
Aby však WMS služba na serveru fungovala, je nutné na něj nainstalovat speciální aplikaci. Tyto aplikace, které slouží ke komunikaci mezi uživatelem a mapovými daty, WMS službu nainstalují, spojí s webovým serverem a spustí. Aplikací pro WMS server existuje velké množství pro nejrůznější platformy. Mezi komerčními můžeme nalézt například ArcGIS for Server, či Autodesk MapGuide Enterprise. V rámci zachování nízkého rozpočtu pro potřeby malé obce však byla použita OpenSource řešení. Prvním z otestovaných byl MapGuide OpenSource, který však postrádal kvalitně zpracovanou dokumentaci a bez této dokumentace nebylo možné jej nastavit tak, aby WMS služba fungovala. Z tohoto důvodu byla vybrána aplikace UMN MapServer, která byla vyvinuta Minnesotskou univerzitou v Minneapolis.

3.1.3.1 UMN MapServer

MapServer je opensource aplikace, spravována v rámci organizace OSGeo a je spustitelná na většině základních platforem, jakými jsou Linux, Windows, či Mac OS X.

3.1.3.1.1 Instalace

MapServer potřebuje ke svému fungování několik dalších aplikací, mezi které patří webový server Apache a PHP pro webovou správu. Instalace v rámci práce proběhla na hostovaný server s operačním systémem Windows Server 2008 R2 Datacenter edition. Pro instalaci bylo použito sestavení MapServer for Windows (MS4W), které uživatele provede instalací, výběrem součástí a nastavením portů krok za krokem v podobě standardního Windows instalátoru. Po instalaci je nutné povolit port, na který je MapServer nastaven, jinak nebude server při přístupu zvenčí fungovat. Fungování serveru po instalaci lze ověřit na adrese localhost:port daného serveru, přičemž za port dosadíme číslo portu zvoleného v rámci instalace. Zvenčí se poté adresa localhost nahradí za IP adresu serveru.



Obr. 2 Ukázka úspěšně spuštěného MapServeru za pomoci MS4W

3.1.3.2 Vložení dat

Pro testování mapového serveru byla použita mapová data obce Kájov ve formátu shapefile. Pro použití s MapServerem je nutné data popsat v rámci souboru mapfile, který serveru určuje, jak má daná data zpracovat. Pro vytvoření tohoto souboru byl použit opensource nástroj Quantum GIS (QGIS) ve verzi 1.6.0. Ten je sice dostupný již ve verzi 2, ale ta má oproti předchozí verzi nevýhodu. Plugin, který zajišťuje export mapového souboru mapfile pro MapServer není pro tuto verzi dostupný a není z ní tedy možné mapfile exportovat. Mapfile obsahuje veškeré informace o vrstvách, jejich definici, oblast i jejich souřadnicový systém. Připravený mapfile je poté součástí WMS dotazu a určuje, jaká data má dotaz zpracovávat. Mapfile určíme v rámci dotazu po proměnné map, jak je vidět v následujícím příkladu.

```
map=C:/Data/mapa2.map
```

Po vytvoření mapfile souboru jsou data připravena k publikaci za pomoci WMS služby.

3.2 Příprava porovnání rychlosti mapového serveru a lokálních souborů

Příprava mapového serveru je oproti nahrání souborů KML náročnější, ale předpokladem je, že by mapový server mohl tuto námahu vynahradit v podobě rychlejšího zpracování. Otázkou je, zda bude tento předpoklad platit i pro data malé obce, tedy data, která mají v průměru do 3MB.

3.2.1 Použité vrstvy

Pro toto měření byly publikovány dvě WMS vrstvy, které odpovídají dvěma vzorkům KML souborů použitích v rámci aplikace obce Kájov. První z nich jsou funkční plochy, které reprezentují WMS vrstvu složenou z jednoho zdrojového souboru. KML soubor ekvivalentní této vrstvě má velikost 576 kB. Tento soubor reprezentuje průměrnou velikost souborů obsažených na serveru, není tedy největší, ani nejmenší.

Druhou testovací vrstvou je Telefonní síť, která je složena ze tří zdrojových souborů. KML soubor ekvivalentní této vrstvě má velikost 2111 kB. Je tak tou největší vrstvou, kterou data obce Kájov obsahují.

3.2.2 Způsob porovnání

Pro porovnání byla nejdříve připravena speciální verze aplikace, která neobsahuje žádné podkladové vrstvy, které by ovlivňovaly délku načítání jednotlivých vrstev. Poté následovala publikace výše zmíněných vrstev.

Po přípravě technických potřeb byly stanoveny tři operace, které budou měřeny a použity v porovnání. Tyto operace jsou těmi základními, které mohou při načítání mapy nastat, první z nich je načtení čisté vrstvy. Tato operace ukazuje, jak dlouho trvá vrstvě její načtení, pokud je pouze zapnuta, aniž by byla jakkoliv uložena ve vyrovnávací paměti prohlížeče. Další dvě operace spolu souvisí, jelikož se jedná o přiblížení a oddálení mapy, které testují, jak dlouho bude načtení dané vrstvy jednotlivým technologiím trvat, pokud již je nějaká úroveň přiblížení načtena. V rámci porovnání nebyla mapa posouvána, tudíž se vždy jednalo o načtení stejné lokace.

Pro měření jednotlivých operací byly použity dva prohlížeče pro systém Windows. Prvním s nich byla Mozilla Firefox, ve které byl použit doplněk Firebug, který nabízí možnost sledovat časy načtení jednotlivých nových požadavků, které vyvolá načtení webové stránky, či její části. Druhý prohlížečem použitým v porovnání je Google Chrome, který taktéž ve své vývojářské konzole nabízí možnost sledovat načítání jednotlivých nových požadavků na stažení dat z Internetu.

Pro minimalizaci vlivu mezipaměti prohlížeče a dalších urychlujících prvků bylo měření prováděno v anonymních režimech jednotlivých prohlížečů, které do mezipaměti neukládají žádná data. Načítání bylo provedeno za pomoci kombinace kláves Ctrl a F5, kdy prohlížeče svou mezipaměť vyčistí a pro jistotu byla po každém načítání mezipaměť vymazána společně s historií a cookies. Tyto úkony zajistily, že se v rámci měření nevyskytne žádná část webové aplikace, která by zůstala načtená ve vyrovnávací paměti či ovlivňovala načtení měřených prvků.

Samotné měření probíhalo načtením každé operace. Toto načtení proběhlo vždy sedmkrát, společně s výše zmíněným vymazáním mezipaměti, a z hodnot, které byly naměřeny, byla za pomoci mediánu vybrána střední hodnota. Tato hodnota byla poté zanesena do porovnání.

3.2.3 Technické parametry

Mapová data a aplikace mapového serveru byly umístěny na hostovaný server. Jedná se o virtuální server, který se nachází u poskytovatele služby a uživatel se na něj vzdáleně připojuje, například za použití Vzdálené plochy. Hostovaný server byl použit na základě požadavků na nízké finanční náklady celého řešení. Server je vybaven jednojádrovým procesorem o frekvenci 1,8 GHz a 1GB paměti RAM. Jeho operačním systémem je již zmíněný Windows Server 2008 R2.

Pro samotné porovnání byly vybrány čtyři sestavy s rozdílným výpočetním výkonem a rychlostí internetového připojení. Na každé z těchto sestav proběhlo úplně měření za použití výše zmíněných postupů a prohlížečů. První ze sestav byl laptop se čtyřjádrovým procesorem Intel Core i7 s frekvencí 2,2 GHz a 8GB paměti RAM. Tento laptop byl připojen internetovým připojením o rychlosti downloadu přibližně 60 Mbit/s. Upload byl naměřen přibližně 5 Mbit/s. Operačním systémem tohoto laptopu je Windows 7 SP1. V rámci porovnání bude tato sestava nazývána Sestava 1.

Druhou sestavou byl přímo samotný hostovaný server, kde bylo měření prováděno za pomoci připojení přes vzdálenou plochu. Server má již zmíněnou konfiguraci s operačním systémem Windows Server 2008 R2 a rychlost jeho připojení k internetu je přibližně 98 Mbit/s u downloadu a 75 Mbit/s u uploadu. V porovnání se server vyskytuje pod označením Sestava 2.

Další sestavou, která se v porovnání vyskytuje pod názvem Sestava 3, je stolní počítač vybavený jednojádrovým procesorem Intel Pentium s frekvencí 2,7 GHz a 4GB RAM. Rychlost připojení této sestavy je přibližně 2 Mbit/s downloadu a 274 kbit/s uploadu. Operačním systémem jsou v tomto případě Windows XP.

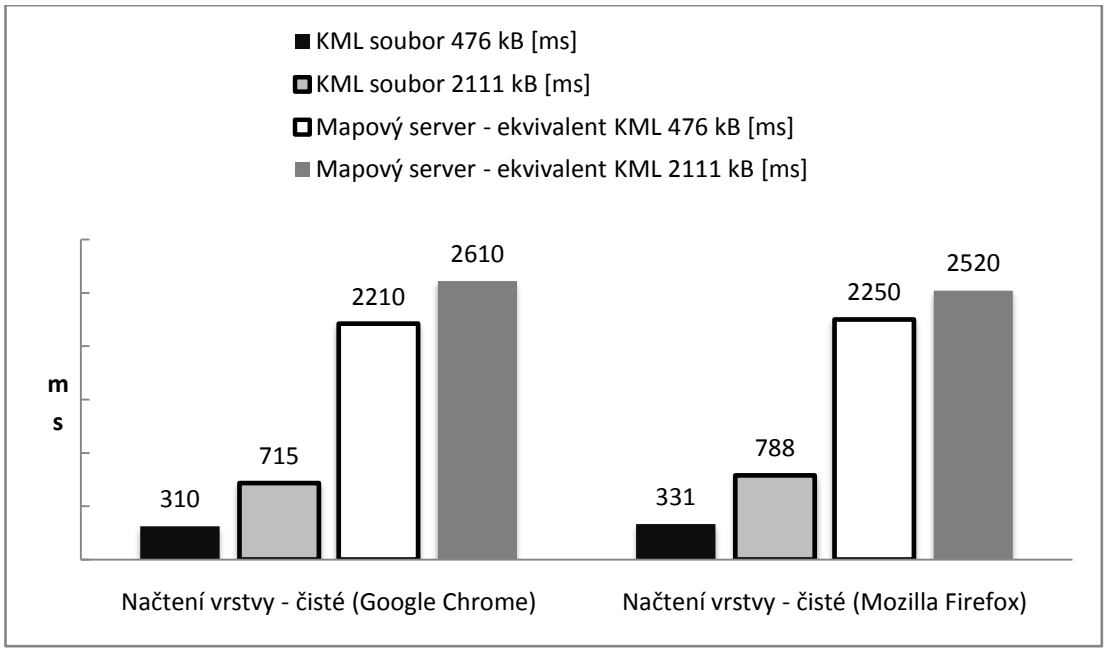
Poslední sestavou je Sestava 4, která je vybavená procesorem Intel Core 2 Duo s frekvencí 3,16 GHz a 4GB RAM. Internetové připojení této sestavy má rychlost downloadu přibližně 80 Mbit/s a uploadu 28 Mbit/s. Operační systém je taktéž Windows XP.

3.3 Vyhodnocení porovnání

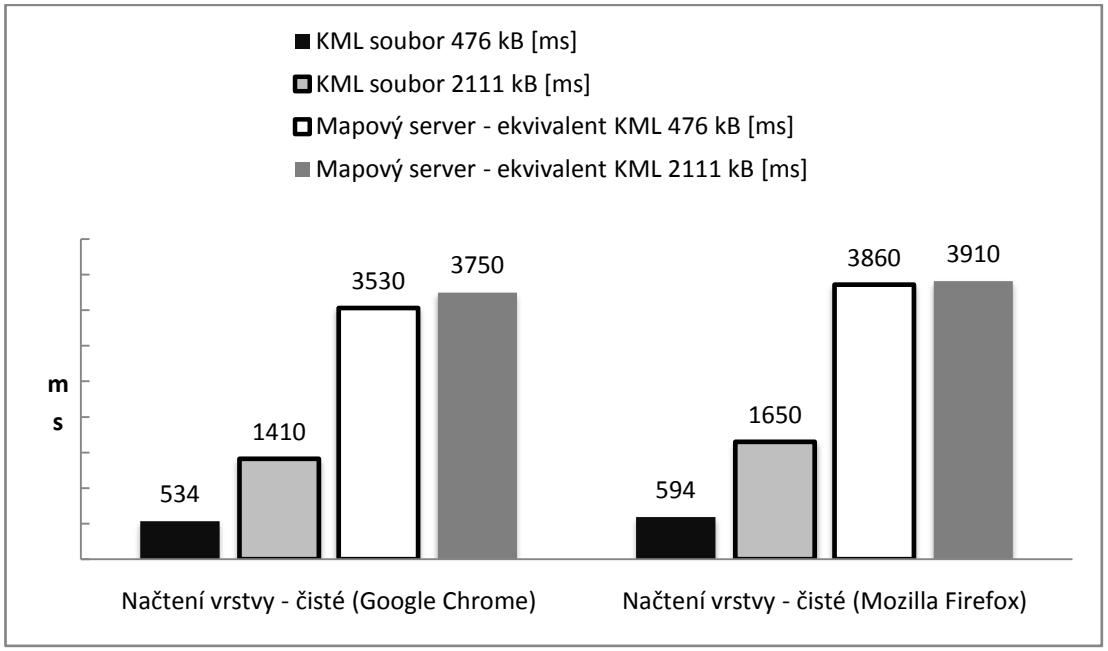
V rámci vyhodnocení byly sestaveny grafy, které znázorňují jednotlivé naměřené hodnoty.

3.3.1 Čisté načtení vrstvy

V rámci načtení čisté vrstvy byly většinou KML soubory v načítání rychlejší, než samotný mapový server. Načítání KML souborů nebylo příliš ovlivněno výkonem sestavy. Na nejslabší sestavě virtuálního serveru došlo k navýšení času načtení u menšího z KML souborů přibližně o 50% a u většího na přibližně dvojnásobnou hodnotu. Tento nárůst však znamenal hodnoty vyšší o časy do jedné sekundy. Rozdíly v délce načtení KML souboru nebyly zaznamenány také mezi oběma prohlížeči, kdy se lišily o hodnoty do 200 ms. Co ovšem velice ovlivnilo načítání KML souborů bylo pomalé internetové připojení u sestavy číslo 3. Doba načítání KML souborů pomalým internetovým připojením bylo oproti slabší sestavě číslo 2 delší přibližně třikrát.



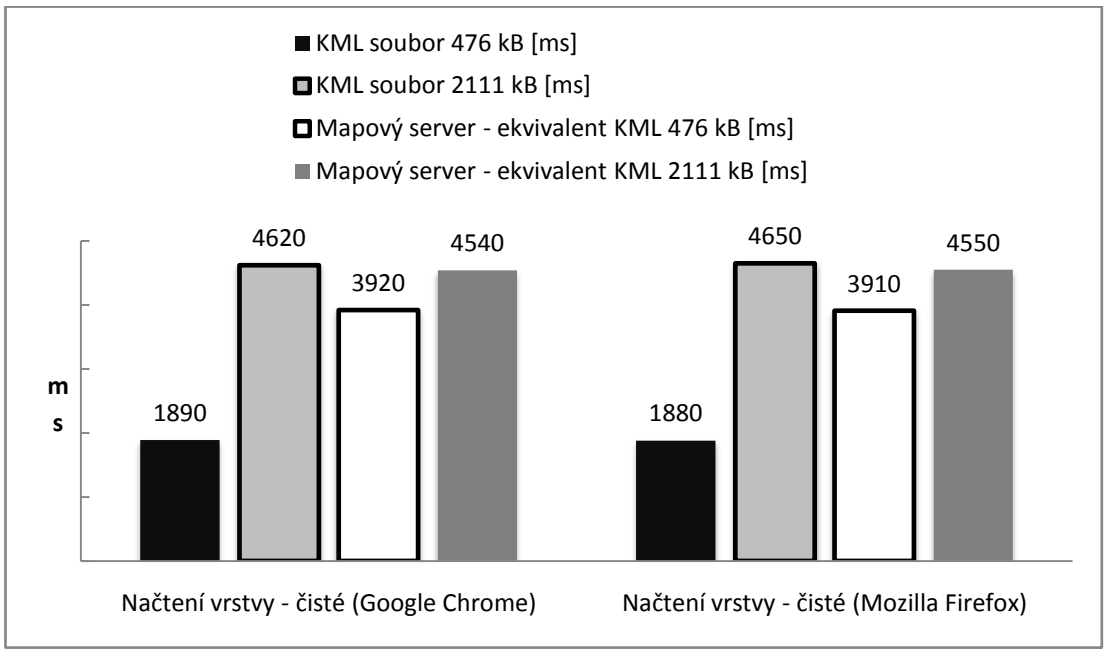
Obr. 3 Načtení čisté vrstvy - Sestava 1



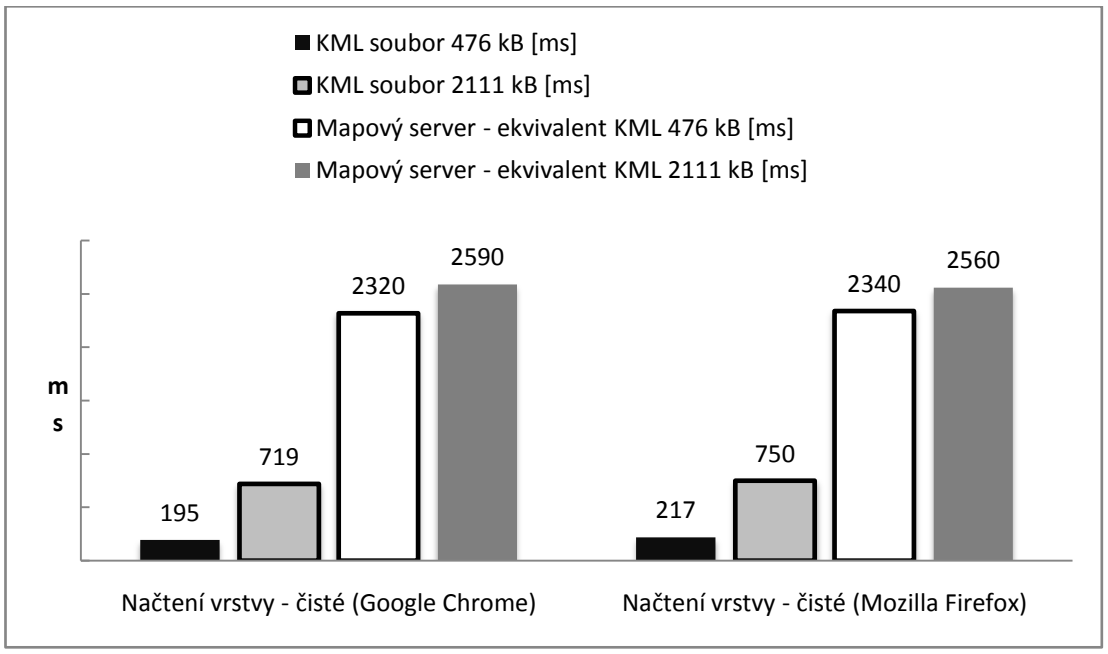
Obr. 4 Načtení čisté vrstvy - Sestava 2

Při načítání dat z mapového serveru byla i hodnota naměřená na sestavách s rychlým internetovým připojením razantně pomalejší. Rozdíl byl v některých případech propastný, a to až o tři celé sekundy. Načítání z mapového serveru ovlivňuje také výkon sestavy, kdy na pomalé sestavě s rychlým připojením (Sestava 2) bylo načítání dat ze serveru přibližně 1,5 - 2 sekundy pomalejší, než na rychlých sestavách se srovnatelným připojením. Pomalé připojení u Sestavy 3 navýšilo rozdíl již jen

o málo, přibližně 400-800 ms oproti sestavě číslo 2. Pro připomenutí, rozdíl načítání KML souborů u těchto sestav byl přibližně 3,2 sekundy.



Obr. 5 Načtení čisté vrstvy - Sestava 3

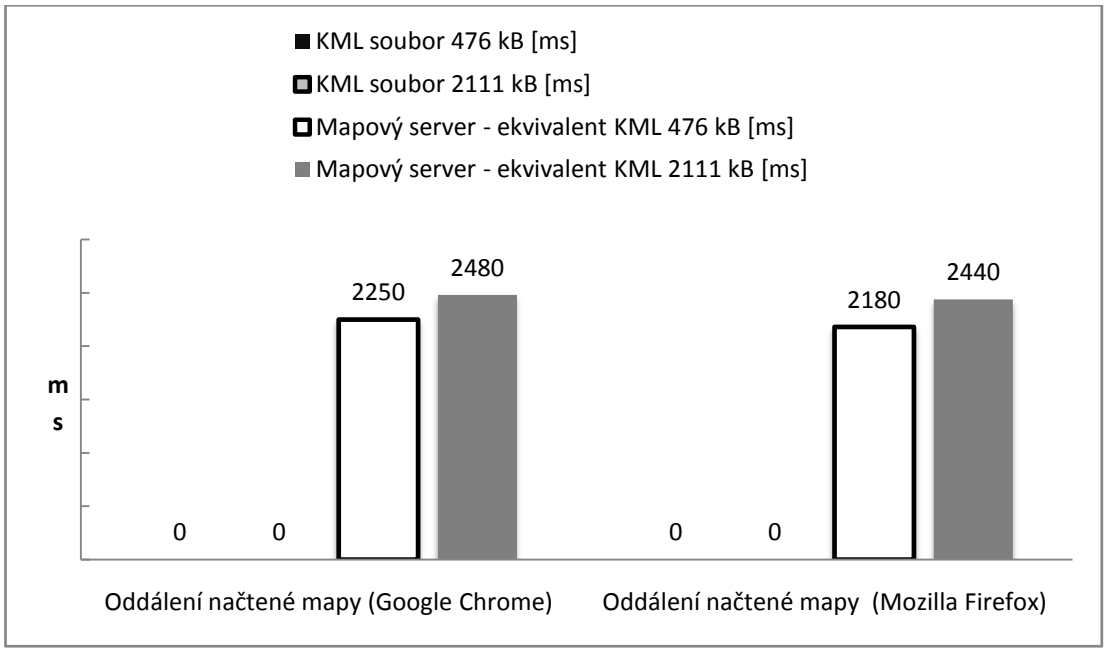


Obr. 6 Načtení čisté vrstvy - Sestava 4

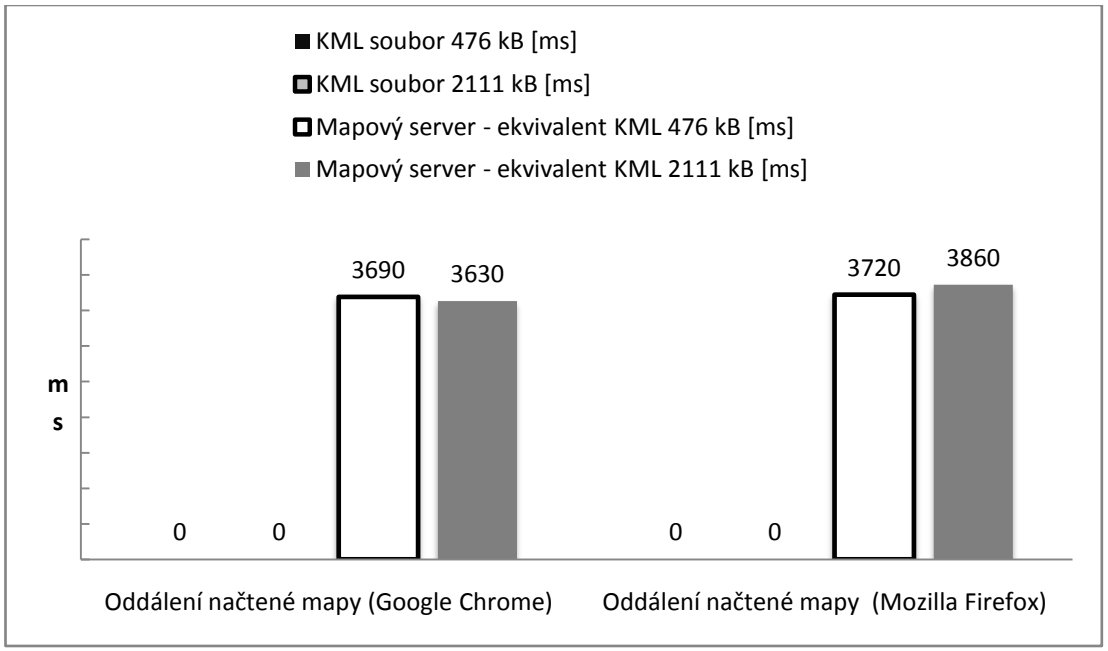
3.3.2 Oddálení a přiblížení mapy

Oddálení a přiblížení načtené mapy jsou další operace, které byly měřeny. Právě v těchto operacích se ukázala výhoda KML souborů. Pokud již jednou načteme KML soubor a zobrazíme jej, tak ani vymazání veškeré mezipaměti, historie a cookies

nenechá zobrazený soubor skrýt a ten tak zůstane zobrazený. V rámci tohoto chování KML souboru zůstane zobrazen i při přechodu přiblížení, a to na větší, i na menší úroveň. Tudíž při přechodu na odlišnou úroveň vrstvy zobrazené KML soubory nenačítají žádná nová data, ale pouze zaostří zobrazenou vrstvu. Tato činnost je prakticky okamžitá.

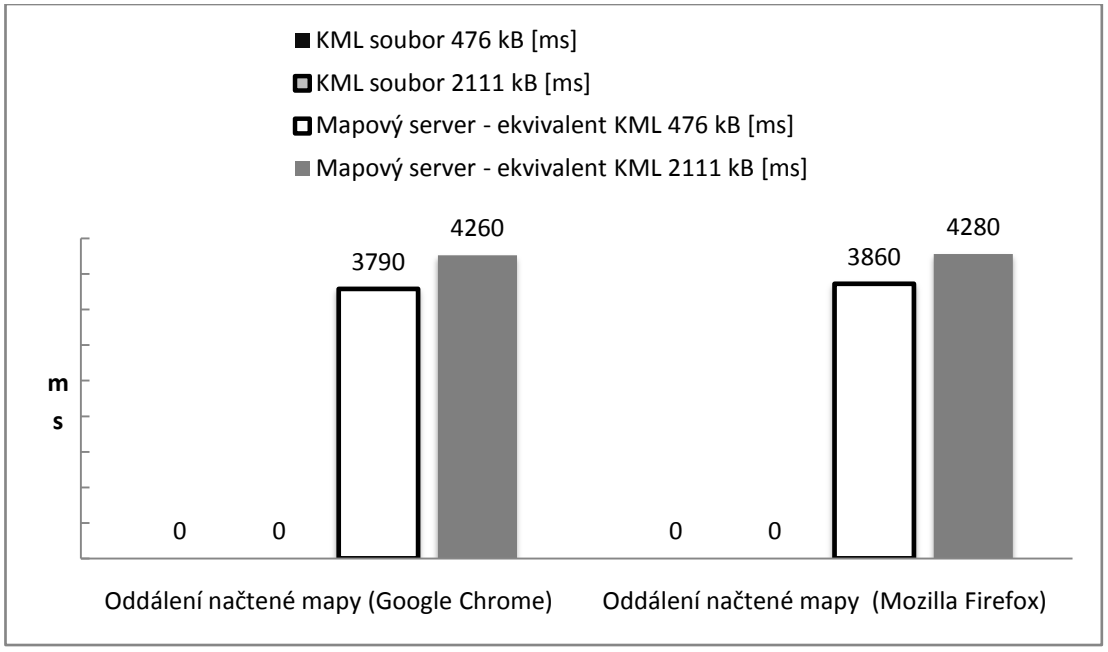


Obr. 7 Oddálení mapy - Sestava 1



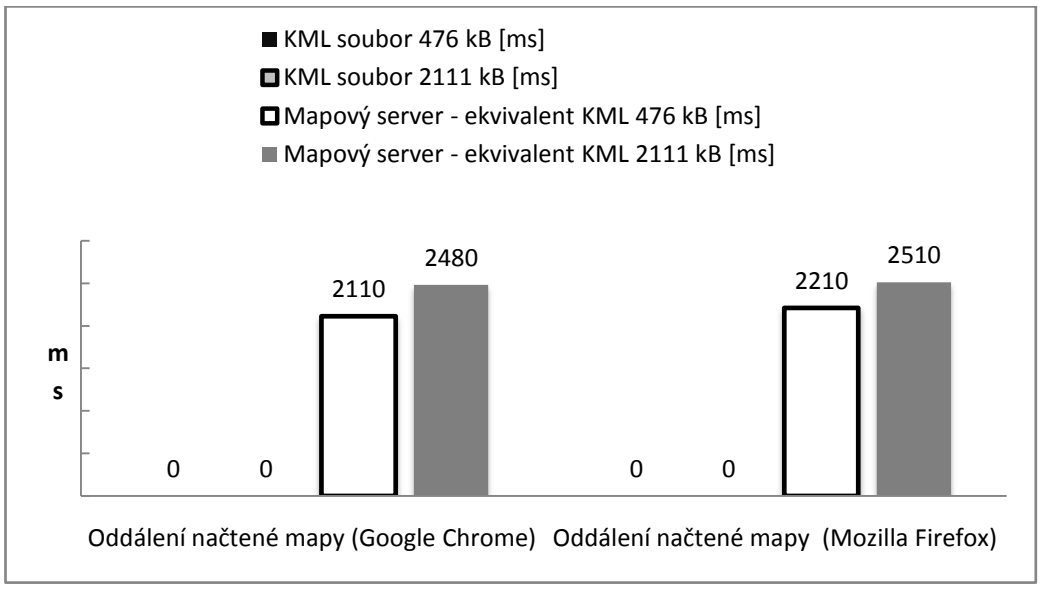
Obr. 8 Oddálení mapy - Sestava 2

Na rozdíl od KML souboru, vrstvy zobrazené z mapového serveru při přiblížení a oddálení nová data načítají. Tato skutečnost je důsledkem toho, že data zobrazená za pomoci WMS vrstvy jsou načtené rastrové obrázky, a ty jsou načtené vždy pouze pro jednu úroveň přiblížení.

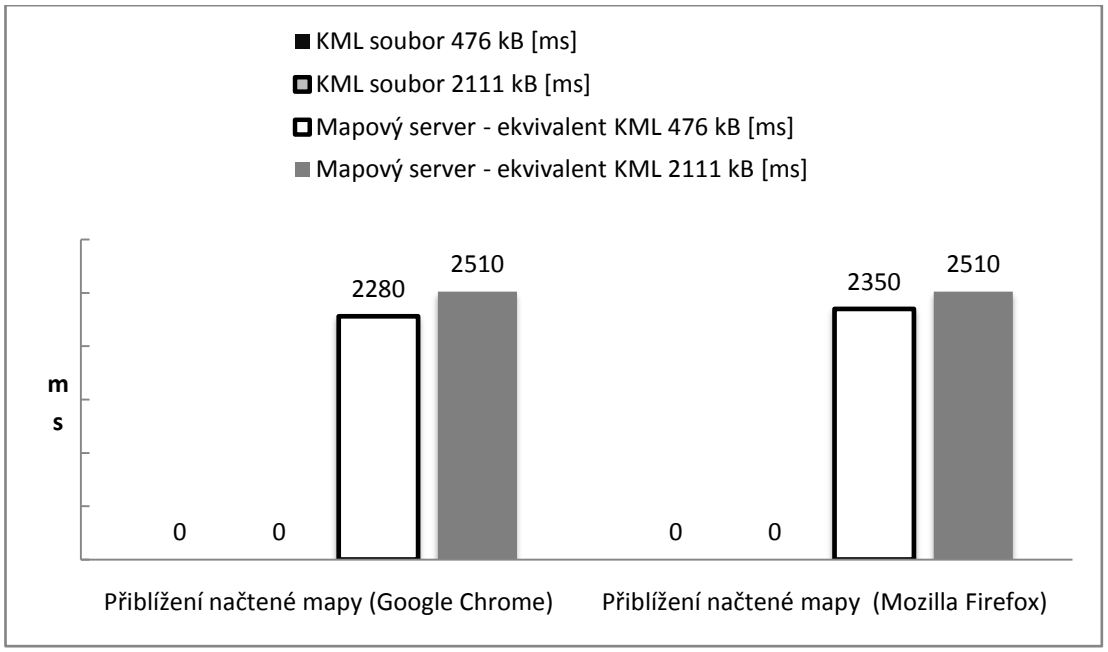


Obr. 9 Oddálení mapy - Sestava 3

Pokud tedy nebude využita mezipaměť prohlížeče, bude vždy překreslení mapového serveru stahovat nová data ve formě jednotlivých čtverců, ať se již jedná o přiblížení, oddálení nebo posun mapy.

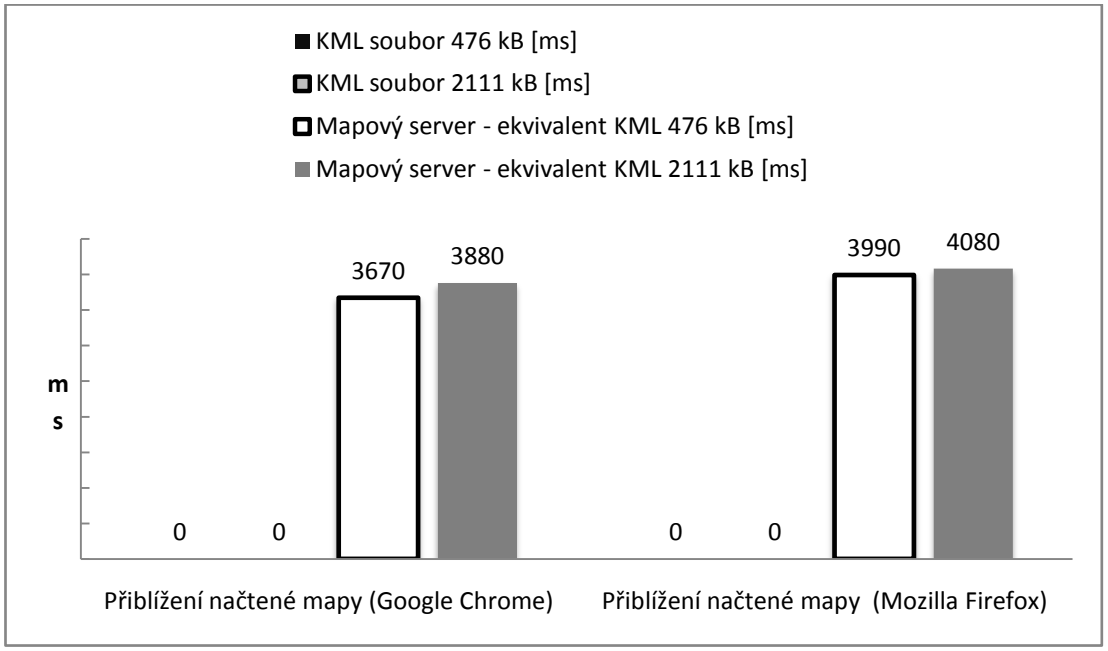


Obr. 10 Oddálení mapy - Sestava 4



Obr. 11 Přiblížení mapy - Sestava 1

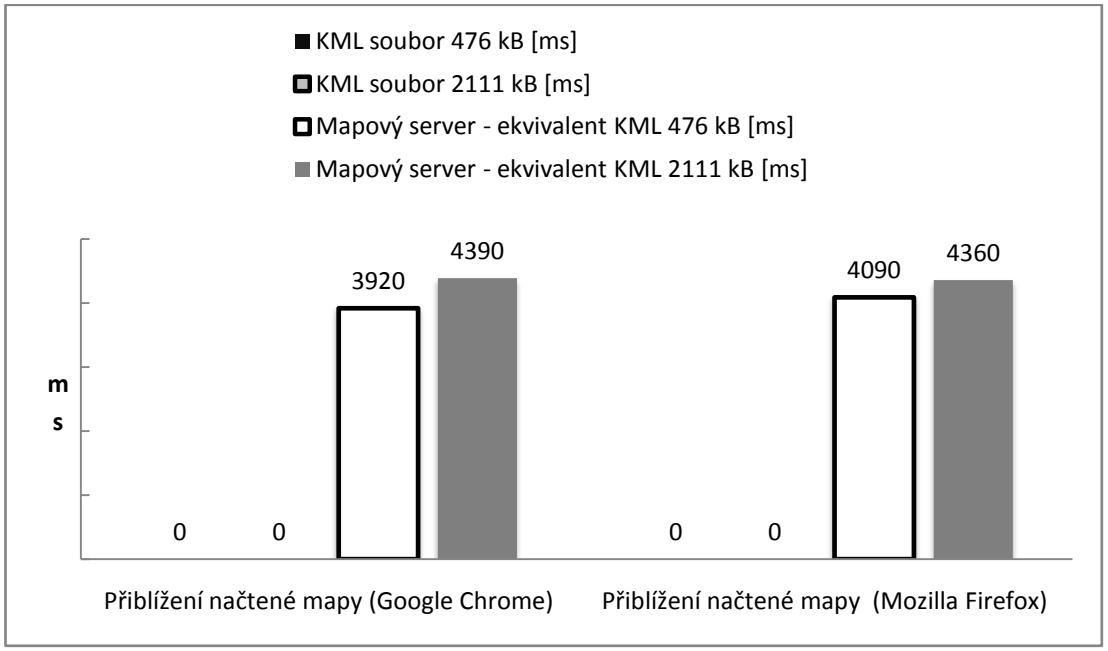
Tato skutečnost má za následek, že časy naměřené při přiblížení a oddálení mapy jsou velice podobné načtení čisté vrstvy, jelikož server provádí při obou operacích prakticky stejnou činnost, a to překreslení celé zobrazené vrstvy.



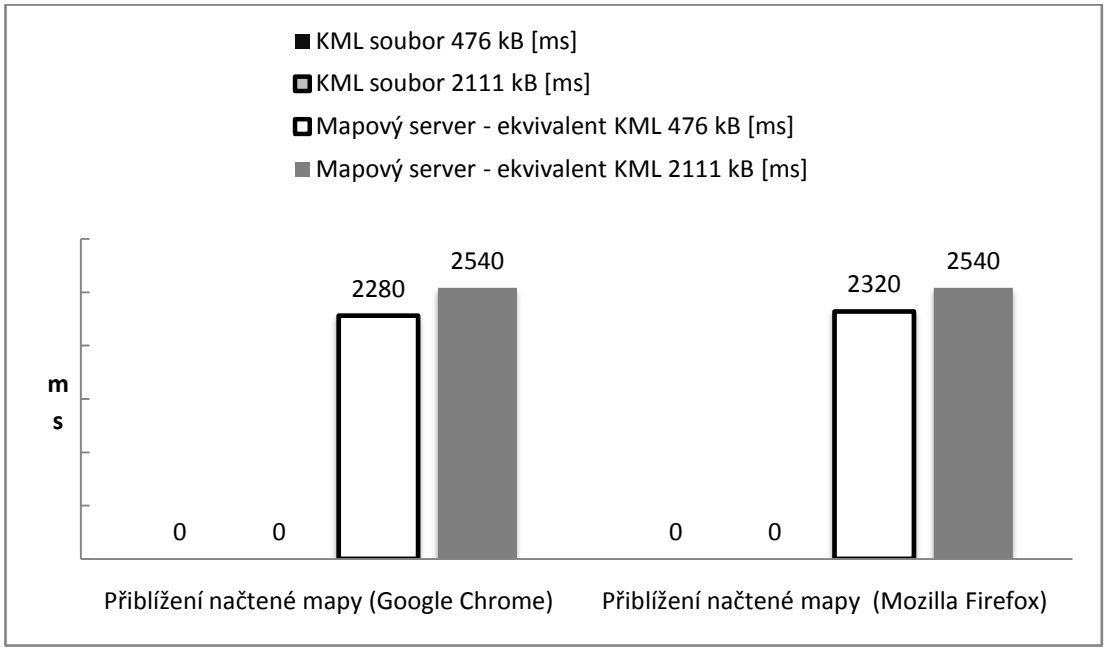
Obr. 12 Přiblížení mapy - Sestava 2

I při měření přiblížení a oddálení se potvrdilo, že načítání z mapového serveru není tolik ovlivněno rychlostí internetového připojení, ale spíše výkonem dané sestavy. Sestava 2, tedy slabší počítač s rychlým internetovým připojením je oproti těm

s větším výpočetním výkonem pomalejší o přibližně 1,5 sekundy, zatímco rozdíl pomalejšího počítače s pomalejším internetovým připojením činil jen o 300-600 ms více.



Obr. 13 Přiblížení mapy - Sestava 3



Obr. 14 Přiblížení mapy - Sestava 4

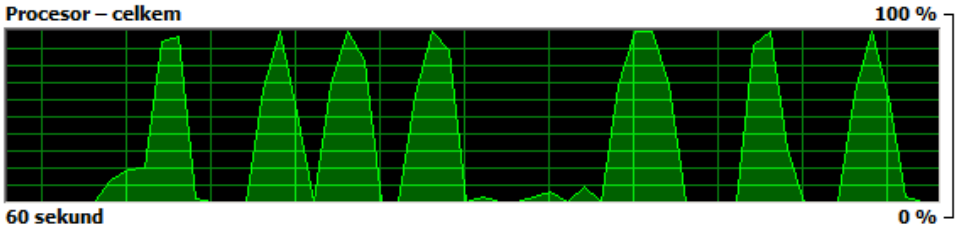
Výsledek porovnání je tedy překvapivý, jelikož, ač se jednalo o menší, či větší KML soubor, tak na třech ze čtyř sestav bylo načítání rychlejší, než v případě WMS vrstev z mapového serveru. Pouze v jednom případě, kdy bylo internetové připojení

razantně pomalejší, než v ostatních případech, tak se server načítání z KML souborů vyrovnal, ale nepředstihl jej.

Mapový server se tedy při rychlejším internetovém připojení na datech malé obce nevyplatí a je časově výhodnější využít souborů KML. Pokud však bude předpokladem provoz aplikace na pomalém připojení, přibližně 3 Mbit/s downloadu, tak se může stát, že zobrazení KML souborů bude pomalejší, jelikož data, která server zasílá, jsou v průměru v řádů stovek kB, na rozdíl od KML souborů, které na testovacích datech nabývaly hodnot až 2 MB.

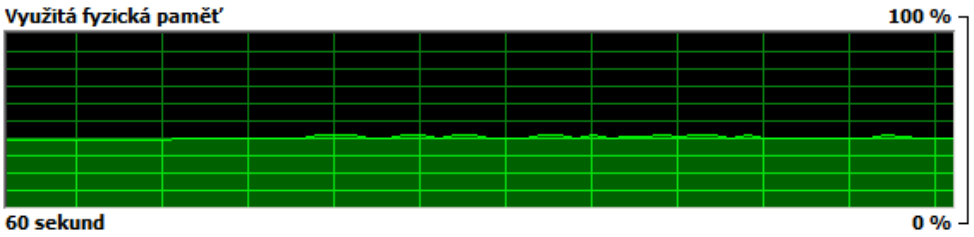
Dá se také předpokládat, že při zvětšení KML souborů na více než dvojnásobek dojde ke zpomalení jejich doby načítání, což je ovlivněno samotnou velikostí daného souboru. Data z mapového serveru si však svou velikost zachovávají a tudíž se nebudou příliš lišit.

Další možností je využití výkonnějšího serveru, což však může mít za následek zvýšení finanční náročnosti celého řešení, ať se již jedná o vyšší měsíční poplatky za pronájem serveru, či nákup dedikovaného serveru pod správou obce. Při jednotlivých požadavcích totiž procesor hostovaného serveru indikoval krátkodobé vytížení 100%, jak je vidět na přiloženém snímku z aplikace Sledování prostředků, lze snadno odhadnou, kdy došlo k dotazu na server. Více jader procesoru by tedy mohlo zvýšit výkonnost serveru a zkrátit načítací časy.



Obr. 15 Využití procesoru serveru při sedmi požadavcích

Při porovnání se zdá, že výše paměti RAM serveru výkon a časy nezlepší, jelikož, jak je vidět na snímku sledování paměti RAM, její využití se radikálně nezměnilo.



Obr. 16 Využití paměti při osmi požadavcích

Pro malá data tedy není nutně potřebné realizovat chod mapového serveru, jelikož KML soubory vytvořené z těchto dat jsou dostatečně malé, aby jej v načítání předčily.

4 Klientská aplikace

Ukázková aplikace pro zobrazení mapových dat byla vytvořena převážně v jazyce javascript. Její funkce byly vytvořeny s ohledem na požadavky na mapový portál malé obce. Zahrnují například měření vzdáleností a plochy, zobrazení katastrální mapy a zajištění odkazů do náhledu katastru nemovitostí. Aplikace také obsahuje implementaci aktivní vrstvy, výběru z ní a následné zobrazení dat, které daný prvek obsahuje. Spolu se zobrazením dat umožňuje aktivní vrstva také implementaci obrázkové legendy jednotlivých vrstev. Mezi funkce patří také strom vrstev s možností tvorby skupin, které umožňují jednotlivé vrstvy sdružovat dle obdobného obsahu.

4.1.1 Použité knihovny

V rámci vytvořené klientské aplikace bylo použito několik knihoven tak, aby byla zajištěna požadovaná funkčnost a reprezentační vzhled. Mezi tyto knihovny patří Openlayers ve verzi 2.13.1, Ext JS verze 3.4.0, Proj4js a GeoExt verze 1.1.

4.1.1.1 Openlayers 2.13.1

Openlayers je javascriptové API, které bylo poprvé uvedeno v roce 2006 a od roku 2007 je součástí neziskové nadace Open Source Geospatial Foundation, který se zabývá podporou volných geoinformačních technologií. Openlayers slouží k vytvoření mapové aplikace pro použití na webových stránkách a nabízí podporu velkého množství formátů a technologií. Openlayers je prohlížeči podporováno velmi dobře, jelikož stačí podpora jazyka JavaScript. Jak již bylo napsáno v rámci porovnání, tak Openlayers je API dostupné zdarma v rámci BSD licence.

4.1.1.2 Ext JS 3.4.0

Ext JS je javascriptová knihovna, která umožní vytváření interaktivních webových aplikací s velkým množstvím ovládacích prvků, jakými jsou nejrůznější tlačítka, panely a nástroje. Ext JS tak umožňuje vytvoření profesionálně vypadajícího layoutu a zlepší pohodlí při ovládání webové stránky. Ext JS je dostupné v rámci dvou licencí, open source licence a licence pro komerční využití. V případě použití open source řešení se licence řídí GPL licencí ve verzi 3[5], která mimo jiné nařizuje volný

[5] The GNU General Public License v3.0 - GNU Project - Free Software Foundation. *Operační systém GNU* [online]. 2007 [cit. 2014-03-08]. Dostupné z: http://www.gnu.org/copyleft/gpl.html

přístup k vytvořenému kódu a možnost editace a úpravy kódu dle potřeb uživatele. Pokud však aplikace využívá jiných knihoven, například Openlayers, které je licencované na základě BSD licence, je nutné zkontrolovat, zda splňuje podmínky výjimky, kterou tvůrci Ext JS, společnost Sencha Inc., umožňují licencovat výslednou aplikaci pod méně striktní licencí. Text výjimky je dostupný na webových stránkách Ext JS[6]a její poslední verze 1.0.4 byla vydána na začátku roku 2013. Pokud nechceme, aby byl zdrojový kód aplikace volně dostupný, je nutné zaplatit komerční licenci, jejíž podmínky lze opět nalézt na webových stránkách Ext JS[7].

4.1.1.3 GeoExt 1.1

GeoExt je javascriptová knihovna, která slouží jako most mezi API Openlayers a knihovnou Ext JS. Umožňuje tedy vytvořit aplikaci pro prohlížení map Openlayers v layoutu Ext JS a obsahuje nástroje pro toto spojení. GeoExt tedy umožňuje využití stromu pro mapové vrstvy, včetně možností aktivní vrstvy a seskupení vrstev, panelů pro mapu i jednotlivé textové informace, či pokročilé popupy. Knihovna je dostupná na základě BSD licence a je tedy zdarma.

4.1.1.4 Proj4js

Knihovna Proj4jsje konverze knihovny PROJ.4[8], která byla napsána v jazyce C, do jazyka JavaScript. Knihovna se zabývá transformací souřadnic mezi jednotlivými souřadnicovými systémy. Použití Proj4js je v aplikaci nutné pro definici souřadnicového systému S-JTSK pro použití s katastrem nemovitostí ČÚZK, který v Openlayers chybí. Proj4js je k dispozici zdarma pod MIT licencí.

4.1.2 HTML část aplikace

Základem webové aplikace je samozřejmě webová stránka. Ta je vytvořena za pomoci HTML jazyka a je umístěna v souboru index.html. Pro bezproblémové použití Ext JS aplikace ve všech prohlížečích je doporučeno definovat doctype dokumentu jako HTML5 doctype.

---

[6] Exception for Applications | Open Source FAQ | Legal | Sencha. *HTML5 App Development for Desktop and Mobile. JavaScript Frameworks and Dev Tools from Sencha. | Home | Sencha* [online]. 2013 [cit. 2014-03-08]. Dostupné z: http://www.sencha.com/legal/open-source-faq/open-source-license-exception-for-applications/

[7] Commercial License Information | Legal | Sencha. *HTML5 App Development for Desktop and Mobile. JavaScript Frameworks and Dev Tools from Sencha. | Home | Sencha* [online]. 2014 [cit. 2014-03-08]. Dostupné z: http://www.sencha.com/legal/license-overview

[8] *PROJ.4* [online]. 2013 [cit. 2014-03-08]. Dostupné z: http://trac.osgeo.org/proj/

```
<!DOCTYPE html>
```

V hlavičce dokumentuje poté standardně definován titul stránky a nastavena znaková sada pro UTF-8, pomoci elementů title a meta.

Následuje připojení jednotlivých knihoven, které byly použity k vytvoření aplikace za pomoci elementů script. Pro Openlayers je nutné připojit soubor OpenLayers.js, který se nachází v hlavní složce staženého Openlayers.

```
<script src="OpenLayers/OpenLayers.js"
type="text/javascript"></script>
```

Pro použití knihovny Ext JS je nutné v hlavičce připojit tři soubory, dva javascriptové a jeden CSS soubor. Soubor ext-all.js lze nalézt v hlavní složce po stažení Ext JS a soubor ext-base.js se nachází v podložkách adapter a ext. CSS soubor ext-all.css se nachází v podložkách resources a css. Soubory poté připojíme standardním způsobem.

```
<script src="ext-3.4.0/adapter/ext/ext-base.js"
type="text/javascript"></script>

<script src="ext-3.4.0/ext-all.js" type="text/javascript"></script>

<link rel="stylesheet" type="text/css" href="ext-
3.4.0/resources/css/ext-all.css" />
```

Připojení knihovny GeoExt je obdobné, jako v případě Openlayers a potřebuje pouze jeden soubor, konkrétně GeoExt.js, který se nachází v podsložce lib staženého GeoExt.

```
<script src="GeoExt/lib/GeoExt.js" type="text/javascript"></script>
```

Poslední knihovnou, kterou je pro běh aplikace nutné implementovat je převodní knihovna Proj4js. Tuto knihovnu k aplikaci připojíme za pomoci jediného souboru proj4js-combined.js, který se nachází v podsložce lib ve staženém archivu Proj4js.

```
<script src="proj4js/lib/proj4js-combined.js"></script>
```

Pokud je aplikace vytvořena v externím souboru, je nutné jej v hlavičce také připojit. V případě této aplikace se tento soubor nazývá kajov.js a nachází se v kořenovém adresáři společně se souborem index.html.

```
<script type="text/javascript" src="kajov.js" charset="UTF-
8"></script>
```

Pro potřeby exportu mapy je dále připojen soubor ExportMap.js, který obsahuje třídu vytvořenou v rámci institutu Camptocamp.[9] Tento soubor, dostupný z OpenLayers Wiki[10], byl zveřejněn Tobiasem Sauerweinem V rámci Clear BSD licence[11] a je tedy dostupný jako OpenSource. V rámci aplikace je tento soubor uložen v kořenovém adresáři a připojen stejným způsobem, jako soubor kajov.js.

```
<script src="ExportMap.js" type="text/javascript"></script>
```

V rámci hlavičky je poté ještě připojen soubor styl.css, který obsahuje styly pro jednotlivé panely aplikace. Tento soubor může být nahrazený libovolným CSS souborem, ve kterém se budou nacházet definice stylů pro obsah panelů.

```
<link rel="stylesheet" type="text/css" href="styl.css"/>
```

Po hlavičce je nutné připravit také element body, čili tělo html stránky, který obsahuje několik elementů div, následně využitých v rámci aplikace. Jedná se o elementy, které jsou potřebné k vytvoření jednotlivých panelů pro použití s grafickým rozhraním Ext JS.

Div, který má nastaveno id legenda je navázán na panel s legendou, do kterého se poté zobrazí. Elementy, které mají id mereni_vzdalenost a mereni_plocha,jsou propojeny s panely měření vzdálenosti a plochy, které poté požadovanou hodnotu na základě obsahu těchto divů zobrazí. Tyto divy lze formátovat v rámci CSS a to dle jejich id v přiloženém CSS souboru styl.css.

```
<div id="legenda">V tomto panelu naleznete případnou legendu dané
vrstvy</div>

<div id="mereni_vzdalenost">V tomto panelu naleznete výsledky měření
vzdálenosti</div>

<div id="mereni_plocha">V tomto panelu naleznete výsledky měření
plochy</div>

<div id="multiVyber">V tomto panelu naleznete informace o prvcích
v případě, že využijete hromadný výběr.</div>
```

[9] OpenLayers and HTML5. SAUERWEIN, Tobias. *OpenLayers Wiki* [online]. 2010 [cit. 2014-03-18]. Dostupné z: http://trac.osgeo.org/openlayers/wiki/Future/OpenLayersAndHTML5

[10] http://trac.osgeo.org/openlayers/browser/sandbox/camptocamp/canvas/openlayers/lib/OpenLayers/Control/ExportMap.js

[11] http://svn.openlayers.org/trunk/openlayers/license.txt

Poslední částí elementu body je div obsahující HTML5 element canvas s id exportovanyObrazek, který slouží k exportu prvků do obrázku a jeho následném zpracování v rámci exportu. Element canvas je poté zabalen do div elementu, který má nastavenou CSS třídu skryty z důvodu skrytí canvas elementu tak, aby se nezobrazoval v rámci aplikace, ale pouze při exportu. Definice CSS třídy skryty se nachází v souboru styl.css.

4.1.3 Javascriptová část aplikace

Hlavní část aplikace se nachází v souboru kajov.js, který po načtení zobrazí veškeré ovládací prvky, i mapu samotnou. Spuštění aplikace obstará funkce Ext.onReady(), ve které je převážná část kódu zabalena. Ta zajistí vykreslení všech prvků v tu chvíli, kdy jsou připraveny. Mimo tuto funkci jsou definovány funkce, které jsou potřebné k běhu aplikace a jsou v rámci funkce Ext.onReady() volány.

4.1.3.1 Proměnné a dodatečné definice

Pro běh aplikace je nutné definovat několik základních proměnných. V ukázkové aplikaci se nachází na začátku kódu a patří mezi ně například proměnná pro mapu, mapový panel, či aktivní vrstvu. Tyto proměnné jsou většinou pouhé názvy, které jsou použity dále v kódu. Výjimku tvoří proměnné format, nastrojeListy, aktivniVrstvaLayer a aktivniVrstvaNazev, které je nutné také inicializovat. Proměnná nastrojeListy je inicializována na prázdné pole, do kterého se poté přidávají jednotlivé nástroje z horní nástrojové lišty. Proměnné aktivniVrstvaNazev a aktivniVrstvaLayer jsou nastaveny na hodnotu null z důvodu odstranění chyby, kdy byly proměnné nedefinované. Proměnná aktivniVrstvaNazev obsahuje textovou reprezentaci jména aktivní vrstvy tak, jak se zobrazuje v přepínači vrstev. AktivniVrstvaLayer obsahuje objekt typu OpenLayers.Layer a je možné s ní pracovat jako s jakoukoliv vrstvou. Toto rozdělení bylo použito k odlišení názvu od objektu a zjednodušení čtení kódu aplikace. Proměnná format je určena pro vektorové vrstvy ve formátu kml a její parametry extractStyles a extractAttributes určují, zda se z kml souboru mají zobrazit také jeho styly a atributy. Oba parametry jsou nastaveny na hodnotu true.

```
var aktivniVrstvaNazev = null;
var aktivniVrstvaLayer = null;
var format = new OpenLayers.Format.KML({
      extractStyles: true,
      extractAttributes: true
});
```

Mimo proměnných je nutné také definovat souřadnicový systém S-JTSK pod kódem EPSG:5514[12], který je dále využit v rámci nahlížení do katastru nemovitostí. Tento souřadnicový systém v Openlayers chybí a je potřeba jej definovat za pomoci knihovny Proj4js.

```
Proj4js.defs["EPSG:5514"] = "+proj=krovak +lat_0=49.5
+lon_0=24.83333333333333 +alpha=30.28813972222222 +k=0.9999 +x_0=0
+y_0=0 +ellps=bessel
+towgs84=570.8,85.7,462.8,4.998,1.587,5.261,3.56 +units=m +no_defs";
```

Následuje definice prototypů pro zavedení proměnných filtr a legendaCesta a modifikaci funkce selectBox. Prototyp umožní přidat objektu, či funkci nové vlastnosti, které lze poté využít.

Prototyp filtr je zaveden pro potřeby třídění vrstev, aby je šlo zobrazit v rámci skupin v přepínacím panelu. Parametr filtr předáme třídě OpenLayers.Layer a je tedy dostupný pro všechny její instance a podtřídy.

```
OpenLayers.Layer.prototype.filtr = null;
```

Prototyp legendaCesta je stejně jako filtr předán třídě OpenLayers.Layer a je určen pro uložení cesty k obrázkovému souboru s legendou pro použití v panelu legenda. Parametr legendaCesta je poté využit v rámci přepínání aktivní vrstvy.

```
OpenLayers.Layer.prototype.legendaCesta = null;
```

Posledním prototypem, který se v aplikaci nachází je selectBox. Jedná se o modifikaci funkce selectBox, která se nachází ve třídě OpenLayers.Control.SelectFeature. SelectBox umožní vybrat více prvků z mapy za pomoci výběrového obdélníku. Aplikace poté může prvky zpracovat hromadně.

[12] EPSG:5514: S-JTSK / Krovak East North -- SJTSK. *EPSG.io: Coordinate Systems Worldwide* [online]. 2014 [cit. 2014-03-20]. Dostupné z: http://epsg.io/5514-1623

Oproti původní funkci se v prototypu nachází několik úprav. První z nich je zavedení nových proměnných vybranePrvky a vybran. Proměnná vybranePrvky je pole, které uchovává jednotlivé vybrané prvky a vybran určí, zda byl vybrán alespoň jeden prvek.

```
var vybranePrvky = [];

var vybran = false;
```

Další modifikací je přidání podmínky, která zajistí přeskočení prvků s nulovou geometrií. Tyto prvky, které závisí na dodaných datech, způsobovaly chybu a nevybrání všech ostatních prvků. Při výběru je tedy nutné je přeskočit.

```
if (feature.geometry === null) {
      continue;
}
```

V případě, že je prvek vybrán, dojde na další modifikaci, kdy je zkontrolováno, zda je z aktivní vrstvy a následně přidán do pole vybraných prvků. Proměnná vybran je následně nastavena na true.

```
if (feature.layer.name === aktivniVrstva) {
      vybranePrvky.push(feature);
      vybran = true;
}
```

Na závěr funkce je přidáno volání funkce zpracovatVybrane, které je předáno pole vybraných prvků a uvolnění nástroje za pomoci příkazu unsellectAll`()`. Uvolnění je nutné provést z toho důvodu, aby při novém výběru nebyly do pole vloženy také prvky z minulých měření.

```
if (vybran === true) {
      zpracovatVybrane(vybranePrvky);
}
this.unselectAll();
```

Dále jsou v kódu definovány doplňkové funkce, které umožňují použití některých přidaných vlastností.

4.1.3.2 Dodatečné funkce

Pro nové vlastnosti aplikace bylo nutné přidat několik funkcí, které umožní jejich fungování. Definice těchto funkcí jsou uvedeny v souboru kajov.js a nachází se mezi definicemi proměnných a začátkem funkce Ext.onReady().

4.1.3.2.1 Export prvků

V rámci aplikace je možné nechat si vyexportovat informace a náhled prvků aktivní vrstvy, které jsou momentálně viditelné. Export prvků využívá již zmíněného souboru ExportMap.js, bez něhož nebude její část, jenž zajišťuje export prvků ve formě obrázku, fungovat.

V rámci exportu prvků jsou použity tři zásadní funkce. První z nich je exportMapy(), která ve dvou if podmínkách kontroluje, zda je aktivní vrstva nastavena a viditelná. Pokud je aktivní vrstva nastavena a viditelná, je zavolána hlavní funkce vybratPrvkyProExport(). V opačném případě je uživatel upozorněn chybovým hlášením.

```
function exportMapy() {
	if (aktivniVrstvaNazev !== null) {
		if (aktivniVrstvaLayer.getVisibility() !== false) {
			vybratPrvkyProExport();
		}
		else {
			alert("Zobrazte aktivní vrstvu");
		}
	}
	else {
		alert("Vyberte aktivní vrstvu");
}
```

VybratPrvkyProExport() je hlavní funkcí exportu prvků, která sestavuje z viditelných prvků novou webovou stránku, kterou nakonec zobrazí. Ve svém průběhu využívá funkce převážně proměnné exportOkno, která reprezentuje html kód nově vytvářeného okna. Tento kód je do proměnné ukládán v řetězcové podobě. Pokud se během vytváření html struktury vyskytne část začínající class=\", následuje název CSS třídy, pomoci které se dá daný prvek stylovat.

```
function vybratPrvkyProExport() {
      exportOkno = "<html><head>";
      exportOkno += "<meta http-equiv=\"Content-Type\"
      content=\"text/html\">";
      exportOkno += "<meta http-equiv=\"Content-Type\"
      content=\"text/html; charset=utf-8\">";
      exportOkno += "<link rel=\"stylesheet\" type=\"text/css\"
      href=\"styl_export.css\"></link>";
      exportOkno += "<title>Export</title>";
      exportOkno += "</head>";
      exportOkno += "<body>";
      exportOkno += "<h1 class=\"nadpisHlavni\">Export prvků
      z mapy</h1>";
      exportOkno += "<h2
      class=\"podnadpisVypis\">Výpis informací</h2>";
      exportOkno += "<table class=\"tabulkaPrvku\">";
```

Po přípravě začátku html struktury je v rámci funkce nutné vybrat viditelné prvky, z nichž se bude výpis skládat. Funkce projde veškeré prvky aktivní vrstvy a porovná, zda mají nenulovou informaci o své geometrii. Pro nenulové prvky poté za pomoci funkce onScreen() určí, zda se nacházejí v zobrazené části mapy a pokud ano, tak daný prvek přidá do pole vybranePrvky.

```
      var vybranePrvky = [];
      for (var i = 0; i < aktivniVrstvaLayer.features.length; ++i) {
            if (aktivniVrstvaLayer.features[i].geometry === null) {
                  continue;
            }
            if (aktivniVrstvaLayer.features[i].getVisibility() ===
            true && aktivniVrstvaLayer.features[i].onScreen() ===
            true) {
                  vybranePrvky.push(aktivniVrstvaLayer.features[i]);
            }
      }
```

Po dokončení výběru prvků metoda vytvoří další část tabulky, do které poté cyklem vypíše popisy jednotlivých prvků. Tento zápis je prováděn v rámci for cyklu. Uvnitř cyklu je kontrolována podmínka, která vyřadí nedefinované popisy. Samotný zápis je realizován jednotlivými buňkami tabulky, přičemž proměnná radek určuje, kolik prvků se vypíše na jeden řádek. V ukázkové aplikaci je tato hodnota nastavená na číslici tři. Na konci každého řádku se zapíše také konec tr elementu a začátek nového. Proměnná řádek se poté vynuluje.

```
exportOkno += "<tr>";
var radek = 0;
for (var j = 0; j < vybranePrvky.length; j++) {
      if (typeof vybranePrvky[j].attributes.description !==
      "undefined") {
            exportOkno += "<td>" +
            vybranePrvky[j].attributes.description + "</td>";
            radek++;
            if (radek === 3) {
                  exportOkno += "</tr><tr>";
                  radek = 0;
            }
      }
}
```

Poté, co jsou všechny vybrané prvky zapsány do proměnné exportOkno dokončí funkce html definici tabulky a vytvoří html elementy pro exportovaný obrázek.

```
exportOkno += "</tr>";
exportOkno += "</table>";
exportOkno += "<h2 class=\"podnadpisZnazorneni\">Znázornění
prvků z vrstvy</h2>";
exportOkno += "<div id=\"exportObrazekDiv\"> ";
exportOkno += "<img id=\"exportObrazek\" /> ";
exportOkno += "</div> ";
exportOkno += "</body></html>";
```

Posledním krokem v rámci funkce vybratPrvkyProExport() je vytvoření prázdného okna uloženého v proměnné oknoExport a jeho otevření. Do tohoto okna je poté zapsán obsah proměnné exportOkno, tedy vytvořený html kód. Po dokončení okna je následně zavolána funkce ziskatMapu, která jako parametr požaduje právě vytvořené okno.

```
      var oknoExport = window.open("");
      oknoExport.document.write(exportOkno);
      ziskatMapu(oknoExport);
}
```

Funkce ziskatMapu je založena na použití již zmíněného souboru ExportMap.js a vytváří obrázek ve formátu png, získaného z předpřipraveného elementu canvas, na který jsou vykreslovány jednotlivé vektorové vrstvy.

```
function ziskatMapu(okno) {
```

V první řadě je v rámci exportu obrázku nutné uložit element canvas, který je v html souboru aplikace uveden s id exportovanyObrazek do proměnné canvas, se kterou bude nadále pracováno.

```
var canvas = OpenLayers.Util.getElement("exportovanyObrazek");
```

Následuje předání této proměnné ovládacímu prvku nastrojExportMapa a jeho spuštění za pomoci funkce trigger, které je předána právě zmíněná proměnná canvas.

```
nastrojExportMapa.trigger(canvas);
```

Po spuštění exportu je do proměnné odkazObrazku uložen url odkaz na proměnnou canvas, který je následně předán jako zdroj pro element img, který byl v exportním okně vytvořen s id exportObrazek. Náhled prvků se poté zobrazí v rámci okna na jeho konci.

```
var odkazObrazku = canvas.toDataURL();
okno.document.getElementById("exportObrazek").src =
odkazObrazku;
}
```

Elementy v rámci okna exportu je možné stylovat standardními CSS příkazy. Použité styly pro exportní okno jsou uloženy v souboru styl_export.css a stylovat je možné za použití id, která byla přidána během tvorby okna jednotlivým elementům.

4.1.3.2.2 Nahlížení do katastru nemovitostí

Nahlížení do katastru nemovitostí[13] je bezplatná funkce, kterou nabízí Český úřad zeměměřický a katastrální (ČÚZK). V rámci nahlížení do katastru je možné vyhledat volně dostupné informace z katastru nemovitostí nejrůznějších parametrů, jakými je například číslo parcely, nebo číslo popisné. Nahlížení do katastru však také umožňuje vyhledávání dle zeměpisných souřadnic. Tohoto způsobu ukázková aplikace využívá pro svou funkci katastr, při které uživatel klikne na mapu a v novém okně se otevře náhled do katastru na dané souřadnice.

V rámci ukázkové aplikace jsou s nahlížením do katastru svázány dvě funkce. První z nich je zapnoutKlikKatastr(), která po kliknutí na tlačítko Katastr registruje událost

[13] Nahlížení do katastru nemovitostí. *ČUZK* [online]. 2014 [cit. 2014-03-29]. Dostupné z: http://nahlizenidokn.cuzk.cz/

pro kliknutí na mapu. Funkce zapnoutKlikKatastr() je volána v rámci tlačítka nástrojové lišty.

```
function zapnoutKlikKatastr() {
```

Registrace události click probíhá funkcí events.register, která je volána za pomoci mapy, na kterou je událost registrována. Prvním parametrem je název události, v tomto konkrétním příkladu se jedná o událost "click". Druhým parametrem je proměnná mapy, pro kterou je událost registrována, tedy proměnná mapa. Posledním parametrem je název funkce, která se po kliknutí má zavolat. V rámci ukázkové aplikace se jedná o funkci katastr(), jejíž název bez závorek je uveden jako poslední parametr.

```
    mapa.events.register("click", mapa, katastr);
```

Funkce po registraci nastaví parametr klikZapnuty na hodnotu true a dá tedy najevo, že je kliknutí aktivní.

```
    klikZapnuty = true;
}
```

Hlavní v rámci nahlížení do katastru nemovitostí je funkce katastr(e), která zpracovává kliknutí, převádí jej na zeměpisné souřadnice a otevírá okno s náhledem.

Funkce je volána s parametrem e, ve kterém je uložena samotná událost kliknutí, její parametry, souřadnice, atd.

```
function katastr(e) {
```

Prvním krokem ve zpracování kliknutí je získání zeměpisných souřadnic ze souřadnic jednotlivých pixelů kliknutí. K tomuto účelu nabízí OpenLayers funkci getLonLatFromViewPortPx, které je předána proměnná xy události e. Funkce poté do proměnné sirkaDelka uloží souřadnice v souřadnicovém systému, ve kterém je zobrazena mapa. V ukázkové aplikaci se jedná o souřadnicový systém EPSG:900913.

```
    var sirkaDelka = mapa.getLonLatFromViewPortPx(e.xy);
```

Pro zobrazení náhledu do katastru je však nutné využít souřadnicového systému, ve kterém aplikace ČÚZK souřadnice přijímá. Jedná se systém S-JTSK, který je v rámci EPSG definován pod kódem 5514. Pro samotnou transformaci souřadnic nabízí

OpenLayers funkci transform, která je zavolána pomocí tečkové notace z proměnné sirkaDelka. Prvním parametrem funkce je projekce, ve které je zobrazená mapa, čili EPSG:900913. Druhým parametrem je cílová projekce, tedy EPSG:5514. Pro tento krok nesmí být zapomenuta zmíněná definice projekce 5514 za pomoci Proj4js.

```
sirkaDelka.transform(new OpenLayers.Projection("EPSG:900913"),
new OpenLayers.Projection("EPSG:5514"));
```

Dalším krokem je získání souřadnice x a y z proměnné sirkaDelka. Souřadnici x lze získat za pomoci proměnné lon a souřadnici y za pomoci proměnné lat. Tyto souřadnice je poté nutné zaokrouhlit na celá čísla za pomoci funkce Mat.round(), jelikož nahlížení do katastru nemovitostí nezvládá zpracovat desetinná čísla.

```
var sourdadX = Math.round(sirkaDelka.lon);
var sourdadY = Math.round(sirkaDelka.lat);
```

Na závěr je nutné otevřít okno s odkazem na nahlížení do katastru. Odkaz je tvořen dle následujícího příkazu a je nutné do něj vložit získané souřadnice x a y. Souřadnici x vložíme za část **&x=-** a y za část **&y=-** .Tvorba odkazu probíhá za pomoci standardní práce s řetězcem v jazyce Javascript.

```
window.open("http://nahlizenidokn.cuzk.cz/MapaIdentifikace.asp
x?l=KN&x=-" + sourdadX + "&y=-" + sourdadY);
}
```

4.1.3.2.3 Nastavení legendy

O nastavení obrázkové legendy v rámci panelu Legenda se v rámci ukázkové aplikace stará funkce legenda(), která do předpřipraveného html elementu legenda umístí obrázkový element img s cestou, která byla v rámci vrstvy uložena v parametru legendaCesta. Zobrazení legendy je vázáno na aktuální aktivní vrstvu.

```
function legenda() {
```

Získání elementu legenda a jeho uložení do parametru legenda je realizováno za pomoci funkce document.getElementById(), jejímž parametrem je právě html id hledaného elementu.

```
	var legenda = document.getElementById('legenda');
	alert(aktivniVrstvaNazev + " je nyní aktivní.");
```

Podmínka následně určí, zda má aktivní vrstva parametr legenda nastaven a pokud ano, nastaví parametr innerHTML, čili vnitřní html kód elementu daného proměnnou legenda na řetězcovou reprezentaci img elementu spojeného s parametrem legendaCesta jako cesty v atributu src.

```
	if (aktivniVrstvaLayer.legendaCesta !== null) {
		legenda.innerHTML = "<img src=\"" +
		aktivniVrstvaLayer.legendaCesta + "\">";
	}
```

Pokud nemá aktivní vrstva parametr nastaven a podmínka if je tedy nepravdivá, upozorní uživatele na tuto skutečnost ve větvi else. Obsah elementu legenda je poté nastaven na text, který se zde nacházel při načtení aplikace.

```
	else {
		legenda.innerHTML = "V tomto panelu naleznete případnou
		legendu dané vrstvy";
		alert(aktivniVrstvaNazev + " nemá nastaven parametr
		legenda.");
	}
}
```

4.1.3.2.4 Multivýběr

V rámci aplikace je možné vybírat prvky z mapy kliknutím, ale také multivýběrem v podobě tažení výběrového boxu. Multivýběr využívá mimo již zmíněného prototypu funkce selectBox ještě dvě další funkce.

Funkci zpracovatVybrane(vybranePrvky) je jako parametr předáno pole prvků, které byly vybrány v rámci funkce selectBox. Funkce následně vynuluje proměnnou popis, aby nedošlo k zobrazení nechtěných informací. Proměnná popis obsahuje html kód pro tabulku, která bude následně zobrazena v panelu Multivýběr.

```
function zpracovatVybrane(vybranePrvky) {
      popis = "";
      popis = popis + "<table>";
```

Funkce projde v cyklu for pole prvků a pro každý z nich zavolá funkci pridatPopis(prvek), která přidá jejich popis k již vytvořené tabulce.

```
      for (var j = 0; j < vybranePrvky.length; j++) {
            pridatPopis(vybranePrvky[j]);
      }
```

Funkce následně uzavře tabulku a za pomoci funkce document.getElementById() získá element s id multiVyber, který uloží do proměnné popisDiv. V posledním kroku nastaví proměnnou popis jako vnitřní html kód pro element multiVyber.

```
      popis = popis + "</table>";
      var popisDiv = document.getElementById('multiVyber');
      popisDiv.innerHTML = popis;
}
```

V rámci funkce zpracovatVybrane() byla již zmíněna druhá funkce, kterou pro multivýběr aplikace využívá. Jedná se o jednoduchou funkci pridatPopis(prvek) přidávající do proměnné popis jeden řádek html tabulky obsahující popis prvku, který byl předán funkci jako parametr.

```
function pridatPopis(prvek) {
      popis = popis + "<tr><td>" + prvek.attributes.description +
      "</tr></td>";
}
```

4.1.3.2.5 Vytvoření popupu

V ukázkové aplikaci je kromě hromadného výběru možné prvky vybírat také kliknutím. Po tomto kliknutí je následně zobrazeno vyskakovací okno, neboli popup, které obsahuje informace o daném prvku. Toto zobrazení obstarává funkce vytvoritPopup(prvek), jejímž parametrem je prvek vybraný za pomoci události featureselected. Pro samotný popup je připravena stejnojmenná proměnná popup, do které je následně uložen.

```
var popup;
function vytvoritPopup(prvek) {
```

Funkce v první řadě zkontroluje, zda je původem prvku aktivní vrstva. V případě, že ano, je do proměnné popup vytvořená nová instance třídy GeoExt.Popup, která je založena na třídě Ext.Window. Tato třída nabízí pro vytvoření popupu několik parametrů, které ovlivní jeho vzhled a funkce.

```
    if (prvek.layer.name === aktivniVrstvaNazev) {
    popup = new GeoExt.Popup({
```

Parametr title nastavuje název popupu, který se zobrazí v jeho hlavičce. V ukázkové aplikaci je nastaven na jméno prvku, ze kterého se popup vytváří.

```
            title: prvek.attributes.name,
```

V parametru location určujeme, kde se popup zobrazí. Tento parametr je nastaven na samotný prvek, tudíž se popup zobrazí na místě, které je mu relevantní.

```
            location: prvek,
```

Parametr width určuje šířku vytvořeného popupu v pixelech. Je možné použít také parametr height, který nastavuje stejným způsobem také výšku. Ten však v aplikaci není použit.

```
            width: 250,
```

V rámci parametru html je nastaven samotný obsah popupu, který je nastaven na popis prvku z jeho atributů. Tento parametr, jak jeho název napovídá, přijímá také html elementy, které pak zobrazí v těle popupu.

```
            html: prvek.attributes.description,
```

Parametr maximizable určuje svými hodnotami true, či false možnost popup roztáhnout na celý panel, čili jej maximalizovat.

```
            maximizable: true,
```

Collapsible je taktéž parametr, který může nabývat hodnot true a false a určuje, zda je možné popup minimalizovat na pouhý stavový řádek.

```
            collapsible: true,
```

Posledním použitým parametrem je autoScroll, který, pokud je nastaven na true, umožní zobrazení scrolovacích lišt, pokud je popup zmenšen do formy, kdy je text příliš obsáhlý, aby se do něj vešel.

```
            autoScroll: true
        });
```

Popup je následně otevřen jeho metodou show a po jeho otevření je výběrový nástroj uvolněn metodou unselectAll(), která z něj odstraní aktuálně vybraný prvek tak, aby bylo možné znovu vybírat.

```
        popup.show();
        vyberovyNastroj.unselectAll();
        }
```

Pokud se vybraný prvek neshoduje s aktivní vrstvou, bude aplikace postupovat dle větve else, ve které je opět nutné výběrový nástroj uvolnit, aby bylo možné dále vybírat.

```
        else {
            vyberovyNastroj.unselectAll();
        }
}
```

Obr. 17 Popup vytvořený za pomoci třídy GeoExt.Popup

4.1.3.2.6 Funkce pro nastavení horní lišty

Grafické rozhraní ukázkové aplikace nabízí také horní lištu, ve které se nachází tlačítka pro jednotlivé nástroje. V rámci těchto tlačítek je nutné v některých případech zajistit vypnutí, či zapnutí dané funkce, převážně těch, které jsou naprogramovány mimo jednotlivé třídy GeoExt. Funkce nastaveniListy() je použita právě pro tento případ. V rámci každého tlačítka je přidáno její volání, tudíž je

možné kód spravovat na jediném místě a není nutné jej měnit u každého tlačítka zvlášť.

```
function nastaveniListy() {
```

V rámci ukázkové aplikace je ve funkci zajištěno pouze vypnutí kliknutí na mapu za účelem náhledu do katastru nemovitostí. Podmínka využije proměnné klikZapnuty, která má v sobě uloženou informaci, zda je kliknutí na mapu zapnuté. Pokud ano, tak deregistruje událost click na mapě pro funkci katastr. Proměnná klikZapnuty je poté nastavena na hodnotu false.

```
      if (klikZapnuty) {
            mapa.events.unregister("click", mapa, katastr);
            klikZapnuty = false;
      }
}
```

4.1.3.3 Vytvoření mapy a přidání ovládacích prvků

Stěžejním prvkem celé aplikace je samozřejmě zobrazená mapa. Pro její vytvoření je nutné deklarovat a inicializovat proměnnou, do které se mapa uloží. V rámci aplikace se jedná o proměnnou mapa.

```
var mapa;

mapa = new OpenLayers.Map({
      projection: new OpenLayers.Projection("EPSG:4326"),
      displayProjection: new OpenLayers.Projection("EPSG:900913"),
      controls: [new OpenLayers.Control.PanZoomBar(), new
      OpenLayers.Control.Navigation(), new
      OpenLayers.Control.Attribution()]
});
```

Vytvoření mapy obsahuje několik nastavení, které definují její vzhled. Parametr projection určuje souřadnicový systém mapy a parametr displayProjection umožňuje mít druhý souřadnicový systém, který lze použít pro zobrazení různých souřadnicových systémů v rámci aplikace, příklad: pokud máme mapové podklady v zobrazení, například mercatorově, ale chceme textově zobrazit souřadnice v jiném formátu, například WGS 84.

V parametru controls se nachází pole, které obsahuje jednotlivé ovládací prvky mapy. Do pole můžeme vložit jednotlivé ovládací prvky a to za použití proměnných, které si vytvoříme předem, či za pomoci operátoru new přímo v jednotlivých prvcích

pole. Pokud parametr controls vynecháme, bude mapa vytvořena s ovládacími prvky OpenLayers.Control.Zoom a OpenLayers.Control.Navigation, které zajistí pohyb mapou a ovládací prvek pro přiblížení a oddálení mapy. Pokud v parametru použijeme prázdné pole ([ ]), nebude mapa obsahovat žádné ovládací prvky.

Ovládací prvky lze do mapy přidat také mimo konstruktor mapy za pomoci funkce addControl, například ovládací kříž a přibližovací nástroj.

```
mapa.addControl(new OpenLayers.Control.PanZoomBar());
```

V rámci ukázkové aplikace jsou přidány tři ovládací prvky, a to konkrétně PanZoomBar, zajišťující přidání čtyř tlačítek pro posun mapou a nástroj přiblížení, Navigation, který přidá ovládání za pomoci myši, a Attribution, který je nutný pro zobrazení copyrightu v rámci OpenStreetMap.

Dalším ovládacím prvkem, který je však přidám mimo definici mapy, je ovládací prvek exportu. Ten využívá souboru ExportMap.js a slouží k exportu vektorových prvků jako obrázku. Ovládací prvek je vytvořen jako instance právě třídy OpenLayers.Control.ExportMap a uložen v proměnné nastrojExportMapa. Následně je přidán do mapy za pomoci funkce addControl.

```
nastrojExportMapa = new OpenLayers.Control.ExportMap();

mapa.addControl(nastrojExportMapa);
```

4.1.3.4 Vytvoření mapového panelu

V rámci grafického rozhraní Ext JS je nutné za pomoci GeoExt vytvořit pro mapu mapový panel, zobrazující mapu a ovládací prvky, které jí byly předány v rámci jejího vytvoření. Tento mapový panel je vytvořen za pomoci třídy GeoExt.MapPanel a v rámci ukázkové aplikace je uložen v proměnné mapaPanel.

```
var mapaPanel;

var stred = new OpenLayers.LonLat(14.2599867, 48.8116842);

stred.transform(new OpenLayers.Projection("EPSG:4326"), new
OpenLayers.Projection("EPSG:900913"));
```

```
mapaPanel = new GeoExt.MapPanel({
      region: "center",
      map: mapa,
      center: stred,
      zoom: 15,
      layers: [
            new OpenLayers.Layer.OSM("OpenStreetMap (Mapnik)")
      ],
      tbar: horniLista
});
```

Mapový panel má několik parametrů, které byly v aplikaci použity. Prvním z nich je parametr region, který určí, kde se má panel vykreslit v rámci prohlížeče. Standardním umístěním je center, které panel umístí na střed. Je možné jej také vykreslit do jiných pozic, které jsou označeny anglickými názvy světových stran, north pro sever, west pro západ, south pro jih a east pro východ. Pokud však pro mapový panel použijeme jiný sektor, než center, je nutné do něj implementovat jiný panel, jelikož sektor center je povinný.

Dalším parametrem je map, který, jak již název napovídá, určuje, jaká mapa se má v panelu vykreslit. Parametru můžeme předat již vytvořenou mapu ve formě proměnné, či ji v jeho rámci vytvořit za pomoci konstruktoru. V rámci ukázkové aplikace předáváme proměnnou mapa.

Parametr center určí, kde má být nastaven střed mapy. V ukázkové aplikaci je střed nastaven za pomoci proměnné stred, která je instancí třídy OpenLayers.LonLat(zeměpisná šířka, zeměpisná délka). Následně je tento střed transformován do zobrazovací projekce mapy, v případě ukázky do souřadnicového systému EPSG:900913. Poté je střed předán mapovému panelu, který dle něj nastaví mapu.

V parametru zoom nastavíme mapě počáteční přiblížení. Toto přiblížení je počítáno od nuly, až po limit zobrazené mapy, přičemž platí, že čím větší hodnota, tím větší přiblížení mapy.

V rámci parametru layers lze nastavit pole vrstev, které bude mapa obsahovat. Tyto vrstvy můžeme definovat za použití konstruktoru, jako u vrstvy OpenStreetMap uvedené v příkladu, či předáním proměnné, ve které je vrstva uložena. Tyto vrstvy

jsou odděleny od vrstev, které jsou uloženy v rámci mapy, tudíž mohou být například vyňaty z registrace výběrového nástroje.

Posledním parametrem mapového panelu je tbar, který obsahuje vytvořenou horní nástrojovou lištu. Pokud chceme lišty ve spodní části panelu, je nutné parametr tbar změnit na parametr bbar, který je funkčně stejný, ale lištu vykreslí právě v dolní části panelu.

4.1.3.5 Přidání vrstev mapy

Pokud chceme, aby bylo v rámci aplikace možné zobrazit více druhů mapových dat, je nutné vytvořit pro tato data vrstvy a ty následně do aplikace přidat. První způsobem, jak vrstvy vložit je již zmíněné pole v rámci parametru layers mapového panelu. Pokud však chceme využít výběrového nástroje, je nutné vrstvy přidat přímo mapové proměnné za pomoci funkcí mapa.addLayer(vrstva), či mapa.addLayers([vrstva1, vrstva2]). Ačkoliv OpenLayers umožní přidání vrstvy bez proměnné, kdy vložíme jako parametr funkce přímo vytvoření vrstvy za pomoci operátoru new, je však výhodnější, aby vrstvy byly uloženy v proměnných, které budou poté uvedeny, jako parametry funkce addLayer.

4.1.3.5.1 OpenStreetMap vrstva

Ukázková aplikace obsahuje několik druhů vrstev. První z nich je již zmíněná vrstva OpenStreetMap. Jedná se o vrstvu, která do mapy přidá mapové podklady právě z tohoto projektu. OpenStreet map nabízí volné mapové podklady v rámci licence CC-BY-SA, čili Creative Commons Uveďte autora-Zachovejte licenci 2.0.[14] Při použití OpenStreetMap vrstvy je navíc nutné zobrazit odkaz, který odkazuje na licenci projektu.[15] Tento odkaz je v rámci OpenLayers vkládán automaticky, pokud je do mapy přidán ovládací prvek OpenLayers.Control.Attribution. Copyright OpenStreetMap se poté zobrazí v pravém dolním rohu. Samotné vytvoření vrstvy se odehrává v rámci třídy OSM. Jako parametr je použit název, pod kterým se bude třída zobrazovat v přepínači. Pokud nebude název uveden, bude použit defaultní OpenStreetMap. OSM vrstvu lze inicializovat za pomoci proměnné, či přímo použitím operátoru new.

[14]Uveďte autora-Zachovejte licenci 2.0 Generic: CC BY-SA 2.0. *Creative Commons* [online]. [2013] [cit. 2014-03-21]. Dostupné z: http://creativecommons.org/licenses/by-sa/2.0/deed.cs

[15] Autorská práva a licence. *OpenStreetMap* [online]. [2013] [cit. 2014-03-21]. Dostupné z: http://www.openstreetmap.org/copyright

```
var osm = new OpenLayers.Layer.OSM("OpenStreetMap (Mapnik)");

new OpenLayers.Layer.OSM("OpenStreetMap (Mapnik)")
```

4.1.3.5.2 WMS vrstva

Dalším typem jsou WMS vrstvy, které v aplikaci zajišťují napojení na vrstvy ČÚZK, konkrétně na ortofoto a katastr nemovitostí. Pro zobrazení WMS vrstvy obsahuje OpenLayers třídu OpenLayers.Layer.WMS.

```
new OpenLayers.Layer.WMS("Katastr nemovitostí (ČUZK)",
      "http://wms.cuzk.cz/wms.asp?service=WMS&",
      {
            layers: "KN",
            transparent: true,
            format: 'image/png'
      },
      {
            isBaseLayer: false,
            visibility: false,
            projection: mapa.displayProjection,
            filtr: "Hranice"
      })
```

Prvním parametrem při tvorbě WMS vrstvy je její název, který se zobrazí v přepínači vrstev. Jako druhý parametr je použita základní webová adresa, která odkazuje na WMS server. Ve třetím parametru se uvádějí doplňující požadavky na dotaz GetMap. V příkladu katastru nemovitostí se jedná vyžádání vrstvy KN za pomoci parametru layers, požadavek na průhlednost vrstvy parametrem transparent a její formát v parametru format. OpenLayers poté z těchto doplňujících parametrů a základního url serveru vytvoří WMS dotaz, který navrátí mapové podklady pro zobrazení.

Posledním parametrem jsou doplňující nastavení, která určí, jak se bude vrstva zobrazovat v rámci OpenLayers. Doplňující nastavení jsou samostatný objekt, který se skládá ze svých parametrů, prvním z nich je isBaseLayer. Ten určí, zda se jedná o základní, či překryvnou vrstvu. OpenLayers dělí vrstvy právě na tyto dva typy, přičemž základní vrstvu, kdy je parametr nastaven na hodnotu true, lze zobrazit vždy pouze jednu a je v rámci přepínače vykreslena s tlačítkem v podobě radio buttonu. Pokud bude vrstva nastavena jako překryvná, parametr nabývá hodnoty false, lze ji zobrazit společně se základní a dalšími překryvnými vrstvami. V rámci přepínače se

překryvná vrstva zobrazí společně s checkboxem, který ji umožní vypnout a zapnout a v případě ukázkové aplikace také s radio buttonem aktivní vrstvy. Druhým parametrem je visibility, který umožní nastavit, zda je vrstva viditelná ihned po načtení aplikace, nebo bude skryta a uživatel si ji zapne až po kliknutí na příslušné tlačítko. Parametr může nabývat hodnot true, kdy bude vrstva viditelná po načtení a false, kdy bude skryta. Další parametr projection určuje, ve kterém souřadnicovém systému bude vrstva zobrazena. Hodnota tohoto parametru závisí na souřadnicovém systému zdrojových dat. V rámci ukázky je vrstvě nastavena stejná projekce, jakou má mapa ve svém parametru displayProjection. Pokud by byl tento parametr chybně nastaven, vrstva by se nezobrazila. Posledním uvedeným parametrem v rámci doplňujícího nastavení je již zmíněný filtr. Jeho hodnota nabývá názvu skupiny, který je poté využit v rámci načtení skupiny pro zobrazení v přepínači vrstev. U WMS vrstvy lze použít také parametr legendaCesta, pokud by bylo pro vrstvu nutné zobrazit obrázkovou legendu. Parametr nabývá řetězcové hodnoty, která uvádí cestu k danému souboru s legendou.

4.1.3.5.3 Vektorová vrstva

Posledním typem vrstvy, která se v rámci ukázkové aplikace vyskytuje, je vektorová vrstva zobrazující data z kml souborů. V rámci OpenLayers se vektorová data zobrazují za pomoci třídy OpenLayers.Layer.Vector.

```
var ZFP = new OpenLayers.Layer.Vector("Funkční plochy",
{
      strategies: [new OpenLayers.Strategy.Fixed()],
      projection: mapa.projection,
      isBaseLayer: false,
      filtr: "Stavby",
      legendaCesta: "images/legendZFP.png",
      visibility: false,
      protocol: new OpenLayers.Protocol.HTTP({
            url: "./mapy/ZFP/ZFP.kml",
            format: format
      }),
      renderers: ["Canvas", "SVG", "VML"]
}
);
```

Některé parametry vektorové vrstvy jsou identické s parametry WMS třídy, jmenovitě se jedná o visibility, projection, isBaseLayer a filtr, které fungují identicky i zde. Oproti ukázkové WMS vrstvě je u této vektorové navíc použit parametr legendaCesta, který obsahuje cestu k souboru s legendou.

Parametru protocol, který u vektorové vrstvy zajišťuje spojení se souborem, je předána nová instance třídy OpenLayers.Protocol.HTTP. Tato třída má při svém vytvoření použity dva další parametry a to url, který udává adresu k souboru, jenž se má zobrazit a parametr format, kterému je předána proměnná format. Ta poté určí, jaký formát se má na daný soubor aplikovat.

Dalším novým parametrem vektorové vrstvy je strategies. Tento parametr obsahuje pole strategií, které určí, jakým způsobem se má soubor zpracovat. Použitá strategie je instancí třídy OpenLayers.Strategy.Fixed, jenž umožňuje načtení celého souboru najednou a dále v jeho použití nenačítá žádná nová data. Pro vektorové vrstvy existuje ještě strategie BBOX, která načítá postupně prvky, které jsou momentálně zobrazeny. Jenže tato strategie nefunguje tak, jak by měla a pokaždé načítá celý soubor. V rámci strategie fixed je tedy soubor načten pouze jednou a proto je její použití výhodnější.

Posledním odlišným parametrem je renderers. Tento parametr je použit na základě již zmíněné úpravy pro export mapy a určuje způsob, jakým se vektorové prvky vykreslí. V rámci parametru renderers je použito pole tří hodnot, konkrétně Canvas, SVG a VML. Tyto hodnoty je nutné použít pro export prvků a jejich plynulé zobrazení. Parametr renderers je potřeba použít u každé vektorové vrstvy.

4.1.3.6 Vytvoření výběrových nástrojů

Výběrové nástroje zajišťují výběr prvků z mapy. V rámci ukázkové aplikace jsou použity dva, odlišný pro výběr kliknutím a pro výběr pomoci tzv. boxu. Nástroj pro výběr kliknutím se nachází v proměnné vyberovyNastroj a pro výběr tažením boxu se nachází v proměnné vyberovyNastrojBox a oba dva jsou instancemi třídy OpenLayers.Control.SelectFeature.

```
vyberovyNastroj = new OpenLayers.Control.SelectFeature(
      [ZFP, NAT, USES],
      {
            clickout: true,
            box: false
      }
);
```

Na příkladu proměnné vyberovyNastroj bude vysvětleno fungování obou nástrojů. V rámci konstruktoru v první řadě předáváme pole vrstev, které mají mít možnost, aby z nich bylo vybíráno. Druhým parametrem konstruktoru je objekt doplňujících nastavení, které určí, jak se má nástroj chovat, a ten se skládá ze dvou doplňujících parametrů. Prvním z nich je clickout, který, pokud je nastaven na hodnotu true, umožní, aby byl výběr zrušen. Pokud uživatel klikne mimo prvek, bude jeho vybrání zrušeno. Druhý parametr box svou hodnotou true, či false ovlivňuje právě tu vlastnost, kdy výběrový nástroj funguje jako kliknutí, či jako box. Právě tímto parametrem se odlišují proměnné vyberovyNastroj a vyberovyNastrojBox, kdy vyberovyNastrojBox má parametr box nastaven na hodnotu true.

Mimo samotné výběrové nástroje je nutné jednotlivým vrstvám také přidat událost featureselected, která zajistí spuštění funkce v případě, že je prvek vybrán. Tato registrace probíhá v cyklu for, který pro každou vrstvu za pomoci funkce vrstva.events.on() registruje událost featureselected, jejíž hodnotou je funkce, kterou událost po vybrání prvku, spustí. V rámci ukázkové aplikace tato funkce zkontroluje, zda je výběrový nástroj pro kliknutí aktivní a pokud ano, zavolá funkci pro vytvoření popupu. Následně pro jistotu uvolní oba výběrové nástroje.

```
for (var i = 0; i < mapa.layers.length; i++) {
      mapa.layers[i].events.on({
            featureselected: function(e) {
                  if (vyberovyNastroj.active) {
                        vytvoritPopup(e.feature);
                  }
                  vyberovyNastroj.unselectAll();
                  vyberovyNastrojBox.unselectAll();
            }
      });
}
```

Jak již bylo dříve zmíněno, funkce, která zpracovává výběr pomocí boxu, se jmenuje zpracovatVybrane(vybranePrvky) a je volána v rámci prototypu funkce selectBox.

Obr. 18 Výběr prvků za pomoci parametru box nastaveného na hodnotu true

4.1.3.7 Nástroje měření

Nástroje měření se v rámci ukázkové aplikace vyskytují dva, měření vzdálenosti a měření plochy. Oba nástroje jsou si velice podobné a oba jsou instancí třídy OpenLayers.Control.Measure. Rozdíly jsou pouze v prvním parametru, kdy je nástrojům přidán různý handler, který odlišuje, zda se jedná o vzdálenost (OpenLayers.Handler.Path), či plochu (OpenLayers.Handler.Polygon), a ve způsobu zpracování dat v rámci událostí. Ostatní parametry mají stejnou funkčnost. Příklad bude vysvětlen na variantě měření vzdálenosti.

```
var vzdalenost = new
OpenLayers.Control.Measure(OpenLayers.Handler.Path, {
```

Parametr displayClass určuje CSS třídu, která vizuálně formátuje nástroj. V ukázkové aplikaci byla využita třída, která je v Openlayers obsažena.

```
    displayClass: 'olControlMeasureDistance',
```

Následují parametry, které určují chování měřícího nástroje. Parametr persist nastavený na hodnotu true určí, že má nákres měřené trasy (oblasti) zůstat na mapě viditelný i po ukončení měření. Bude z ní vymazán až po začátku nového měření, či vypnutí nástroje. Parametr immediate povolí okamžité měření, které se za pomoci události measurepartial bude zobrazovat okamžitě, jakmile uživatel pohne kurzorem

myši. Další parametr geodesic určí, zda se má použít rovinné (hodnota false), či geodetické (hodnota true) měření. Pro reálnou hodnotu je nutné v rámci aplikace použít právě geodetické měření.

```
persist: true,
immediate: true,
geodesic: true,
```

Na závěr je nutné určit obsah parametru eventListeners, čili dvě události, které zajistí zobrazení výsledků v rámci příslušných panelů. První z nich je measure, která je použita v případě, že se nejedná o okamžité měření. Druhá, measurepartial, je využita v případě, že se o okamžité měření jedná. Parametrem událostí je proměnná evt, která obsahuje dané naměřené hodnoty v proměnné evt.measure a jednotky měření v proměnné evt.units. Výraz, ve kterém je proměnná evt.measure umístěna zajistí zaokrouhlení naměřené hodnoty na tři desetinná místa. Aplikace poté nalezne html element s id mereni_vzdalenost a jeho obsahem bude poté řetězec, který byl nastaven do proměnné innerHTML.

```
eventListeners: {
        measure: function(evt) {
                var mereni_vzdalenost =
                document.getElementById('mereni_vzdalenost');
                mereni_vzdalenost.innerHTML = "Naměřená
                vzdálenost: " + "<br />" +
                (Math.round((evt.measure)*1000)/1000) + " " +
                evt.units;
        },
        measurepartial: function(evt) {
                var mereni_vzdalenost =
                document.getElementById('mereni_vzdalenost');
                mereni_vzdalenost.innerHTML = "Naměřená
                vzdálenost: " + "<br />" +
                (Math.round((evt.measure)*1000)/1000) + " " +
                evt.units;
        }
}
});
```

Měření plochy je v zásadě identické. Rozdílem je již zmíněné použití jiného handleru pro měření (OpenLayers.Handler.Polygon) a nalezení jiného html elementu v rámci jednotlivých událostí s id mereni_plocha. Samozřejmostí je také jiný text zobrazené zprávy, ve které musí být za jednotky měření přidán horní index pro jednotky plochy.

```
mereni_plocha.innerHTML = "Naměřená plocha: " + "<br />" +
evt.measure + " " + evt.units + "<sup>2</sup>";
```

4.1.3.8 Nástrojová lišta

Nástrojová lišta sdružuje veškeré nástroje, které se v ukázkové aplikaci nachází. Samotná lišta se nachází v proměnné horniLista, která je instancí třídy Ext.Toolbar a obsahuje dva parametry.

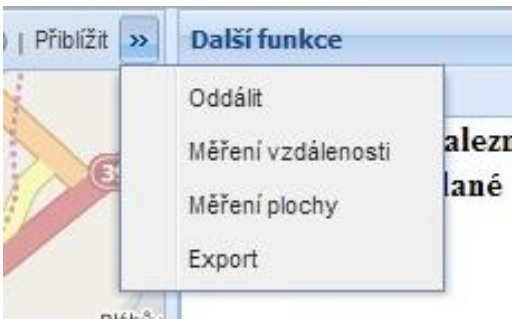
```
horniLista = new Ext.Toolbar({
```

Parametr items obsahuje pole, ve kterém jsou jednotlivé prvky umístěny. V rámci ukázkové aplikace se toto pole nalézá v proměnné nastrojeListy, která je poté parametru items předána.

```
    items: nastrojeListy,
```

Pokud je lišta větší než panel, ve kterém je umístěna, umožní vlastnost enableOverflow, nastavená na hodnotu true, skrytí její části, která přesahuje a zobrazení této části formou rolovacího menu.

```
    enableOverflow: true
});
```



Obr. 19 Část lišty, jež se nevešla do panelu, zobrazena za pomoci parametru enableOverflow nastaveného na hodnotu true

4.1.3.9 Funkce nástrojové lišty

Funkce nástrojové lišty se v rámci ukázkové aplikace nachází v proměnné nastrojeListy. Tato proměnná je standardní pole a nástroje se do ní vkládají za pomoci funkce nastrojeListy.push(nastroj). Jednotlivé nástroje jsou většinou

instancemi třídy GeoExt.Action, až na nástroje katastr a export, které jsou instancemi třídy Ext.Button.

Jednotlivá tlačítka, která jsou instancí třídy GeoExt.Action, využívají v rámci nástrojové lišty funkcí activate a deactivate, které jednotlivé nástroje nabízí. Tyto funkce zajistí, aby byl daný nástroj zapnut na kliknutí tlačítka a vypnut na jeho přepnutí do neaktivního stavu. Akci pro tlačítko třídy Ext.Button je nutné nastavit v rámci parametru handler.

Nástroj třídy GeoExt.Action bude vysvětlen na příkladu výběrového nástroje, který z mapy vybírá za pomoci kliknutí.

```
vyberNastrojAction = new GeoExt.Action({
```

Parametr text obsahuje řetězec, který se má zobrazit jako text tlačítka. Pokud nechceme využít textového tlačítka, lze parametr text vyměnit za parametr icon, který bude obsahovat cestu k ikoně, kterou chceme použít jako obrázek tlačítka.

```
    text: "Výběr kliknutím",
```

V parametru control je určeno, jaký nástroj se má po kliknutí na tlačítko spustit. V tomto případě se jedná o výběrový nástroj uložený v proměnné vyberovyNastroj.

```
    control: vyberovyNastroj,
```

Parametr map určuje, které mapové proměnné se nástroj týká.

```
    map: mapa,
```

Parametry toogleGroup a group určují skupinu tlačítek, které jsou mezi sebou provázány. Vždy jedno z tlačítek skupiny může být aktivní, takže parametry zajistí, aby se vždy tlačítko, které je zapnuté vypnulo, pokud klikneme na jiné ze skupiny. Hodnoty parametrů musí být identické pro všechna tlačítka ve skupině.

```
    toggleGroup: "ovladani",
    group: "ovladani",
```

Pokud je parametr allowDepress nastaven na hodnotu false, není možné tlačítko vypnout bez aktivování jiného tlačítka ze skupiny. Nastavení na hodnotu false tedy neumožní situaci, kdy budou všechna tlačítka vypnuta.

```
    allowDepress: false,
```

Parametry tooltip a tooltipType nastavují popisek tlačítku, který se zobrazí, pokud na něj najede uživatel myší. Pro tuto funkci je nutné nastavit tooltipType na hodnotu "title", pokud by nebyl na tuto hodnotu nastaven, byl by tooltip instancí třídy qtip, který se po najetí myší nezobrazí.

```
tooltip: "Aktivuje nástroj pro výběr z mapy kliknutím",
tooltipType: "title",
```

V parametru handler se nachází funkce, která je spuštěna po kliknutí na tlačítko. V příkladu se jedná o funkci nastaveniListy(), která již byla dříve zmíněna.

```
handler: function() {
      nastaveniListy();
      }
});
```

Po vytvoření nástroje je nutné jej přidat do pole nástrojů za pomoci funkce push. Následuje přidání oddělovače, který v nástrojové liště zobrazí svislou čáru mezi nástroji a zvýší tak její přehlednost.

```
nastrojeListy.push(vyberNastrojAction);
nastrojeListy.push("-");
```

Vytvoření nástrojů, které jsou instancemi třídy Ext.Button, je velice podobné. Rozdílem u nástroje katastr je nepoužití parametru control, jelikož není nástroj, který by tomuto tlačítku posloužil. Druhým rozdílem je zavolání jiné funkce v parametru handler, a to již zmíněné funkce zapnoutKlikKatastr().

```
handler: function() {
      zapnoutKlikKatastr();
}
```

Nástroj export je oproti ostatním chudší o parametry control, allowDepress, toogleGroup a group, a to z toho důvodu, že se jedná o pouhé klikací tlačítko a není tedy nutné jej zařazovat do skupiny, nebo nastavovat jeho chování v případě jeho deaktivace. V rámci parametru handler je použita funkce exportMapy().

```
var exportTlacitko = new Ext.Button({
      text: "Export",
      tooltip: "Exportuje prvky ma mapě ve formě jednoduché
      stránky",
      tooltipType: 'title',
      handler: function() {
            exportMapy();
      }
});
```

4.1.3.10 Grafické rozhraní

Grafické rozhraní je v ukázkové aplikaci zařízeno převážně prvky z knihovny ExtJs kombinované s prvky z knihovny GeoExt. Rozhraní se dělí na tzv. panely, které jsou kombinovány do instance třídy Ext.Viewport.

Prvním panelem grafického rozhraní je již zmíněný mapový panel, který se nachází v proměnné mapaPanel. Dalším panelem je panel vrstev, který je instancí třídy Ext.tree.TreePanel. Panel vrstev v ukázkové aplikaci obsahuje a rozšiřuje části obsažené v tutoriálu na webových stránkách GeoExt[16].

Pro potřeby panelu vrstev je nejdříve nutné deklarovat proměnnou LayerNodeUI, která je potřebná pro rozšíření panelu vrstev o dodatečné radio buttony pro přepínání aktivní vrstvy na základě zmíněného tutoriálu. Proměnná musí být deklarována za použití funkce Ext.extend(), jejíž dva parametry jsou GeoExt.tree.LayerNodeUI a nová instance třídy GeoExt.tree.TreeNodeUIEventMixin.

```
var LayerNodeUI = Ext.extend(GeoExt.tree.LayerNodeUI, new
GeoExt.tree.TreeNodeUIEventMixin());
```

Samotný panel vrstev je v rámci ukázkové aplikace uložen v proměnné vrstvyPanel a má formu stromového zobrazení.

```
var vrstvyPanel = new Ext.tree.TreePanel({
```

První parametry, které jsou v rámci vytvoření panelu použity, určují jeho vzhled a pozici. Parametr title obsahuje nadpis, který se zobrazí v hlavičce panelu. Parametr region určí, kde se v rámci aplikace panel zobrazí. Nabývat může hodnot názvů světových stran v anglickém jazyce, doplněných o hodnotu center pro střed.

[16] Layer Tree Tutorial. *GeoExt: JavaScript Toolkit for Rich Web Mapping Applications* [online]. 2010 [cit. 2014-03-30]. Dostupné z: http://geoext.org/tutorials/layertree-tutorial.html

V ukázkové aplikaci se panel nachází vlevo, čili na západě (west). Parametr split určí, zda je možné panelu měnit šířku za pomoci tažením jeho hrany.

```
title: "Vrstvy",
region: "west",
split: true,
```

Parametr lines umožňuje zobrazit čáry, které spojují jednotlivé uzle stromu. V ukázkové aplikaci jsou tyto čáry vypnuty. Parametrem width lze nastavit počáteční šířku panelu v pixelech. Parametrem rootVisible se nastavuje, zda je kořenový uzel stromu v panelu viditelný, či nikoliv. Za pomoci parametru collapseMode lze nastavit, zda má mít panel schopnost minimalizace. Hodnota mini panel minimalizuje na vzhled úzké hrany se zobrazeným tlačítkem pro opětovné zobrazení panelu. Poslední hodnotou, která ovlivňuje vzhled panelu je autoScroll, který, nastavený na hodnotu true, umožní automatické zobrazení rolovacích lišt, pokud je obsah větší, než aktuální rozměr panelu.

```
lines: false,
width: 250,
rootVisible: true,
collapseMode: "mini",
autoScroll: true,
```

Obsahem parametru plugins je pole pluginů, které jsou do panelu vrstev přidány. V ukázkové aplikaci se jedná o plugin radiobuttonů, který je přidán za pomoci nové instance třídy GeoExt.plugins.TreeNodeRadioButton, jejíž parametr listeners obsahuje registraci události radiochange, která je spuštěna při kliknutí na libovolný radio button v panelu. Funkce obsažená v rámci této události obsahuje parametr uzel, který obsahuje referenci na uzel, jenž událost spustil. Funkce v první řadě nastaví proměnné pro aktivní vrstvu na hodnotu uzel.layer a uzel.layer.name a poté nastaví výběrovému nástroji vrstvu, ze které má vybírat za pomocí funkce setLayer. Tato funkce nastaví index vrstvy na nejvyšší hodnotu a ta se následně vykreslí nad všemi ostatními a nástroj k ní může následně přistoupit. Pro tuto skutečnost je nutné modifikovat funkci onChangeLayer, která se nachází v souboru GeoExt/lib/data/LayerStore.js. V základní funkci má přesun indexu vrstvy za následek její přesun v rámci stromu. Aby se tedy vrstvy neustále neměnily, je vyřazen z činnosti řádek s příkazem this.remove(record); a následně

this.insert(layerIndex, [record]);. Touto změnou je dosaženo požadované funkčnosti a vrstvy zůstanou v rámci přepínače na svém místě. Funkci setLayer je předána vrstva, kterou obsahuje vybraný uzel, tedy proměnná uzel.layer. Jako poslední je poté volána zmíněná funkce legenda(), která nastaví pro aktivní vrstvu panel Legenda.

```
plugins: [
      new GeoExt.plugins.TreeNodeRadioButton({
            listeners: {
                  "radiochange": function(uzel) {
                        aktivniVrstvaNazev = uzel.layer.name;
                        aktivniVrstvaLayer = uzel.layer;
                        vyberovyNastroj.setLayer(uzel.layer);
                        legenda();
            }
      }})
],
```

V dalším parametru loader se nachází instance třídy Ext.tree.TreeLoader, která je využita pro určení, zda má strom vrstev v tomto panelu používat rozšíření uložené v proměnné LayerNodeUI. Parametr applyLoader musí být nastavený na hodnotu false, aby zabránil zasahování tohoto loaderu do podřízených uzlů a narušení jejich vlastnímu načtení. Parametr "layernodeui" je řetězec, který spojí uzel právě s proměnnou LayerNodeUI a uvádí se u každého uzlu, který má mít vykreslené radiobuttony pro aktivní vrstvu.

```
loader: new Ext.tree.TreeLoader({
      applyLoader: false,
      uiProviders: {
            "layernodeui": LayerNodeUI
      }
}),
```

V parametru root se nachází nastavení kořenového uzle stromu. První vlastností kořene je jeho název, který se nachází v parametru text. Hodnota parametru expanded určí, zda má být kořen při načtení aplikace otevřen, či nikoliv a parametr children obsahuje odkaz na pole, ve kterém se nachází uzle, které jsou na kořen navázány. Toto pole je v ukázkové aplikaci uloženo v proměnné strom.

```
        root: {
                text: "Kájov",
                expanded: true,
                children: strom
        }
});
```

Proměnná strom je již zmíněné pole, ve kterém se nachází jednotlivé uzle stromu. Tyto uzle jsou v rámci pole reprezentovány vždy několika parametry. V rámci ukázky proměnné byly použity dva druhy uzlů, které se v aplikaci používají. Prvním z nich je uzel s názvem Základní vrstvy. Typ tohoto uzlu je v parametru nodeType nastaven na gx_baselayercontainer, tudíž uzel schraňuje všechny základní vrstvy, které se v aplikaci vyskytují. Obdobným typem je gx_overlaylayercontainer, který obsahuje naopak všechny překryvné vrstvy. Ten však není v aplikaci použit. Parametr expanded určuje, zda má být uzel rozbalen, či zabalen. Parametr text obsahuje již zmíněný název uzlu.

```
var strom = [
{
        text: "Základní vrstvy",
        expanded: true,
        nodeType: "gx_baselayercontainer"
},
```

Druhým typem uzle, který v aplikaci najdeme, je uzel, který obsahuje překryvné vrstvy. Jeho typ je však nastaven na gx_layercontainer, tudíž uzel obsahující všechny vrstvy. V rámci parametru loader a jeho parametru filter je však použita funkce na filtrování vrstev obsažená v již zmíněném tutoriálu, která umožní na základě prototypu filtr, který byl definován na začátku aplikace, vytvářet uzly, které odpovídají jednotlivým tematickým skupinám jednotlivých vrstev. Stěžejní částí této funkce je pro daný uzel parametr u funkce indexOf, který musí být shodný s textem v parametru filtr u dané vrstvy.

```
{
      text: "Stavby",
      nodeType: "gx_layercontainer",
      loader: {
            filter: function(record) {
                  if (record.get("layer").filtr !== null) {
                        return
                        record.get("layer").filtr.indexOf("Stavby")
                        !== -1;
                  }
            },
```

Druhým parametrem v rámci loaderu je baseAttrs, který prvkům daného uzlu přidá požadované radio buttony pro zpracování aktivní vrstvy. U parametru uiProvider je nutné použít stejný řetězec, který figuruje v parametru uiProviders uvedeného v rámci parametru loader u samotného panelu. Parametr radioGroup určuje název skupiny, v rámci které budou dané radio buttony spojeny. Vždy jeden radio button z dané skupiny může být aktivní. Aby byl parametr použit správně, musí být pro všechny uzly stejný.

```
            baseAttrs: {
                  radioGroup: "aktivni",
                  uiProvider: "layernodeui"
            }
      }
}]
```

V ukázkové aplikaci je obsaženo více uzlů, ty jsou již jen obdobami druhého typu a obsahují pouze změněné názvy a texty pro filtrování. Funkčně jsou identické.

Dalším panelem je pravý panel dodatečných funkcí, který je tvořen skupinou menších panelů sdružených ve formě tzv. accordion layoutu. Ten umožní zpracování velkého množství panelu na malém prostoru a efektivně tak využije prostor, který mu byl dán. Jednotlivé panely jsou instancemi třídy Ext.Panel a obsahují tři parametry. Prvním z nich je title, který obsahuje název daného panelu, jenž se zobrazí v jeho hlavičce. Následuje contentEl, určující id html elementu, který tvoří obsah panelu. Posledním z parametrů je autoScroll, který již byl zmíněn u panelu vrstev a umožňuje automatické zobrazení rolovacích lišt v případě potřeby na základě obsahu panelu.

```
var legenda_panel = new Ext.Panel({
      title: 'Legenda',
      contentEl: "legenda",
      autoScroll: true
});
```

Jednotlivé panely jsou poté sdruženy ve zmíněné proměnné pravy_panel, která je také instancí třídy Ext.Panel. Tento panel však obsahuje více parametrů, které určují, jak se bude v rámci aplikace zobrazovat. Některé parametry již byly zmíněny u levého panelu a i zde mají stejnou funkčnost. Například parametr title nastavuje název panelu, width jeho šířku, split schopnost změny jeho šířky a collapseMode možnost jeho minimalizace. Region panelu je nastaven na východ (east) a tedy doprava.

```
var pravy_panel = new Ext.Panel({
      title: "Další funkce",
      region: 'east',
      width: 250,
      split: true,
```

Nově použitým parametrem je layoutConfig, který nastaví panelu schopnost minimalizace kliknutím kamkoliv do hlavičky parametrem titleCollapse a animaci přechodu při přepínání jednotlivých panelů za pomoci parametru animate.

```
      layoutConfig: {
            titleCollapse: true,
            animate: true
      },
      collapseMode: "mini",
```

V rámci parametru layout je nastaven výsledný layout panelu. Ten je v ukázkové aplikaci nastaven na již zmíněný accordion, čili jakési "roletkové" zobrazení.

```
      layout: 'accordion',
```

Posledním a pravděpodobně nejdůležitějším parametrem je items, který obsahuje pole jednotlivých panelů, které pravý panel tvoří. Tyto panely byly vytvořeny za pomoci předchozí metody a liší se pouze ve svých názvech a použitých html elementech.

```
        items: [legenda_panel, mereni_panel, mereni_plocha,
        vyber_panel]
});
```

4.1.3.11 Zkompletování GUI

Posledním krokem při vytváření aplikace je zkompletování grafického rozhraní v rámci tzv. viewportu. Ten je uveden v závěru kódu a tvoří jej instance zmíněné třídy Ext.Viewport. Viewport není přiřazen do žádné proměnné a je vytvořen pouze za pomoci operátoru new. Jedná se prakticky o kontejner, který obsahuje veškeré prvky vytvořené v rámci aplikace. Automaticky se přizpůsobuje velikosti okna a překresluje také veškerý jeho obsah.

```
new Ext.Viewport({
```

V rámci parametru layout je nastaven layout fit pro celý kontejner. Tento layout umožní efektivní využití celé plochy prohlížeče každého rozlišení a vykreslení aplikace bez rolování stránky, či překrývání jednotlivých prvků. Parametr hideBorders umožní skrýt okraje viewportu.

```
        layout: "fit",
        hideBorders: true,
```

Poslední parametr items obsahuje dodatečná nastavení pro jednotlivé prvky viewportu. Stanovuje jim border layout, který poté napomáhá již zmíněnému rozmístění prvků podle světových stran.

```
        items: {
                layout: "border",
```

Parametr items, který je použit v rámci dodatečných nastavení, poté obsahuje pole tří základních panelů celé aplikace, a to mapového panelu, panelu vrstev a panelu dalších funkcí.

```
                items: [mapaPanel, vrstvyPanel, pravy_panel]
        }
});
```

4.1.3.12 Umístění aplikace na webový hosting

Na závěr je samozřejmě nutné vytvořenou aplikaci zpřístupnit na webových stránkách. Je ji nutné nahrát kompletní včetně všech adresářů použitých knihoven. Celková velikost aplikace je přibližně 135 MB včetně dat pro obec Kájov. Samotná aplikace bez dat má přibližně 120 MB a s touto velikostí je tedy nutné v rámci její publikace počítat. Posledním krokem, který je nutné po nahrání aplikace provést, je nastavit knihovnám a souborům aplikace oprávnění na hodnotu 777. Bez tohoto nastavení nebude aplikace fungovat.

5 Závěr

Při zpracování této bakalářské práce se ukázalo, že nejen drahé a náročné řešení může vytvořit kvalitně vypadající a fungující aplikaci pro zobrazení mapových dat. Při porovnání aplikací na základě kritérií se dokonce ukázalo, že Opensource řešení vyšlo lépe, než řešení velkých firem, jakými společnosti Google a Seznam jsou. Ovšem práce také ukázala, že Google Maps API a Mapy API jsou API, které mají trochu odlišné zaměření, než Openlayers a má domněnka o tom, že nebudou vhodná pro vytvoření plnohodnotné mapové aplikace pro potřeby malé obce, se potvrdila.

Při tvorbě aplikace, která byla součástí práce, se ukázalo, že nejdůležitější částí každého API a aplikace je dokumentace. Ačkoliv mělo Openlayers dokumentaci kvalitně zpracovanou, během programování se ukázalo, že spousta pokročilých funkcí přeci jen není tak kvalitně popsána, jak se na první pohled zdálo. Nedostatek dokumentace tak může mít za následek zdržení až několik desítek hodin, které vývojář může investovat do jiných problémů. Proto doufám, že popis aplikace, který byl v rámci práce sepsán, napomůže při úpravě aplikace pro další obce, či jejímu obohacení o další možné funkce. Problematiku nedostatečné dokumentace a celkové složitosti vývoje aplikace by potenciálně mohl vyřešit WYSIWYG, který by tvorbu aplikace mohl přiblížit také úplným laikům.

V rámci práce se překvapivě nepotvrdil předpoklad, že mapová data malé obce bude lepší umístit na mapový server. Ačkoliv mapový server vykázal lepší výkonnost při pomalém internetovém připojení, tak postupný úbytek těchto pomalých připojení a jejich celkovému zrychlování i v mobilním sektoru ukazuje, že je na takto malých datech výhodnější a jistě pohodlnější využít souborů KML. V rámci problematiky mapového serveru lze však nadále testovat, zda lepší hardware, či optimalizace řešení nevylepší server natolik, že se i pro tato malá data vyplatí.

V rámci této bakalářské práce jsem tedy splnil stanovené cíle, porovnal jsem jednotlivé API kritérii přímo pro danou problematiku, srovnal jsem mapový server a KML data přímo na autentických vzorcích obce Kájov a na závěr vytvořil aplikaci pro zobrazení těchto dat, kterou jsem také, jak již bylo zmíněno, popsal k budoucímu využití a úpravám.

6 Seznam použitých zdrojů

[1] FU, Pinde, SUN, Jiulin. Web GIS: Principles and Applications. ESRI Press, Redlands, 2011. ISBN 978-1-58948-245-6

[2] FLANAGAN, David. JavaScript: kapesní příručka. Gliwice: Helion, 2004. ISBN 83-7361-466-4.

[3] MITCHELL, Tyler. Web mapping illustrated: [using open source GIS toolkits]. 1st ed. Beijing [u.a.]: O'Reilly, 2005. ISBN 978-059-6008-659.

[4] GOOGLE. Google Maps API: Google Developers [online]. 2012 [cit. 2013-03-06]. Dostupné z: https://developers.google.com/maps/

[5] SEZNAM.CZ. API Mapy.cz [online]. 2013 [cit. 2013-03-06]. Dostupné z: http://api4.mapy.cz/

[6] OPENLAYERS. What is OpenLayers? [online]. 2008 [cit. 2013-03-06]. Dostupné z: http://docs.openlayers.org/index.html#

[7] Welcome to MapServer. *Welcome to MapServer* [online]. 2014 [cit. 2014-03-20]. Dostupné z: http://www.mapserver.org/index.html

[8] Ext JS 3.4.0 - Sencha Docs. *HTML5 App Development for Desktop and Mobile. JavaScript Frameworks and Dev Tools from Sencha. | Home | Sencha* [online]. 2013 [cit. 2014-03-20]. Dostupné z: http://docs.sencha.com/extjs/3.4.0/

[9] Web Map Service. *Open Geospatial Consortium | OGC(R)* [online]. 2014 [cit. 2014-03-21]. Dostupné z: http://www.opengeospatial.org/standards/wms

[10] Metody stanovení vah kriterií. *Rozhodovací procesy | UPCE* [online]. 2011 [cit. 2014-03-21]. Dostupné z: http://www.rozhodovaciprocesy.cz/vicekriterialni-rozhodovani/2-1-metody-stanoveni-vah-kriterii.html

[11] OpenLayers: JavaScript Mapping Library. *OpenLayers* [online]. [2014] [cit. 2014-03-25]. Dostupné z: http://dev.openlayers.org/apidocs/files/OpenLayers-js.html

[12] GeoExt v1.1: JavaScript Toolkit for Rich Web Mapping Applications. *GeoExt v1.1* [online]. 2010 [cit. 2014-03-25]. Dostupné z: http://geoext.org/index.html

[13] OpenLayers Examples: JavaScript Toolkit for Rich Web Mapping Applications. *OpenLayers: Free Maps for the Web* [online]. [2014] [cit. 2014-03-25]. Dostupné z: http://openlayers.org/dev/examples/

[14] DOLEŽEL, Jan. *Datové formáty pro prezentaci map na webu*. Praha, 2005. Dostupné z: http://geo3.fsv.cvut.cz/~soukup/dip/dolezel/index.html. Diplomová práce. ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ v PRAZE.

[15] PEREZ, Antonio Santiago. *OpenLayers cookbook*. Birmingham: Packt Publishing, 2012, iii, 284 p. ISBN 978-1-84951-784-3.

[16] HAZZARD, Erik. *OpenLayers 2.10 beginner's guide: create, optimize, and deploy stunning cross-browser web maps with the OpenLayers JavaScript web-mapping library*. Birmingham, U.K.: Packt Open Source, 2011, xii, 351 p. ISBN 978-1-849514-12-5.

[17] Seriál Mapový server snadno a rychle. *Root.cz - informace nejen ze světa Linuxu* [online]. 2005 [cit. 2014-04-01]. Dostupné z: http://www.root.cz/serialy/mapovy-server-snadno-a-rychle/#ic=serial-box&icc=title

[18] Openlayers 2.6 (Example Transform and Labels). *Gis.ibbeck.de - Homepage der webbasierten GeoInformations Systemen des Ibbeck* [online]. [2014] [cit. 2014-04-01]. Dostupné z: http://gis.ibbeck.de/ginfo/apps/OLExamples/OL26/examples/labels_radiobtn.html

[19] Html5 - How to use Canvas element with Openlayers? - Geographic Information Systems Stack Exchange. In: *Geographic Information Systems Stack Exchange* [online]. 2013 [cit. 2014-04-02]. Dostupné z: http://gis.stackexchange.com/questions/71922/how-to-use-canvas-element-with-openlayers

[20] Future/OpenLayersWithCanvas – OpenLayers with Openlayers? - Geographic Information Systems Stack Exchange. In: *OpenLayers Wiki* [online]. [2014] [cit. 2014-04-02]. Dostupné z: http://trac.osgeo.org/openlayers/wiki/Future/OpenLayersWithCanvas

[21] OpenLayers and HTML5. In: *OpenLayers Wiki* [online]. [2014] [cit. 2014-04-05]. Dostupné z: http://trac.osgeo.org/openlayers/wiki/Future/OpenLayersAndHTML5

[22] SAUERWEIN. *Evaluation of HTML5 for its Use in the Web Mapping Client OpenLayers*. Kaiserslautern, 2010. Dostupné z: http://www.tobias-sauerwein.de/files/Tobias%20Sauerwein%20-%20Evaluation%20of%20HTML5%20for%20its%20Use%20in%20the%20Web%20Mapping%20Client%20OpenLayers.pdf. Bakalářská práce. Fachhochschule Kaiserslautern - University of Applied Sciences.

[23] Setting up MapServer WMS and use it with OpenLayers - Geographic Information Systems Stack Exchange. *Geographic Information Systems Stack Exchange* [online]. 2012 [cit. 2014-04-06]. Dostupné z: http://gis.stackexchange.com/questions/18364/setting-up-mapserver-wms-and-use-it-with-openlayers

[24] *Welcome to the QGIS project!* [online]. 2014 [cit. 2014-04-06]. Dostupné z: http://www.qgis.org/en/site/index.html

[25] *JavaScript Tutorial* [online]. [2014] [cit. 2014-04-06]. Dostupné z: http://www.w3schools.com/js/default.asp

[26] *Proj4js* [online]. [2014] [cit. 2014-04-06]. Dostupné z: http://trac.osgeo.org/proj4js/

[27] S-JTSK / Krovak East North - SJTSK - EPSG:5514. *EPSG.io: Coordinate Systems Worldwide* [online]. 2014 [cit. 2014-04-06]. Dostupné z: http://epsg.io/5514-1623

[28] Nahlížení do Katastru nemovitostí - nápověda. *Nahlížení do katastru nemovitostí* [online]. 2014 [cit. 2014-04-06]. Dostupné z: http://nahlizenidokn.cuzk.cz/Napoveda/index.htm

[29] Documentation - Version 2.0 | MapGuide Open Source. *MapGuide Project Home | MapGuide Open Source* [online]. [2014] [cit. 2014-04-06]. Dostupné z: http://mapguide.osgeo.org/2.0/documentation.html

[30] How do i fire a function after all features have been selected by a box in OpenLayers?. *Rqna Questions and Answers* [online]. 2012 [cit. 2014-04-06]. Dostupné z: How do i fire a function after all features have been selected by a box in OpenLayers?