VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

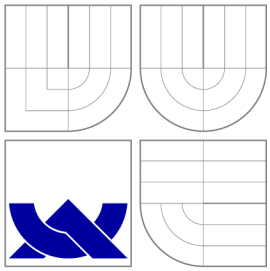
NATIVNÍ ROZHRANÍ PRO GOOGLE READER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

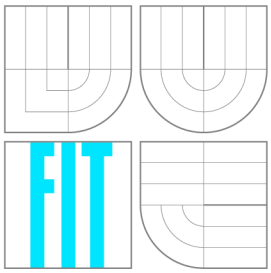
AUTOR PRÁCE
AUTHOR

TOMÁŠ POPELA

BRNO 2010



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

NATIVNÍ ROZHRANÍ PRO GOOGLE READER

NATIVE INTERFACE FOR GOOGLE READER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

TOMÁŠ POPELA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. SVATOPLUK ŠPERKA

BRNO 2010

Abstrakt

Cílem této bakalářské práce byl návrh a vývoj nativního rozhraní, které by umožňovalo odebírání syndikovaných zdrojů a položek ze služby Google Reader. Aplikace byla implementována za použití frameworku Qt a funguje na běžných operačních systémech.

Abstract

The goal of this bachelor's thesis was to design and develope native user inteface, which enables subscribing of syndicated feeds and entries from Google Reader service. Application was implemented with usage of Qt framework and it's working on common operation systems.

Klíčová slova

online syndikace, Google Reader, GUI, Qt, RSS, Atom, XML

Keywords

online syndication, Google Reader, GUI, Qt, RSS, Atom, XML

Citace

Tomáš Popela: Nativní rozhraní pro Google Reader, bakalářská práce, Brno, FIT VUT v Brně, 2010

Nativní rozhraní pro Google Reader

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Svatopluka Šperky

.....
Tomáš Popela
18. května 2010

Poděkování

Tímto bych chtěl rád poděkovat panu Ing. Svatopluku Šperkovi za jeho pomoc při realizaci této práce.

Obsah

Seznam obrázků

Seznam tabulek

Kapitola 1

Úvod

Se stránkami, které nabízí odebírání jejich obsahu pomocí standardů pro online syndikaci se setkáváme prakticky denně. Jsou to například různé zpravodajské weby, internetové obchody či blogy našich přátel.

Díky použití standardů pro online syndikaci obsahu můžeme soustředit všechny informace z oblíbených zdrojů na jednom místě. Při využití online syndikace můžeme ušetřit spoustu našeho drahoceného času. Vezměme čas, který trávíme dnes a denně procházením našich oblíbených webů a hledáním nových zpráv. Pokud jej porovnáme se spuštěním aplikace, která nám stáhne a zobrazí všechny nové zprávy z našich oblíbených webů, jednoznačně vše hovoří ve prospěch druhého způsobu získávání aktuálních informací. Nebo můžeme využít služeb jako například Google Reader, které nabízejí stejnou, ne-li lepší funkcionalitu než desktopové aplikace.

Cílem této bakalářské práce bylo implementovat nativní uživatelské rozhraní, které by umožnilo odebírání položek ze zdrojů pomocí služby Google Reader.

V následující kapitole se dozvíme o nejrozšířenějších standardech pro online syndikaci. Zmíníme jejich historii a hlavní rozdíly.

Třetí kapitola bude věnována službě Google Reader. Vysvětlíme si přístup ke službě za pomoci Google Reader API, které si následně blíže popíšeme.

V pořadí čtvrtá kapitola se bude zabývat návrhem uživatelského rozhraní aplikace gReader, která je výsledkem praktické části bakalářské práce. Popíšeme si způsoby ovládání aplikace, které do ní byly implementovány. Na závěr kapitoly si popíšeme jednotlivé prvky uživatelského rozhraní a uvedeme jeho ukázku.

Pátá kapitola bude věnována implementaci programu gReader. Budou zmíněny nástroje a technologie použité při vývoji aplikace a popsány třídy, které byly implementovány. Pokud je ve třídě zmíněna komunikace se službou Google Reader, je odesílaný požadavek detailně popsán a vysvětlen. Dále si zmíníme o testování výsledné aplikace.

Následující, v pořadí již šestá kapitola, bude věnována zamyšlení nad možnostmi dalšího vývoje aplikace.

Sedmá a závěrečná kapitola obsahuje závěr.

Kapitola 2

Online syndikace obsahu

Při procházení internetových stránek můžeme narazit na ikonu, která je zobrazena na obrázku 2.1. Ikona nám prozrazuje, že navštívená stránka nabízí svůj obsah k odebírání pomocí standardů pro online syndikaci. Obsah můžeme odebírat pomocí desktopových aplikací (v současnosti nejčastěji pomocí internetových prohlížečů) či pomocí specializovaných webů jako například Google Reader. Mezi nejznámější a nejpoužívanější standardy patří RSS či Atom, které si blíže popíšeme v následující kapitole.

Obrázek 2.1: Ikona označující možnost odebírání obsahu [1]

2.1 XML

Extensible Markup Language neboli XML je značkovací jazyk, který slouží k popisu a následnému uchování či výměně dat. Samotný jazyk vychází ze značkovacího jazyka SGML[1], z něhož také vychází například jazyk pro tvorbu webových stránek HTML. Vývoj začal roku 1996 na popud konsorcia W3C[2]. O dva roky později v roce 1998 byla schválena verze 1.0 [7].

Při popisu dat se využívá značek, které ale nejsou předdefinované a jejich tvorba je přenechána programátorovi. Jak vidíme na níže uvedeném příkladu 2.1, tak každý tag neboli značka má k sobě párový ukončovací tag (`<jmeno></jmeno>`). Mezi těmito tagy jsou uchována data jimi definovaného typu. XML elementem nazýváme vše od počátečního tagu, až po jeho ukončovací tag (`<jmeno>Petr</jmeno`). Může jim být také prázdný tag, který nemá počáteční tag, ale zapisuje se pouze ukončovacím (`</jmeno>`). Dále každý element může obsahovat atributy, které mohou dále obsahovat metadata a jejich příklad můžeme nalézt v elementu `telefon` [15].

[1] Univerzální značkovaní metajazyk, jehož rozšíření zabránila jeho složitost.
[2] Mezinárodní instituce pro správu standardů webu.

```
<?xml version="1.0"encoding="utf-8"? >
<vizitka>
    <jmeno>Petr</jmeno>
    <prijmeni>Novák</prijmeni>
    <adresa>
        <ulice>Skácelova</ulice>
        <cp>1140</cp>
        <mesto>Brno</mesto>
        <psc>61600</psc>
    </adresa>
    <telefon typ="mobilni">123456789</telefon>
</vizitka>
```

Tabulka 2.1: Příklad kontaktu popsaného jazykem XML

Jazyk XML nachází své využití prakticky kdekoliv, ale nejčastěji je spojován s internetovými službami. Slouží například pro popis syndikovaných zdrojů ve standardech RSS a Atom nebo z něj vychází jazyky jako XHTML. V praktické části této bakalářské práce je XML využito pro ukládání nastavení aplikace.

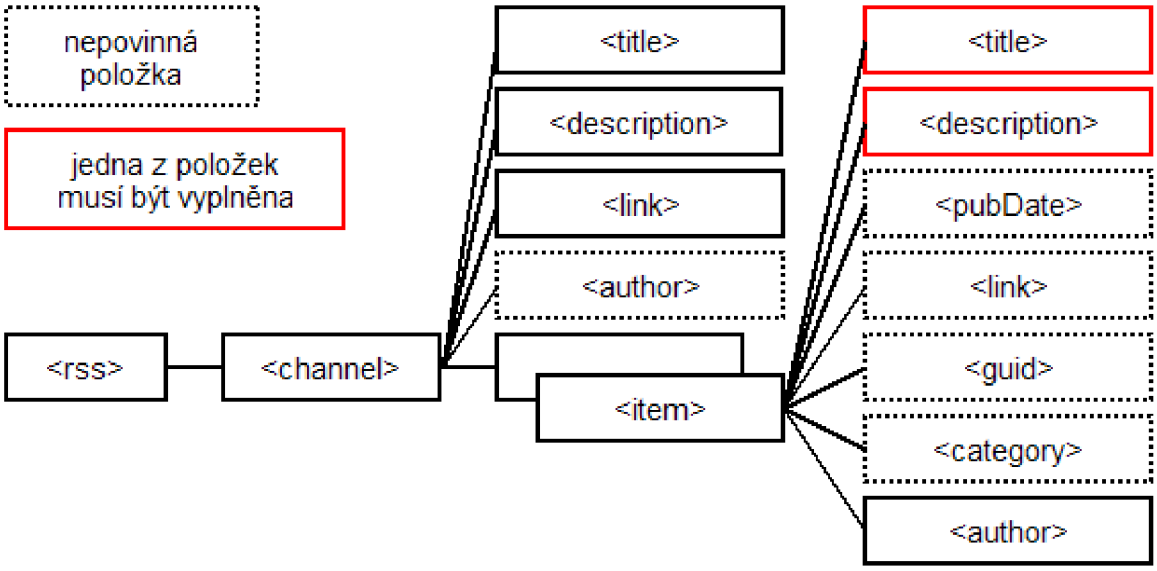
2.2 RSS a ATOM standard

Mezi nejpoužívanější standardy pro online syndikaci obsahu se řadí standardy RSS a Atom. Vývoj RSS (Really Simple Syndication či Rich Site Summary [6]) počal v roce 1999, kdy firma Netscape navrhla jeho první verzi s označením 0.90. Nedlouho poté její zaměstnanec Dan Libby navrhl vylepšenou verzi 0.91. Ve stejnou dobu navrhl vylepšení verze 0.90 také Dawe Winer ze společnosti UserLand a vydal je pod stejným označením 0.91. Byly tedy uvolněny dvě různé verze protokolu, avšak obě se stejným označením. Takto vznikl jev, který je RSS často vytýkán a to neuspořádaný vývoj s mnoha navzájem nekompatibilními verzemi [3].

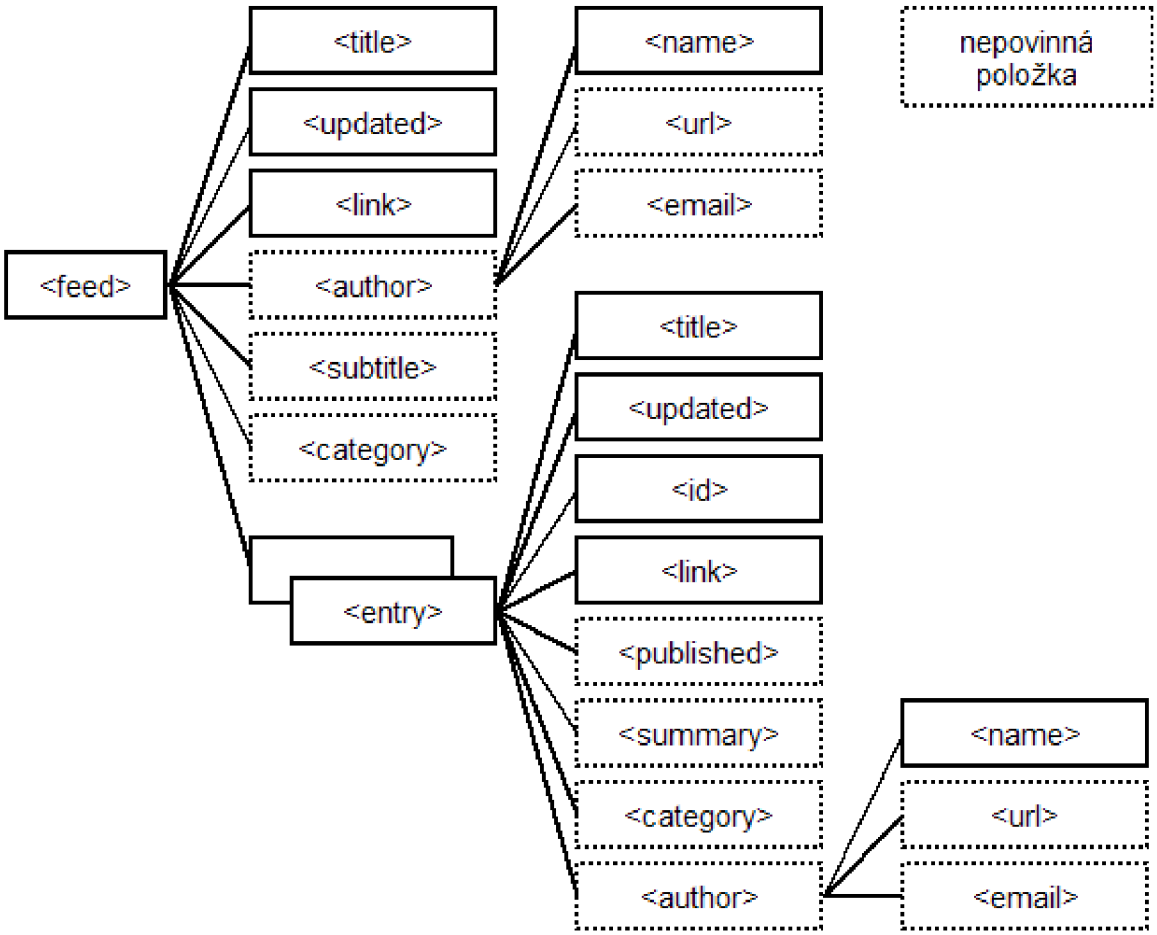
Proto jako alternativa vznikl roku 2005 standard Atom definovaný v RFC 4287 [16]. Atom se liší od RSS v několika aspektech a to například v podpoře pro přenášená data. Standard RSS umožňuje přenos dat ve formátu plain text a HTML. Atom přidává podporu pro XHTML, XML a binární data ve formátu base64. Dále přináší širší možnosti pro popis autorů jednotlivých syndikovaných položek a rozšíření počtu elementů přes jmenné prostory [17].

Na obrázcích 2.2 a 2.3 jsou uvedeny příklady elementů standardů RSS 2.0 a Atom používaných pro syndikaci obsahu. Jedná se o dokumenty ve formátu XML. Můžeme si všimnout rozšíření elementu `author` ve standardu Atom o nové prvky pro jeho popis, oproti RSS. Dále musí být v RSS syndikované zdroje a položky uzavřeny ve speciálním kontajneru `rss`. Standard Atom přidává element `updated`, který udává čas poslední modifikace položky.

Specifikace standardu RSS je narozdíl od Atomu uzavřená a přidávání nových význačných změn není umožněno. Prává na licenci RSS specifikace vlastní Hardvardská univerzita [5]. Od verze 0.3 používá standard Atom firma Google ve svém Google Data Protokolu.



Obrázek 2.2: Schéma elementů standardu RSS 2.0[11]

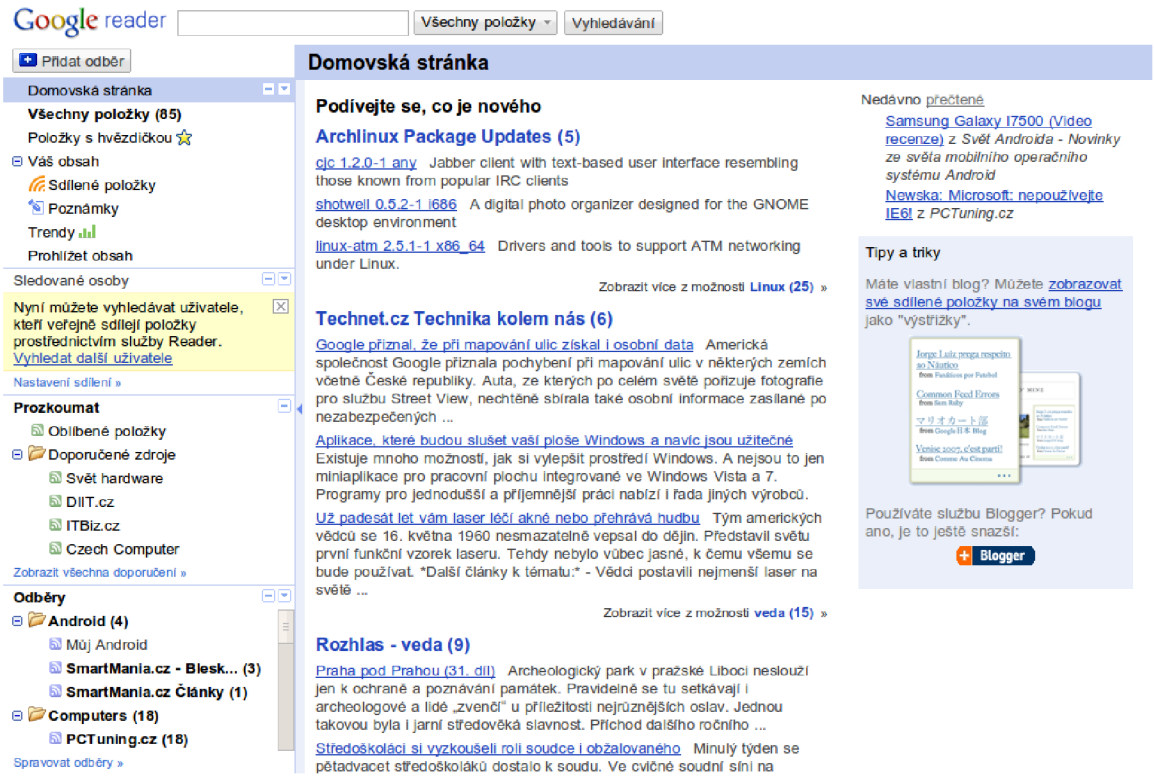


Obrázek 2.3: Schéma elementů standardu Atom[11]

Kapitola 3

Google Reader

Google Reader dostupný na adrese `www.google.com/reader/` je jednou z širokého portfolia služeb, které tato význačná firma nabízí. Oproti jiným webovým aplikacím firmy Google (Google Mail, Google Kalendář) není mezi běžnými uživateli sítě Internet tak rozšířena. Dle mého názoru je v pozadí neprávem, přestože nabízí možnost odebírat obsah webů a celý jej soustředit na jednom místě. A to celé bez závislosti na operačním systému uživatele, kterému stačí k přístupu pouhý internetový prohlížeč.



Obrázek 3.1: Google Reader

Služba byla spuštěna v testovacím provozu 7. října 2005 a po dvou letech vývoje 7. září 2007 jí byl odebrán status „beta".

Google Reader nabízí možnost odebírání zdrojů z internetových serverů ve formátu RSS a Atom. Zdroje můžeme organizovat pomocí štítků a za jejich pomoci je umísťovat do

různých složek. Zajímavou funkcí je zajisté také nabízení nových zdrojů odběru, podle již odebíraných, kdy služba předpokládá, že by mohly navrhované zdroje uživatele zaujmout. Dále může Google Reader sloužit i jako sociální síť, kdy můžeme ostatním uživatelům doporučovat odebírané zdroje či sdílet položky mezi určitou skupinou uživatelů.

3.1 Hypertext Transfer Protocol

Hypertext Transfer Protocol nebo zkráceně HTTP protokol slouží pro komunikaci mezi dvěma subjekty v síti Internet, zpravidla mezi serverem a klientem (například internetový prohlížeč). Protokol je implementován na aplikační vrstvě modelu ISO/OSI a při přenosu využívá transportní protokol TCP. HTTP se používá již od roku 1990 a v dnešní době narazíme nejčastěji na verzi 1.0 [8] a 1.1 [10]. Protokol pracuje na principu dotaz/odpověď, kdy se klient dotazuje serveru, který mu následně odpovídá. Protokol HTTP je bezstavový a tudíž si neudržuje informace o již provedených spojeních [9].

HTTP protokol obsahuje několik metod určených pro komunikaci se serverem.

- GET – Nejpoužívanější metoda, která slouží pro získání objektů ze serveru. Můžeme k němu také připojit data pokud na konec cílové adresy přidáme znak „?“ a za něj vložíme jednotlivá data oddělená znakem „&“.
- POST – Používáno nejčastěji pro nahrávání dat na server. Data jsou vložena do těla dotazu a tím jsou narozdíl od metody GET před běžným uživatelem skryta, což je zajisté lepší i z hlediska bezpečnosti.
- HEAD – Obdobná metoda jako GET, ale v odpovědi od serveru se nepřenáší tělo dotazu. Dotaz je používán pro ověření, zda je požadovaný objekt na stránce opravdu dostupný.
- PUT – Vytvoří objekt nový objekt na serveru.
- DELETE – Smaže objekt ze serveru.
- OPTIONS – Slouží pro získání informací o metodách HTTP protokolu, které server podporuje.
- TRACE – Slouží ke sledování cesty odeslaného dotazu. V odpovědi jsou navrácené dotazy jednotlivých systému, kterými požadavek procházel.
- CONNECT – Slouží k tunelování HTTP protokolu (například v protokolu SSL) [18].

3.2 Google Reader API

Google Reader API na rozdíl od ostatních služeb firmy Google zatím nebylo uveřejněno, přestože vysocí představitelé firmy uveřejnění několikrát přislíbili [14]. Existují však projekty, které přístup ke službě Google Reader využívají a svoje získané poznatky uveřejnili. Mezi ně patří například aplikace Pyrfeed [12], ze které jsou čerpány níže uvedené informace.

Celé API je odvozeno od Google Data protokolu, který využívají například webové aplikace Google Mail či Google Kalendář. Jedná se o komunikaci klient/server, kdy klient serveru odesílá dotazy protokolu HTTP, na které server následně navrací odpověď. Klient může specifikovat v jakém formátu mu bude server vracet odpovědi. Může volit mezi formátem JSON či Atom.

API služby Google Reader je rozděleno do tří vrstev. První vrstva je zodpovědná za získávání informací z jednotlivých zdrojů. Tato vrstva je kompatibilní se všemi dosud uveřejněnými verzemi standardů RSS a Atom. Na výstupu této vrstvy používají všechny zdroje standard Atom. Pro přístup k informacím této vrstvy se využívá adresa `http://www.google.com/reader/atom/`. Tato vrstva slouží k získání jednotlivých zdrojů či položek. Jsou zde uchovávány všechny zdroje s položkami, které služba Google Reader nabízí k odebírání. Můžeme zde například získat položky z učitého zdroje při zaslání požadavku, který je uveden v následující tabulce 3.1.

Cílová adresa:	http://www.google.com/reader/atom/feed/`parametr_1`
Příklad:	http://www.google.com/reader/atom/feed/http://www.archlinux.org/feeds/
`parametr_1`	internetová adresa požadovaného zdroje

Tabulka 3.1: Požadavek získání položek z určitého zdroje

Tento požadavek není vázán na uživatelský účet, a proto jej můžeme provést i bez přihlášení ke službě Google Reader. Dalším požadavkem můžeme získat položky ze zdrojů označených společným štítkem. Použitý dotaz ukazuje tabulka 3.2.

Cílová adresa:	http://www.google.com/reader/atom/user/`parametr_1`/label/`parametr_2`
Příklad:	http://www.google.com/reader/atom/user/-/label/Linux
`parametr_1`	identifikace uživatele
`parametr_2`	název štítku

Tabulka 3.2: Požadavek získání položek označených určitým štítkem

Identifikací uživatele se zde myslí dvacetimístné číslo, kterým je reprezentován uživatel služby Google Reader. Toto číslo si však nemusíme pamatovat, ale vložíme místo něj znak „–“ a dotaz se bude vztahovat k právě přihlášenému uživateli. Pokud chceme například získat seznam všech přečtených položek nebo seznam odebíraných zdrojů vložíme do výše uvedeného požadavku za název štítku `read` respektive `reading-list`.

Druhá vrstva slouží jako databáze. Jsou zde uloženy všechny přijaté zdroje s položkami a informacemi o nich (zda jsou označené jako přečtené či mají přiřazený štítek). Pro přístup se využívá adresy pro první vrstvu a navíc také adresa `http://www.google.com/reader/api/0/`. Tato vrstva se dále dělí na dvě podvrstvy. Jednou je vrstva, jejímž účelem je upravovat seznam odebíraných zdrojů. Operace prováděné touto vrstvou jsou například přidání nového zdroje do seznamu odebíraných zdrojů či jeho odebrání. Dále přidání nebo odebrání štítku zdroje. Funkcí následující a zároveň poslední podvrstvy je získání seznamu všech štítků (`tag/list`), odebíraných zdrojů (`subscription/list`) a počtu nepřečtených položek pro jednotlivé zdroje a štítky(`unread-count`). Informace uvedené v závarokách připojíme k adrese pro přístup k vrstvě. Parametrem `preference/list` získáme seznam nastavení vztahujících se ke službě Google Reader. Formát dotazů pro jednotlivé operace je uveden ve třídě FeedNetworking – kapitola 5.2.4.

Poslední vrstva zajišťuje přístup k uživatelskému rozhraní. Pro přístup jsou využity adresy `http://www.google.com/reader/view/` a `http://www.google.com/reader/settings/`. V praktické části bakalářské práce je využit přístup k prvním dvou vrstvám API.

Kapitola 4

Návrh uživatelského rozhraní

Při tvorbě a návrhu uživatelského rozhraní aplikace gReader jsem vycházel z několika východisek. Hlavním východiskem pro mě bylo perfektní ovládání za pomoci klávesnice a tím i minimalizovat použití polohovacích zařízení jako myš či touchpad. Ty shledávám při používání značně neefektivní. Představme si situaci, ve které máme dokument, na jehož levé straně je jeho obsah. Pokud chceme přejít na další kapitolu a nacházíme se na začátku předchozí, zbývají nám tři možnosti jak toho docílit. První je posouvat kolečkem myši do té doby, než dojdeme na stránku s požadovanou kapitolou. Další možností je přejet kurzorem z jeho aktuální polohy na panel s obsahem a zde zvolit požadovanou kapitolu. Jelikož při práci s počítačem máme většinu času obě ruce na klávesnici, je velmi neefektivní z ní ruce sundávat a používat myš či touchpad. Proto se nám nabízí ještě poslední možnost. Použití klávesových zkratek. Pro přechod na další položku nám v případě, že to program umožňuje stačí pouhé stisknutí zpravidla dvou kláves. Podle mého názoru je tento způsob ovládání programu velmi efektivní. Dále jsem se snažil minimalizovat použití polohovacích zařízení i při takových úkonech, jako přidání nového zdroje obsahu či změnu nastavení aplikace. Jakým způsobem je vyřešen tento problém, bude vysvětleno v následující kapitole. Pro zachování přehlednosti a jisté funkcionality spojené s výše uvedeným typem ovládání, jsem volil přehledné a v jistém slova smyslu minimalistické rozvržení prvků aplikace. Dle mého názoru musí mít aplikace toho typu jednoduché, přehledné uživatelské rozhraní s co největší možnou zobrazovací plochou pro jednotlivé položky. Výsledek je uveden na obrázku 4.1 a bude popsán v kapitole 4.1.2.

4.1 Ovládání aplikace

Při návrhu ovládání jsem vycházel z předpokladů uvedených výše. V následujících dvou bodech si jednotlivé metody ovládání přiblížíme.

4.1.1 Režim příkazů pro ovládání programu

Tento režim vychází z textového editoru VIM[1], který je znám svým promyšleným systémem příkazů a kompletním využitím klávesnice pro jeho ovládání. Pro aktivaci režimu zadávání příkazů je třeba v editoru VIM stisknout klávesu „:“. Nyní můžeme zadávat jednotlivé příkazy, které se provedou po stisknutí klávesy Enter. Stejný systém jsem zakomponoval i do své aplikace. Po aktivování režimu pro zadávání příkazů (také klávesou „:“) se ve spodní

[1] Viz http://www.vim.org

části hlavního okna programu zobrazí pole pro zadávání příkazů. Přes příkazy můžeme měnit nastavení aplikace (například změna uživatelského jména) či procházet mezi jednotlivými položkami, zdroji nebo štítky. Program má také historii příkazů, kterou vyvoláme po stisknutí kláves nahoru či dolů při aktivovaném poli pro zadání příkazů.

Příkaz	Funkce
a, add	Zobrazí dialog pro přidání nového zdroje
a adresa_zdroje název štítek add adresa_zdroje název štítek	Přidá zdroj se zadanou adresou, názvem a štítkem
c, check	Aktualizuje seznam zdrojů a položek
m, mark	Označí zdroj/štítek jako přečtený
mi, minimize	Minimalizuje aplikaci
n, next	Přejde na následující položku
nf	Přejde na následující zdroj
nl	Přejde na následující štítek
o, options	Zobrazí dialog s nastavením programu
p, prev	Přejde na předchozí položku
pf	Přejde na předchozí zdroj
pl	Přejde na předchozí štítek
q, wq, quit	Ukončí aplikaci
rm, remove	Zobrazí dialog pro odebrání zdrojů
rm this remove this	Odstraní zdroj, který je aktuálně zvolen
rm label remove label	Odstraní štítek zvolené položky
set autohide on/off	Zapne/vypne skrývání seznamu zdrojů
set password hodnota	Nastaví heslo k Google Reader účtu
set time hodnota	Nastaví čas automatické aktualizace
set username hodnota	Nastaví uživatelské jméno aktivního účtu
set toolbar on/off	Zapne/vypne lištu nástrojů
set menu on/off	Zapne/vypne menu
tt	Zobrazí/skryje lištu nástrojů
tm	Zobrazí/skryje menu aplikace

Tabulka 4.1: Přehled příkazů aplikace gReader

4.1.2 Klávesové zkratky

Další z možností jak ovládat aplikaci gReader bez nutnosti používat myš nebo touchpad jsou klávesové zkratky. V aplikaci jsou již klávesové zkratky přednastaveny, ale uživateli je ponechána možnost jejich změny. Jejich hlavní využití je dle mého názoru především při procházení mezi jednotlivými položkami/zdroji či štítky. Pokud chceme například při skrytém panelu se seznamem zdrojů přejít na následující či předchozí zdroj/štítek nemusíme jej znovu zobrazovat, ale můžeme jednoduše přejít pomocí klávesových zkratek. Některé z klávesových zkratek jako například „ZZ“ pro ukončení programu či „/“ pro vyhledávání jsou převzaty z editoru VIM.

Klávesa	Funkce
Ctrl + .	Přejde na následující položku
Ctrl + ,	Přejde na předchozí položku
Ctrl + §, Ctrl + ’	Přejde na následující zdroj
Ctrl + ;, Ctrl + ů	Přejde na předchozí zdroj
Ctrl +], Ctrl +)	Přejde na následující štítek
Ctrl + [, Ctrl + ú	Přejde na předchozí štítek
Ctrl + \	Zobrazí/skryje seznam zdrojů
Ctrl + N	Zobrazí dialog pro přidání nového zdroje
Ctrl + R	Zobrazí dialog pro odebrání zdrojů
Ctrl + M	Označí zdroj/štítek jako přečtený
Ctrl + P	Zobrazí dialog s nastavením programu
Ctrl + F, /	Zobrazí pole pro vyhledávání položek
ZZ	Ukončí aplikaci
F5	Aktualizuje seznam zdrojů a položek

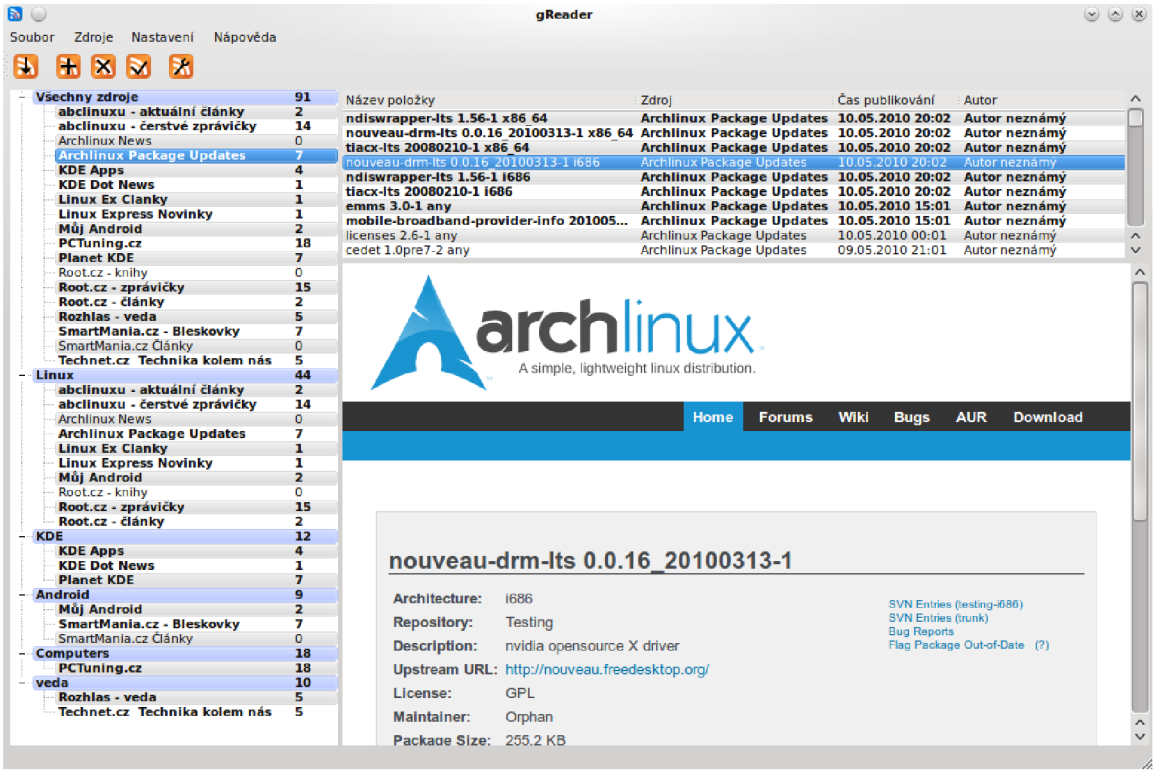
Tabulka 4.2: Přehled výchozích klávesových zkratek aplikace gReader

Použitím kombinace dvou výše zmíněných metod dosáhneme dle mého názoru nejefektivnějšího způsobu ovládání aplikace. Pro pohyb mezi položkami, zdroji či štítky využijeme klávesové zkratky, naopak pro změnu nastavení či rychlého přidání/odebrání zdrojů využijeme příkazový řádek.

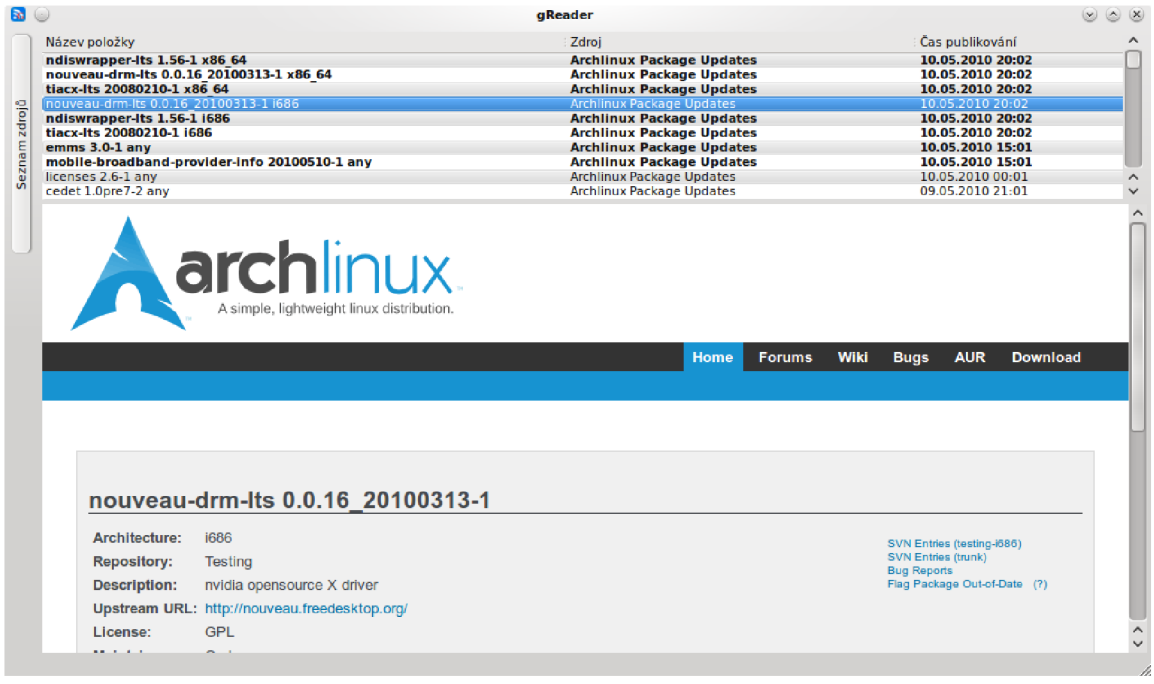
4.2 Rozložení a popis prvků uživatelského rozhraní

Hlavnímu oknu aplikace dominuje prvek, který slouží jako prohlížeč obsahu položek. Nad ním se nachází panel se seznamem jednotlivých zpráv. Pro každou položku se zobrazuje její název, název zdroje ze kterého pochází, datum publikování a jméno autora. V levé části okna se nalézá panel se seznamem všech odebíraných zdrojů. Tento prvek je implementován ve formě stromu, kdy nadřazenými prvky jsou všechny přiřazené štítky a do nich vnořené jednotlivé zdroje, které jim náleží. Dále jsou zde uvedeny počty nepřečtených položek z jednotlivých zdrojů. Pro lepší orientaci mezi položkami a zdroji jsou v panelech barevně odlišeny sudé a liché řádky. Pokud je položka označena jako nepřečtená nebo zdroj obsahuje nepřečtené položky, je tučně zvýrazněna. V hlavním okně se dále nachází panel s lištou nástrojů, kde jsou položky s ikonami pro funkce jako aktualizovat, přidat či odebrat zdroje, označit jako přečtené a nastavení. Dále aplikace obsahuje menu a ikonu v systémovém panelu. Pro zvětšení prostoru vyhrazeného pro prohlížeč se dají jak menu, tak panel s lištou nástrojů skrýt.

Toto rozložení prvků hlavního okna aplikace bohužel není vhodné pro zařízení s nízkým rozlišením displeje. Při zachování tohoto rozmístění budou rozměry prostoru vyhrazeného pro zobrazení obsahu položky velmi malé. Z tohoto důvodu je implementována možnost automatického skrývání panelu se zdroji. Po zvolení položky se panel skryje a na levé straně se místo něj zobrazí tlačítko. Po stisknutí tohoto tlačítka či klávesové zkratky, která je uvedena dále v textu, se panel se seznamem zdrojů a štítků opět zobrazí.



Obrázek 4.1: Hlavní okno aplikace gReader



Obrázek 4.2: Minimalistické zobrazení aplikace gReader

Kapitola 5

Implementace – aplikace gReader

V následující kapitole je popsán vývoj aplikace gReader, jakožto nativního rozhraní pro službu Google Reader.

5.1 Použité technologie a nástroje

Před samotnou implementací aplikace jsem se musel rozhodnout jaký jazyk či framework využiji. Moje volba padla bez váhání na framework Qt, který při dodržení určitých pravidel zaručoval, že výsledná aplikace bude multiplatformní. Výběr zajisté ovlivnily také mé sympatie k jazykům C a C++. Ze získaných zkušeností z předchozích projektů, které jsem zpracovával během studia, se jevilo jako nutnost použít systém pro správu verzí zdrojového kódu. Doposud jsem používal pouze systém SVN a již dlouho jsem chtěl vyzkoušet jeho alternativu, kterou je systém Git. Proto padla volba na tento systém.

5.1.1 Qt

Qt je multiplatformní framework pro tvorbu aplikací s grafickým či konzolovým rozhraním. Vývoj začal již roku 1991 společností Quasar Technologies, později známou jako Trolltech [13]. V roce 2008 firmu odkoupila společnost Nokia, která se nyní stará o stále probíhající vývoj. Qt je oficiálně vydáváno pro všechny nejrozšířenější platformy na trhu jmenovitě například pro Linux/Unix, Microsoft Windows či MAC OS X [4]. Samotné Qt může být využíváno i s jinými jazyky než C++ jehož je nadstavbou, ale také například jazyky Python (PyQt) či Java (Qt Jambi).

Qt obsahuje například multimedální API Phonon či QWebKit jako implementaci jádra internetových prohlížečů WebKit[1]. Qt využívá řada známých produktů jako Video Lan Client (VLC)[2], Google Earth[3] či například VirtualBox[4]. Komplexnost tohoto frameworku dokazuje například multiplatformní desktopové prostředí KDE[5], které je celé napsáno právě v Qt.

Při vývoji aplikace gReader bylo použito Qt SDK[6] ve verzi 4.6.2, která je aktuální k datu 11. května 2010. Sada nástrojů Qt SDK obsahuje vývojové prostředí QtCreator nebo

[1] Viz http://webkit.org
[2] Viz http://www.videolan.org/vlc/
[3] Viz http://earth.google.com
[4] Viz http://www.virtualbox.org/
[5] Viz http://www.kde.org
[6] Viz http://qt.nokia.com/products

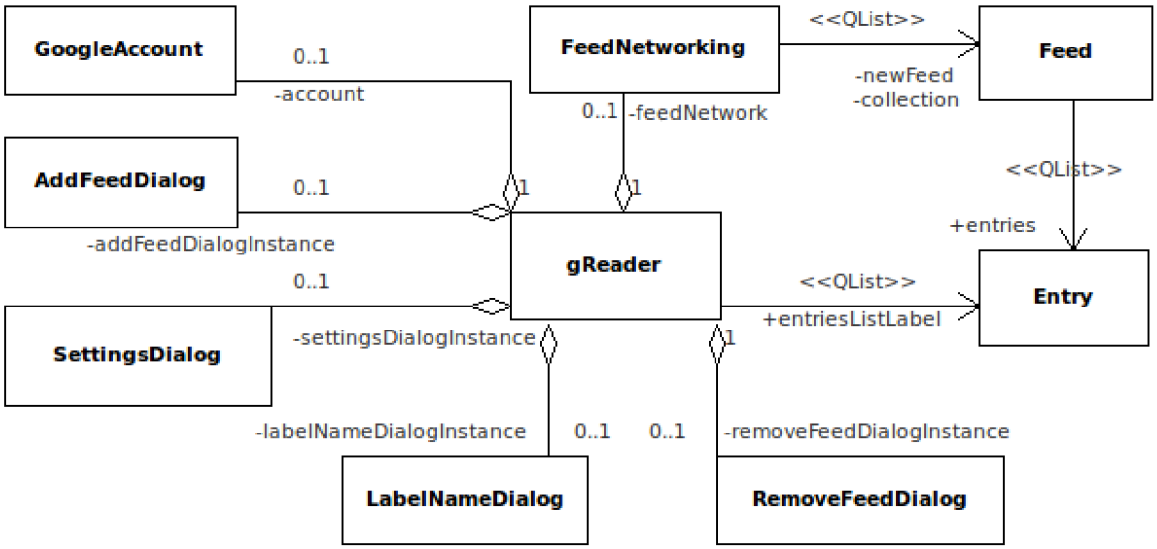
například aplikaci QtDesigner, který slouží pro návrh uživatelského rozhraní. Do aplikace QtCreator je navíc integrována kompletní verze dokumentace jazyka Qt, která je dle mého názoru zpracována na velmi vysoké úrovni a patří mezi přednosti tohoto jazyka.

5.1.2 Git

Při vývoji programu gReader bylo využito systému Git[7], který slouží pro rychlou a efektivní správu verzí zdrojových kódů aplikace. Systém Git byl vyvinut Linusem Torvaldsem a jeho hlavní úlohou měla být správa vývoje jádra systému Linux. Systém se ale stal oblíbeným a rozšířil se dále mezi uživatele. Git využívá při svém vývoji řada známých projektů jako například již zmíněné jádro systému Linux, desktopové prostředí GNOME či linuxová distribuce Fedora [2]. Pro vývoj toho projektu bylo využito repozitáře, který po registraci nabízí server Gitorious[8].

5.2 Struktura aplikace

Zdrojový kód aplikace gReader se skládá ze tříd, jejichž vztah je ukázán v UML diagramu 5.1.



Obrázek 5.1: UML diagram aplikace gReader

V následujících sekcích výše uvedené třídy blíže popíši.

5.2.1 Třída GoogleLogin

Třída `GoogleLogin` představuje účet ke službě Google Reader. Obsahuje metody pro přihlášení a navrácení získaných informací o přihlášení.

Přihlášení probíhá ve dvou krocích. V prvním kroku odešleme přihlašovací informace zadané uživatelem pomocí HTTP dotazu, ke kterému připojíme metodou POST uživatelské jméno a heslo a celek následně odešleme na specifikovanou adresu.

[7] Viz http://git-scm.com
[8] Viz http://gitorious.org

Cílová adresa:	https://www.google.com/accounts/ClientLogin
POST data:	service=reader&Email=`parametr_1`&Passwd=`parametr_2` &Source=`parametr_3`
`parametr_1`	uživatelské jméno ke službě Google Reader
`parametr_2`	heslo k zadanému uživatelskému jménu
`parametr_3`	identifikace aplikace

Tabulka 5.1: Dotaz pro získání Session ID

Po odeslání výše uvedeného dotazu nám server služby Google Reader navrátí několik údajů ve tvaru *klíč* = *hodnota*, ze kterých si uložíme pouze údaj označený jako SID (Session ID). SID je textový řetězec, který identifikuje přihlášení uživatele a jeho platnost vyprší po jeho odhlášení.

Vytvoříme si HTTP cookie a vložíme do ní získané SID. Dále nastavíme doménu a cestu, čímž vznikne výsledná cookie, která je uvedena v tabulce 5.2. Tuto cookie budeme potřebovat i dále v aplikaci, kdy jí budeme přikládat ke každému požadavku odesílaného službě Google Reader.

Proměnná	Hodnota
SID	hodnota SID
Domain (doména)	.google.com
Path (cesta)	/

Tabulka 5.2: Cookie potřebná pro získání tokenu

Nyní vytvoříme HTTP dotaz do jehož hlavičky vložíme výše specifikovanou cookie a celek odešleme na adresu `http://www.google.com/reader/api/0/token`. Odpověď od serveru bude obsahovat textový řetězec o délce 22 znaků, který identifikuje probíhající relaci. Jeho platnost je narozdíl od SID časově omezena. Tento řetězec dále uváděný jako „token“ se přikládá k dotazům, ke kterým budeme přidávat data metodou POST.

5.2.2 Třída Entry

Třída `Entry` implementuje samotnou položku v odebíraném zdroji. Pro každou položku se ukládá její identifikační řetězec, název, autor a datum publikování. Dále pak internetová adresa, na které byla položka původně publikována. Jako poslední informace o položce se ukládá příznak, jestli zda položka již přečtena (tzn. je jí přiřazen štítek `read`) nebo nikoliv.

5.2.3 Třída Feed

Třída reprezentující zdroj položek. Každý zdroj je identifikován svým jedinečným identifikačním řetězcem. Dále třída uchovává jeho název a názvy štítků, které jsou zdroji přiřazeny. Třída `Feed` obsahuje seznam objektů třídy `Entry`, který reprezentuje položky náležící do uvedeného zdroje. Ukládá se také počet nepřečtených položek ve zdroji, čas nejnovější položky a čas první položky ve zdroji, oba dva údaje v unixové reprezentaci času.

5.2.4 Třída FeedNetworking

V této třídě je implementována komunikace se službou Google Reader. Skládá se z několika dílčích metod, ve kterých jsou zasílány HTTP dotazy v daném formátu. Proto si jednotlivé metody blíže popíšeme a uvedeme podrobnosti o specifickém formátu jednotlivých dotazů. Ke všem požadavkům musíme také připojit cookie uvedenou v tabulce 5.2. Formát níže uvedených dotazů pochází z uveřejněného API projektu Pyrfeed [12].

Metoda getFeeds

Cílem této metody je získat seznam odebíraných zdrojů právě příhlášeného uživatele ke službě Google Reader. Pro jeho získání odešleme HTTP požadavek na adresu `http://www.google.com/reader/api/0/subscription/list?output=xml` a do hlavičky dotazu přidáme cookie vytvořenou při přihlášení.

Od serveru následně obdržíme odpověď, která bude obsahovat data ve formátu XML. Příklad získaných dat reprezentujících odebíraný zdroj můžeme vidět v tabulce 5.3.

```
<object>
    <list name="subscriptions">
        <object>
            <string name="id">feed/Adresa zdroje</string>
            <string name="title">Název zdroje</string>
                <object>
                    <string name="id">/user/-/label/štítek</string>
                    <string name="label">štítek</string>
                </object>
            </list>
            <string name="sortid">hodnota</string>
            <string name="firstitemmsec">čas první položky</string>
        <object>
    </list>
</object>
```

Tabulka 5.3: Data ve formátu XML obsahující popis zdroje

V navráceném souboru se nalézají nejen odebírané zdroje, ale také i seznam uživatelem vytvořených štítků. Pro každý takto získaný zdroj vytvoříme instanci třídy `Feed` a naplníme ji získanými informacemi.

Metoda getUnreadFeeds

Cílem této metody je získat počet nepřečtených položek v jednotlivých zdrojích, které uživatel odebírá.

Cílová adresa:	http://www.google.com/reader/api/0/subscription/edit
GET data:	client=`parametr_1`&allcomments=true&output=`parametr_2` &ck=`parametr_3`
Příklad:	client=gReader&allcomments=true&output=xml&ck=1273950979
`parametr_1`	identifikace aplikace
`parametr_2`	formát navrácených dat – JSON nebo XML
`parametr_3`	čas odeslání požadavku v unixovém formátu

Tabulka 5.4: Požadavek pro získání počtu nepřečtených položek ve zdrojích

Pokud zvolíme jako formát odpovědi XML, bude příchozí odpověď ve tvaru jaký ukazuje tabulka 5.5.

```
<object>
    <number name="max">1000</number>
    <list name="unreadcounts">
        <object>
            <string name="id">feed/adresa zdroje</string>
            <string name="count">počet nepřečtených položek</string>
            <string name="newestItemTimestampUsec">
                čas nejnovější položky v unixovém formátu
            </string>
        <object>
    </list>
</object>
```

Tabulka 5.5: Zdroj s nepřečtenými položkami

Metoda nám navrátí nejen počet nepřečtených položek ve zdrojích, ale také ve štítkách. Nově získané informace o zdrojích přiřadíme k objektům, které jsme vytvořili v metodě `getUnreadFeeds`.

Metoda getEnries

Cílem této metody je získat seznam položek ve všech zdrojích. Maximální počet je 1000 položek v jedné odpovědi. Pokud chceme přijmout více položek musíme z odpovědi získat hodnotu tagu <`gr:continuation`> a připojit ji k novému požadavku jako hodnotu parametru `c`. Pokud k dotazu nepřipojíme parametr s počtem položek, které budeme chtít navrátit, je nastaven výchozí počet navrácených položek na dvacet.

Cílová adresa:	http://ww.google.com/reader/atom/user/`parametr_1`/state/com.google/reading-list?n=`parametr_2`
Příklad:	http://ww.google.com/reader/atom/user/-/state/com.google/reading-list?n=500
`parametr_1`	identifikace uživatele (znak „–“ pro právě přihlášeného uživatele)
`parametr_2`	počet položek, které chceme navrátit

Tabulka 5.6: Požadavek pro získání definovaného počtu položek ze zdroje

Po odeslání tohoto požadavku nám server služby Google Reader navrátí definovaný počet položek, kde každá položka bude ve tvaru jaký uvádí tabulka 5.7. Položky jsou navráceny seřazené podle času od nejnovější po nejstarší.

```
<entry gr:crawl-timestamp-msec="čas položky">
    <id gr:original-id="adresa položky">
        id položky - tag:google.com,2005:reader/item/5d94ab2d0dcf9204
    </id>
    <category term="user/-/state/com.google/reading-list"
        scheme="http://www.google.com/reader"label="reading-list"/>
    <category term="user/-/label/název přiřazeného štítku"
        scheme="http://www.google.com/reader"label="reading-list"/>
    <title type="html">titulek položky</title>
    <published>datum publikovaní - 2010-05-10T20:02:30Z</published>
    <link rel="alternate"href="adresa položky"type="text/html">
    <summary xml:base="adresa zdroje"type="html»
        stručné shrnutí položky
    </summary>
    <author>
        <name>jméno autora</name>
    </author>
    <source gr:stream-id="feed/adresa zdroje">
        <id>
            id zdroje - tag:google.com,2005:reader/feed/adresa zdroje
        </id>
        <title type="html">titulek zdroje</title>
        <link rel="alternate"href="adresa zdroje"type="text/html"/ >
    </source>
</entry>
```

Tabulka 5.7: Položka navrácená ze serveru ve formátu XML

Pro každou takto získanou položku vytvoříme objekt třídy `Entry`, naplníme jej získanými informacemi a vložíme do odpovídajícího zdroje.

Metoda getAllEntriesFromFeed

Použitím této metody získáme seznam zvoleného počtu položek v daném zdroji.

Cílová adresa:	http://www.google.com/reader/atom/feed/`parametr_1`?n=`parametr_2`
Příklad:	http://www.google.com/reader/atom/feed/http://www.archlinux.org/feeds/news/?n=50
`parametr_1`	id zdroje, ze kterého chceme získat položky
`parametr_2`	počet položek, které chceme navrátit

Tabulka 5.8: Požadavek pro získání definovaného počtu položek ze zdroje

V odpovědi od serveru získáme seznam položek ve tvaru uvedeném v tabulce 5.7.

Metoda markEntryAsRead

Tato metoda má jediný vstupní parametr, kterým je identifikační řetězec položky, kterou chceme označit jako přečtenou. Operace se provádí pomocí HTTP dotazu, ke kterému jsou vložena metodou POST data a celek je odeslán na určenou adresu jak uvádí tabulka 5.9.

Cílová adresa:	http://www.google.com/reader/api/0/edit-tag?client=`parametr_1`
POST data:	i=`parametr_2`&a=user/-/state/com.google/read/&ac=edit &T=`parametr_3`
Příklad:	i=tag:google.com,2005:reader/item/e8402d4fb8f99544&a=user/-/state/com.google/read/&ac=edit &T=2aJkkICSwj1vRxR968gq1Q
`parametr_1`	identifikace aplikace
`parametr_2`	id položky pro označení
`parametr_3`	token získaný při přihlášení

Tabulka 5.9: Požadavek pro označení položky jako přečtené

Pokud je položka úspěšně označena jako přečtená, je navrácena odpověď „OK“, jinak popis chyby, která při provádění nastala.

Metoda addFeed

Metoda požaduje tři vstupní parametry a to internetovou adresu nově přidávaného zdroje, uživatelem definovaný název zdroje a název štítku, do kterého má být zdroj přiřazen. Poslední dva parametry mohou být vloženy prázdné.

Cílová adresa:	http://www.google.com/reader/api/0/subscription/edit?client=`parametr_1`
POST data:	s=feed/`parametr_2`&ac=subscribe&t=`parametr_3` &a=user/label/`parametr_4`&T=`parametr_5`
Příklad:	s=feed/http://www.archlinux.org/feeds/news/&ac=subscribe&t=Archlinux &a=user/label/Linux&T=2aJkkICSwj1vRxR968gq1Q
`parametr_1`	identifikace aplikace
`parametr_2`	adresa zdroje
`parametr_3`	uživatelem definovaný název zdroje
`parametr_4`	název štítku, do kterého má být zdroj přiřazen
`parametr_5`	token získaný při přihlášení

Tabulka 5.10: Požadavek pro přidání nového zdroje

Pokud je nový zdroj úspěšně přidán, je navrácena odpověď „OK“, jinak popis chyby, která při provádění nastala.

Metoda removeFeed

Jediným vstupním parametrem této metody je identifikační řetězec zdroje, který chceme odstranit ze seznamu odebíraných zdrojů.

Cílová adresa:	http://www.google.com/reader/api/0/subscription/edit?client=`parametr_1`
POST data:	s=feed/`parametr_2`&ac=unsubscribe&T=`parametr_3`
Příklad:	s=feed/http://www.archlinux.org/feeds/news/&ac=unsubscribe &T=2aJkkICSwj1vRxR968gq1Q
`parametr_1`	identifikace aplikace
`parametr_2`	identifikační řetězec zdroje
`parametr_3`	token získaný při přihlášení

Tabulka 5.11: Požadavek pro odebrání zdroje

Pokud je zdroj úspěšně odebrán, je navrácena odpověď „OK“, jinak popis chyby, která při provádění nastala.

Metody addFeedLabel, removeFeedLabel

Vstupními parametry těchto dvou metod jsou identifikační čísla zdrojů, kterým chceme odebrat štítek, jehož název je definován pomocí druhého parametru.

Cílová adresa:	http://www.google.com/reader/api/0/subscription/edit?client=`parametr_1`
POST data pro přidání štítku:	s=feed/`parametr_2`&ac=edit&a=user/-/label/`parametr_3`&T=`parametr_4`
POST data pro odebrání štítku:	s=feed/`parametr_2`&ac=edit&r=user/-/label/`parametr_3`&T=`parametr_4`
Příklad pro přidání štítku:	s=feed/http://www.archlinux.org/feeds/news/&ac=edit&r=user/-/label/Linux&T=2aJkkICSwj1vRxR968gq1Q
`parametr_1`	identifikace aplikace
`parametr_2`	identifikační řetězec zdroje
`parametr_3`	název štítku pro odebrání
`parametr_4`	token získaný při přihlášení

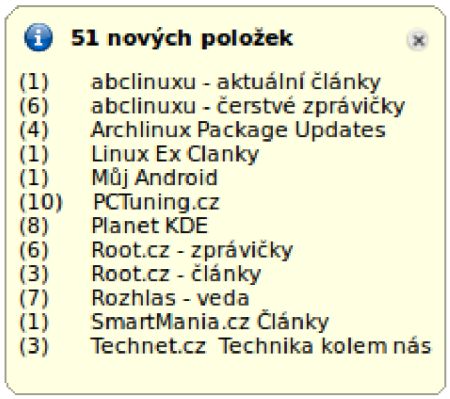
Tabulka 5.12: Požadavek pro přidání/odebrání štítku

Pokud je štítek zdroji úspěšně přidán nebo odebrán, je navrácena odpověď „OK“, jinak popis chyby, která při provádění nastala.

5.2.5 Třída gReader

V této třídě je implementována logika a funkčnost uživatelského rozhraní. O funkčnosti, která je implementována v této třídě bych zmínil několik informací.

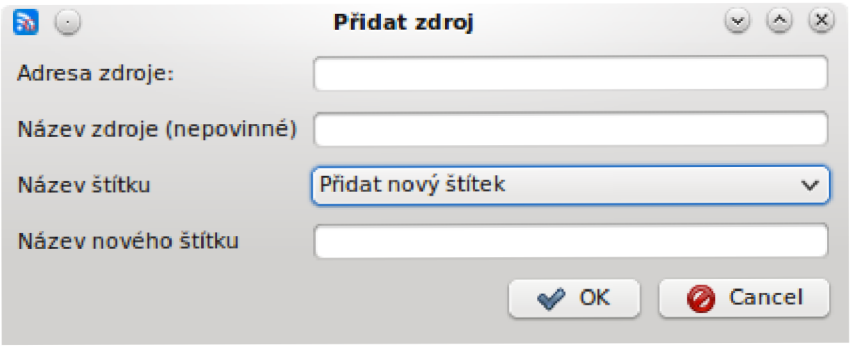
- Prohlížeč obsahu položek vychází z oblíbeného enginu pro internetové prohlížeče WebKit.
- Pokud uživatel dojde na poslední položku v panelu se seznamem položek, automaticky se stáhne dalších 5 záznamů z aktuálně zvoleného zdroje. Toto řešení mi přijde výhodnější, než stahovat úplně celou historii všech položek, protože hlavní funkčností tohoto programu by mělo být především informovat uživatele o nových položkách a ne sloužit jako archiv. Navíc pokud bychom stahovali celou historii, znatelně by se zvětšilo množství potřebného času pro start aplikace.
- Aplikace umožňuje vyhledávání v položkách dle jejich názvu.
- Při obdržení nových nepřečtených položek je na ikoně v systémové oblasti zobrazeno oznámení, které je uvedeno na obrázku 5.2.



Obrázek 5.2: Oznámení o nových položkách

5.2.6 Třída AddFeedDialog

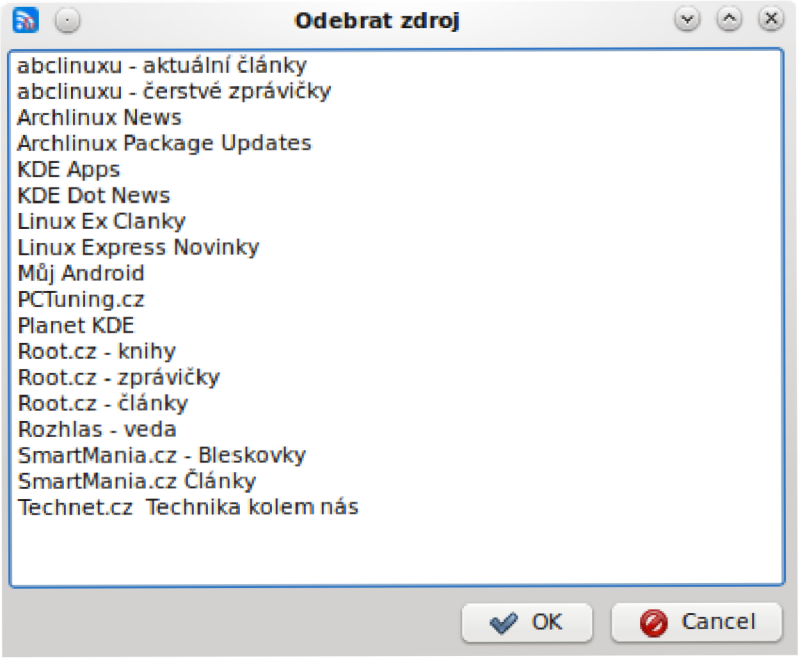
Třída implementuje dialog pro přidání nového zdroje do seznamu odebíraných zdrojů. Pro úspěšné přidání postačí vyplnit správnou adresu zdroje. Pokud chceme můžeme vložit vlastní název pro zdroj nebo jej označit štítkem.



Obrázek 5.3: Dialog pro přidání nového štítku

5.2.7 Třída RemoveFeedDialog

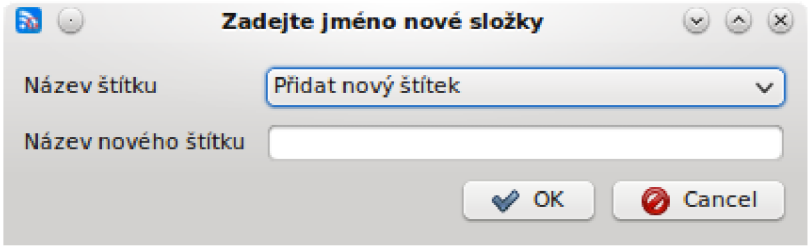
V této třídě se nachází implementace dialogu pro odebrání zdroje ze seznamu odebíraných zdrojů.



Obrázek 5.4: Dialog pro odebrání zdroje

5.2.8 Třída LabelNameDialog

V této třídě je implementován dialog pro přidání nového štítku ke zdroji.



Obrázek 5.5: Dialog pro přidání nového štítku

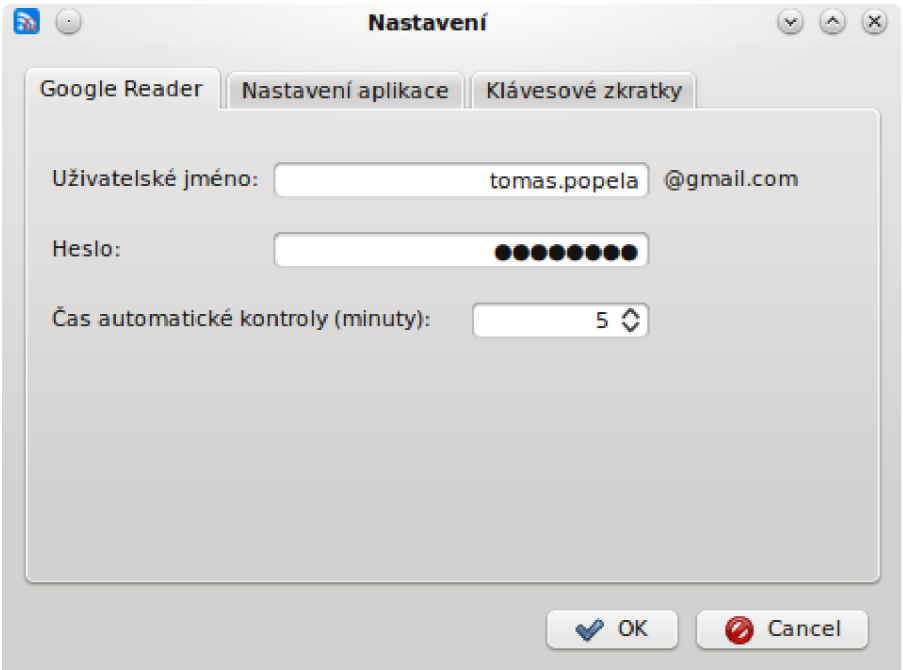
5.2.9 Třída SettingsDialog

Tato třída implementuje dialog s nastavením aplikace. Ten je rozdělen do tří záložek podle zaměření jednotlivých nastavení.

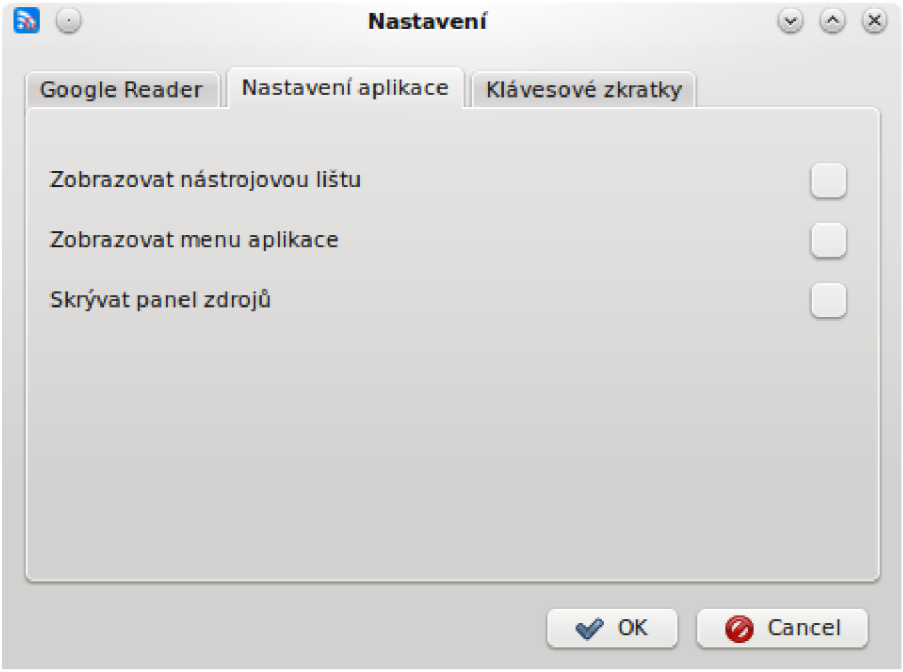
V první záložce nastavujeme uživatelské jméno a heslo ke službě Google Reader a dále čas automatické aktualizace odebíraných zdrojů.

V následující záložce najdeme volby týkající se vzhledu aplikace. Můžeme skrýt menu a nástrojovou lištu nebo vynout automatické skrývání panelu se seznamem zdrojů.

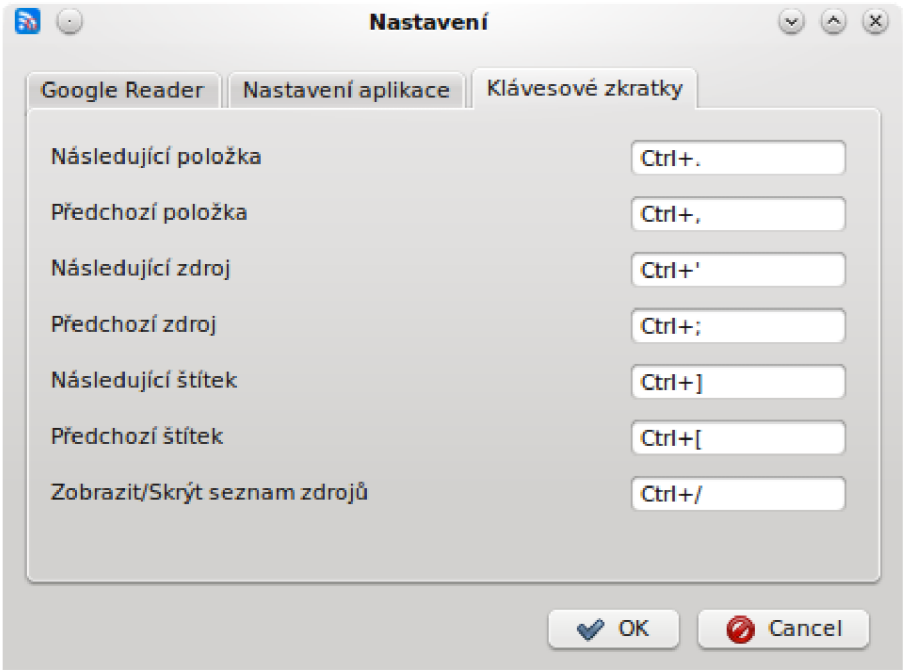
V poslední záložce se nachází pole s možností změny klávesových zkratek aplikace. Klávesové zkratky jsou nastavovány pomocí kliknutí do příslušného pole a stiskutí příslušné klávesové zkratky. Podporovány jsou klávesy Ctrl a Meta (klávesa s oknem nalevo od mezerníku) jako modifikátory a k nim zbytek znaků klávesnice.



Obrázek 5.6: Záložka s nastavením uživatelského účtu



Obrázek 5.7: Záložka s nastavením vzhledu aplikace



Obrázek 5.8: Záložka s nastavením klávesových zkratek

Pro nastavení programu můžeme také použít příkazový řádek, jehož příkazy jsou uvedeny v tabulce 4.1

Celá konfigurace je poté ve třídě gReader uložena do XML souboru settings.xml v následujícím formátu.

```
<?xml version="1.0"encoding="utf-8"? >
<settings>
    <userName>josef.novak</userName>
    <password>heslo</password>
    <checkFeedTime>300000</checkFeedTime>
    <toolBar>false</toolBar>
    <autoHideFeedPanel>true</autoHideFeedPanel>
    <menu>false</menu>
    <shortcutNextEntry>Ctrl+.</shortcutNextEntry>
    <shortcutPrevEntry>Ctrl+,</shortcutPrevEntry>
    <shortcutNextFeed>Ctrl+'</shortcutNextFeed>
    <shortcutPrevFeed>Ctrl+;</shortcutPrevFeed>
    <shortcutNextLabel>Ctrl+]</shortcutNextLabel>
    <shortcutPrevLabel>Ctrl+[</shortcutPrevLabel>
</settings>
```

Tabulka 5.13: Položka ze seznamu odebíraných zdrojů

Bohužel jelikož framework Qt nenabízí algoritmy pro šifrování/dešifrování, je heslo uloženo v originální textové podobě.

5.3 Testování

Vývoj aplikace probíhal pod operačním systémem Linux (distribuce Archlinux 64bit) a s Qt verze 4.6.2. Při testování byla aplikace dále kompilována a testována v linuxové distribuci Fedora (32bit) a na operačních systémech MAC OS X 10.6.3, Microsoft Windows XP(32bit) a Windows 7(32bit). Ve všech operačních systémech se aplikace chovala dle očekávání.

Kapitola 6

Možnosti dalšího vývoje

Dle mého názoru je aplikace gReader již připravena ke každodennímu použití. Při dalším vývoji programu by bylo vhodné implementovat níže specifikovaná rozšíření či vylepšení.

6.1 Uživatelské rozhraní

V otázce rozložení prvků aplikace není dle mého názoru více co zlepšovat. Naproti tomu by bylo vhodné více propracovat systém příkazů a zkratek pro ovládání. Jedno z vylepšení by mohl být pohyb mezi seznamy prvků a zdrojů nebo v prohlížeči pomocí kláves h, j, k, l jako v editoru VIM.

6.2 Funkcionalita

Co se týče funkcionality, tak zde vidím největší možnosti pro rozšíření. Přidat podporu pro označování položek „hvězdičkami“ nebo při přidávání nového zdroje nabízet seznam doporučených zdrojů, jak je tomu v originální internetové aplikaci.

6.3 Bezpečnost

Jak jsem již dříve uvedl, heslo pro Google účet je ukládáno do konfigurace ve formě plaintextu. Bylo by proto vhodné implementovat nějaký šifrovací algoritmus a ukládat heslo v šifrované podobě nebo využít správců hesel, které nabízejí jednotlivé platformy (např. KWallet pro desktopové prostředí KDE).

6.4 Lokalizace

Aplikace v podobě v jaké je odevzdávána je lokalizována pouze do českého jazyka. Avšak všechny uživatelovi viditelné popisky či oznámení jsou definovány pomocí funkce `tr()`, čímž je umožněn překlad aplikace pomocí programu *Linguist*.

Kapitola 7

Závěr

Cílem mé bakalářské práce bylo navrhnout a implementovat nativní rozhraní pro službu Google Reader. Rozhraní aplikace gReader bylo implementováno ve frameworku Qt a je kompatibilní s Qt verze 4.6 a vyšší.

Aplikace umí získat a zobrazit položky a zdroje, které odebírá přihlášený uživatel. Dále si uživatel může nové zdroje přidávat, stávající odebírat či organizovat pomocí štítků. Aplikace implementuje rozumnou podmnožinu funkcí, které nabízí služba Google Reader, a proto je dle mého názoru připravena pro běžné použití. Potencionální uživatele jistě zaujmou způsoby, kterými je možné aplikaci ovládat či použitelnost aplikace na zařízeních s nízkým rozlišením displeje.

V rámci vývoje aplikace jsem si osvojil práci s frameworkem Qt a nástroji, které Qt SDK nabízí. Jmenovitě nástroje Qt Designer pro návrh uživatelského rozhraní a Qt Creator, který slouží jako vývojové prostředí s integrovanou dokumentací jazyka Qt, která je zpracována na velmi vysoké úrovni a při vývoji jsem ji využíval prakticky neustále. Dále jsem získal zkušenosti se systémem verzí zdrojového kódu Git. Díky službě Gitorious jsou zdrojové kódy přístupné veřejnosti na adrese `http://gitorious.org/bc_gr/`.

Při vývoji aplikace mne daná problematika zaujala natolik, že se budu dále věnovat vývoji aplikace gReader.

Literatura

[1] Feed icon. [online], [cit. 2010-05-14].
URL http://en.wikipedia.org/wiki/File:Feed-icon.svg

[2] Git - Fast Version Control System. [online], [cit. 2010-05-13].
URL http://git-scm.com/

[3] History of RSS. [online], [cit. 2010-04-25].
URL http://www.rss-specifications.com/what-is-rss.htm

[4] Qt - A Cross-platform application and UI framework. [online], [cit. 2010-06-28].
URL http://qt.nokia.com/

[5] Rss20AndAtom10Compared. [online], [cit. 2010-04-25].
URL http://www.intertwingly.net/wiki/pie/Rss20AndAtom10Compared

[6] What is RSS. [online], [cit. 2010-04-25].
URL http://www.rss-specifications.com/what-is-rss.htm

[7] XML - The History of Xml. 2010, [online], [cit. 2010-05-13].
URL http://www.totalxml.net/history-xml.php

[8] Berners-Lee, T.; Fielding, R.; Frystyk, G.: Hypertext Transfer Protocol – HTTP/1.0. 1996, [online], [cit. 2010-05-11].
URL http://www.ietf.org/rfc/rfc1945.txt

[9] Dostálek, L.: HTTP protokol. 1997, [online], [cit. 2010-05-13].
URL http://www.cpress.cz/knihy/tcp-ip-bezp/HTTP/http1-1.htm

[10] Fielding, R.; Gettys, J.; Frystyk, G.; aj.: Hypertext Transfer Protocol – HTTP/1.1. 1999, [online], [cit. 2010-05-11].
URL http://www.ietf.org/rfc/rfc2616.txt

[11] Gagnon, R.: Create a RSS feed (part 1) - Real's Java How-to. [online], [cit. 2010-05-13].
URL http://www.rgagnon.com/javadetails/java-0556.html

[12] Gissehel: GoogleReaderAPI - pyrfeed. [online], [cit. 2010-04-25].
URL http://code.google.com/p/pyrfeed/wiki/GoogleReaderAPI

[13] Jasmin Blanchette, M. S.: *C++ GUI programming with Qt4*, kapitola A Brief History of Qt. Prentice Hall, 2008, ISBN 0-13-235416-0.

[14] Kenedy, N.: Google Reader API. [online], [cit. 2010-04-25].
URL http://www.niallkennedy.com/blog/2005/12/google-reader-api.html

[15] Kolář, D.: Builder - Zápis správné syntaxe XML dokumentů. 2001, [online], [cit. 2010-05-13].
URL http://www.builder.cz/art/html/xml_syntaxe.html

[16] Nottingham, M.; Sayre, R.: The Atom Syndication Format. 2005, [online], [cit. 2010-05-11].
URL http://www.ietf.org/rfc/rfc4287.txt

[17] Snell, J.: An overview of the Atom 1.0 Syndication Format. [online], [cit. 2010-04-25].
URL http://www.ibm.com/developerworks/xml/library/x-atom10.html

[18] Zapletal, L.: Protokol HTTP 1.1 pod lupou - Root.cz. 2001, [online], [cit. 2010-05-13].
URL http://www.root.cz/clanky/protokol-http-1-1-pod-lupou/

Seznam příloh

- Příloha A – Obsah DVD

Obsah DVD

Přiložené dvd obsahuje zdrojové kódy aplikace a další potřebné soubory ke spuštění aplikace. Dále je přiložena kopie této bakalářské práce.

Adresářová struktura:

`/implementation`	aplikace gReader
`binary_win32/`	spustitelný soubor pro OS Windows
`library/`	potřebné soubory k běhu programu
`qt_sdk/`	vývojové prostředí QtSdk verze 4.6.2
`source/`	zdrojové kódy aplikace gReader
`/thesis`	bakalářská práce
`source/`	zdrojové kódy bakalářské práce ve formátu LaTeX
`bachelors_thesis.pdf`	bakalářská práce
`bachelors_thesis_print.pdf`	bakalářská práce – verze pro tisk
`install.txt`	postup při instalaci
`readme.txt`	obsah DVD