UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Knihovna Google API

Michal Mikyska

Bakalářská práce
2013

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2012/2013**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Michal Mikyska**
Osobní číslo:	**I10139**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Knihovna Google API**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

Teoretická část bude zaměřena na zmapování možností použití prostředí Google API. Zejména na použití v komerčním prostředí. Nejdůležitější vybrané komponenty API budou popsány a navrženy možné způsoby jejich využití. U každé části budou doplněny příklady. Dále bude teoretická část zaměřena na možnosti licencování produktů, které využívají Google API a provedena rešerše v oblasti využití API v komerční sféře.
Výstupem z praktické části práce bude knihovna, která bude obsahovat řešené příklady použití komponent, které budou popsány v teoretické části práce. Cílem knihovny bude usnadnění přístupu k hotovým řešením pro vývojáře.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

SVENNERBERG, Gabriel. Beginning Google Maps API 3. New York, NY: Apress, c2010, xvi, 310 p. Expert's voice in Web development. ISBN 978-143-0228-028.
CHOW, Shu-Wai. Programujeme Mashup aplikace pro Web 2.0 v PHP. Vyd. 1. Brno: Computer Press, 2008, 280 s. ISBN 978-80-251-2057-6.
https://developers.google.com/

Vedoucí bakalářské práce: **Ing. Jiří Zechmeister**
Katedra informačních technologií

Datum zadání bakalářské práce: **21. prosince 2012**
Termín odevzdání bakalářské práce: **10. května 2013**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 29. března 2013

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 7. 5. 2013 Michal Mikyska

Poděkování

Rád bych poděkoval jak vedoucímu mé bakalářské práce Ing. Jiřímu Zechmeisterovi, tak mým přátelům a kolegům, kteří mi vždy byli nápomocni v průběhu zpracování této práce. Jejich tolerance, ochota a rady byly vždy impulsem k úspěšnému dokončení této bakalářské práce. Děkuji také mé rodině, která mě po celou dobu studia neustále podporovala.

Anotace

Tato práce se zabývá možnostmi využití aplikačního rozhraní u služeb společnosti Google. Cílem je seznámit čtenáře s licencemi a s komerčním využitím aplikačního rozhraní u nejpopulárnějších a nejrozšířenějších Google služeb. Použití aplikačního rozhraní je demonstrováno na praktických příkladech tak, aby byla problematika přiblížena především vývojářům internetových aplikací.

Klíčová slova

Rozhraní, API, Google, knihovna, licence, aplikační rozhraní, synchronizace, kalendář, mapy, překladač, analytika, web, aplikace, vývoj

Title

The Google API Library

Annotation

This bachelor thesis deals with possibilities of application programming interface of Google services. The objective is to introduce with the license and commercial use application interface of the most popular and the most widely used Google services. The programming interface is demonstrated on practical examples. Issues should be approached especially developers of Internet applications.

Keywords

Interface, API, Google, library, license, application interface, synchronization, calendar, maps, translate, analytics, web, application, development

Obsah

Seznam zkratek

API	Application Programming Interface
WWW	World Wide Web
HTTP	Hypertext Transfer Protocol
URL	Uniform Resource Locator
URI	Uniform Resource Identifier
GPS	Global Positioning System
GUI	Graphical User Interface
iCAL	iCalendar
HTML	HyperText Markup Language
XML	Extensible Markup Language
PHP	PHP: Hypertext Preprocessor
JS	JavaScript language
CSS	Cascading Style Sheets
REST	Representational State Transfer
ID	Identificator

Seznam obrázků

Seznam tabulek

1 Úvod

1.1 Motivace

Internet je v současné době nejoblíbenější a nejrozšířenější prostředek pro výměnu informací. Pomocí nejrůznějších nástrojů a služeb můžeme z pohodlí domova nakupovat, sledovat aktuální dění ve světě nebo komunikovat s přáteli. Abychom si mohli v běžném životě pomocí Internetu usnadnit práci při získávání informací, musí být ony nástroje někým vytvořeny. Aktuálně existuje mnoho velkých společností, které díky svým nápadům a projektům mohou udávat směr vývoje Internetu.

Jednou z těchto společností je Google. Díky svým pokročilým službám udává směr Internetu tím, že svá díla nabízí méně zkušeným vývojářům k drobným modifikacím. Vývojáři si díky speciálně vyvinutému rozhraní „ohýbají“ produkty k obrazu svému. Tím vzniká stále širší komunita lidí využívající Google služby. Firma díky této komunitě stále roste. Využitím Google služeb docílíme mnohem modernějších a více dynamických vlastních internetových stránek bez nutnosti programovat vlastní mapový, či vyhledávací systém.

1.2 Zadání

Cílem této práce je seznámit čtenáře s problematikou využití speciálního rozhraní společnosti Google, tzv. Google API.

Teoretická část bude zaměřena na zmapování možností použití prostředí Google API. Zejména na použití v komerčním prostředí. Nejdůležitější vybrané komponenty API budou popsány a navrženy možné způsoby jejich využití. U každé části budou doplněny příklady. Dále bude teoretická část zaměřena na možnosti licencování produktů, které využívají Google API a provedena rešerše v oblasti využití API v komerční sféře.

Výstupem z praktické části práce bude knihovna, která bude obsahovat řešené příklady použití komponent, které budou popsány v teoretické části práce. Cílem knihovny bude usnadnění přístupu k hotovým řešením pro vývojáře.

2 Základní pojmy

Tato kapitola seznamuje čtenáře se základními pojmy, které jsou v následujících částech práce použity. Pochopení základních pojmů vede je nutné pro další rozvoj práce.

2.1 API, aplikační rozhraní

Aplikační rozhraní je sběr funkcí, tříd, či metod nějaké knihovny, případně jiného programu. Používá se tedy při vývoji aplikací. Rozhraní zároveň předepisuje, jak mají být funkce volány.

2.2 GPS, souřadnice, geolokace

GPS je polohový systém, díky kterému může být přístroj opatřený modulem lokalizován kdekoliv na světě. Tato metoda pro zjišťování polohy se nazývá geolokace. Systém funguje na principu radiových vln. K identifikaci pozice se využívají tzv. souřadnice, což je číselné označení zeměpisné šířky a zeměpisné délky. Kombinací těchto dvou údajů lze určit kterýkoliv bod ve světě. Polohovací systém aktuálně zažívá největší rozmach v mobilních telefonech nebo cestovních navigacích.

2.3 Google účet

Většina internetových aplikací vyžaduje registraci uživatele. Nejinak tomu je i u služeb společnosti Google. Protože Google vlastní několik aplikací a služeb, dává uživateli při registraci možnost využít jednotné přihlášení. Tím vzniká uživatelský účet zastřešující přihlašování do všech služeb, které jsou navzájem propojeny.

2.4 Vývojář

Vývojář je pokročilý uživatel, který vytváří kanály pro výměnu informací (například v podobě programu, či aplikace). Vývojář je tedy tvůrcem vzhledu, chování a funkčnosti vytvářené aplikace, kterou musí umět ovládat méně zkušení uživatelé.

2.5 Uživatel

Uživatel může být osoba bez hlubších znalostí z oblasti informačních technologií. Uživatel je ve velké většině případů konzumentem informací, které získává skrze aplikace vyvinuté vývojářem.

2.6 Mobilní telefon, smartphone

Mobilní telefon je kapesní přístroj sloužící především pro výměnu informací mezi uživateli. Pokročilejší přístroj s bohatšími funkcemi a vlastním operačním systémem se nazývá „smartphone“, neboli chytrý telefon. Do chytrého telefonu lze pohodlně instalovat doplňkové aplikace. Mobilním telefonem je v této práci myšlen chytrý telefon s operačním systémem Android, za jehož vývoj zodpovídá společnost Google.

2.7 Gadget

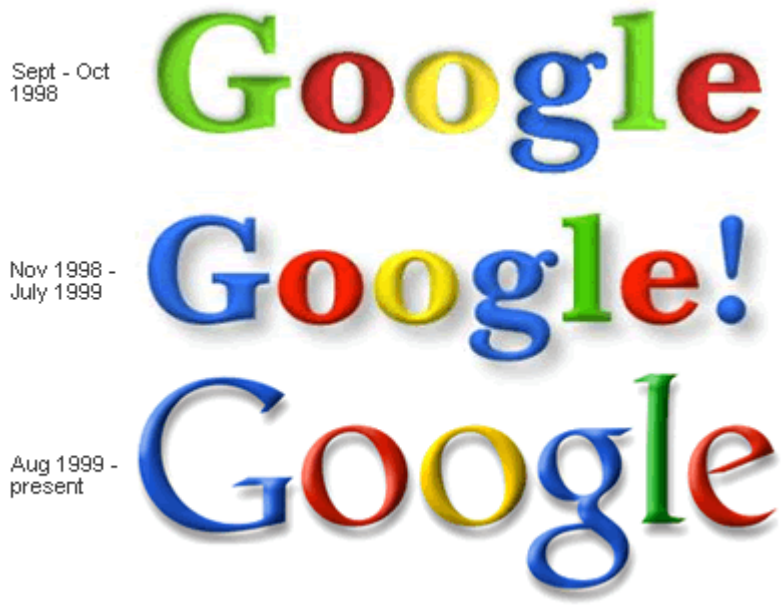
V internetových aplikacích může být často použitý interaktivní prvek, který sbírá informace z jiného zdroje. Zároveň je tento prvek takřka nezávislý na okolních prvcích v aplikaci. Často se jedná o miniaplikaci, v praxi si lze takový gadget představit jako vestavěný postranní kalendář, který pouze informuje o aktuálním dnu v měsíci. Tato miniaplikace nemusí být nijak doprogramována, zdrojový kód lze jednoduše od tvůrce převzít a nasadit do své vyvíjené aplikace.

3 Google, produkty a služby, úvod do Google API

3.1 Společnost Google, Inc.

Společnost Google, Inc. (dále pouze Google) je jednou z největších společností pohybujících se v oblasti informačních technologií. Založena byla v roce 1998 a jejím v té době jediným posláním bylo propagovat svůj vlastní neustále vyvíjený stejnojmenný vyhledávač. V současnosti společnost nabízí mnoho produktů a služeb, které se snaží propojit do jednoho „ekosystému“. Google disponuje vlastními mapovými podklady, vlastní sociální sítě, operační systémy, platební a reklamní systémy a nespočet dalších služeb.

V následujících podkapitolách jsou uvedeny nejpopulárnější služby, jež mají své vlastní API. Nejrozšířenější API rozhraní budou v dalších kapitolách podrobněji prezentovány na praktických příkladech. Výčet není úplný, protože Google podporuje a neustále rozvíjí stovky projektů.



Obrázek 1 - Vývoj loga společnosti Google (zdroj: [1])

3.1.1 Mapy Google

Aplikace Mapy Google je volně dostupná pro nekomerční využití na Internetu. Základem jsou mapové podklady vytvořené samotnou společností. Aplikace nabízí tři mapové vrstvy – vektorově kreslenou, bitmapovou satelitní a terénní. Mapy Google umožňují například volně měřit vzdálenosti, plánovat trasy, vyhledávat existující podniky, nebo informovat o aktuální dopravní situaci.

3.1.2 Google Mail – Gmail

Gmail je služba, která poskytuje uživatelům bezplatně využívat svoji e-mailovou schránku. Aplikace má své uživatelské prostředí, dostupné prostřednictvím Internetu. Tato služba je dále propojena s ostatními službami, typicky se sociálními sítěmi, případně jinými osobními účty u jiných Google produktů.

3.1.3 Vyhledávání Google

Je nejstarší a nejvyužívanější služba, která umožňuje uživatelům vyhledat vždy relevantní informace na základě krátkých dotazů, tzv. klíčových slov. V pozadí aplikace běží pokročilé a těžko nahraditelné algoritmy. Díky nim vzniknul populární a ve své době bezkonkurenční vyhledávač nabízející výsledky, které uživatel očekává. Tato základní vyhledávací řádka může být využita i v aplikacích třetích stran, aby byla ze strany Google podpořena stále zvyšující se rozšířenost a popularita.

3.1.4 Google kalendář

Google kalendář je další online aplikace svázaná s Google účtem každého uživatele. Při vytvoření účtu se zakládá uživateli online kalendář s širokými možnostmi nastavení. Do kalendáře lze například vkládat, exportovat a sdílet události. Aplikace disponuje webovým rozhraním, ovládat lze však i z mobilního telefonu. Uživatel s touto službou získává svůj osobní diář.

3.1.5 Google Analytics

Google Analytics je služba, která si klade za cíl měřit návštěvnost vlastních internetových stránek. Služba je zdarma a díky implementaci krátkého zdrojového kódu do vlastních webových stránek je Google schopen poskytnout například zdroje návštěvnosti, což může být stránka nebo doména předcházející té navštívené. Mezi další údaje může patřit tok návštěvníků, nebo nejvyužívanější zařízení, která internetovou stránku zobrazila. Díky této službě získává vývojář zpětnou vazbu, díky které může svůj web optimalizovat.

3.1.6 Google+

Google+ je sociální síť propojující uživatelské Google účty. Každý uživatel si buduje vlastní síť přátel, se kterými může sdílet informace nebo osobní údaje.

3.1.7 Google Latitude

Google Latitude je služba kombinující prvky sociální sítě a geolokační hry. Uživatel má možnost skrze mobilní telefony sdílet a oznamovat svoji polohu se svými přáteli. Do služby jsou zahrnuty různé firmy, kina, či restaurace. Tyto podniky mohou uživatelé veřejně hodnotit. Uživatelé mohou své výkony sdílet nebo mezi sebou v rámci aplikace soupeřit

3.1.8 Google Translate

Google překladač je online aplikace, která umožňuje přeložit psaný, či mluvený text z jednoho jazyka do jazyka druhého. Uživatelé mohou zadávat jak slovní spojení, tak internetové URL adresy. Aplikace při této volbě přeloží celou internetovou stránku do požadovaného jazyka.

3.1.9 Google Street View

Doplněk pro aplikaci Mapy Google, který je v aplikaci přímo vestavěný. Pokud existují patřičné podklady, je možné se díky 3D panoramatickým snímkům pohybovat po reálných ulicích v nejrůznějších zemích světa. Snímky jsou pořizovány speciálními zařízeními. Tam, kde 3D snímky nejsou dostupné, jsou k dispozici alespoň realistické náhledy, což jsou ověřené fotografie uživatelů, které dané místo navštívili..

3.2 Úvod do Google API

V současné době se každý vývojář webových aplikací pokouší vynaložit určité úsilí na to, aby ze svého produktu vytěžil maximum. Aby tvůrce aplikace splnil náročné požadavky uživatelů, často se ocitá v situaci, kdy potřebuje vložit do stránek aktivní prvky. Příkladem může být mapa s adresou sídla (v případě, že se jedná o internetovou prezentaci firmy). Protože vývojář nemá své vlastní mapové podklady, musel by být nucen vložit jednoduchý textový URL odkaz, potažmo statický obrázek mapy s funkčností odkazu na existující mapové portály. Bez využití API by však byl odkaz ochuzen o dynamičnost, tzn., že by mapa nemohla být v daném obrázku ovládána.

Společnosti Google patří jeden z mapových portálů[1], které nabízí vlastní API rozhraní. Díky tomuto rozhraní se může z obyčejného textového odkazu stát interaktivní mapa s vlastními ikonami a ovládacími prvky. Google uvolňuje různá API pro většinu svých služeb. Jejich použití je popsáno dále na konkrétních příkladech.

3.3 Registrace, API klíč, licence

K tomu, aby vývojář mohl využívat pro své projekty Google API, je nutné mít vlastní tzv. Google účet, tedy být zaregistrovaný. Vývojář si skrze svůj účet nechá vygenerovat svůj unikátní API klíč. Registrace je možná na příslušné internetové adrese[2]. API klíč slouží k jednoznačné definici spojení mezi servery Google a serverem, kde běží vývojářova aplikace. Tím je do jisté míry zajištěna i kontrola provozu ze strany Google.

Registrace a generování API klíče je bezplatné. Není však možné využívat API například u komerčních, či technicky náročných projektů. Ke každému aplikačnímu rozhraní se vztahují licence, které jsou ve většině případů odvozené od náročnosti na provoz.

3.3.1 Licence Google Maps API

Bezplatnou licenci lze využít pro malé a středně velké webové stránky. Ze stránek, kde se vyskytuje prvek využívající API rozhraní, nesmí mít vývojář žádný finanční zisk. API nesmí být ani využíváno pro vnitrofiremní účely, například ve školním nebo firemním intranetu. Z hlediska náročnosti na provoz je bezplatná licence omezena na počet požadavků směřujících ze serveru vývojáře na server Google (například požadavek aplikace pro překlad textové adresy na číselné souřadnice). Počet požadavků je omezen na 2 500 denně. V případě, že vývojář tento limit překročí, musí provést jedno z následujících opatření:

- Modifikovat svoji aplikaci tak, aby počet požadavků nedosahoval stanovené hranice 2 500 požadavků za den,
- Nechat si od společnosti Google účtovat určitou částku za každý požadavek, který je nad limitem,
- Zakoupit tzv. Business licenci.

[1] https://maps.google.cz/
[2] https://code.google.com/apis/console

Business licence poskytuje vývojářům parametry pro využití, provoz a zpracování požadavků z aplikačního rozhraní. Příkladem může být:

- 100 000 požadavků denně,
- Možnost použití statické mapy o rozlišení 2048 x 2048px (oproti 640 x 640px v bezplatné licenci),
- Možnost zpětného sledování návštěvnosti a provozu prostřednictvím služby Google Analytics,
- Technická podpora ze strany Google, a další.

Kompletní podmínky používání a veškeré podrobnosti o licencích jsou dostupné na portále určenému vývojářské komunitě společnosti Google[3].

3.3.2 Licence Google Calendar API

U Google Kalendáře neexistuje placená licence, služba je poskytována zdarma. I přesto zde existují limity, aby nebyla aplikace zneužívána a zahlcována. Limitem je maximální počet 10 000 požadavků za kalendářní den. Požadavkem se rozumí vložení, editace nebo smazání události, potvrzení schůzky, aktualizace dat ve sdíleném kalendáři, apod.

Pokud jsou limity nedostačující, lze prostřednictvím internetového formuláře[4] zažádat o navýšení limitů. V žádosti se uvádí odůvodnění, která Google zohlední a posléze limity navýší, či nikoliv.

3.3.3 Licence Google Translate API

Tato služba nedisponuje bezplatnou licencí, a to od 1. prosince 2011, kdy první verze API rozhraní byla nahrazena verzí novější. Namísto bezplatné licence nabízí Google tvůrcům webových aplikací možnost umístit gadget[5], což je hotová miniaplikace, kterou nelze nijak hlouběji editovat. I tento prvek však může vývojářům postačit, neb splňuje základní očekávání uživatele.

Z hlediska placené licence jsou limity určeny počtem přeložených znaků. Za přeložené znaky je určena i výše placené licence. Základní poplatek je ve výši dvaceti amerických dolarů za jeden milion přeložených znaků. Vývojář má možnost aktivovat detekci jazyka zadávaného textu. Tato funkce je zpoplatněna zvlášť, a to ve stejné výši, tedy 20 $ za 1 milion detekovaných znaků.

K placené licenci se vztahuje i limit vytíženosti služby. Zákazník nesmí překročit limit dvou milionů přeložených znaků během jednoho kalendářního dne. Tento limit lze rozšířit až na 50 milionů znaků za den. V případě, že i tento limit zákazníkovi nedostačuje, přistupuje se k individuálnímu jednání o rozšíření.

[3] https://developers.google.com/maps/licensing
[4] K formuláři se přistupuje z účtu vývojáře, formulář nemá vlastní pevnou URL adresu
[5] Dostupný ke stažení na WWW: https://translate.google.com/manager/website/

3.3.4 Licence Google Analytics API

Tak jako u Google Kalendáře je i tato služba dostupná výhradně zdarma. I přesto zde existují limity, které chrání především dostupnost serverů Google před potenciálním zneužitím. Limitem je 50 000 dotazů denně, přičemž dotazem se rozumí žádost o stažení návštěvnosti, vykreslení grafu, apod. Existuje zde také omezení maximálního počtu dotazů za sekundu v rámci jedné IP adresy. Toto omezení je standardně na hodnotě jednoho dotazu za sekundu, avšak je možné tento limit navýšit až na deset dotazů za sekundu. V případě překročení limitu je při každém dalším požadavku, který je nad limitem, vrácen chybový stavový kód HTTP protokolu s číslem 402.

Veškeré limity lze samozřejmě rozšířit i nad uvedené hodnoty, to je však možné pouze cestou žádosti o povolení, tak jako je tomu u Google Calendar API.

3.3.5 Licence Google Custom Search API

Vlastní vyhledávač lze využít maximálně k 100 požadavkům denně. Jakmile je limit překročen a zároveň má majitel API klíče povoleny bankovní transakce, je každý další dotaz hrazen. Pokud transakce není ze strany majitele API klíče povolena, vyhledávání, resp. vyhledávací požadavek se nezdaří. Výše poplatků je stanovena na 5 $ za každých 1 000 dotazů za den. I hrazená licence má však omezení, maximálně lze přenést 10 000 dotazů za den.

Tabulka 1 – Přehled limitů pro popisovaná API

API	limit na jednoho uživatele	celkový limit licence zdarma	rozšířená licence	zúčtovatelný limit
Analytics API	1 dotaz za sekundu na uživatele	50 000 dotazů za den	na požádání	-
Calendar API	5 dotazů za sekundu na uživatele	10 000 dotazů za den	na požádání	-
Custom Search API	1 dotaz za sekundu na uživatele	100 dotazů za den	placená	1 000 dotazů za den
Maps API	1 dotaz za sekundu na uživatele	25 000 dotazů za den	placená	500 000 dotazů za den
Translate API	100 znaků za sekundu na uživatele	0 znaků za den	placená	2 000 000 znaků za den

4 Google Maps API

V této kapitole jsou na praktických příkladech znázorněny možnosti využití Google Maps API. K těmto účelům je zapotřebí vytvořit webovou stránku a vlastnit API klíč. Webová stránka by měla splňovat veškeré standardy. Z hlediska znalostí je vyžadována pokročilá znalost HTML, základní znalost kaskádových CSS stylů a základní znalost jazyka JavaScript. Tato kapitola se nebude zabývat popisem jednotlivých HTML značek, popsán bude převážně kód psaný v jazyce JavaScript, v němž je aplikační rozhraní definováno. Příklady jsou strukturovány dle složitosti.

4.1 Vytvoření základní mapy

K zobrazení základní plnohodnotné mapy je zapotřebí připojit do hlavičky základního HTML souboru URL adresu obsahující klíč. Tím se web vývojáře propojí se servery Google. Adresa se vkládá jako parametr do značky `<script>`. Výsledek může vypadat takto:

```
<script src="http://maps.googleapis.com/maps/api/js?
key=AIzaSyC_5c5jcCeK3hOuDOROkNjmve4JcXX00XX&sensor=false"></script>
```

Hodnota za rovnítkem u parametr `key` představuje zmiňovaný API klíč. Parametr `sensor` je jediným povinným parametrem. V případě, že k aplikaci přistupuje uživatel například z mobilního telefonu vybaveným GPS modulem, může být (pokud si to aplikace vyžaduje) vyzván ke sdílení své polohy. Jestliže vývojář nepotřebuje pracovat s polohou uživatele, nastaví hodnotu parametru na `false`.

Základní mapu vývojář vytvoří opět v HTML značce `<script>` vhodně umístěnou v HTML dokumentu. K definici základní mapy se standardními ovládacími prvky postačí vývojáři tento zdrojový kód:

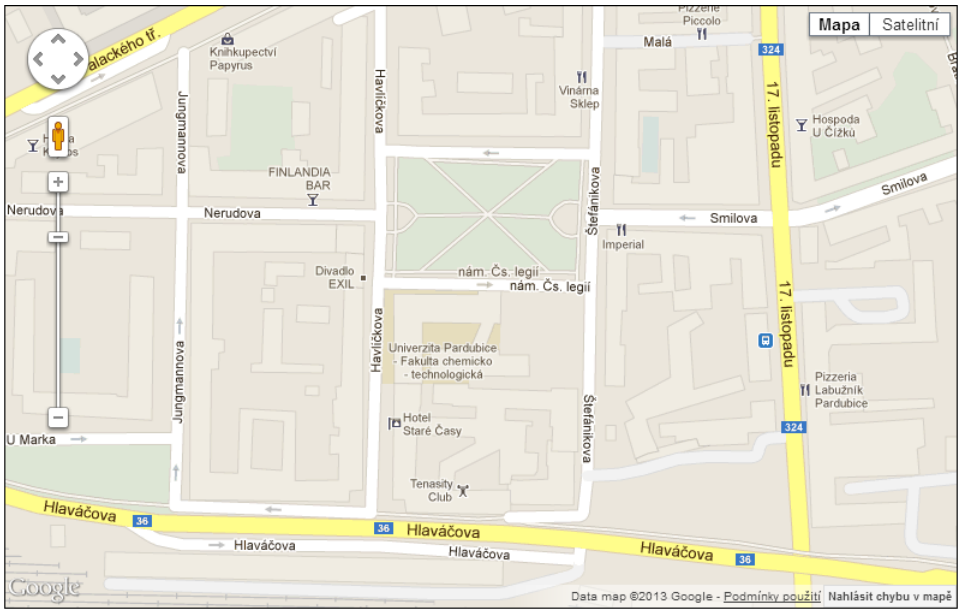
```
var mapaJS = new google.maps.Map(document.getElementById('mapa'), {
      zoom: 17,
      center: new google.maps.LatLng(50.033445, 15.767472);
      mapTypeId: google.maps.MapTypeId.ROADMAP,
});
```

Proměnná `mapaJS` je instance třídy `google.maps.Map`. Argumentem objektu je hodnota HTML identifikátoru, nad kterým se má mapa vytvořit. Další argument je uvedený ve složených závorkách, jako vlastnosti vytvářené mapy. Mezi ně patří počáteční přiblížení mapy, přičemž hodnota může být v rozsahu 0-19. Parametr `center` udává střed mapy v číselných souřadnicích. Posledním parametrem je `mapTypeId` a ten udává typ vrstvy mapy. Je možné využít hodnoty:

- `ROADMAP` (pro standardní vektorovou mapu),
- `SATELITE` (pro bitmapové satelitní snímky),
- `HYBRID` (satelitní kombinovanou s vektorovými cestami),
- `TERRAIN` (mapu znázorňující zdrsnění terénu).

V prohlížeči se mapa zobrazí mezi blokovými HTML značkami, příkladem může být značka `<div>` s parametrem označující identifikátor mapy:

```
<div id="mapa"></div>
```



Obrázek 2 - Mapa se základními vlastnostmi (zdroj: vlastní)

4.2 Přidání značek (markerů) do mapy

4.2.1 Vložení implicitní značky

Marker neboli značku, či pozici v mapě, lze přidávat několika způsoby. Implicitní značku, značku s předem definovanými vlastnostmi, definuje vývojář v kódu obdobně jako celou mapu.

```
var znacka = new google.maps.Marker({
      position: new google.maps.LatLng(50.033445, 15.767472),
      map: mapa
});
```

V parametru `position` je definována souřadnice, kde se marker zobrazí. V druhém parametru se objevuje název objektu (mapy), který byl deklarován v předešlé kapitole. Značku lze do mapy umístit i metodou `setMap` následovně:

```
znacka.setMap(mapa);
```

4.2.2 Vložení vlastní značky

Vývojář má možnost do mapy vkládat i vlastní obrázky, nebo ikony a použít je jako značku. V takovém případě stačí přidat parametr `icon` a v apostrofech uvést cestu k obrázku, který musí ležet na serveru vývojáře. Analogicky lze ke značce přidat parametr `shadow`, který simuluje vržený stín vkládané značky.

4.2.3 Agregace velkého počtu značek v mapě

V krajních případech lze mapu zahltit vysokým počtem značek, což nese za následek výrazné zpomalení odezvy při ovládání mapy. Mapa může být zároveň nečitelná, protože velké množství značek překrývá lokační údaje v mapě. Této situaci lze předejít několika způsoby [2].

- **Implementace vyhledávání** – při vyhledání zájmového bodu se zobrazí takové území, aby nedošlo k zahlcení mapy značkami,
- **Implementace filtrování** – značky jsou rozděleny do kategorií, přičemž uživatel si může zapínat a vypínat zobrazení značek z dané kategorie,
- **Shlukování značek** – značky se shlukují do grafických prvků, které jsou od sebe dostatečně vzdáleny. Při přibližování mapy se značky automaticky přeorganizovávají. Tato metoda vyžaduje implementaci knihoven třetích stran.

4.3 Vložení informační bubliny

Velmi často se ke značkám připojují informační bubliny sloužící například k uvedení adresy nebo kontaktních údajů zobrazovaného místa. Již známým způsobem je deklarována proměnná informačního okna a k ní je přiřazen parametr `content`, do jehož hodnoty se uvádí v apostrofech vkládaný text. Hodnotou může být i HTML kód obalený ve značce `<div>` s identifikátorem, ke kterému se nevztahují kaskádové styly nadřazeného HTML dokumentu. Tím můžeme docílit vložení obrázků, odkazů, apod.

```
var informace = new google.maps.InfoWindow({
      content: 'Text v informační bublině'
});
```

K vytvořené informaci může být přiřazena událost, která nastane po kliknutí na značku.

```
google.maps.event.addListener(znacka, 'click', function() {
      informace.open(mapa, znacka);
});
```

V této části kódu je implementován listener na prvek `znacka`. Po kliknutí (parametr `'click'`) se vykoná otevření informační bubliny vztažené ke značce (druhý řádek kódu). Není však nutné bublinu vztahovat vždy k nějakému prvku. Pokud je uveden parametr `position`, může se bublina rozevřít prakticky na kterémkoliv místě. Souřadnice se musí zadat ve stupních souřadnic GPS:

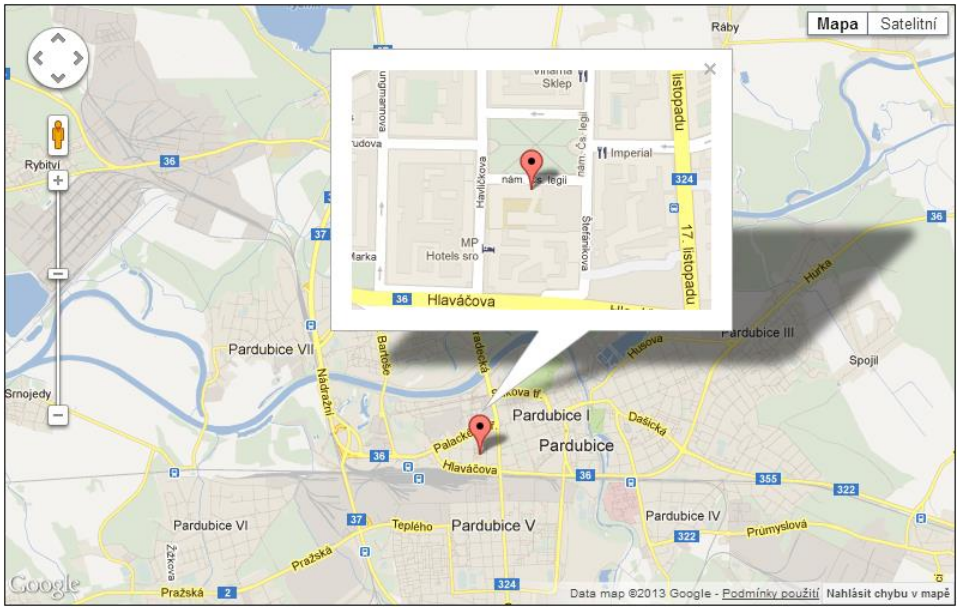
```
position: new google.maps.LatLng(50.033445, 15.767472)
```

4.3.1 Detail mapy jako obsah informační bubliny

Informační bublina se dá využít jako prostor pro vložení jiné mapy. Ta může sloužit například jako detailní náhled uváděného místa.

Nejdříve je zapotřebí vytvořit a definovat velikost interaktivního boxu v informační bublině. Následně prostřednictvím příslušné funkce je definovaný box připojen k nadřazené mapě.

Dále je deklarována nová proměnná, která inicializuje vlastností vnořené mapy. Této mapě se přiřadí stejné vlastnosti jako mapě nadřazené, pouze je změněno přiblížení na vyšší hodnotu. Z vnořené mapy se vytvoří nová instance. Parametrem je vytvořený box a proměnná s definovanými vlastnostmi mapy. Zbývá už jen známým způsobem vytvořit značku nadřazené mapě a přiřadit k ní vytvořenou instanci detailní mapy.



Obrázek 3 - Detail mapy v informační bublině (zdroj: vlastní)

4.4 Trasy, úsečky a polygony v mapě

Na známých mapových portálech lze mimo jiné i vyhledat a zvýraznit trasu z bodu A do bodu B dle požadovaných kritérií (nejkratší, nejrychlejší cesta, apod.). Zvýrazněná trasa není nic jiného než série desítek, stovek, někdy i tisíců propojených úseček. Počáteční bod se rovná koncovému bodu předchozí úsečky v sérii. Tyto „ohyby“ reprezentují reálné zatáčky a změny směrů ve vyhledávané trase.

API rozhraní nám umožňuje do map vkládat vlastní trasy. Pro vložení jedné úsečky je nutné znát souřadnice počátečního a koncového bodu. Tyto souřadnice se zahrnou do jednorozměrného pole, kde prvkem pole je objekt typu `google.maps.LatLng`.

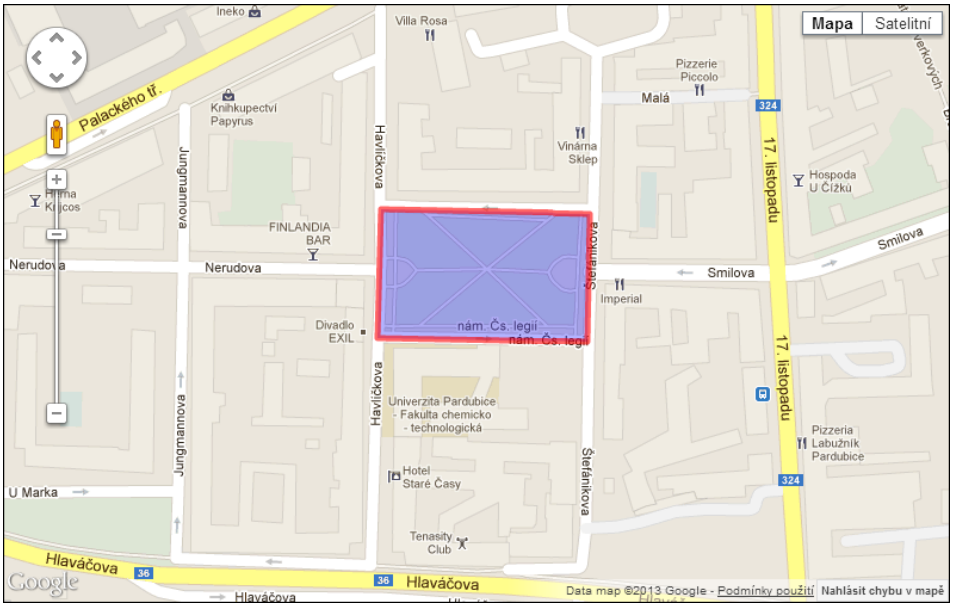
```
var cesta = [
      new google.maps.LatLng(50.034319, 15.76658),
      new google.maps.LatLng(50.033577, 15.766553),
      new google.maps.LatLng(50.033553,15.768442),
      new google.maps.LatLng(50.034294,15.768463)
];
```

Samotná trasa je odvozená od třídy `Polyline`, která poskytuje konstruktor s jedním parametrem. V tomto argumentu se přenáší vlastnosti vytvářené trasy. A protože je výše uvedená definice bodů jen jednou z vlastností vytvářené trasy, je nutné definovat proměnnou nesoucí veškeré vlastnosti pohromadě.

```
var vlastnostiTrasy = {
path: cesta
};
var trasa = new google.maps.Polyline(vlastnostiTrasy);
```

Nakonec zbývá vytvořené trase přiřadit mapu, do které se má promítnout. Tím docílíme pomocí metody `setMap`, kde proměnná s vytvořenou trasou je parametrem metody. Pro rozšíření lze samozřejmě přidat i více vlastností, mezi nimiž může být například tloušťka, barva a průhlednost vykreslované čáry.

Analogicky lze promítnout do mapy i libovolný polygon (mnohoúhelník). Do pole se pouze přidá více souřadnic a při vytváření instance je zapotřebí odvozovat objekt od třídy `Polygon`.



Obrázek 4 - Vyznačení mnohoúhelníku v mapě (zdroj: vlastní)

4.5 Zjišťování adres a zájmových bodů

Často dochází k situaci, kdy uživatel zná hledaný bod podle adresy, ale nezná souřadnice. Dochází i k opačné situaci, kdy uživatel zná souřadnice, ale potřebuje na mapě zobrazit, kde se hledané místo nachází. V Google Maps API jsou zahrnuty oba způsoby hledání. K tomu, aby uživatel mohl hledat na základě vstupních dat, je nutné vytvořit alespoň základní HTML formulář minimálně s jedním textovým polem a odesílacím tlačítkem. Samozřejmostí je i definice boxu, ve kterém se bude zobrazovat mapa s výsledkem. V části zdrojového kódu, kde se vyskytuje JavaScript, se již známým způsobem mapa deklaruje, nastaví se její počáteční vlastnosti a zobrazí se.

```
var vlastnosti = {
      zoom: 3,
      center: new google.maps.LatLng(37.09, -95.71),
      mapTypeId: google.maps.MapTypeId.ROADMAP
};
mapa = new google.maps.Map(document.getElementById('mapa'), vlastnosti);
```

V další části je nutné do proměnné získat hodnotu z formulářového pole. Využít můžeme následující metodu:

```
var formular = document.getElementById('htmlIdentifikatorFormulare');
```

Po vyvolání události `onsubmit` se vykoná blok, který zjistí vyplněnou hodnotu z formulářového pole. Ta je použita jako argument pro funkci, která spouští samotný proces hledání.

```
formular.onsubmit = funkce() {
      var adresa = document.getElementById('htmlIdTextPoleAdresa').value;
      zjistitSouradnice(adresa);
      return false;
}
```

V těle funkce pro zjištění souřadnic je inicializována proměnná odvozená od třídy `Geocoder`. Tím, že je proměnná tohoto typu, může být použita metoda `geocode`, která se stará o vyhledání a navrácení výsledku.

```
function zjistitSouradnice (adresa) {
      geocoder = new google.maps.Geocoder();
      var geocoderPozadavek = {address: adresa}

      geocoder.geocode(geocoderPozadavek, function(results, status) {
            if (status == google.maps.GeocoderStatus.OK) {
            mapa.setCenter(results[0].geometry.location);
            znacka = new google.maps.Marker({
                  map: mapa
            });
            znacka.setPosition(results[0].geometry.location);
            infoBublina = new google.maps.InfoWindow();
            var obsahBubliny = '<strong>' + results[0].formatted_address
                  + '</strong><br />';
            obsahBubliny += 'Zem. šířka: ' +
                  results[0].geometry.location.lat() + '<br />';
            obsahBubliny += 'Zem. délka: ' +
                  results[0].geometry.location.lng();
            infoBublina.setContent(obsahBubliny);
            infoBublina.open(mapa, znacka);
            }
      });
}
```

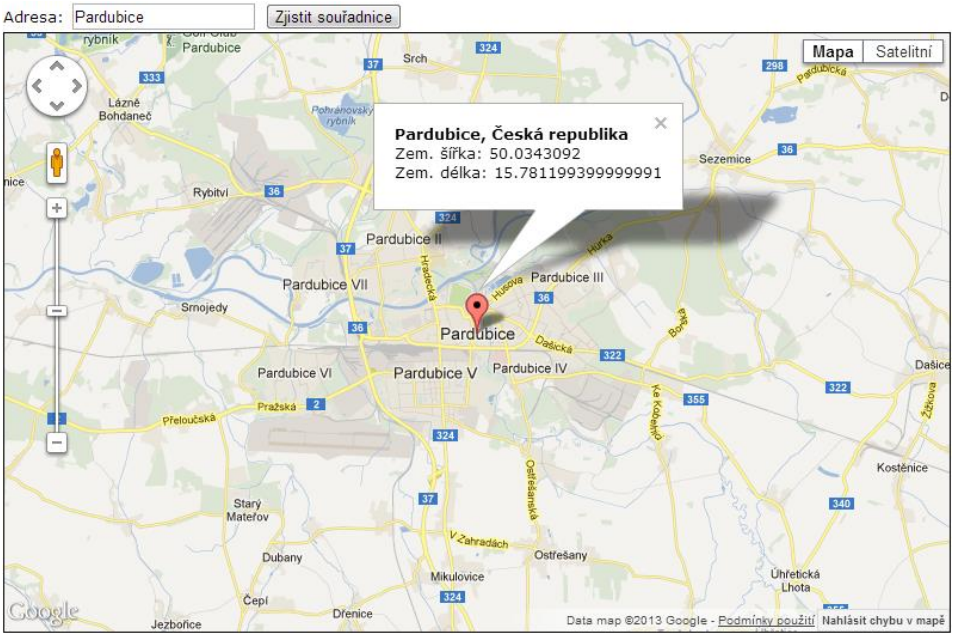
Metoda `geocode` obsahuje dva argumenty. Prvním je proměnná nesoucí informaci, že bude hledáno podle adresy. Druhým argumentem je funkce se zpětným voláním. Metoda `geocode` vrací dvě hodnoty, které je nutné zpracovat. Hodnota `status` nese stav výsledku hledání. Výsledek může nabývat následujících hodnot [2]:

- `OK` – indikuje, že vše proběhlo v pořádku a byly vráceny konkrétní výsledky,
- `ZERO_RESULTS` – dotaz byl zadán v pořádku, ale nebyl vrácen žádný výsledek,
- `OVER_QUERY_LIMIT` – indikuje, že byl přesáhnut maximální limit požadavků, který je možno v rámci licence dosáhnout,

- `REQUEST_DENIED` – z nějakého důvodu byl požadavek serverem odmítnut,
- `INVALID_REQUEST` – nesprávně zadaný příkaz (souřadnice, či adresa není ve správném tvaru).

Hodnota `results[0]` vrací první prvek pole z nalezených výsledků, pokud tedy nějaký výsledek existuje. Z těchto prvků pole se dá získat jak adresa, tak souřadnice nalezeného prvku.

Pro úplnost je ještě ve výše uvedeném kódu deklarována nová informační bublina, do které je již známým způsobem připojen obsah, v tomto případě výsledek hledání, tedy nalezené souřadnice. Níže je na obrázku uvedený výstup popsané funkce.



Obrázek 5 - Zobrazení výsledku hledání (zdroj: vlastní)

4.6 Zjišťování a sdílení polohy uživatele

Na mapových portálech se nevyhledávají jen zájmové body, sídla firem, či podniků. Mapové aplikace bývají stále častěji obohacovány o funkce, které souvisí s aktuální polohou uživatele nebo přenositelného přístroje. Takový přístroj musí nejprve, se schválením uživatel, sdílet svoji polohu. Díky sdílení polohy může být uživatel navigován po trase, protože vidí svoji aktuální polohu zobrazenou v mapové aplikaci. Sdílení může probíhat přes GPS modul nebo prostřednictvím bezdrátové sítě, na kterou je uživatel s přístrojem napojen. V takovém případě je sdílena poloha bezdrátové sítě.

Sdílením polohy se otevírají obrovské možnosti. V současnosti jsou v rozmachu různé geolokační hry a služby. Aby mohla taková služba vzniknout, musí tomu být přizpůsobena i mapová aplikace.

Google Maps API umožňuje získat polohu uživatele, pokud ji ovšem sdílí. O informace získané od uživatele nese objekt `google.loader.ClientLocation`. Použity mohou být tyto metody:

- `latitude` – aktuální zeměpisná šířka zařízení,
- `longitude` – aktuální zeměpisná délka zařízení,
- `address.city` – název města odvozená od aktuálních souřadnic,
- `address.country` – název země odvozená od aktuálních souřadnic,
- `address.country_code` – specifické označení země (dle standardu ISO 3166-1),
- `address.region` – název oblasti, v níž se zařízení nachází.

Do mapové aplikace se po zjištění souřadnic může vložit například značka s informační bublinou, kde se zařízení, resp. uživatel nachází. Vkládání a zobrazení probíhá obvyklým, již popsaným způsobem, stačí si uchovat v pomocných proměnných zeměpisnou šířku a délku „hledaného“ zařízení. Pokud není poloha známa, jsou vráceny nulové souřadnice.

5 Google Calendar API

Mezi další populární služby patří bezesporu Google Kalendář. Každému uživatelskému účtu přiděluje Google možnost využít online aplikaci[6] pro správu událostí, úkolů nebo schůzek vztahující se k určitému datu. Uživatel si může vytvořit hned několik kalendářů. Typické bývá rozdělení na osobní a pracovní kalendáře, přičemž libovolný kalendář může být sdílený s kolegy, potažmo s přáteli. Kalendář může být dokonce i veřejně vyvěšený v nějaké aplikaci třetí strany.

I k této aplikaci uvolnil Google vlastní rozhraní, aby ovládání aplikace nemuselo probíhat v GUI prostředí výchozí internetové aplikace. API umožňuje například přidávat události přímo z desktopových nebo internetových aplikací třetích stran. Možností pro správu kalendářů prostřednictvím API je hned několik.

5.1 Viditelnost kalendářů, sdílení

Ne všechny kalendáře jsou veřejné, nastavení sdílení závisí pouze na uživateli. Google umožňuje uživateli nastavit několik úrovní sdílení:

- Soukromý kalendář - nelze jej využít z hlediska API rozhraní, k soukromému kalendáři se nelze připojit.
- URL adresa kalendáře pro soukromé účely – tato úroveň slouží pro osobní správu a propojení. Data kalendáře jsou ve formátu XML, příp. iCAL a jsou určená pouze pro čtení. Nevyžaduje se autorizace.
- Sdílený kalendář s konkrétními osobami – kalendář je viditelný a upravovatelný konkrétním osobám. Proto je nutná před spojením s kalendářem autorizace.
- Veřejný kalendář – je veřejně dostupný, a to v několika formátech. Není nutná autorizace uživatele.

5.2 Interakce s kalendářem – REST architektura

Architektura REST je založena na stavovém hlášení serveru na základě požadavku HTTP protokolu ze strany klienta. Klient při dotazu může použít jakoukoliv z metod HTTP protokolu, některé však jsou pro interakci s kalendářem zakázané [4]:

- GET – získání zdroje, například XML soubor,
- POST – vytvoření a předání nové zprávy serveru (často s autorizací uživatele),
- PUT – operace změny (v aktuální verzi není operace povolena, využívá se POST),
- DELETE – smazání zdroje (v aktuální verzi není operace povolena),
- HEAD – poskytuje metadata o požadovaném cíli (velikost, datum změny, apod.).

Pokud je kalendář dostupný pod URL adresou, lze například z prostřední konzolové aplikace (využívající komunikaci se serverem na úrovni protokolů) vznést požadavek. Ten je v následujícím tvaru, přičemž `userID` se rovná názvu Google účtu a `unikatniKlic` se rovná

[6] https://www.google.com/calendar

vygenerovanému sledu znaků, který lze zjistit například v nastavení sdílení daného kalendáře[7].

```
GET http://www.google.com/calendar/feeds/userID/private-unikatniKlic/full
```

Protože jde o GET požadavek HTTP protokolu, lze situaci simulovat v adresové řádce webového prohlížeče, pochopitelně bez zadání příkazu GET. Klient je následně přesměrován na jinou URL adresu, ve které je uložena i připojovací session. Pokud je poté vrácen stavový kód „200 OK“, obdrží klient data o kalendáři, včetně všech událostí, a to ve formátu XML.

Pokud jsou v kalendáři povoleny veřejné úpravy, lze XML kód patřičně upravit a obdobným způsobem, ovšem za použití metody POST, odeslat zpět na server, který změny v kalendáři zaznamená.

5.3 Interakce s kalendářem – klientské knihovny

Daleko pohodlnější a rozmanitější přístup ke kalendářům zajišťuje tzv. „Client library“. V překladu klientská knihovna je soubor tříd a metod sloužící ke komunikaci s Google kalendářem. Knihovna je dostupná pro většinu vyšších programovacích jazyků. Na oficiálních stránkách[8] lze stáhnout knihovnu například pro jazyk Java, Python, PHP, C# nebo Ruby. Vývojář má k dispozici i knihovnu psanou v jazyce JavaScript, ta ovšem není dostupná ke stažení a volnému používání. Veškerá interakce s kalendářem probíhá v grafickém rozhraní v internetové aplikaci[9] vyvinuté spol. Google.

Pro praktické ukázky je použita knihovna pro jazyk PHP.

5.3.1 Výpis událostí bez autorizace

Nejjednodušším příkladem pro znázornění komunikace je výpis události, který je dostupný skrze URL odkaz na XML soubor. Pro tento případ není nezbytné využít klientskou knihovnu, vývojáři postačí pouze PHP metoda `simplexml_load_file()`.

```
$kal = simplexml_load_file('http://www.google.com/calendar/feeds/...');
foreach ($kal->entry as $udal) {
      echo "<h2>" . $udal->title . "</h2><p>" . $udal->content . "</p>";
}
```

Ze zdrojového kódu je patrné, že se přistupuje ke každé položce s názvem `entry` ze staženého XML souboru. Pro všechny tyto události je předepsáno, že se vypíše nejprve název události a následně obsah. Na výstupní HTML stránce se události řadí tak, že nejprve se vypíší uplynulé události a po nich se řadí události od časově nejvzdálenější po nejaktuálnější. Pořadí řazení lze samozřejmě příslušnou modifikací skriptu upravit.

[7] V prostředí online aplikace lze sdílení a nastavení kalendáře vyvolat rozbalením roletky ve výpisu kalendářů v pravé části obrazovky.
[8] https://developers.google.com/gdata/docs/client-libraries
[9] https://developers.google.com/apis-explorer/#p/calendar/v3/

Obrázek 6 - Možný výstup výpisu událostí (zdroj: vlastní)

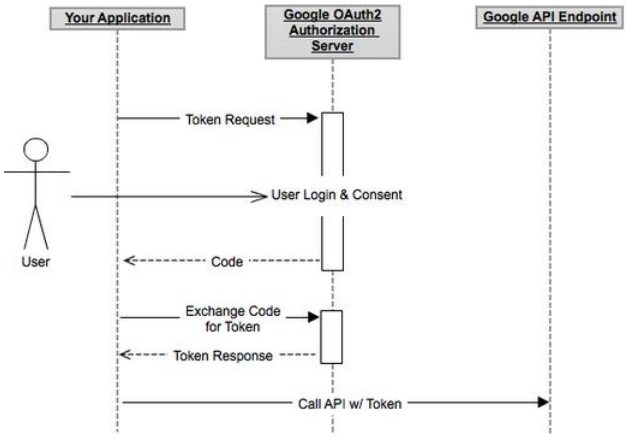
5.3.2 Autentizace a autorizace uživatelů

Kalendáře uživatelů mohou být i částečně sdílené. Může existovat například internetová aplikace, která umožňuje uživatelům vkládat nebo zobrazovat události ve svých kalendářích. V takovém případě je nutné implementovat proces autentizace (jednoznačné určení uživatele) a autorizace (proces ověřující oprávnění k přístupu). Po ověření přes Google účet se propojí aplikace třetí strany s Google kalendářem uživatele. Tento proces zajišťuje tzv. OAuth 2.0 protokol.

Aby mohl vývojář pracovat s procesem autentizace, musí nejprve vyvíjenou aplikaci zaregistrovat[10]. Při registraci je nutné uvést název a druh aplikace (desktopová, internetová), dále může uvést logo a domovskou www adresu projektu. Po úspěšné registraci je aplikaci přiřazen jednoznačný identifikátor s ověřovacím kódem. Společně s API klíčem je vyvíjená aplikace provázána se servery Google.

Ve výsledku je uživatel při prvním přístupu do vyvíjené aplikace přesměrován na autentizační URL adresu, která je generována ze strany Google ve volané metodě z klientské knihovny. Na této adrese se uživatel přihlašuje do svého Google účtu. Následně je uživatel upozorněn, která osobní data zpřístupňuje aplikaci (v tomto případě data z kalendáře). Pokud uživatel souhlasí, dojde po stisknutí tlačítka myši k autorizaci na základě tzv. tokenu, jehož hodnotou je generovaný sled znaků. Hodnota tokenu slouží jako session, tzn., že na základě hodnoty tokenu je uživatel v průběhu celé interakce s aplikací ověřován.

[10] https://code.google.com/apis/console, poté výběr položky z menu „API Access“.



Obrázek 7 – Use case diagram využití OAuth 2.0 protokolu (zdroj: [5])

5.3.3 Výpis kalendářů s využitím autentizace

V rámci klientských knihoven nabízí Google možnost nahlédnout do jednoduchých příkladů použití. V případě Google Kalendáře je k dispozici skript znázorňující implementaci autorizačního procesu.

```
require_once "google-api-php-client/src/Google_Client.php";
require_once "google-api-php-
      client/src/contrib/Google_CalendarService.php";
$client = new Google_Client();
    $client->setApplicationName($appname);
    $client->setClientId('1234567890.apps.googleusercontent.com');
    $client->setClientSecret('xXxXxXxXxXxXxX');
    $client->setRedirectUri('http://localhost/callback.php');
    $client->setDeveloperKey('APIkey123');
    $cal = new Google_CalendarService($client);
```

V prvním případě je nutné definovat cestu ke třídám, od kterých jsou odvozovány vytvářené objekty. Prvním z nich je objekt `client`, který uchovává identifikátory pro spojení vyvíjené aplikace se servery Google. Je nutné nastavit jméno aplikace a jednoznačný identifikátor s ověřovacím kódem (získaný při registraci). Dále je nastavována URI adresa, na kterou je směrován uživatel v případě úspěšné autorizace, v poslední řadě je nastaven API klíč. Objekt `client` je posléze použit jako argument při vytváření objektu, která reprezentuje konkrétní službu.

```
if (isset($_GET['code'])) {
    $client->authenticate($_GET['code']);
    $_SESSION['token'] = $client->getAccessToken();
   header('Location:http://'.$_SERVER['HTTP_HOST'].$_SERVER['PHP_SELF']);
}
if (isset($_SESSION['token'])) {
    $client->setAccessToken($_SESSION['token']);
}
if ($client->getAccessToken()) {
    $_SESSION['token'] = $client->getAccessToken();
} else {
    $authUrl = $client->createAuthUrl();
    echo "<a class='login' href='$authUrl'>Připojit se k účtu</a>";
}
```

Při prvním připojení uživatele je vykonán poslední rozhodovací blok. Protože zatím není nastavený token, nebude vyhověno podmínce. Zobrazí se tedy výzva pro přihlášení uživatele. Adresa odkazu se generuje zavoláním metody z třídy `Google_Client`. Na této adrese je prováděno přihlášení. Po přihlášení je uživatel přesměrován zpět do aplikace na již vyplněnou URI adresu. Skript se tedy provede celý znovu. Protože je už v URL adrese přenášen parametr `code`, vykoná se první rozhodovací blok a s ním i autentizace uživatele. Metoda `authenticate` zajistí v případě úspěchu uložení tokenu. Ten se zároveň uloží i do session. Druhý rozhodovací blok zajišťuje pouze vždy aktuální provázanost session a tokenu.

Jakmile je získán platný token, může být zavolána metoda `listCalendarList()` pro uložení seznamu kalendářů. Tato metoda vrací kalendáře v třírozměrném poli (informace o feedu, číselné označení kalendářů, vlastnosti konkrétního kalendáře). Výpis pole může být v kódu implementován následovně.

```
if (isset($_SESSION['token'])) {
        $calList = $cal->calendarList->listCalendarList();
        echo "<pre>".print_r($calList, true)."</pre>";
}
```

5.3.4 Výpis událostí v určitém rozsahu

Obdobným způsobem lze získat seznam událostí z jakéhokoliv kalendáře přihlášeného uživatele. Následující příklad vyhledává události dle zadaného časového rozsahu, a to v kalendáři, jehož název je „TEST_BP“.

O vše se stará metoda `listEvents()`, jejímž parametrem je ID kalendáře a pole nepovinných parametrů, mezi které může patřit časový rozsah, nebo maximální počet vrácených výsledků. Protože uživatel zná spíše název kalendáře namísto identifikátoru, je použita část skriptu z předchozího příkladu, tedy metoda `listCalendarList()`. Metoda `listEvents()` opět vrací vícerozměrné pole. Výpis událostí lze převzít z předchozího příkladu.

```
$minCheck = date(DATE_ATOM, mktime(0, 0, 0, 1, 1, 2013));
$maxCheck = date(DATE_ATOM, mktime(0, 0, 0, 12, 31, 2013));
$calendarName = "TEST_BP";
$res = Array();
$calID = null;

if (isset($_SESSION['token'])) {
      $calList = $cal->calendarList->listCalendarList();
      for ($i = 0; $i < sizeof($calList['items']); $i++) {
            if ($calList['items'][$i]['summary'] == $calendarName) {
                  $calID = $calList['items'][$i]['id'];
            }
      }
      if($calID == null) {
            die("Zadaný kalendář neexistuje.");
      }
      $res = $cal->events->listEvents($calID, array('timeMin' =>
        $minCheck, 'timeMax' => $maxCheck, 'maxResults' => $maxResults));
}
```

Skript může být modifikován tak, že vstupní parametr nebude název kalendáře, ale jednoznačný identifikátor kalendáře. Pokud zná uživatel ID kalendáře jiného uživatele, může nahlížet do jeho událostí. Podmínkou však je, že takový kalendář musí být veřejně sdílený.

5.3.5 Vložení události s použitím HTML formuláře

Data nemusí být pouze stahována a zobrazována. Uživatel samozřejmě může data i tvořit. Příkladem může být vkládání úplně nové nebo upravení již existující události. Z hlediska Google API lze vkládání události demonstrovat na příkladu, ve kterém se bude událost vytvářet na straně uživatele v HTML formuláři aplikace. Formulář může mít mnoho položek sestavující vlastnosti vytvářené události. Následující příklad obsahuje pouze základní vlastnosti, a to název události společně s datem a časem začátku a konce události. Událost musí být vložena do nějakého konkrétního kalendáře. Těch může mít uživatel více. Proto je ve formuláři očekáván prvek, který daný kalendář identifikuje. Příkladem může být roletka vypisující seznam kalendářů přihlášeného uživatele. Výpis kalendářů byl vysvětlen v předchozích kapitolách, proto byl z následující ukázky vynechán.

```
if (isset($_SESSION['token'])) {
      $udalost = new Google_Event();
      $nazevUdalosti = $_POST['nazev'];
      $udalost->setSummary($nazevUdalosti);
      $start = new Google_EventDateTime();
      $start->setDateTime('2013-02-03T10:00:00.000-00:00');
      $udalost->setStart($start);
      $konec = new Google_EventDateTime();
      $konec->setDateTime('2013-02-03T14:00:00.000-00:00');
      $udalost->setEnd($konec);
}
```

Samozřejmostí je ověření prostřednictvím platného tokenu. Následuje vytvoření objektu `$udalost` odvozený od třídy reprezentující událost kalendáře. Příslušnými metodami lze hodnoty vytvořeného objektu nastavit. Použít lze samozřejmě POST metodu odesílání formuláře, a tedy i přebírat hodnoty z příslušných proměnných.

Datum a čas musí být nastaven ve správném formátu. Může se využít PHP konstanta DATE_ATOM reprezentující potřebný časový formát, který je v ukázce zdrojového kódu znázorněn. Datum a čas je samostatný objekt třídy `Google_EventDateTime`.

Pokud jsou události předány veškeré informace z formuláře, může se přikročit k samotnému procesu vložení.

```
$vytvorenaUdalost = $cal->events->insert($calID, $udalost);
echo "Vytvořeno pod ID: " . $vytvorenaUdalost["id"];
```

V proměnné `$calID` je uložen ID kalendáře. Tento argument pro funkci `insert` je povinný. Druhým parametrem je objekt `$udalost` vytvořený v předchozí ukázce. V případě, že je funkce vykonána v pořádku, je proměnné `$vytvorenaUdalost` vráceno asociativní pole s vytvořenou událostí a jejími vlastnostmi. Pomocí klíče `id` lze informovat uživatele, pod jakým jednoznačným identifikátorem byla jeho událost vytvořena. V oficiální internetové

aplikaci je událost zobrazena téměř okamžitě. Uživateli je v poli vrácen i unikátní URL identifikátor do aplikace Google Kalendář na vytvořenou událost. Tento prvek se skrývá pod klíčem `htmlLink`.

V rámci vkládání událostí lze příklad mnohonásobně rozšířit, či modifikovat. Události lze přiřadit seznam účastníků, místo konání, barvu nebo status zaneprázdněnosti uživatele. Obdobně lze události upravovat, případně mazat, dle zadaných kritérií.

6 Google Analytics API

Patrně nejužitečnější Google služba zdarma nejen pro vývojáře poskytuje možnost měření toku návštěvníků na internetové aplikaci. Z hlediska implementace je nasazení služby velmi jednoduché – vývojář zaregistruje svůj web, nechá si vygenerovat část zdrojového kódu a ten vloží do každé stránky, která má být měřena. Služba nabývá na oblibě, protože má tvůrce webu zdarma dostupnou zpětnou vazbu, která svými kvalitami předbíhá i některé marketingové společnosti soustřeďující se na statistiky podobného typu.

I k této službě vydala společnost Google vlastní API rozhraní. Pomocí API rozhraní lze exportovat naměřená data, statistiky nebo grafy, aniž by byl uživatel nucen se přihlašovat do oficiální webové aplikace Google Analytics. Tím se otevírají možnosti pro tvorbu vlastních reportů, které mohou být majiteli stránek zasílaný například e-mailem.

6.1 Využití klientských knihoven

Tak jako u Google Calendar API, lze i u Analytics API využít klientských knihoven. Postup při implementaci a nastavení spojení je analogický. Nastavení spojení je shodné, jak bylo popsáno v jedné předchozí kapitole. Proces autentizace a autorizace probíhá taktéž shodně. Rozdíl nastává ve chvíli, kdy se vytváří instance třídy, která reprezentuje Google službu. V případě Analytics API je objekt odvozován od třídy `Google_AnalyticsService`. Veškeré metody, které lze při práci s klientskými knihovnami využít, lze nalézt ve stejnojmenném souboru uloženým ve složce `src/contrib` stažené klientské knihovny `google-api-php-client` (v případě, že je použita PHP klientská knihovna).

6.2 Easy Dashboard JavaScript Library

Dalším způsobem, jak získávat grafy, je použití knihovny psané v jazyce JavaScript, tzv. Google Analytics Easy Dashboard JavaScript Library[11]. Tato knihovna nebyla vyvinuta přímo společností Google. Na tvorbě této knihovny se podílel tým šesti vývojářů, kteří si kladli za cíl usnadnit získávání grafů z hlediska náročnosti implementace.

K získání základního grafu návštěvnosti postačí k HTML stránce připojit tři externí JavaScript soubory.

```
<script src="https://www.google.com/jsapi"></script>
<script src="http://analytics-api-
samples.googlecode.com/svn/trunk/src/reporting/javascript/ez-ga-
dash/gadash-1.0.js"></script>
<script src="https://apis.google.com/js/client.js?onload=gadashInit">
</script>
```

V dalším kroku je třeba svázat aplikaci s API klíčem a autorizačním protokolem OAuth. Externě je definována proměnná `gadash`, proto stačí zavolat funkci pro nastavení spojení.

[11] Bližší informace o knihovně: http://analytics-api-samples.googlecode.com/svn/trunk/src/reporting/javascript/ez-ga-dash/docs/user-documentation.html

Jako argument bude použit API klíč a klientský identifikátor vygenerovaný při registraci do OAuth 2.0 protokolu.

```
gadash.configKeys({'apiKey': 'ABCDE12345', 'clientId': '12345.xyz.com'});
```

Následně stačí vytvořit nový objekt reprezentující graf. V konstruktoru jsou přenášené veškeré parametry grafu, které jsou z hlediska syntaxe v souladu s oficiálními parametry. Při vytváření objektu může být rovnou zavolána funkce `render()` pro vykreslení grafu.

```
var myChart = new gadash.Chart({
  'type': 'LineChart',
  'divContainer': 'line-chart-example',
  'query': {
    'ids': 'ga:1234567',
    'metrics': 'ga:visitors',
    'dimensions': 'ga:date',
    'start-date': '2013-03-01',
    'end-date': '2013-04-15'
  }
}).render();
```

V uvedeném příkladu jsou nastaveny povinné parametry grafu. Mezi ně patří typ grafu a identifikátor značky `<div>`, do které bude graf vykreslen. Parametr `query` definuje parametry samotného dotazu. Metrikou jsou noví návštěvníci vzhledem k ohraničenému datu. Parametr `ids` je identifikátor uživatelského profilu, který lze vyčíst z URL adresy po přihlášení do aplikace Google Analytics. Tvar adresy končí ve tvaru `aXXXwXXXpXXX`. ID profilu je číselné označení za znakem „p“.

Před prohlížením výstupu je nutná autentizace pomocí OAuth protokolu. Může být použito tlačítko, či odkaz s použitím HTML parametru `id="authorize-button"`. V následujícím obrázku je znázorněn výstup.

Obrázek 8 - Graf návštěvnosti v časovém intervalu (zdroj: vlastní)

7 Google Custom Search API

Také nejstarší a nejrozšířenější Google služba má své API rozhraní. Protože se jedná pouze o vyhledávač, je rozmanitost tohoto rozhraní znatelně menší v porovnání s Maps API, či Calendar API. I principiálně je služba jednoduchá. Uživatel odešle na stranu serveru dotaz obsahující klíčová slova. Na základě klíčových slov musí vrátit co nejrelevantnější výsledky.

Aplikace může být s touto službou propojena na základě vyhledávací řádky a prostoru pro navrácené výsledky hledání. Protože není propojení implementačně náročné, má vývojář několik možností, jak „svůj“ vyhledávač do své aplikace implementovat.

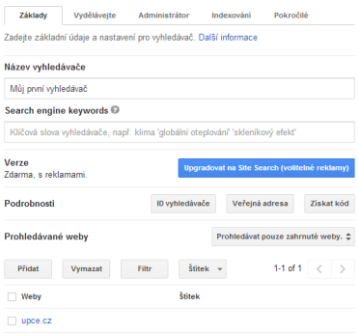
7.1 Vlastní vyhledávač za chodu

První nejjednodušší možností bez znalosti API rozhraní je implementovat část HTML kódu, který lze zkopírovat z internetové adresy[12] pro vytvoření vlastního vyhledávače za chodu. Po implementaci do HTML kódu je vytvořen vyhledávač z odkazů na vývojářově webu. Pokud je provedeno hledání, zobrazí se výsledky webů, na které stránka vývojáře odkazuje.

7.2 Vytvoření plnohodnotného vlastního vyhledávače

Pokud vývojář od vyhledávače očekává širší možnosti úprav a nastavení, může si vlastní vyhledávač více přizpůsobit v internetové aplikaci, která představuje průvodce nastavení vlastního vyhledávače[13].

Na úvodu konfigurace je vyžadován název vyhledávače, jazyková mutace vyhledávače a doména, nebo konkrétní URL adresa, na které bude vyhledáváno. Po nastavení těchto prvních kroků je vývojáři vygenerován identifikační klíč společně s veřejnou adresou vygenerovaného vyhledávače. Průvodce nastavením umožňuje i vygenerovat JavaScript kód, který stačí vložit do té části HTML stránky, kde má vyhledávač fungovat. Správu prohledávaných webů, jazykovou mutaci a možnost zapnutí vyhledávání obrázků lze nastavit i po vytvoření. V pokročilejším nastavení lze i prohledávané weby vylučovat.



Obrázek 9 - Nastavení vlastností vlastního vyhledávače (zdroj: vlastní)

[12] http://www.google.com/cse/tools/create_onthefly
[13] http://www.google.com/cse/all

7.3 Využití klientských knihoven

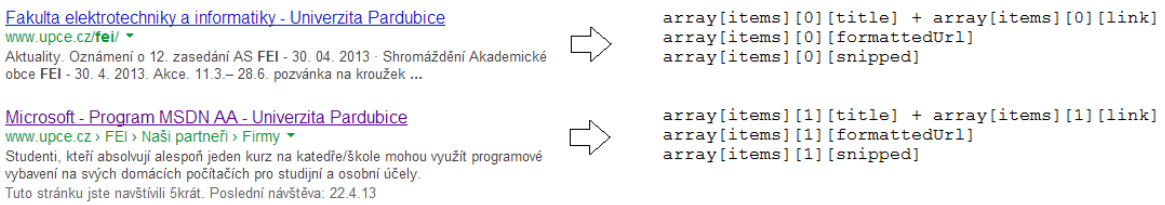
Klientské knihovny může vývojář použít i u rozhraní pro vlastní vyhledávač. Tato volba umožňuje lépe optimalizovat nejen parametry vyhledávacího dotazu, ale také výsledky hledání. Ve vyvíjené aplikaci tak může být výstup hledání naformátován zcela odlišně oproti klasickému výpisu, který je znám z oficiální webové aplikace Google Search.

Princip použití knihovny je velmi obdobný, jako u předchozích příkladů. Při nastavování spojení je nutné uvést pouze API klíč. Dalším parametrem je vygenerovaný identifikátor vlastního vyhledávače, ten se však posílá na stranu serveru společně s vyhledávacím dotazem jako jeden z parametrů.

V následující ukázce je již známým způsobem vytvořena instance třídy `Google_Client`. Této instanci je přiřazena vlastnost v podobě API klíče. Dále je vytvořena služba vytvořením instance třídy `Google_CustomsearchService`. Metoda `listCse` zajišťuje vyhledání dotazu. Metoda má dva parametry, první je text hledaného dotazu, v uvedeném příkladu je to klíčové slovo `fei`, a druhým parametrem je asociativní pole vlastností upřesňující vyhledávací dotaz. Klíč `cx` reprezentuje identifikátor vlastního vyhledávače.

```
$klient = new Google_Client();
$klient->setApplicationName('První vyhledávač');
$klient->setDeveloperKey($apiKlic);
$vyhledavac = new Google_CustomsearchService($client);
$array = $vyhledavac->cse->listCse('fei', array('cx' => $idVlVyh));
```

Odpověď ze strany serveru je uchovávána opět ve vícerozměrném asociativním poli `array`. V poli obsahuje rekapitulaci zadávaného dotazu, statistiku hledání (počet výsledků, délka hledání) a samozřejmě podrobné výsledky hledání. Výsledky jsou řazeny podle relevance. Například titulek stránky z první pozice lze zjistit s použitím klíče `array[items][0][title]`. Vývojář si může kompletní pole vypsat, zhodnotit, které položky z asociativního pole využije, a ty ve své aplikaci naformátovat dle libosti.



Obrázek 10 - Výsledky hledání v asociativním poli (zdroj: vlastní)

8 Využití Google API v komerční sféře

Aplikační rozhraní může být využíváno i v komerčním prostředí. Jedinou podmínkou je dodržení licence k danému produktu. Licence, které jsou zdarma, většinou dovolují použít API k vlastním potřebám. Místa, kde je API použito (webová stránka, desktopová aplikace), nesmí generovat podnikům ani vývojářům finanční zisk. Ve většině případů nesmí být volná licence použita ani pro vnitrofiremní účely a intranety.

8.1 Příklad využití Google Maps API

Aktuálně nejvíce využívané je Google Maps API, což svědčí i o podpoře ze strany Google. Nejvíce verzí aplikačního rozhraní existuje právě ke službě Google Maps. Protože Google vlastní a propaguje své mapové podklady, které v takové kvalitě nevlastní ani mnoho kartografických společností, je služba na té nejvyšší úrovni. Veškeré mapové podklady jsou dostupné skrze API, a proto mohou být použita na prakticky jakékoliv webové stránce. Aplikační rozhraní může být například využíváno v redakčním systému firemních webových stránek. Typicky lze uvést zobrazení několika markerů v jedné mapě ve stránce s firemními kontakty, kde jednotlivé značky znázorňují adresy poboček nebo obchodních partnerů firmy.

Google Maps API může být použito v tzv. mashup aplikacích. Tento druh aplikací sdružuje data z více zdrojů. Konkrétním příkladem je služba Instagram[14], což je sociální síť, která umožňuje rychlé sdílení fotografií. Fotografie jsou pořizovány pomocí aplikace v chytrých telefonech. Ty jsou vybaveny GPS modulem a ke každé fotografii je možné přiřadit polohu pořízené fotografie. Google Maps API je použito například při náhledu na mapu, ve které markery znázorňují místo pořízené fotografie.

Obrázek 11 - Využití Maps API v aplikaci Instagram (zdroj: vlastní)

[14] http://instagram.com

8.2 Příklad využití Google Calendar API

Aplikační rozhraní služby Google Kalendář má své využití u desktopových aplikací i v mobilních telefonech. Vývojář má možnost vytvořit zjednodušenou správu kalendáře, aniž by se přihlašoval do webového rozhraní. S použitím Calendar API mohou být i vyvíjeny aplikace pro synchronizaci kalendářů na různých platformách. Příkladem může být synchronizace webové aplikace Google Kalendář a kalendáře dostupného v desktopové verzi e-mailového klienta Microsoft Outlook[15].

8.3 Příklad využití Google Custom Search API

Vlastní vyhledávač lze využít především na vlastní webové prezentaci nebo eshopu. Vývojář tak nemusí implementovat složité algoritmy pro hledání produktů sám. Jednoduše v konfiguraci vlastního vyhledávače propojí vyhledávání se všemi stránkami v rámci celé domény. Se správnou volbou klíčových slov v jednotlivých stránkách tak vzniká vyhledávač, který generuje výsledky na stejném principu jako plnohodnotný Google Vyhledávač.

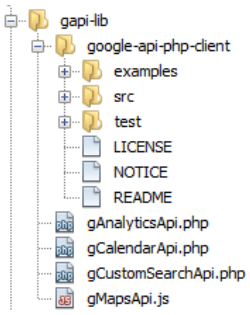
8.4 Příklad využití Google Analytics API

Analytické služby v oblasti internetového marketingu jsou v současnosti velice poptávané. Google Analytics API může být velice užitečným nástrojem právě pro společnosti poskytující služby internetového marketingu. V případě, že marketingová společnost nabízí služby, které garantují návštěvnost webových stránek, může Analytics API využít pro tvorbu pravidelných reportů návštěvnosti doplněných o grafy a další statistiky. Koncový zákazník tak dostává důvěryhodná a neovlivnitelná data pořízená metrikou Google Analytics. Tím může stoupat i důvěryhodnost služby marketingové společnosti.

[15] Aplikace je dostupná ke stažení na adrese: http://google-calendar-sync.m.en.softonic.com/free-download/launch

9 Praktická část

Praktická část této práce je věnována implementaci veškerých popisovaných příkladů použití Google API. Příklady byly strukturovány do funkcí s možností ovlivnit výstup příkladu za použití vstupních parametrů. Cílem praktické části bylo vytvořit knihovnu řešených příkladů, které by měly usnadnit vývojářům práci se základními komponenty Google API.



Obrázek 12 - Struktura souborů knihovny Google API (zdroj: vlastní)

Knihovna byla navržena tak, aby vývojáři stačilo pouze stáhnout soubor `gapi-lib.zip`, ten rozbalit a nahrát do adresářové struktury vyvíjené aplikace. Aby mohl vývojář využívat připravené funkce, je nutné zdrojové soubory z knihovny propojit se zdrojovými soubory aplikace. Pro PHP soubory je to příkaz `require`, pro JS HTML značka `<script src="soubor.js">`.

V adresáři Knihovny Google API se nachází složka `google-api-php-client`, která představuje klientskou knihovnu vyvinutou společností Google pro jazyk PHP. V této části knihovny nebyl proveden zásah do zdrojových souborů, v případě zájmu má vývojář možnost využít API u všech ostatních služeb, které práci s klientskými knihovnami podporují.

Společně s knihovnou byla vytvořena i webová stránka, která má sloužit nejen jako zdroj pro stažení vytvořené knihovny, ale také i jako dokumentace doplněná o názorné příklady použití.

Závěr

Cílem práce bylo přiblížit problematiku aplikačního rozhraní u nejrozšířenějších služeb společnosti Google.

V teoretické části byla představena společnost Google společně s jejími nejvyužívanějšími Google službami. Popsán byl i úvod do aplikačního rozhraní, které Google uvolnil k většině svých služeb. Z těchto služeb bylo vybráno pět, na kterých byla postupně vysvětlena problematika ohledně licencí, limitů a komerčního použití. Vybrány byly služby Google Maps, Google Calendar, Google Analytics, Google Search a Google Translate. Na čtyřech službách byly uvedeny příklady použití svého aplikačního rozhraní. Google Translate API bylo záměrně vynecháno, jelikož Google k této službě nenabízí možnost bezplatného používání. Ke každé službě se tedy vztahují jiné licenční podmínky a podmínky pro užívání.

Příklady použití jednotlivých komponent jsou vysvětlovány tak, aby byly pochopeny především vývojářem, který buduje svoji aplikaci. Ten se však musí nejdříve zaregistrovat pod Google účtem do vývojářského programu, aby mu byl přidělen API klíč, který slouží k autorizaci a monitorování požadavků v komunikaci mezi aplikací a servery Google.

V kapitole o Google Maps API je znázorněno, jak vytvořit vlastní mapu s možností vložení značek, informačních bublin, apod. Popsáno je i zakreslování úseček a mnohoúhelníků do vytvořené mapy. Kapitola je zakončena příkladem kombinující GPS souřadnice se zájmovými body (městy, úřady, apod.).

V části o Google Calendar API jsou popsány dvě možnosti využívání Google API. Blíže popisována je možnost využití tzv. klientských knihoven, což je připravený soubor tříd vyvinutý ze strany Google. Klientské knihovny jsou vyvinuty pro populární programovací jazyky a kladou si za cíl usnadnit vývojáři komunikaci mezi aplikací a servery Google. Na názorných příkladech je vysvětlena práce s klientskými knihovnami pro jazyk PHP.

V části o Google Analytics API, který z hlediska implementace je podobný Google Calendar API, je popsána alternativní možnost implementace prostřednictvím knihovny nepřímo vyvinuté společností Google. V kapitole o Google Custom Search API je přiblížena možnost vytvoření vlastního vyhledávače bez použití API.

Praktická část uzavírala veškeré popisované příklady do jednotné knihovny. Cílem je, aby vývojář mohl co nejsnáze využít některé řešené příklady, aniž by musel oplývat vědomostmi o komponentech Google API. Ke knihovně byla sestavena i webová stránka, která slouží především jako dokumentace a manuálové stránky k vytvořené knihovně.

Z celkového pohledu byla práce vytvořena v souladu se zadáním. S ohledem na rozsáhlost a celkový počet nemohly být v této uvedeny všechny Google služby které mají vlastní API. Realizovaná Knihovna Google API však může být v budoucnu rozšiřována o další funkce, parametry a příklady užití. Knihovna by také měla být udržována s ohledem na časté změny ve verzích API u jednotlivých služeb.

Literatura

[1] SANTOSO, A. *Neatorama: The Evolution of Tech Companies' Logos* [online]. 7. Únor. 2008 [cit. 2013-Leden-07]. Dostupné z: http://www.neatorama.com/2008/02/07/the-evolution-of-tech-companies-logos/

[2] SVENNERBERG, G. *Beginning Google Maps API 3*. New York, NY: Expert's voice in Web development, 2010. ISBN 978-143-0228-028.

[3] GOOGLE INC. *Google Developers: The Google Geocoding API - Google Maps API Web Services* [online]., Last updated February 13, 2013 [cit. 2013-Únor-8]. Dostupné z: https://developers.google.com/maps/documentation/geocoding/

[4] MALÝ, M. *Zdroják: REST: architektura pro webové API* [online]. 3. Srpen. 2009 [cit. 2013-Březen-24]. Dostupné z: http://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/

[5] GOOGLE INC. *Google Developers: Google Accounts Authentication and Authorization* [online]., Last updated February 16, 2013 [cit. 2013-Duben-24]. Dostupné z: https://developers.google.com/accounts/docs/OAuth2

[6] CHOW, S.W. *Programujeme Mashup aplikace pro Web 2.0 v PHP*. Brno: Computer Press, 2008. ISBN 978-80-251-2057-6.

[7] CLIFTON, B. *Advanced Web Metrics with Google Analytics, 2nd Edition*. San Francisco, CA: Sybex, 2012. ISBN 978-0470562314.

[8] GOOGLE INC. *Google Developers: Configure your App - Google Apps Platform* [online]., Last updated December 20, 2012 [cit. 2013-Duben-26]. Dostupné z: https://developers.google.com/google-apps/calendar/instantiate

[9] INC., G. *Google Developers: Tutorial: Hello Analytics API* [online]., Last updated February 20, 2013 [cit. 2013-Duben-28]. Dostupné z: https://developers.google.com/analytics/solutions/articles/hello-analytics-api

[10] GOOGLE INC. *Google Developers: CSE: list - Custom Search* [online]., Last updated April 9, 2012 [cit. 2013-Duben-28]. Dostupné z: https://developers.google.com/custom-search/v1/cse/list

[11] MIHAILOVSKI, N. *Google Analytics Sample Code - Google Project Hosting: Google Analytics Easy Dashboard Javascript Library* [online]. [cit. 2013-Duben-29]. Dostupné z: http://analytics-api-samples.googlecode.com/svn/trunk/src/reporting/javascript/ez-ga-dash/docs/user-documentation.html

[12] GOOGLE INC. *Google Developers: Google Calendar API Usage Limits* [online]., Last updated March 25, 2013. [cit. 2013-Leden-7]. Dostupné z: https://developers.google.com/google-apps/calendar/pricing

[13] GOOGLE INC. *Google Developers: Google Maps JavaScript API v3* [online]., Last updated March 14, 2013 [cit. 2013-Leden-7]. Dostupné z: https://developers.google.com/maps/documentation/javascript/tutorial

[14] GOOGLE INC. *Google Developers: Overview - Custom Search* [online]., Last updated August 3, 2012 [cit. 2013-Leden-8]. Dostupné z: https://developers.google.com/custom-search/v1/overview

[15] GOOGLE INC. *Společnost – Google: Podrobná historie společnosti* [online]. 5. Leden. [cit. 2013]. Dostupné z: http://www.google.com/about/company/history/

[16] GOOGLE INC. *Google Developers: Pricing - Google Translate API* [online]., Last updated April 20, 2012 [cit. 2013-Leden-7]. Dostupné z: https://developers.google.com/translate/v2/pricing

Příloha A – Popis přílohy

V této příloze je uveden zdrojový kód funkce, která je volána v případě, že chce vývojář implementovat do vytvořené Google mapy informační bublinu znázorňující vloženou mapu s detailním náhledem na umístění zájmového bodu.

```
var map;
var markers = [];

function vlozInfoDetailMapu(position){
    var infoMapa;
    google.maps.event.addListener(markers[position-1], 'click',
function() {

        var detailDiv = document.createElement('div');
        detailDiv.style.width = '300px';
        detailDiv.style.height = '200px';
        map.getDiv().appendChild(detailDiv);
        var options = {
            zoom: 16,
            center: markers[position-1].getPosition(),
            mapTypeId: map.getMapTypeId(),
            disableDefaultUI: true
        };
        var detailMap = new google.maps.Map(detailDiv, options);
        new google.maps.Marker({
            position: markers[position-1].getPosition(),
            map: detailMap,
            clickable: false
        });
        if (!infoMapa) {
            infoMapa = new google.maps.InfoWindow();
        }
        infoMapa.setContent(detailDiv);
        infoMapa.open(map, markers[position-1]);
    });
}
```

Příloha B – CD se zdrojovými kódy a obrázky