Gymnázium Jana Keplera

Maturitní práce - předmět Informatika

2014/2015

Školní aplikace pro Android

Autor:
Dalimil Hájek

Vedoucí:
Pavel Zbytovský

12. března 2015

1 Anotace

Hlavní téma maturitní práce je školní aplikace určená pro Gymnázium Jana Keplera, kterou lze používat na mobilní platformě Google Android.

Aplikace umožňuje uživateli okamžitý přístup ke změnám v rozvrhu jeho třídy, dennímu menu školní jídelny a dalším užitečným službám (odkazy na systém STUDY, Docházku, rozvrh tříd, ...)

Student má díky aplikaci přehled o změnách ve svém rozvrhu a má možnost být o těchto změnách automaticky notifikován. Dále se mu zobrazuje přehledné menu školní jídelny s volitelným množstvím zobrazovaného obsahu včetně alergenů obsažených v jednotlivých jídlech.

Tyto informace aplikace ukládá, a tak je uživatel má dostupné i v době, kdy není připojen k internetu.

2 Přesné zadání

Úkolem je vytvořit školní mobilní aplikaci pro operační systém Android. Aplikace bude umět uživatelům zobrazovat aktuální suplování a změny v rozvrhu týkající se jejich třídy. Dále bude ukazovat jídelníček pro aktuální týden.

3 Prohlášení

Prohlašuji, že jsem jediným autorem této maturitní práce a všechny citace, použitá literatura a další zdroje jsou v práci uvedené.

Tímto dle zákona 121/2000 Sb. ve znění pozdějších předpisů uděluji bezúplatně škole Gymnázium Jana Keplera, Praha 6, Parléřova 2 oprávnění k výkonu práva na rozmnožování díla (§ 13) a sdělování díla veřejnosti (§ 18) na dobu časově neomezenou a bez omezení územního rozsahu.

4 Návod na instalaci

K instalaci aplikace je potřeba mít nainstalovaný mobilní operační systém Android verze 4.0.3 nebo vyšší.
Aplikaci lze pak stáhnout z distribuční služby Google Play. [1]

Alternativně si lze celou aplikaci zkompilovat a sestavit z původních zdrojů.
Celý projekt je dostupný na GitHubu: `https://github.com/Dalimil/Kepler-GJK`

Ke kompilaci a sestavení byly použity následující nástroje a knihovny:

- Android SDK Tools 24.0.2
- Android SDK Platform-tools 21
- Android SDK Build-tools 21.0.2
- Android Support Library 21.0.3

[1] https://play.google.com/store/apps/details?id=gjk.kepler

5 Dokumentace

Základní myšlenkou při návrhu mobilní aplikace bylo oddělit část kódu, která bude získávat data o suplování a jídelně, od samotné aplikace. Android aplikace nestahuje data přímo ze školních stránek, nýbrž komunikuje se serverem, na kterém běží PHP skript, který tato data o suplování a jídelně získává a předzpracovává. Toto PHP API tedy funguje jako prostředník při komunikaci mezi aplikací a školním webem.

5.1 PHP API

Základem PHP skriptu je návrh API, které je neměnné a mobilní aplikace na jeho neměnnost spoléhá. Výhodou tohoto přístupu je, že při změnně formátu jídelníčku nebo suplování se nemusí měnit kód Android aplikace a nemusí tak být spoléháno na aktualizaci aplikace ze strany uživatele. Stačí změnit zdrojový kód PHP skriptu a změny se na straně uživatele projeví okamžitě bez toho, aniž by si uživatel něčeho všiml.

Data skript vrací v JSON formátu. Výstup je vracen v závislosti na parametrech v URL (např. `http://adresa-serveru.org/?type=suplovani&trida=1.A`). PHP API je navrženo takto:

```
parametry:
type= /* Android aplikace vnitřně nastavuje jednu ze dvou hodnot */
    suplovani
    jidelna

type=suplovani & trida=4.A  (string - povinné)
/** Pokud je specifikován type "suplovani", musí být uveden parametr "trida"
 *  Ta je nastavena uživatelem v nastavení aplikace a je to povinný údaj
 */
ODPOVĚĎ PHP skriptu (např. pro ?type=suplovani&trida=1.A):
JSON
  {
   "type": "suplovani",
   "trida": "1.A",
   "dny":[
          {
          "den": "Čtvrtek 23.10.2014 (lichý týden)",
          "info": "Dnes se koná burza učebnic.",
          "hodiny":[
            {"hodina": 4, "predmet": "Nj NF4", "zmena": "(PCH) supluje Kučera František"},
            {"hodina": 6, "predmet": "Lab lab2", "zmena": "odpadá (Koc)"},
            ...
           ]
          },

          {
           "den": "Pátek 24.10.2014 (lichý týden)",
           ...
          },

          ...
      ]
 }
```

```
type=jidelna
/** Pro type "jidelna" je vrácen jídelníček školní jídelny spolu s alergeny
 *   O zobrazení/nezobrazení alergenů se stará Android aplikace
 */
ODPOVĚĎ PHP skriptu (např. pro ?type=jidelna):
 JSON
 {
  "type": "jidelna",
  "dny":[
         {
          "den": "Pondělí 20.10.",
          "polevka": {"nazev": "Krupicová s vejcem", "alergeny": "lepek, vejce, celer"},
          "jidla":[
                   {"nazev": "Kuřecí maso na paprice, dušená rýže", "alergeny": "lepek"},
                   {"nazev": "Zapečené šunkové flíčky, okurka", "alergeny": "vejce"},
                   ...
                   ]
           },

           {
            "den": "Úterý 21.10.",
            ...
           },

           ...

        ]
 }
```

5.2 PHP skript

5.2.1 Suplování

Suplování generované systémem Bakaláři na školním webu má v HTML strukturu tabulek, například:

Změny v rozvrzích tříd:							
1.A	0.- 12. hod AvŠ						
1.B	0.- 12. hod AvŠ						
2.A	4.hod	Aj	A1	(P2.2)	změna	Žďárek Karel	
	4.hod	Aj	A3	(P2.2)	spojí	Žďárek Karel	(Háj)
	4.hod	Aj	A4	(H2.3)	spojí	Koubková Helena	(Hlá)
2.C	1.hod	Zem			odpadá		(Kčr)
	4.hod	Aj	A1	(P2.2)	změna	Žďárek Karel	
	4.hod	Aj	A3	(P2.2)	spojí	Žďárek Karel	(Háj)
	4.hod	Aj	A4	(H2.3)	spojí	Koubková Helena	(Hlá)
3.A	1.hod	Nj	NF3		odpadá		(Gra)
	5.hod	M	M2		odpadá		(Kom)
3.B	1.hod	Nj	NF3		odpadá		(Gra)
	5.hod	M	M2		odpadá		(Kom)

```
<table class="tb_supltrid_3" width="100%">
  <tr class="tr_supltrid_3">
    <td class="td_supltrid_3" width="100%" colspan="8"><p>Změny v rozvrzích tříd:</p></td>
  </tr>
  <tr>
    <td class="td_supltrid_3" width="11%"><p>1.A</p></td>
    <td class="td_supltrid_3" width="89%" colspan="7"><p>0.- 12. hod AvŠ</p></td>
  </tr>
  <tr>
    <td class="td_supltrid_3" width="11%"><p>1.B</p></td>
    <td class="td_supltrid_3" width="89%" colspan="7"><p>0.- 12. hod AvŠ</p></td>
  </tr>
  <tr>
    <td class="td_supltrid_3" width="11%"><p>2.A</p></td>
    <td class="td_supltrid_3" width="7%"><p>4.hod</p></td>
    <td class="td_supltrid_3" width="6%"><p>Aj</p></td>
    <td class="td_supltrid_3" width="6%"><p>A1</p></td>
    <td class="td_supltrid_3" width="8%"><p>(P2.2)</p></td>
    <td class="td_supltrid_3" width="14%"><p>změna</p></td>
    <td class="td_supltrid_3" width="28%"><p>Žďárek Karel</p></td>
    <td class="td_supltrid_3" width="20%"> </td>
  </tr>
  <tr>
    <td class="td_supltrid_3" width="11%"> </td>
    <td class="td_supltrid_3" width="7%"><p>4.hod</p></td>
    <td class="td_supltrid_3" width="6%"><p>Aj</p></td>
    <td class="td_supltrid_3" width="6%"><p>A3</p></td>
    <td class="td_supltrid_3" width="8%"><p>(P2.2)</p></td>
    <td class="td_supltrid_3" width="14%"><p>spojí</p></td>
    <td class="td_supltrid_3" width="28%"><p>Žďárek Karel</p></td>
    <td class="td_supltrid_3" width="20%"><p>(Háj)</p></td>
  </tr>
  ...
```

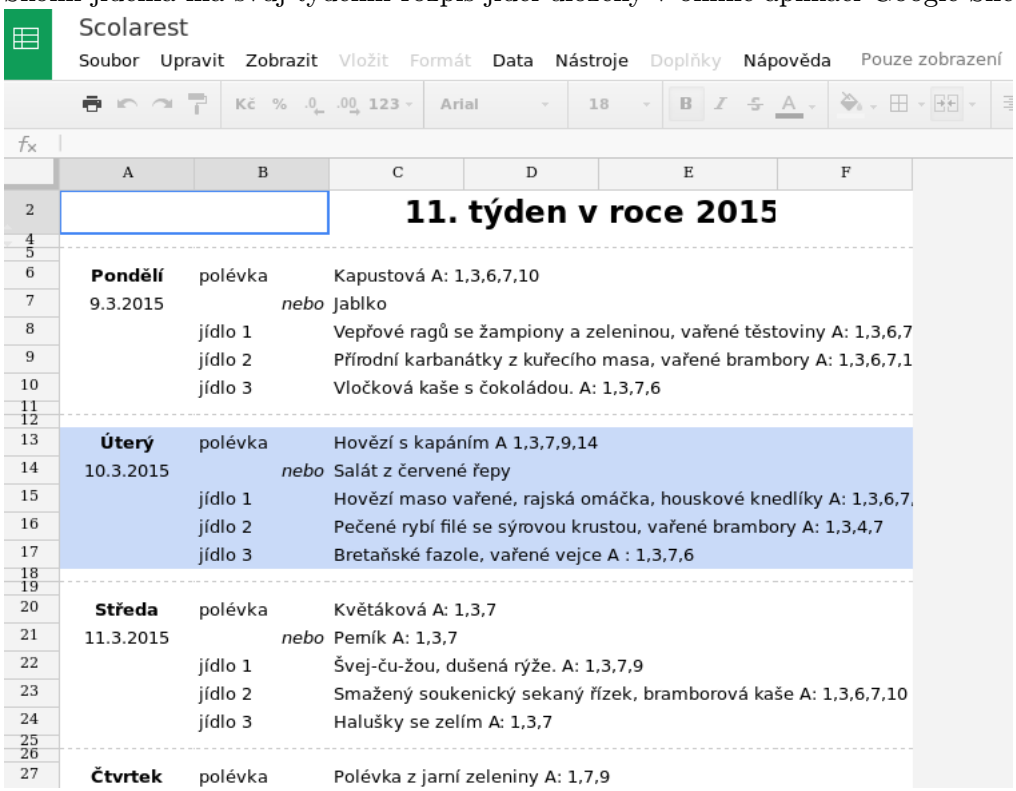
Aby se v programu nemuselo pracovat s HTML kódem jako s obyčejným polem znaků a mohli jsme využít jeho hierarchické uspořádání, převedeme jej pomocí PHP funkce `simplexml_import_dom($doc)` na objekt, se kterým lze pracovat mnohem jednodušeji (podobně jako kdyby se jednalo o čistě XML dokument) – lze např. použit jazyk XPath k adresaci a přístupu ke konkrétním elementům.

Stěžejní část PHP skriptu, která parsuje suplování pak vypadá následovně (trojtečka značí méně důležitou vypuštěnou část kódu):

```
$json = array();
$json["type"] = "suplovani";
$json["trida"] = $trida;
$json["dny"] = array();
$tabNum = 0;
foreach($xml->table as $supl_tab) {

  ...

  $hodiny = array();
  $started = false;
  foreach($supl_tab->tr as $supl_tr){

    ...

    $hodina = array();
    $pocetTD = count($supl_tr->td);
    if($pocetTD < 2) continue; //vadny radek
    if($pocetTD < 5){ //$pocetTD >= 2
      //neni normalni zmena suplovani - napr jen avs
      $zmena = "";
      for($td_num = 1; $td_num < $pocetTD; $td_num++){ //nacist radek
        assert('$supl_tr->td[$td_num]["class"] == "td_supltrid_3"');
        $zmena = $zmena . trim($supl_tr->td[$td_num]->p) . " ";
      }
      $hodina = array(
                "hodina" => -1,
                "predmet" => "",
                "zmena" => trim($zmena));
      $hodiny[] = $hodina;
      continue;
    }
    $td_num = 0;
    $zmena = "";
    //nacist zmenu v suplovani od ctvrte bunky
    for($td_num = 4; $td_num < $pocetTD; $td_num++){
      assert('$supl_tr->td[$td_num]["class"] == "td_supltrid_3"');
      $zmena = $zmena . trim($supl_tr->td[$td_num]->p) . " ";
    }

    $hodina =array(
            "hodina" => trim(str_replace(".hod", "", $supl_tr->td[1]->p)),
            "predmet"=> trim(trim($supl_tr->td[2]->p) ." "
                                                .trim($supl_tr->td[3]->p)),
            "zmena"  => trim($zmena));
    $hodiny[] = $hodina;
  }
  $json["dny"][] = array("den" => $den, "info" => $info, "hodiny" => $hodiny);
}
echo json_encode($json);
```

5.2.2 Jídelna

Školní jídelna má svůj týdenní rozpis jídel uložený v online aplikaci Google Sheets následovně:



Scolarest

Soubor Upravit Zobrazit Vložit Formát Data Nástroje Doplňky Nápověda Pouze zobrazení

	A	B		C
2				**11. týden v roce 2015**
6	**Pondělí**	polévka		Kapustová A: 1,3,6,7,10
7	9.3.2015		*nebo*	Jablko
8		jídlo 1		Vepřové ragů se žampiony a zeleninou, vařené těstoviny A: 1,3,6,7
9		jídlo 2		Přírodní karbanátky z kuřecího masa, vařené brambory A: 1,3,6,7,1
10		jídlo 3		Vločková kaše s čokoládou. A: 1,3,7,6
13	**Úterý**	polévka		Hovězí s kapáním A 1,3,7,9,14
14	10.3.2015		*nebo*	Salát z červené řepy
15		jídlo 1		Hovězí maso vařené, rajská omáčka, houskové knedlíky A: 1,3,6,7,
16		jídlo 2		Pečené rybí filé se sýrovou krustou, vařené brambory A: 1,3,4,7
17		jídlo 3		Bretaňské fazole, vařené vejce A : 1,3,7,6
20	**Středa**	polévka		Květáková A: 1,3,7
21	11.3.2015		*nebo*	Pemík A: 1,3,7
22		jídlo 1		Švej-ču-žou, dušená rýže. A: 1,3,7,9
23		jídlo 2		Smažený soukenický sekaný řízek, bramborová kaše A: 1,3,6,7,10
24		jídlo 3		Halušky se zelím A: 1,3,7
27	**Čtvrtek**	polévka		Polévka z jarní zeleniny A: 1,7,9

PHP skript tyto data získá ve formátu CSV. Google Sheets umožňují stažení v tomto formátu při připojení URL parametru /export?format=csv. V kódu tedy např.

```
file_get_contents('https://docs.google.com/spreadsheets/d/1JpEUpUJ3slFP1y2PgJV1
J_2_sBf5VOek4TUcq90P_Cs/export?format=csv');
```

PHP skript tak dostane data v této podobě:

```
Pondělí,polévka,"Kapustová A: 1,3,6,7,10",,,
9.3.2015,nebo,Jablko,,,
,jídlo 1,"Vepřové ragů se žampiony a zeleninou, vařené těstoviny A: 1,3,6,7,10",,,
,jídlo 2,"Přírodní karbanátky z kuřecího masa, vařené brambory A: 1,3,6,7,10",,,
,jídlo 3,"Vločková kaše s čokoládou. A: 1,3,7,6",,,
,,,,,
Úterý,polévka,"Hovězí s kapáním A 1,3,7,9,14",,,
10.3.2015,nebo,Salát z červené řepy,,,
,jídlo 1,"Hovězí maso vařené, rajská omáčka, houskové knedlíky A: 1,3,6,7,10",,,
,jídlo 2,"Pečené rybí filé se sýrovou krustou, vařené brambory A: 1,3,4,7",,,
,jídlo 3,"Bretaňské fazole, vařené vejce A : 1,3,7,6",,,
,,,,,
```

PHP skript pak jednoduše projde soubor po řádcích a buňkách, které jsou odděleny čárkou. Nejedná se o nic složitého, proto jen odkazuji na zdrojový kód uložený na GitHubu.

`https://raw.githubusercontent.com/Dalimil/Kepler-GJK/master/GJK_API/index.php`

Alergeny jsou reprezentovány identifikačními čísly a jsou převedeny na názvy:

```
function getAlergen($num){
  $alergeny = array("", "lepek", "korýši", "vejce", "ryby", "arašídy", "sója", "mléko",
              "ořechy", "celer", "hořčice", "sezam", "měkkýši", "lupina", "siřičitany");
  if($num >= 1 && $num <= 14) return $alergeny[$num];
  else return "";
}
```

5.3 Android aplikace

Android aplikace se píší v programovacím jazyce Java. Aplikace se skládá z XML dokumentů, které definují uživatelské rozhraní a dále pak funkčních částí – Javovských tříd, které zajišťují veškerou funkcionalitu aplikace.

5.3.1 Uživatelské rozhraní

Grafické rozhraní aplikace je definováno sadou XML souborů. Některé definují barvy, jiné např. pouze popisy tlačítek, většina však popisuje rozložení grafických prvků na obrazovce.

Vzhled aplikace byl navržen v souladu s nejnovějším designovým standardem – **Material Design**, který byl představen spolu s nejnovější verzí Androidu (Lollipop). Android knihovny poskytují metody k vytvoření jednoduchých stavebních prvků – horní lišty, menu rozbalujícího se po poklepání apod. Díky tomu se nemusí každý jednotlivý grafický prvek vytvářet od základů, ale je poskytován nadstavbami Android knihoven.

K zajištění kompatibility se staršími verzemi operačního systému Android byla místy využívána oficiální **Android Support Library**, která k výsledné aplikaci při kompilaci některé důležité soubory z nejnovějších Android knihoven připojí.

XML definice grafického rozhraní hlavní obrazovky vypadá následovně. Definuje horní lištu spolu s navigací po levé straně (`DrawerLayout` spolu s povinným `ListView` potomkem jakožto konkrétní implementací navigace). Tyto grafické prvky se napříč aplikací nemění a proto jsou v hierarchii nad samotným obsahem (`FrameLayout`), který se dynamicky načítá a mění při přepínání stránek v navigaci.

Kepler > app > src > main > res > layout > activity_home.xml

activity_home.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_height="match_parent"
    android:layout_width="match_parent">

    <!-- Horní lišta -->
    <include
        android:id="@+id/toolbar"
        layout="@layout/toolbar" />

    <android.support.v4.widget.DrawerLayout
        android:id="@+id/drawer_layout"
        android:layout_below="@id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:background="@color/background_material_light">

        <!-- Hlavní obsah obrazovky,-->
        <FrameLayout
            android:id="@+id/content_frame"
            android:layout_width="match_parent"
            android:layout_height="match_parent" />

        <!-- Navigation drawer
            musí uvést layout_gravity a height match_parent -->
        <ListView android:id="@+id/navigation_drawer"
            android:layout_width="250dp"
            android:layout_height="match_parent"
            android:layout_gravity="start"
            android:paddingTop="5dp"
            android:choiceMode="singleChoice"
            android:divider="@android:color/transparent"
            android:dividerHeight="0dp"
            android:background="@color/primaryDark"/>

    </android.support.v4.widget.DrawerLayout>
```

Design Text

Nejdůležitějším XML souborem je `AndroidManifest.xml`, ve kterém jsou definované všechny komponenty aplikace – hlavní obrazovka, obrazovka nastavení, služba běžící na pozadí, která kontroluje aktualizace suplování, Java třídy zpracovávající událost změny stavu připojení k internetu atd. Stejně tak jsou zde definovaná povolení, která aplikace vyžaduje – připojení k internetu a povolení *vzbudit* telefon. Tento XML soubor definuje rozhraní mezi Android systémem a samotnou aplikací.

Kepler > app > src > main > AndroidManifest.xml

AndroidManifest.xml

```
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<application
    android:allowBackup="true"
    android:icon="@drawable/ic_launcher"
    android:label="Kepler"
    android:theme="@style/AppTheme" >

    <activity
        android:name=".Home"
        android:label="Kepler" >
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />

            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
    </activity>

    <activity
        android:name=".SettingsActivity"
        android:label="Nastavení"

        android:parentActivityName=".Home">
        <!-- Zpětná navigace v hiearchii -->
        <meta-data
            android:name="android.support.PARENT_ACTIVITY"
            android:value=".Home" />
    </activity>

    <receiver android:name=".AlarmReceiver" />
    <service android:name=".NotificationService" />
    <receiver android:name=".NetworkChangeReceiver" android:enabled="false">
        <intent-filter>
            <action android:name="android.net.conn.CONNECTIVITY_CHANGE" />
        </intent-filter>
    </receiver>
```

TODO 6: Android Terminal

5.3.2 Java třídy

Základní typ funkční komponenty v Android aplikaci je tzv. `Activity`, což je Java třída, která zajišťuje funkcionalitu jedné *obrazovky* (např. hlavní stránka nebo stránka s nastavením).

Protože je důležité mít v aplikaci jednotnou navigaci a horní lištu, vytvořil jsem abstraktní `BaseActivity`, která dědí od Androidovské `ActionBarActivity` a která svým potomkům zajistí přístup k navigaci a horní liště.

Hlavní třídou je `Home`, což je hlavní okno aplikace, ve kterém se ale dynamicky střídá suplování, jídelna a další položky navigace. V této dokumentaci nebudou uvedeny metody, které zajišťují funkcionalitu klikání v navigaci, zaplňování horní lišty ikonami nebo třeba strukturované uspořádání navigační lišty, protože tyto programovací postupy jsou lépe zdokumentované na stránkách samotného Androidu.[2]

Uvedu jen nejpodstatnější část, kterou je dynamické prohazovaní obsahů hlavní stránky. Třída `Home` nejprve řekne třídě `Content`, aby na hlavní stránku vložila obsah odpovídající dané pozici v navigaci (např. jídelna) a současně informaci o tom, zda chce uživatel stáhnout data nová (klikl na tlačítko refresh), nebo data už dříve stažená (např. jen otočil obrazovku zařízení).

```
/* Změní stávající fragment za nový, čímž vytvoří obsah hlavní stránky, parametr download ovlivní stažení nových dat */
private void createContent(int position, boolean download){
    setNewDateID(); //kvůli zneplatnění dříve stahovaných dat
    // Vytvoří nový fragment a nastaví obsah podle argumentu
    Fragment fragment = new Content(); //moje třída Content
    Bundle args = new Bundle();
    args.putInt(Content.ARG_CONTENT_NUMBER, position); //přibalíme argument
    args.putBoolean(Content.ARG_DOWNLOAD_CONTENT, download); //chceme stáhnout nová data?
    fragment.setArguments(args);
    // Vložíme nový fragment nahrazením stávajícího
    getFragmentManager().beginTransaction().replace(R.id.content_frame, fragment).commit();
}
```

Třída `Content` pak na základě argumentu získá a vytiskne data. Současně také získá osobní nastavení uživatele – třídu, kterou na GJK navštěvuje.

```
boolean downloadNew = getArguments().getBoolean(ARG_DOWNLOAD_CONTENT);

type = getArguments().getInt(ARG_CONTENT_NUMBER);
switch(type){
    case 0:
        String oldResult = parentActivity.getSharedPreferences(NotificationService.PREFS_NAME, 0).getString(NotificationService.PREFS_HTTP_RESULT, "");
        if(downloadNew || "".equals(oldResult)){ //vynucené stažení nebo stahujeme poprvé
            String prefClass = PreferenceManager.getDefaultSharedPreferences(parentActivity).getString("pref_class", "");
            getPage("http://kepler-dalimil.rhcloud.com/"+"?type="+content_types[type]+"&trida="+prefClass);
        } else{
            //zobraz naposledy stažené
            show(oldResult, false);
        }
        break;
    case 1:
        String oldFoodResult = parentActivity.getSharedPreferences(NotificationService.PREFS_NAME, 0).getString(NotificationService.PREFS_HTTP_FOOD, "");
        if(downloadNew || "".equals(oldFoodResult)){ //vynucená aktualizace (stažení) dat nebo stahujeme poprvé
            getPage("http://kepler-dalimil.rhcloud.com/"+"?type="+content_types[type]);
        } else{
            //zobrazení dříve staženého
            show(oldFoodResult, false);
        }
        break;
    case 2: //odkazy
        TextView content_text = new TextView(parentActivity);
        content_text.setGravity(0x01);
        content_text.setTextAppearance(parentActivity, R.style.TextAppearance_AppCompat_Display1);
        content_text.setLinkTextColor(getResources().getColor(R.color.primaryDark));
        content_text.setMovementMethod(LinkMovementMethod.getInstance());
        content_text.setText(Html.fromHtml("<a href="http://gjk.cz">GJK.CZ</a><br /> <a href="https..."));
        content_layout.addView(content_text);
        break;
}
```

Metoda `getPage(url)` je pak dále implementována jako obyčejný HTTP GET požadavek na server, kde běží dříve popsaný PHP skript, a stáhne odtamtud JSON data. Aby stahování nezbrzdilo běh aplikace, implementoval jsem HTTP požadavek asynchronně v novém vlákně.

[2] http://developer.android.com

Po získání dat v JSON formátu dojde k jejich přečtení a vypsání. Hlavní část metody, která vypisuje suplování vypadá následovně, přičemž v metodách `createTextView`, `createTextRow` přímo vytvářím grafické textové prvky a měním jim některé vzhledové atributy (viz zdrojový kód [3])

Kepler › app › src › main › java › gjk › kepler › Content

Content.java

```
content_layout.addView(divider);
createVerticalSpace(1);

JSONArray dny = res.getJSONArray("dny");
for (int i = 0; i < dny.length(); i++) {
    JSONObject ob = dny.getJSONObject(i);

    String den = ob.getString("den");
    createTextView(den, R.style.TextAppearance_AppCompat_Subhead, R.color.accent);

    String info = ob.getString("info");
    if(!info.equals("")){
        createTextView(info, R.style.TextAppearance_AppCompat_Caption);
    }

    JSONArray hodiny = ob.getJSONArray("hodiny");
    for (int j = 0; j < hodiny.length(); j++) {
        JSONObject hod = hodiny.getJSONObject(j);
        int hodina = hod.getInt("hodina");
        String predmet = hod.getString("predmet");
        String zmena = hod.getString("zmena");

        if(hodina == -1){//není standardní změna hodiny - např. avš
            createTextRow("Jiná změna:", zmena, true);
        } else{
            createTextRow("" + hodina + ".hod " + predmet, zmena, true);
        }
    }
    if(hodiny.length() == 0){
        createTextRow("Žádné suplování", "", false);
    }

    createVerticalSpace(1);
}

return "";
```

Velmi podobným způsobem vypadá metoda načítající jídelníček.

Kepler › app › src › main › java › gjk › kepler › Content

Content.java

```
JSONArray dny = res.getJSONArray("dny");
for (int i = day_of_week; i < dny.length(); i++) {
    JSONObject ob = dny.getJSONObject(i);

    String den = ob.getString("den");
    createTextView(den, R.style.TextAppearance_AppCompat_Subhead, R.color.accent);

    if(prefSoup) {
        JSONObject polevka = ob.getJSONObject("polevka");
        String polevkaNazev = polevka.getString("nazev");
        if (!"".equals(polevkaNazev)) {
            createTextRow("", "Polévka: "+polevkaNazev, false);
        }
        if (prefFood) {
            String polevkaAlergeny = polevka.getString("alergeny");
            if (!"".equals(polevkaAlergeny)) {
                createTextView("\tAlergeny: " + polevkaAlergeny, R.style.TextAppearance_AppCompat_Caption);
            }
        }
    }

    JSONArray jidla = ob.getJSONArray("jidla");
    for (int j = 0; j < jidla.length(); j++) {
        JSONObject jidlo = jidla.getJSONObject(j);
        String nazev = jidlo.getString("nazev");
        createTextRow("" + (j + 1) + ") ", nazev, false);
        if(prefFood){
            String alergeny = jidlo.getString("alergeny");
            createTextView("\tAlergeny: "+alergeny, R.style.TextAppearance_AppCompat_Caption);
        }
    }
    createVerticalSpace(1);
}
return "";
```

[3] https://raw.githubusercontent.com/Dalimil/Kepler-GJK/master/Kepler/app/src/main/java/gjk/kepler/Content.java

K pohodlí uživatele je součástí aplikace i nastavení automatických notifikacích o změnách suplování. Pokud uživatel danou položku v nastavení zaškrtne, třída `AlarmReceiver` se každou hodinu aktivuje a spustí službu `NotificationService`, která se pokusí stáhnout nová data o suplování a jídelně. Výhodou této služby je, že se spustí i pokud je Android zařízení uspané a aplikace neběží – právě proto aplikace vyžaduje definované povolení `WAKE_LOCK`.

`NotificationService` se postará o stažení dat, jejich uložení do paměti aplikace a případné vytvoření notifikace pro uživatele v případě, že se nové suplování liší od předchozího uloženého.

```
    // Poslat notifikaci
    private void sendNotification(String msg) {
        PendingIntent contentIntent = PendingIntent.getActivity(this, 0, new Intent(this, Home.class), 0);

        NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this)
                            .setSmallIcon(R.drawable.ic_notification)
                            .setLargeIcon(BitmapFactory.decodeResource(getResources(), R.drawable.ic_launcher))
                            .setContentTitle("Nové suplování")
                            .setContentText(msg)
                            .setAutoCancel(true);

        mBuilder.setContentIntent(contentIntent);

        NotificationManager notifMgr = (NotificationManager) this.getSystemService(Context.NOTIFICATION_SERVICE);
        notifMgr.notify(1, mBuilder.build());
    }


}
```

Poslední ze tříd zajišťující funkčnost notifikací je třída `NetworkChangeReceiver`, která je třídou `NotificationService` aktivována v případě, že se nezdařilo stáhnout data (uživatel je například odpojen od internetu). Aplikace následně naslouchá změnám v připojení k síti a ve chvíli, kdy se zařízení připojí k internetu se ihned provede update dat o suplování a `NetworkChangeReceiver` může být zase deaktivován.

Kepler > app > src > main > java > gjk > kepler > NetworkChangeReceiver

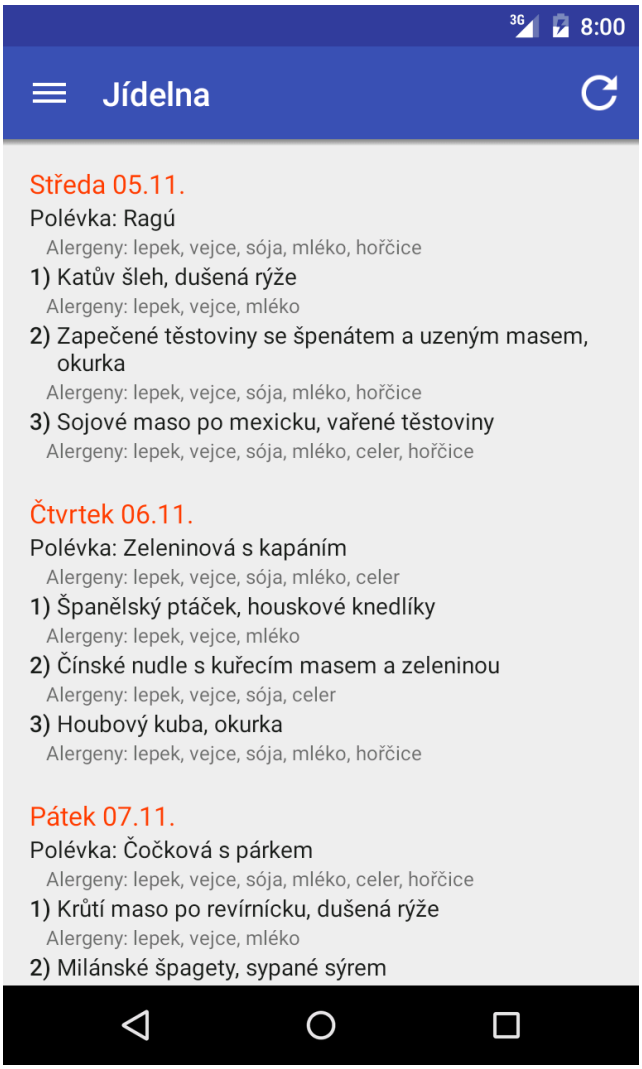
NetworkChangeReceiver.java

```
package gjk.kepler;

import ...

public class NetworkChangeReceiver extends BroadcastReceiver{

    @Override
    public void onReceive(Context context, Intent intent) {
        HTML_Loader html_loader = new HTML_Loader(context);
        if(html_loader.checkConnection()){
            //okamžitý update + případně notifikace (uživatel byl předtím offline a nezdařilo se)
            context.startService(new Intent(context, NotificationService.class));
        }

    }

}
```

6 Závěr a hodnocení

Projekt se celkově povedl a nakonec byl i veřejně publikován ve službě Google Play, kde má v současnosti na základě hodnocení uživatelů skóre 4,64 z 5.

Časově byla práce dobře rozvržena především v první půlce školního roku, kdy postupně vznikaly jednotlivé funkční části. Méně plánované byly zásahy do aplikace v čase, kdy už byla sice funkční, ale přání o drobné změny přicházely od samotných uživatelů.

Současně škola v této době vytvořila nový školní web a kompletně tak změnila tehdejší formát jídelníčku, načež se muselo na tyto změny reagovat přepsáním velké části kódu, která se o získávání dat jídelníčku starala. Zde se velmi dobře uplatnil původní návrh oddělení funkční části aplikace, která získává data o suplování a jídelníčku, protože se tak nemusel měnit kód samotné mobilní aplikace a uživatelé tak žádnou změnu nezaznamenali, a to ani v podobě updatu – protože nebyl nutný.

Díky umístění odkazu na aplikaci na stránkách školy se povedlo aplikaci rozšířit mezi studenty, a ta se tak stala oblíbenou. V době psaní tohoto dokumentu má aplikace přes sto aktivních uživatelů, což je vzhledem k počtu studentů gymnázia úspěch.