UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Vývoj aplikací pod OS Android

Marek Pokorný

Bakalářská práce
2013

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

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Marek Pokorný**
Osobní číslo:	**I09045**
Studijní program:	**B2612 Elektrotechnika a informatika**
Studijní obor:	**Komunikační a mikroprocesorová technika**
Název tématu:	**Vývoj aplikací pod OS Android**
Zadávající katedra:	**Katedra elektrotechniky**

Zásady pro vypracování:

Cílem práce je ukázka možností programování a využití operačního systému android. Teoretická část práce bude obsahovat popis operačního systému android a základních nabízených služeb, přehled použitelných programovacích jazyků a prostředků (překladače, vývojová prostředí). V praktické části bude popsán vývoj aplikace pro android, popis bude obsahovat vše potřebné, od instalace vývojového prostředí, SDK, způsob instalace aplikace do zařízení s androidem, až po ukázkové aplikace. Jako ukázkové aplikace vznikne aplikace pracující se senzory obsažené v zařízení, dále jednoduchá aplikace pro připojení se k jinému zařízení přes bluetooth nebo jiné rozhraní a na závěr plnohodnotná aplikace zjednodušující práci s barevným značením součástek (rezistorů). Součástí práce budou funkční aplikace a dokumentovaný zdrojový kód.

Rozsah grafických prací:

Rozsah pracovní zprávy:

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

Seznam odborné literatury:

http://developer.android.com/index.html
http://www.anddev.org/index.php
Mark L. Murphy : Android 2, Průvodce programováním mobilních aplikací; Computer Press, a.s 2011; ISBN 978-80-251-3194-7

Vedoucí bakalářské práce: **Ing. Pavel Rozsíval**
Katedra elektrotechniky

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

prof. Ing. Simeon Karamazov, Dr.
děkan

L.S.

Ing. Zdeněk Němec, 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 29.4.2013 Marek Pokorný

Poděkování

Rád bych poděkoval panu Ing. Pavlu Rozsívalovi, vedoucímu mé bakalářské práce, za odborné vedení a cenné rady a připomínky, které mi pomohly zpracovat tuto bakalářskou práci.

Anotace

Cílem této práce je představení operačního systému Android od jeho historie až po samotné programování. Po přečtení tohoto dokumentu a pochopení základních příkazů a pravidel, by neměl být problém vytvořit si vlastní jednoduchou aplikaci.

Klíčová slova

Android, Eclipse, Programování, Vývoj

Title

Development of applications under OS Android

Annotation

The aim of this work is to introduce Android from its history to actual programing use. After reading this document and understanding basic commands and rules, there shouldn’t be any problem with creating your own application.

Keywords

Android, Eclipse, Programming, Development

Obsah

Seznam zkratek

JDK	Java Development Kit
SDK	Software Development Kit
ADT	Android Development Tools
OS	Oparační systém
GPS	Global Positioning System

Seznam obrázků

Seznam tabulek

Úvod

Hlavním cílem této práce je vytvoření funkčních aplikací. A to pro určování hodnot rezistorů podle jejich barevného označení. Další aplikace bude pracovat se systémem GPS na určování polohy. Poslední aplikace bude využívat Bluetooth pro připojení k jinému zařízení.

První kapitola se zabývá historií a vznikem operačního systému Android. Zároveň je zde uvedena základní architektura, se kterou tento systém pracuje. Nechybí zde ani přehled jednotlivých verzí Androidu a jejich výhody oproti předešlým.

Druhá kapitola se již týká programování, avšak pouze teoreticky. Jsou zde vysvětleny základní komponenty aplikace, se kterými se setkáváme. Nejdůležitější část této kapitoly se týká aktivit, které se objevují v každé aplikaci a bez nichž nemůžeme vyvíjet pro OS Android.

Třetí kapitola nám ukazuje, jak se nainstaluje vývojové prostředí Eclipse na počítač. Celá část instalace má celkem čtyři kroky. A to instalace JDK, Eclipse, SDK a ADT.

Čtvrtá kapitola se zabývá praktickými ukázkami programování pro OS Android. Jsou zde ukázky základních prvků, které je možno použít při vývoji aplikací. Těmito prvky jsou například tlačítko, popisek, textová pole a přepínače. Za použití těchto komponent lze dosáhnout mnoha aplikací.

V poslední páté kapitole najdeme hotové aplikace. Jsou celkem tři. První, jak již bylo zmíněno, pomáhá určit hodnotu rezistoru podle barevného značení na něm. Druhá využívá GPS na určování naší polohy. Aplikace zobrazuje aktuální polohu pomocí GPS a zároveň nám to ukáže polohu na mapě pomocí internetu. Poslední třetí aplikace slouží k ukázce práce s Bluetooth.

1 Historie

Tato kapitola se zabývá vznikem operačního systému Android. Jeho architekturou a platformami. Celá problematika je dobře vysvětlena v literatuře [2] a [8], z kterých jsem vycházel.

Obrázek 1 - Logo Androidu (obrázky google.cz)

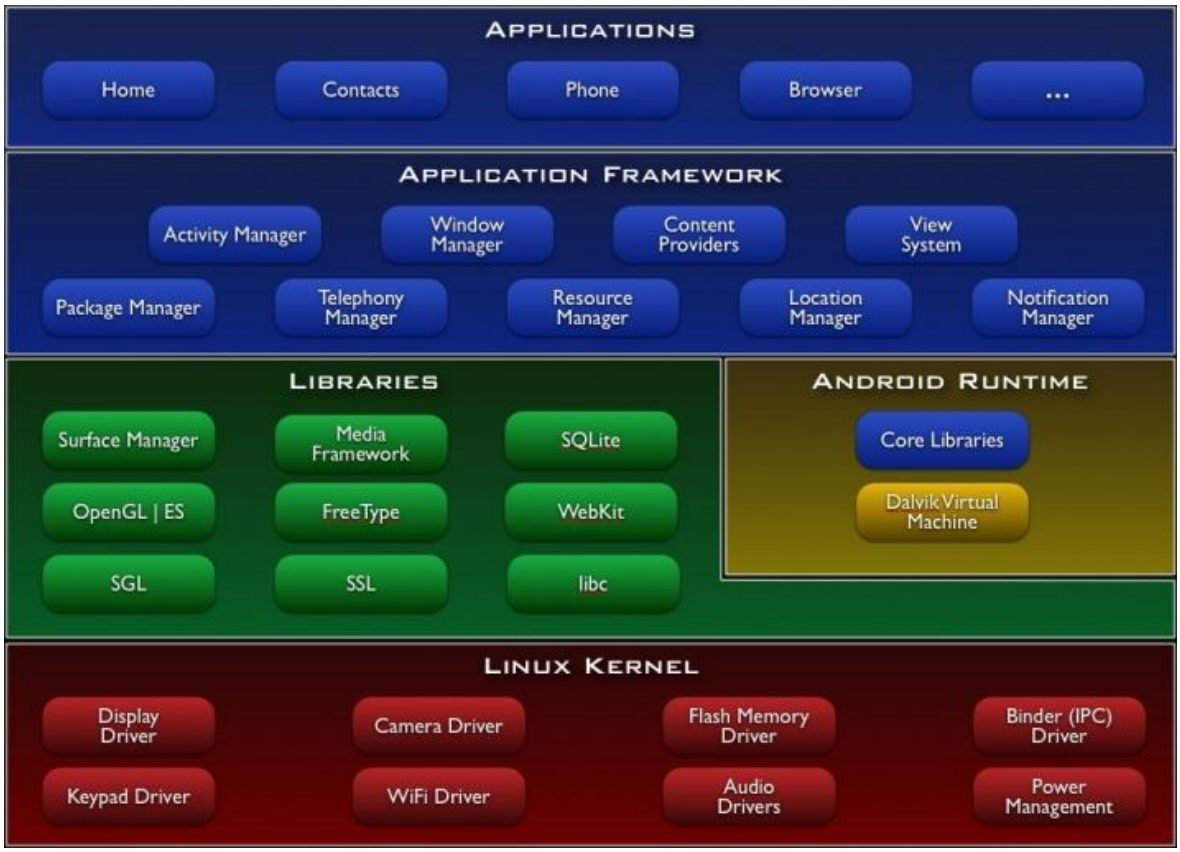
1.1 Vznik Androidu

Na založení společnosti Android Inc. se podílely čtyři osoby. A to sice Andy Rubin, Rich Miner, Nick Sears a Chris White. Ti založili firmu v Kalifornii v říjnu 2003. V srpnu roku 2005 tuto v podstatě neznámou „startup“ firmu odkoupila společnost Google Inc. a stala se z ní dceřiná společnost.

Hlavní zásluhu na vývoji platformy, která je založena na Linuxovém jádře, má Andy Rubin. Díky němu a jeho týmu získala firma Google několik patentů v oblasti mobilních technologií. To ovšem vedlo ke spekulacím na adresu Googlu, že chtějí vstoupit na trh mobilních telefonů.

1.2 Architektura

Operační systém android má architekturu rozdělenou do 5 vrstev. Tyto vrstvy nemusí být přímo odděleny mezi sebou a každá má svůj účel.



Obrázek 2 - Architektura operačního systému Android [2]

Jádro operačního systému je nejnižší vrstva. Ta tvoří abstraktní vrstvu mezi používaným hardwarem a zbytkem softwaru ve zbylých vrstvách. Celé jádro Android systému je založeno na Linuxu ve verzi 2.6. Je zde použito mnoho jeho vlastností, jako je podpora správy paměti, správa sítí, zabudované ovladače nebo správy procesů. Jako příklad je souběžný běh aplikací, které běží s oprávněním systému. Díky tomu je systém stabilní a dá se zajistit jeho ochrana. Jádro Linux bylo použito z důvodu vlastnosti poměrně snadného sestavení na různých zařízeních a tím byla zaručena přenositelnost.

Knihovny jsou další vrstvou systému. Jsou napsány v C/C++ kódu využívající různé komponenty systému. Prostřednictvím Android Application Framework jsou tyto funkce poskytnuty vývojářům. Jako příklad jsou zde uvedeny některé knihovny:

- Media Libraries – slouží k přehrávání audio a video formátů, např. MPEG4, H.264, MP3, AMR, JPG, PNG a ACC
- SQLite – odlehčená relační databázová knihovna
- OpenSSL – secure socket layer
- Free Type – pro rendering bitmapových a vektorových fontů
- OpenGL – na vykreslování 3D grafiky
- LibWebCore – knihovna webového prohlížeče, který podporuje i vložené náhledy webových stránek

Další vrstvou je Android Runtime. Ta obsahuje aplikační virtuální stroj označovaný jako Dalvik a využívá základních vlastností linuxového jádra, jako například správa paměti nebo práce s vlákny. Důvodem vzniku tohoto stroje byla licenční práva, protože jazyk Java a jeho knihovny jsou volně šiřitelné, a taky optimalizace pro mobilní zařízení, a to především kvůli poměru výkonu a úspory energie. V této vrstvě se rovněž nachází základní knihovny programovacího jazyka Java. Ty se blíží platformě Java Standart Edition. Rozdíl je ovšem v tom, že neobsahuje knihovny pro uživatelské rozhraní (AWT a Swing). Ty byly nahrazeny knihovnami pro Android. Při překladu aplikace napsané pro Android dochází ke zkompilování zdrojového kódu v jazyce Java do Java byte kódu pomocí stejného kompilátoru jako v překladu Java aplikací. Následně se překompiluje pomocí Dalvik kompilátoru. Výsledný Dalvik byte kód je spuštěn na DVM (Dalvik Virtual Machine).

Předposlední vrstvu tvoří Application framework. Ta je pro vývojáře nejdůležitější a poskytuje přístup k velkému počtu služeb. Například prvky uživatelského rozhraní, aplikace spuštěné na pozadí nebo zpřístupnění dat v jiných aplikacích a mnoho dalších. Základní sada obsahuje:

- View – slouží k sestavení uživatelského rozhraní, jedná se o seznamy, tlačítka, textová pole, zaškrtávací políčka a další.
- Content providers – pro přístup k datům jiných aplikací nebo sdílení vlastních dat
- Resource manager – přístup k neprogramovým zdrojům (soubory designu a grafiky)
- Notification manager – upozornění ve stavovém řádku
- Activity manager – řídí životní cyklus aplikace

Poslední a nejvyšší vrstvu tvoří aplikace. Jde o aplikace, které využívá běžný uživatel, ale jde i o ty stažené z Android Marketu nebo přeinstalované v zařízení. Jedná se o kalendář, mapy, kontakty, e-mailový klient, zasílání SMS a dalších.

1.3 Historie verzí

S každou verzí Androidu přicházely nové funkce a opravy chyb. Jednotlivé verze se jmenují podle zákusků, a to sice:

- 1.5 (Cupcake) – muffin, košíček
- 1.6 (Donut) - kobliha
- 2.0/2.1 (Eclair) – jemný podélný moučník s vanilkovou náplní a čokoládovou polevou nebo banánek
- 2.2 (Froyo) – mražený jogurt
- 2.3/2.4 (Gingerbread) - perník
- 3.0/3.1/3.2 (Honeycomb) – plástev medu
- 4.0-4.0.4 (Ice Cream Sandwich) – zmrzlinový sandwich
- 4.1-4.2 (Jelly Bean) – barevné bonbóny ve tvaru malých fazolek

Tyto verze si nyní představíme blíže. A podíváme se, co nového nám přinesly jednotlivé verze.

Android 1.5 Cupcake

- Nové widgety a složky
- Bluetooth s podporou automatického Conner Headset (A2DP)
- Klávesnice s dokončováním slov (T9)
- Kamera (REC i PLAY)
- Sdílení multimedií na Youtube a Picasu
- Drobné grafické změny

Android 1.6 Donut

- Podpora pro zařízení s rozlišením WVGA (480x800 bodů)
- Android Market, vyhledávání hlasem
- Vylepšení a nové prostředí multimedií (např. galerie pro správu fotek)
- Quick Search Box – umožňuje vyhledávat historii, záložky, kontakty

Android 2.1 Eclair

- Celkové zrychlení
- Bluetooth 2.1
- Podpora pro Microsoft Exchange
- Grafické a usnadňující (drobné) úpravy: uživatelské prostředí, kontakty, prohlížeč + podpora HTML 5, softwarová klávesnice, podpora přisvětlovací diody, digitální zoom, animované tapety
- Podpora pro větší rozlišení displeje
- Aktualizace Google Map

Android 2.2 Froyo

- Možnost instalovat aplikace na paměťovou kartu
- Adobe Flash 10.1
- Vylepšené řízení RAM (2x až 5x rychlejší)
- Wifi Hotspot
- Nové stavy telefonu – car mode (režim v autě), night mode (noční režim)
- OpenGL ES 2.0, vícebarevný trackball

Android 2.3 Gingerbread

- Podpora více kamer a senzorů
- Vylepšení nativního kódu
- Video v HTML 5
- 3D Google mapy 5.0
- NFC – bezdrátové technologie budoucnosti (zaplacení jízdného v městské dopravě, reproduktory, banka, soubory …)

Android 3.0 Honeycomb (pro Tablety)

- Upravený multi-tasking
- Google E-Books
- Pracovní plochy se předimenzují i do LANDSPACE režimu
- Nový design
- Nový prohlížeč
- Videohovory přes Google Talk

Android 4.0 Ice Cream Sandwich

- Odemčení telefonu obličejem
- Přepracovaný laucher
- Ukazatel přenesených dat
- Zachycení panoramat
- Vylepšený správce kontaktů a rozpoznání hlasu

Android 4.1 Jelly Bean

- Rozpoznání hlasu offline
- Google Now
- Vylepšená informační lišta
- Vylepšená aplikace fotoaparátu
- Více uživatelských účtů

2 Vývoj aplikací

2.1 Co je potřeba pro vývoj aplikací?

Nejdůležitější pro vývoj vlastních aplikací je oficiální podporované prostředí. A to program Eclipse, do kterého je potřeba nainstalovat ADT plugin, který ulehčuje práci s android projektem. Pokud programátor nechce vytvářet v Eclipsu, jsou zde i jiné možnosti. Například pomocí jednoduchého textového editoru a kompilace pomocí příkazového řádku. Osobně shledávám program Eclipse za nejlepší volbu, a to hlavně pro začátečníky. Jak správně nainstalovat program a všechny potřebné komponenty se budu zabývat později v této práci. Nejdříve je nutné porozumět základním částem Androidu. Další potřebnou věcí je mít alespoň nějaké zkušenosti s programovacím jazykem Java. Ti, co nemají, tak nemusí mít strach. Pokud jsou obeznámeni s jiným jazykem, tak naučit se aspoň základy Javy nebude velký problém. Pro jednoduché aplikace není potřeba zvláštních schopností. A poslední důležitá věc je alespoň částečná znalost angličtiny. Android je relativně mladý jazyk a spousta informací je v anglickém jazyce, ale i tak se dá najít mnoho věcí i v rodném jazyce. Bohužel pro větší a složitější projekty je nutné hledat na oficiálních stránkách pro vývoj aplikací, které jsou v jazyce anglickém.

2.2 Základní části aplikace Androidu

Základní vlastností Androidu je možnost využít části jiných aplikací ve své aplikaci. Důvod k tomuto řešení byl prostý – zamezit opakovanému vytváření stejné funkcionality a usnadnit tak vývojářům práci. K tomuto účelu byly aplikace navrženy jako balíky několika komponent, které lze samostatně instanciovat. Existují čtyři typy komponent: activity, service, broadcast receiver a content provider, přičemž každá z těchto komponent slouží k jinému účelu. Postupně si je nyní projdeme. Tyto komponenty jsou dobře vysvětleny na stránkách [3]. Nedílnou součástí hotové aplikace je Android manifest. Ten specifikuje parametry aplikace – jednotlivé komponenty, požadovaná systémová práva a jiné požadavky na běh, aby vše fungovalo jak má.

2.2.1 Activity

Aktivita je základní vizuální komponenta – každá aktivita specifikuje jednu obrazovku aplikace. Představme si třeba e-mailového klienta, kde jedna aktivita zobrazuje seznam přijatých e-mailů, jiná aktivita zobrazuje obsah jednotlivých e-mailů a třetí slouží k jejich psaní. Samozřejmě by takovou aplikaci šlo také napsat jednou aktivitou s tím, že by se vždy kompletně změnil obsah. Takovou architekturu ale nelze doporučit, je to špatná a mnohem pracnější cesta. V čem je tedy kromě snazšího vývoje výhodná popisovaná struktura několika aktivit? Právě v možnosti jednotlivé komponenty samostatně volat i z jiných aplikací. Například, narazíme-li v prohlížeči na link odkazující nás na e-mailovou adresu, lze po kliku přejít do naší aktivity určené pro psaní e-mailů. K tomuto účelu stačí jen aktivitě předat požadovanou e-mailovou adresu. Protože je aktivita vizuální komponentou, má okno, do kterého je možné kreslit. Okno aktivity typicky zaplňuje celou

obrazovku, ale může být také menší – být jen plovoucím dialogem. Obsah okna je specifikován pomocí views. View je vizuální prvek, na obrazovce mu odpovídá obdélníkový prostor. View se řadí do hierarchické struktury – rodičovské prvky, nazývané layouty, organizují rozložení vnitřních prvků. Typ layoutu určuje, jakým způsobem budou jeho podřízené view rozloženy v dostupném prostoru. Příklad základních view je třeba tlačítko nebo obrázek.

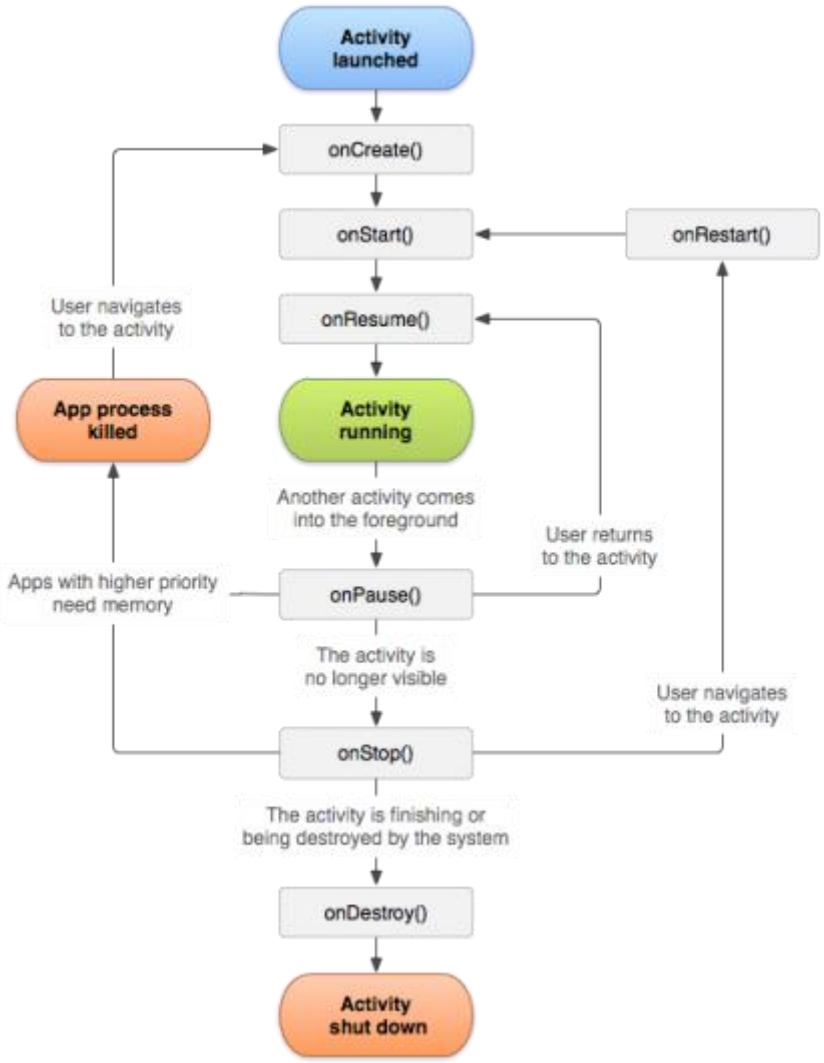
2.2.1.1 Životní cyklus aktivit

Aktivita se v průběhu svého života může nacházet ve třech základních stavech - může být na popředí a mít uživatelský vstup, může být pouze viditelná (třeba jen částečně) nebo může být pozastavená na pozadí. Při přechodu aktivity mezi stavy jsou volány systémová zpětná volání (callbacky). Tyto callbacky vymezují jednotlivé stavy aktivity a definují její životní cyklus. Tento cyklus je zobrazen na následujícím obrázku. Systémové callbacky životního cyklu aktivity jsou: Životní cyklus aktivity je vymezen voláními několika základních funkcí:

- ***onCreate()*** - Volána v momentě vytvoření aktivity. Zde se standardně provádí inicializace aktivity a nastavuje obsah UI.
- ***onStart()*** - Zavolána, když se aktivita stane viditelnou pro uživatele. Nemusí být plně viditelná - její část může být zakryta nebo může být vidět pod poloprůhledným pozadím jiné aktivity.
- ***onResume()*** - Volána, když aktivita začne dostávat uživatelský vstup. V tento moment je tedy na vrcholu zásobníku a není nijak překryta jinou aktivitou.
- ***onPause()*** - Volána těsně před ztrátou uživatelského vstupu.
- ***onStop()*** - Zavolána, když aktivita přestává být viditelná pro uživatele.
- ***onDestroy()*** - Volána před zrušením instance aktivity.
- ***onRestart()*** - Tato funkce je volána těsně předtím než aktivita, která byla zastavena (byla na ní zavolána funkce onStop()), je znovu nastartována - ve funkci onStart().

Celý život aktivity tedy probíhá mezi voláními funkcí onCreate() a onDestroy(). Aktivita je viditelná mezi voláními funkcí onStart() a onStop(). A uživatelský vstup aktivita dostává mezi voláními funkcí onResume() a onPause().

Životní cyklus aktivity nemusí být vždy kompletní. Pokud aktivita prošla funkcí onPause(), systém ji může při nedostatku prostředků ukončit. Funkce onStop() a onDestroy() tudíž nemusí být vždy zavolány. Ve funkci onPause() se často provádí uložení aktuálního stavu aktivity. Ale je třeba se zde vyhnout provádění zdlouhavých operací, neboť je to blokující funkce - pokud je spouštěna nová aktivita, její vytvoření proběhne až po návratu z této funkce.



Obrázek 3 - Životní cyklus aktivit [3]

Jiným typem komponenty je service (služba), která není uživatelsky viditelná a běží na pozadí. Příkladem je třeba komponenta určená k síťové komunikaci. V příkladu e-mailového klienta bychom mohli použít službu pro stahování a odesílání e-mailů na pozadí.

2.2.2 Broadcast receiver

Tato komponenta slouží k „poslouchání“ broadcastových oznámení a k reakci na ně. Broadcast receivery nejsou, stejně jako služby, uživatelsky viditelné. Jako reakci na příchozí oznámení mohou například spustit jiné komponenty. Broadcasty mohou být jak systémové, tak uživatelské – aplikace si mohou vytvářet své vlastní broadcasty. Jednoduchým příkladem může být receiver reagující na systémový broadcast oznamující nízký stav baterie. Reakcí naší aplikace může být například uložení aktuálního stavu a ukončení aplikace. Dalšími příklady mohou být receivery reagující na odpojení nebo připojení SD karty. Jiným příkladem může být receiver reagující na příchozí textovou zprávu tím, že spustí další aktivitu, která danou zprávu zobrazí.

2.2.3 Content provider

Poslední komponentou je content provider, který slouží ke zpřístupnění dat jiným aplikacím. Ty k datům přistupují pomocí instancí třídy ContentResolver. Data lze zpřístupnit ke čtení i k zápisu. Rozlišení toho, kdo může jen číst a kdo může i zapisovat, se provádí podle práv, jež aplikace mají a jež jsou vyžadována. Pomocí content resolveru lze přistupovat například k záznamům v adresáři telefonu, fotkám v galerii telefonu nebo záznamům o příchozích a odchozích hovorech.

2.2.4 Android manifest

Základem jakékoliv aplikace pro android je soubor manifestu, *AndroidManifest.xml*, v kořenovém adresáři vašeho projektu (další položky kořenového adresáře si vysvětlíme později). V něm deklarujeme obsah vaší aplikace – její aktivity, služby atd. Zároveň zde uvádíme informace o tom, jakým způsobem se tyto součásti aplikace propojují s operačním systémem; například indikuje, která aktivita (nebo aktivity) by se měla objevit v hlavním menu zařízení (spouštěči alias launcheru).[1]

Když aplikaci vytvoříte, vygeneruje se automaticky základní manifest. V případě jednoduché aplikace, nabízející jedinou aktivitu a nic jiného, bude zřejmě automaticky vygenerovaný manifest fungovat dobře nebo ho bude nutné pouze mírně upravit. Avšak soubor manifestu pro Android API demo soupravu má více jak 1 000 řádků. Manifesty vašich aplikací pak budou pravděpodobně mezi těmito dvěma extrémy.[1]

V souboru manifestu můžeme najít následující elementy

- ***user-permission*** – Indikují práva, která vaše aplikace bude potřebovat, aby mohla správně fungovat.
- ***permission*** – Deklarují práva, jejichž přidělení jiným aplikacím, aby tyto aplikace mohly používat logiku či data vaší aplikace, mohou aktivity nebo služby vyžadovat.
- ***instrumentation*** – Deklarují zdrojový kód, který by se měl volat při výskytu klíčových systémových událostí, jako například spouštění aktivit za účelem loggingu nebo monitoringu.
- ***uses-library*** – Připojují volitelné android komponenty, jako například mapovací služby.
- ***uses-sdk*** – Indikuje, pro kterou verzi Android SDK byla aplikace přeložena.
- ***application*** – Definuje vnitřnosti aplikace, které manifest popisuje

Příklad manifestu:

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
      package="com.commonsware.android">
      <uses-permission
            android:name="android.permission.ACCESS_LOCATION" />
      <uses-permission
            android:name="android.permission.ACCESS_GPS" />
      <uses-permission
            android:name="android.permission.ACCESS_ASSISTED_GPS" />
      <uses-permission
            android:name="android.permission.ACCESS_CELL_ID" />
      <application>
      ...
      </application>
</manifest>
```

V tomto příkladu manifest používá elementy *uses-permission* k indikaci některých vlastností zařízení, které bude aplikace potřebovat – v tomto případě oprávnění umožňující aplikaci určit aktuální zeměpisnou polohu zařízení. Obsah elementu *application* pak bude popisovat aktivity, služby a cokoliv dalšího, co tvoří svazek aplikace.[1]

3 Instalace vývojového prostředí

Nyní se dostáváme k té nepříjemné a zdlouhavé části. A to sice instalace prostředí. Na to budeme potřebovat trochu času. Návod pro instalaci jednotlivých částí je dobře popsán na stránkách [4] a [5], ze kterých jsem čerpal.

3.1 JDK

Prvním krokem je mít nainstalované JDK (Java Development Kit). Pokud ho máte je vše v pořádku. Pokud ne, je potřeba ho stáhnout a nainstalovat (lze ze stránek [1]).

3.2 Eclipse

Toto je ta nejjednodušší část. Instalace je stejná jako u kteréhokoliv jiného programu. Postupně přes tlačítko next dokončíme instalaci. Program je dostupný na stránkách Eclipse [2].

3.3 SDK

Můžeme se setkat i s označením ADK, neboli Android SDK (Software Development Kit). Tento balíček obsahuje nejrůznější nástroje, které Eclipse využívá. Jedná se o nástroje pro úpravu grafických prvků, testování nebo optimalizaci vzhledu a podobně. Stáhneme ho z [3]. Balíček si uložíme a rozbalíme na libovolné místo na disku (nejčastěji C:\Program Files). Tuto cestu si zapamatujeme a použijeme ji později.

3.4 ADT

Již máme vše potřebné a nyní to dáme všechno dohromady. A k tomu je potřeba ADT (Android Development Tools). To je plugin do prostředí Eclipse, který nám přidá nabídky týkající se androidu. Nejprve si spustíme Eclipse. Otevřeme si nabídku Help- Instal New Software – Add. Jméno si zvolíme libovolné a jako adresu repository přidejme tuto [4] (pokud nebude fungovat, zkusíme vyměnit protokol https za http). V dolním okně by měl být řádek pro výběr Developer Tools. Označíme ho pro stažení a potvrdíme. Jestli vyskočí jakékoliv hlášení o autenticitě nebo validitě softwaru, jednoduše jej potvrdíme souhlasem.

Dalším krokem je správně nakonfigurovat již zmíněný plugin. To uděláme tak, že v Eclipse otevřeme nabídku Windows – Preferences - Android a do pravého políčka zadáme cestu k místu, kde jsme rozbalili SKD z předchozího kroku.

A už se blížíme do finále. Zbývá pouze poslední krok a tostažení konkrétní platformy (verze operačního systému Android) a s tím související emulátor androidu. Tento emulátor slouží jako náhrada za fyzické zařízení a slouží k testování aplikací přímo na PC. Běh

[1] http://www.oracle.com/technetwork/java/javase/downloads/index.html (22.3.2013)
[2] http://www.eclipse.org/downloads/ (22.3.2013)
[3] http://developer.android.com/sdk/index.html (22.3.2013)
[4] https://dl-ssl.google.com/android/eclipse/ (22.3.2013)

aplikace je sice o něco pomalejší a nelze odzkoušet všechny funkce (volání atd.), ale pro většinu aplikací plně postačuje. Vybranou platformu stáhneme přes nabídku Windows-Android SDK Manager. Jsou zde prakticky všechny verze androidu. Nejčastější volba ovšem bude Android 2.3.3, ale je to jen na nás. Stačí nám označit pouze SDK Platform. Další podstatnou věcí je mít stažené položky Tools, Android SDK Tools a Android SDK Platform-tools.

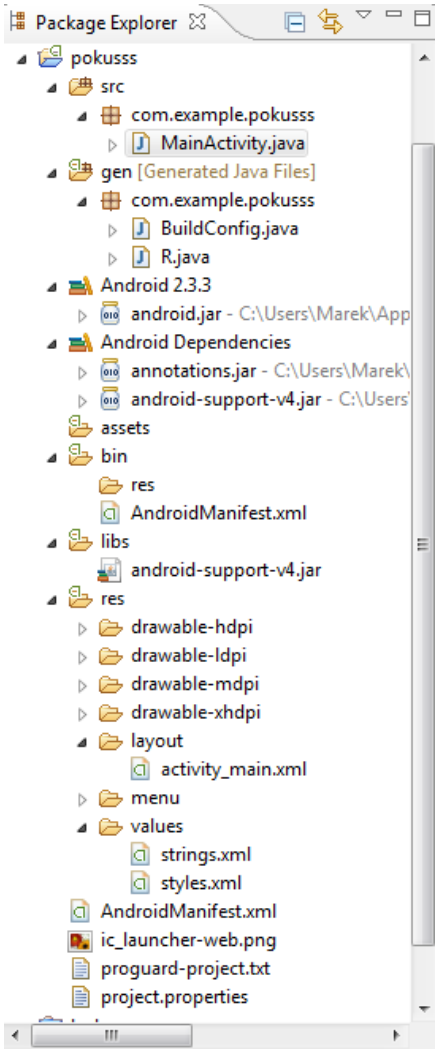
Pokud budeme aplikace testovat přímo v počítači, je nutné ještě nastavit emulátor. To provedeme tak, že v nabídce Windows – Android AVD Manager – New si ho nastavíme. Název může být libovolný, poté zadáme platformu a případně rozlišení displeje. Emulátor spustíme tlačítkem Start. Po jeho spuštění můžeme zvolit velikost displeje tak, aby se přizpůsobil velikosti reálného zařízení pomocí volby Scale display to real size.

Pro testování na skutečném zařízení (telefon, tablet) je potřeba mít správně nainstalované všechny ovladače přístroje. Ty se zpravidla nainstalují sami. Pokud tomu tak není, stačí navštívit internetové stránky výrobce zařízení a potřebné ovladače si stáhnout. Poslední věc je mít v telefonu povoleno ladění USB. Pro verze 4.x se nachází v Nastavení – Možnosti vývojáře - Ladění USB. U starších verzí jako například 2.x se nachází v Nastavení – Aplikace – Ladění USB. Uvidíme neodškrtnuté tlačítko. Odškrtneme ho a už nám nic nebrání testovat aplikace na našem zařízení.

4 Práce s Eclipse

4.1 Vytvoření nového projektu

Veškeré postupy, které zde budu popisovat, platí pro verzi Eclipsu v21.0.1-543035. Pro ostatní verze to bude velmi podobné a možná i stejné. Pro vytvoření nového projektu si otevřeme Eclipse. V nabídce File – New – Android Application Project. Po kliknutí se nám otevře okno pro nový projekt. Do políčka Application Name zadáme libovolné jméno naší aplikace. Ostatní řádky (Project Name a Package Name) se nám doplní automaticky. Dále zde máme, na jakou platformu bude určena naše aplikace. Nachází se zde Minimum Required SDK. Tuto položku raději necháme tak, jak je, protože program Eclipse občas nefunguje správně, pokud se položka změní. Další je Target SDK. To si změníme podle sebe, pro kterou verzi je aplikace určená. Jak jsem již zmínil výše, tak to bude nejčastěji Android 2.3.3. Compile With se opět doplní automaticky. Poslední položkou je Theme. Pokud budeme používat Android 2.3.3, tak nám bude fungovat pouze Theme None. Pro ostatní možnosti je potřeba vyšší verze systému Android. Potvrdíme tlačítkem Next. V dalším okně si můžeme zvolit umístění uložení projektu. Opět bych doporučoval neměnit, ale není problém si zvolit vlastní složku. Opět potvrdíme tlačítkem Next. Nyní si můžeme nastavit ikonu aplikace. Na výběr je Image, Clipart nebo Text. Jednu vybereme a tu můžeme dále upravovat (například velikost, barvu a styl ohraničení). Až budeme hotovi, pokračujeme tlačítkem Next. Další okno opět pouze potvrdíme. Pro začátek je to postačující. Lze zde navolit, jak bude aplikace vypadat na telefonu. Jestli uvidíme stavový řádek a podobně. Necháme být a dáme Next. Posledním oknem nastavení jména naší aktivity (obrazovky) a layoutu (grafické znázornění aktivity). Pokud chceme, můžeme změnit, ale není to potřeba. Dokončíme tlačítkem Finish. Tím se nám vytvoří a otevře náš projekt.

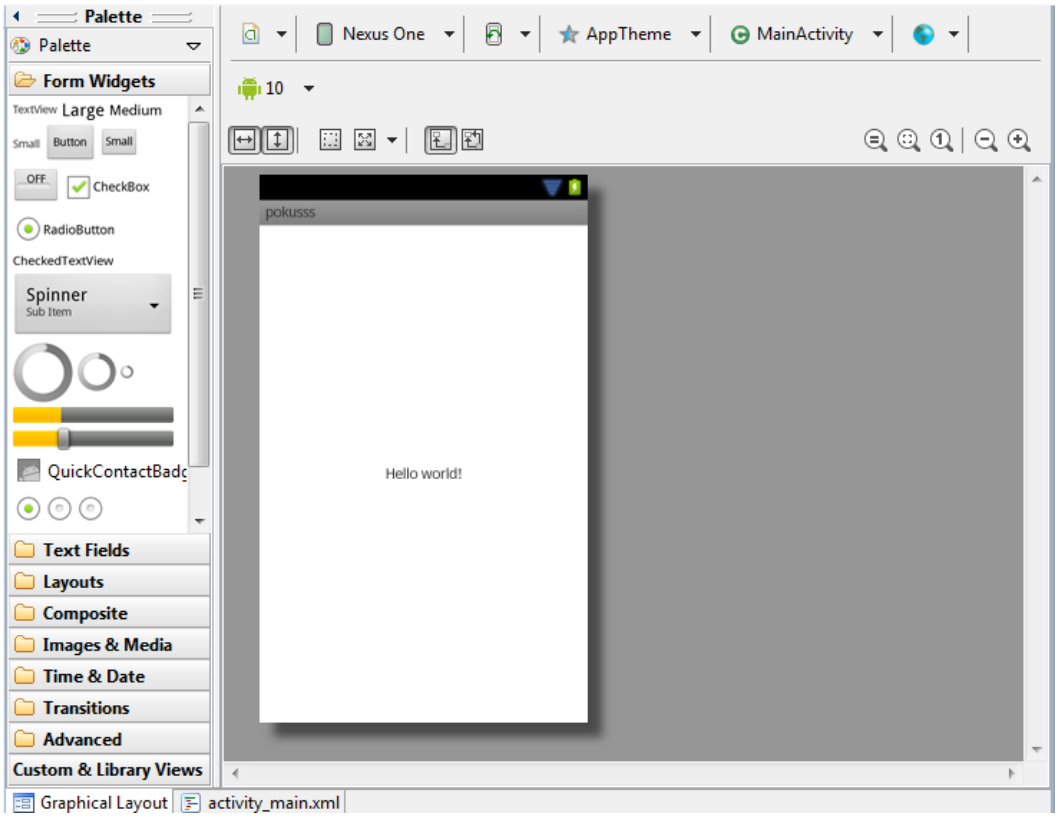


Obrázek 4 - Nový projekt - Adresář

Napravo vidíme adresář našeho projektu. Ten obsahuje různé položky včetně následujících [1]:

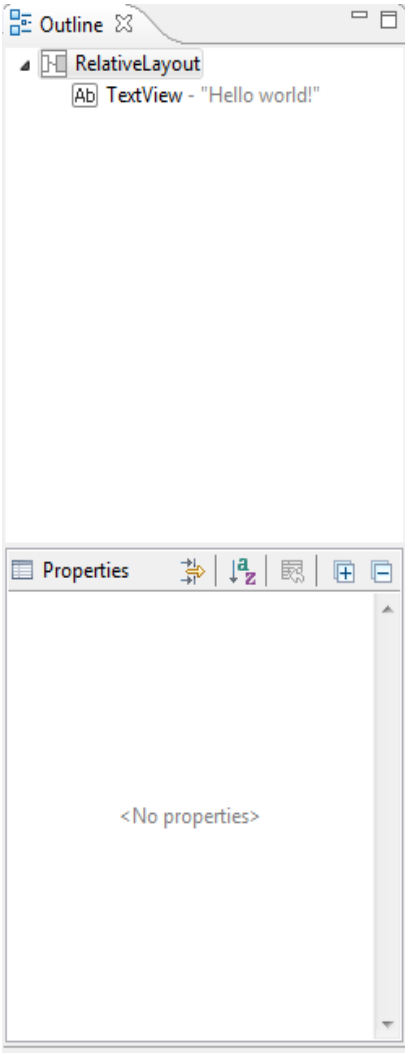
- src/ - složka uchovávající zdrojový kód aplikace v jazyce Java
- gen/ - složka, do které překladové nástroje systému Android umístí zdrojový kód, který vygenerují
- assets/ - složka obsahující ostatní statické soubory, které chcete přibalit k aplikaci pro použití na zařízení
- bin/ - složka, která uchovává aplikaci po jejím zkompilování
- libs/ - složka uchovávající jakékoliv Java archivy třetích stran, které vaše aplikace potřebuje

- res/ - složka uchovávající prostředky (například ikony, návrhy grafického uživatelského rozhraní (GUI) a podobně) zabalené se zkompilovaným Java zdrojovým kódem aplikace
- AndroidManifest.xml – XML soubor popisující vyvíjenou aplikaci a komponenty (aktivity, služby atd.), které aplikace obsahuje



Obrázek 5 - Nový projekt – Uživatelské rozhraní

Uprostřed je nastavení uživatelského rozhraní. Zde si můžeme rozvrhnout umístění jednotlivých prvků (tlačítka, textová pole …). Máme na výběr dvě možnosti, jak tyto prvky vkládat. Buď je přetáhneme z lišty vlevo, a nebo máme možnost je napsat v XML kódu. K tomu slouží záložky vlevo dole (Graphical Layout, nazev_aktivity.xml). Já osobně preferuji první způsob a případné úpravy (velikost, barva, pozadí) provadím již v XML.



Obrázek 6 - Nový projekt - Layout a Properties

Napravo máme dvě okna. Jedno nám ukazuje, jaké prvky v layoutu již máme a druhé slouží k nastavení vlastností (Properties).

4.2 Základní widgety (prvky) a jejich využití

Prvků je celá řada. My si řekneme pouze o těch základních a nejpoužívanějších, které bohatě postačí na jednoduché aplikace. Pokud se chceme dozvědět i o ostatních, doporučil bych navštívit oficiální stránky [5]. Jsou sice v anglickém jazyce, ale je zde i mnoho příkladů použití.

4.2.1 Layouty

Android nemá absolutní pozicování prvků na displeji. Díky tomu je těžké vytvořit pěkně vypadající návrh obrazovky. A k tomu účelu se používají layouty. Ty slouží k uspořádání jednotlivých komponent na obrazovce. Nyní si představíme dva nejpoužívanější.

4.2.1.1 LinerLayout

Tento layout je nejvhodnější pro každého začátečníka. Je relativně jednoduchý, protože všechny prvky, které do něj dáme, jednoduše srovná do řad. Ty se mohou rovnat buď vertikálně (android:orientation =“vertical“), a nebo horizontálně (android:orientation =“horizontal“). Tímto prostředkem lze dosáhnout zajímavých výsledků, ale vznikne bohužel velmi obsáhlý strom vnořených layoutů. A to potom vede k nepřehlednosti pro programátora a zároveň se zvedá náročnost výpočtů pro zařízení. Z tohoto důvodu většina začátečníků brzy přejde na RelativeLayout.

4.2.1.2 RelativeLayout

Jak již bylo zmíněno, tento layout je velmi častá volba k tvorbě obrazovek. Pracuje tak, že všechny vložené prvky pozicuje relativně proti sobě. Tím se myslí to, že se definuje, jestli je prvek pod, nad a nebo vedle jiného vloženého prvku. Nejčastěji se tento layout používá pro základní rozvržení obrazovky a ostatní se potom využijí na pozicování jednotlivých skupin prvků.

4.2.2 Tlačítko (Button)

Asi není nutné vysvětlovat, co tento prvek dělá. Jedná se o základní widget, který bude použit téměř v každé aplikaci. Jak bylo shora uvedeno, existují dvě možnosti použití. Nyní si ukážeme obě varianty, ale později budu používat jen jednu. První možnost je deklarovat tlačítko v XML. Například takto:

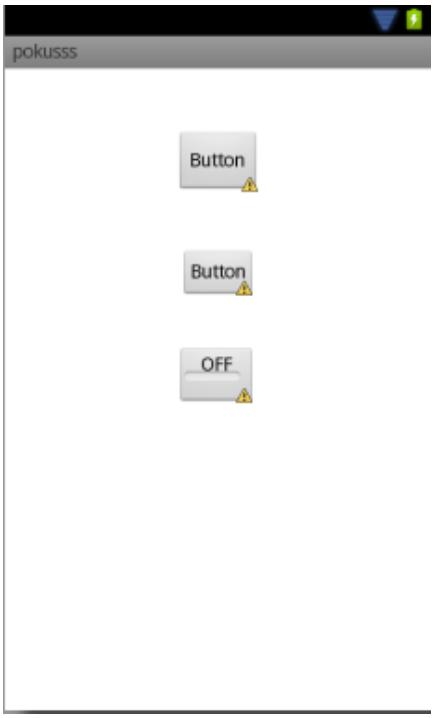
```
<Button
        android:id="@+id/button1"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Button" />
```

První položkou je id. Ta nám bude sloužit v naší aktivitě (při programování) jako identifikátor, o jaký prvek se jedná a jak se má použít. Pokud se chceme dozvědět více, doporučoval bych si nastudovat odbornou literaturu. Pro začátečníky je podstatné, že to takhle prostě funguje. Jediné, co nás zajímá, je text za lomítkem. Jedná se v podstatě o jméno prvku (v našem případě button1), který budeme používat. Jméno je libovolné, ale vyhnul bych se různým znakům (%, = a podobně), protože Eclipse může začít zlobit. Pro přehlednost bych doporučil označovat tlačítka podle jejich funkce (pro potvrzovací tlačítko použít OK a podobně). Další dva řádky slouží k nastavení délky a šířky prvku. Ta může obsahovat číslovku (Například 150dp. Dp je nutno uvést, jedná se o virtuální jednotku používanou androidem. Pracuje s fyzickou velikostí displeje a přepočítává dp na pixely), wrap_content, to znamená, že prvek bude tak velký, aby obsáhl svůj obsah, jako text a podobně, nebo match_parent, kde se prvek roztáhne na celou obrazovku (do verze Androidu 2.2.x existovalo fill_parent, což je to samé a i tam funguje match_parent). Poslední řádek je text na tlačítku (OK, konec, zpět …). Po napsání kódu se nám prvek zobrazí v GUI.

Druhá a jednodušší možnost přidání prvku je tato. Prvek jednoduše přetáhneme z palety na obrazovku a XML kód se vygeneruje automaticky. Tím se dá i rozvrhnout pozice prvků na obrazovce. Vygenerovaný kód vypadá takto:

```
<Button
        android:id="@+id/konec"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="47dp"
        android:text="Button" />
```

Ostatní přidané řádky slouží k umístění prvku na obrazovce. Kde se prvek nachází ve vztahu s obrazovkou. Samozřejmě i tyto řádky můžeme napsat sami, pokud víme, kde se prvek bude nacházet.



Obrázek 7 - Eclipse - příklady tlačítek

Po přidání prvku otevřeme naši aktivitu. Ta je ve složce src/. Po otevření uvidíme automaticky vygenerovaný kód, do kterého připisujeme vlastní řádky.

Pro tlačítko je to například takto:

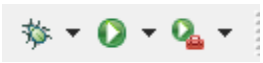
```
protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            Button konec = (Button) findViewById(R.id.konec);
            konec.setOnClickListener(new View.OnClickListener() {

                  public void onClick(View v) {
                        finish();
                  }
            });
      }
```

Pokud to takhle necháme, Eclipse nám ohlásí chybu. A to z důvodu, že je nutné doplnit import na začátek projektu k ostatním importům, co tam již jsou. Je to z důvodu toho, aby Eclipse věděl jak má s tímto prvkem pracovat. Import vypadá takto:

```
import android.widget.Button;
```

Tento import se provádí u všech položek (button, textview…), které používáme. Lze ho buď doplnit ručně, to znamená napsat ho k ostatním importům, a nebo je zde možnost klávesové zkratky CTRL + Shift + O, která nám doplní všechny používané importy automaticky.

Nyní nám Eclipse nehlásí žádnou chybu a můžeme si to vyzkoušet. Na spuštění aplikace se používají tyto symboly:



Obrázek 8 – Eclipse - spustění aplikace

Po spuštění se nám objeví tabulka. Tam se musí zvolit Android Application. Aby Eclipse věděl, co má použít. Poté se buď spustí námi již dříve vytvořený emulátor, a nebo pokud je připojený mobilní telefon, či jiné zařízení, tak se nás zeptá, kterou možnost chceme použít. Jednu si zvolíme a aplikace se nám spustí na vybraném zařízení. Tato aplikace nedělá nic světoborného. Po stisku tlačítka se aplikace zavře. Jde spíše o pochopení práce s tímto prvkem a samotným programem Eclipse.

4.2.3 Textová pole a popisky (EditText a TextView)

Přidání prvku je stejné jako u tlačítka. Již se nebudeme zabývat psaním v XML, ale pouze druhým způsobem přetažení na layout. Nejprve si ukážeme TextView (popisek). Na výběr máme čtyři možnosti, a to sice podle velikosti popisku. Pokud zvolíme základní TextView, potom bude kód v XML vypadat takto:

```
<TextView
        android:id="@+id/vysledek1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/textView11"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="17dp"
        android:text="Large Text"
        android:textColor="#FF0000"
```

Tyto řádky jsou obdobné jako u tlačítka. Je zde navíc barva textu. Ta se zapíše pomocí hexadecimálního kódu (v tomto případě červená). My změníme id prvku (pokud chceme) a text, který nám zobrazuje. Změna textu se dá opět nastavit dvěma způsoby. První je takový, že se v XML přepíše hodnota na řádku android:text. Druhý způsob je nastavení hodnoty přímo v kódu aplikace. To samé se týká i barvy textu. A to například takto:

```
protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            TextView text= (TextView) findViewById(R.id.textView1);
            text.setText(" Ahoj Světe! ");
            text.setTextColor(Color.RED);
      }
```

A opět je důležité připsat import na začátek kódu, to se provádí vždy. Vkládání importů je nejjednodušší pomocí klávesové zkratky zmíněné v kapitole s tlačítkem:

```
import android.widget.TextView;
```

Obrázek 9 – Eclipse - příklady popisků

A teď se podíváme na EditText (textové pole). Na výběr je opět více možností. Jsou tu pole na zadávání textu, nebo pouze číslic (celých i desetinných) a mnoho dalších, které zadá uživatel a aplikace ho zpracuje podle naprogramovaného kódu. Ukážeme si pouze práci s textem a číslicemi. Prvek přidáme na obrazovku a vygenerovaný XML kód vypadá takto:

```
<EditText
        android:id="@+id/editText1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/editText2"
        android:layout_alignParentTop="true"
        android:layout_marginTop="127dp"
        android:ems="10"
        android:text="Zadej text" />
```

Opět se přepneme do MainActivity.java a ukážeme si základní použití těchto textových polích. V kódu lze získat informaci jak textovou, tak i číselnou. V případě čísel je nutné získaný údaj „ naparsovat“. A to v případě celého čísla (int) i desetinného (double). Není to nic složitého.

Příklad je zde:

```
protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            EditText Text= (EditText) findViewById(R.id.editText1);
            String ZiskanyText = Text.getText().toString();
      try   {


            EditText Cislo = (EditText) findViewById(R.id.editText2);
            double ZiskaneCisloDouble =
Double.parseDouble(Cislo.getText().toString());

            int ZiskaneCisloInt =
Integer.parseInt(Cislo.getText().toString());

      }catch (Exception e){
            // co se má vykonat pokud nelze převést na číslo
      }
      }
```

Získávání čísel bude takto fungovat bez problémů. Pokud zvolíme vhodný EditText, kam můžeme zadávat pouze čísla (celá nebo desetinná). Jestli zvolíme pouze základní, kam lze vkládat i text, je nutné, aby se získávání čísla zkoušelo, jestli lze převést tento text na číslo. To se provádí pomocí try a catch viz přiložený kód.

Obrázek 10 – Eclipse - příklady textových polích

4.2.4 Zaškrtnutelná políčka (CheckBox)

Tento prvek má dva stavy. Zaškrtnuto a nezaškrtnuto. Na každou z těchto akcí lze napsat, co má aplikace udělat. Přidáme si prvek a podíváme se jak na to. Xml kód vypadá takto:

```
<CheckBox
        android:id="@+id/checkBox1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="104dp"
        android:text="CheckBox" />
```

Neobsahuje žádné novinky, které již neznáme. Jediná neznámá věc je kontrola, jestli je políčko zaškrtnuté nebo není, a to pomocí metody onCheckedChanged.

```
public class MainActivity extends Activity implements
OnCheckedChangeListener {

      CheckBox check;

      @Override
      protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

            check=(CheckBox) findViewById(R.id.checkBox1);
            check.setOnCheckedChangeListener(this);
      }
      @Override
      public void onCheckedChanged(CompoundButton buttonView, boolean
isChecked) {
            if (isChecked){
                  check.setText(" Toto políčko je zaškrtnuté ");
            }
            else{
                  check.setText(" Toto políčko je odškrtnuté ");
            }
      }
```

Nejdůležitější je nezapomenout připsat do public class MainActivity položku implements OnCheckedChangeListener a zároveň vytvořit proceduru public void onCheckedChanged. Pro začátečníky není podstatné, proč to tak je. Pokud se i přesto chceme dozvědět více, doporučuji navštívit [5] nebo [6]. Tyto stránky jsou sice v anglickém jazyce, ale je zde vše přehledně vysvětleno. Nezapomenout přidat importy a můžeme aplikaci spustit a otestovat.

Výsledná aplikace vypadá takto:

Obrázek 11 - Eclipse - použití CheckBoxu

4.2.5 Kulaté přepínače (RadioButton)

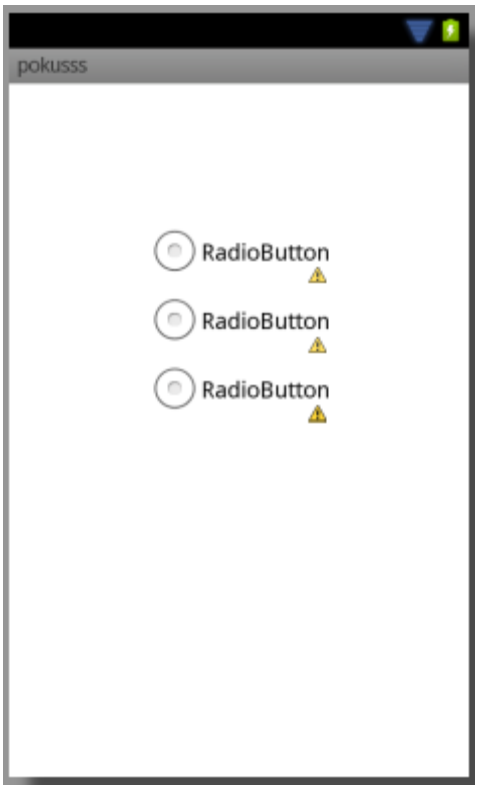
Tento prvek je velmi podobný předchozímu prvku. Jediný rozdíl je v tom, že jednotlivé prvky RadioButton lze seskupovat do RadioGroup. To nám poslouží na výběr jedné možnosti z vícero (jedna ze tří a podobně). Pokud chceme seskupit více přepínačů, musíme do XML připsat začátek a konec RadioGroup. Příklad zde:

```
<RadioGroup >
    <RadioButton
        android:id="@+id/radioButton1"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="94dp"
        android:text="RadioButton" />
    <RadioButton
        android:id="@+id/radioButton2"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/radioButton1"
        android:layout_below="@+id/radioButton1"
        android:text="RadioButton" />
    <RadioButton
        android:id="@+id/radioButton3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignLeft="@+id/radioButton2"
        android:layout_below="@+id/radioButton2"
        android:text="RadioButton" />
</RadioGroup>
```

Po přidání RadioGroup v našem XML návrhu můžeme ke skupině prvků přistupovat v kódu jazyka Java a volat pro ni následující metody:

- check() – Označí specifický přepínač (například group.check(R.id.radio1).
- clearCheck() – Zruší označení všech přepínačů ve skupině, takže žádný z nich nezůstane označený.
- getCheckedRadioButtonId() – Vrátí Id aktuálně označeného přepínače (nebo -1, pokud není označený žádný přepínač).

Metoda check() se používá hlavně proto, že při spuštění aplikace není žádný přepínač označený. A pokud s nimi programátor pracuje v kódu. Mohly by nastat potíže, že není ani jeden označený, tímto se problémům předejde. Samotné programování v Javě je obdobné jako u CheckBoxu, a proto si ho nebudeme ukazovat.



Obrázek 12 – Eclipse - příklad RadioButtonu

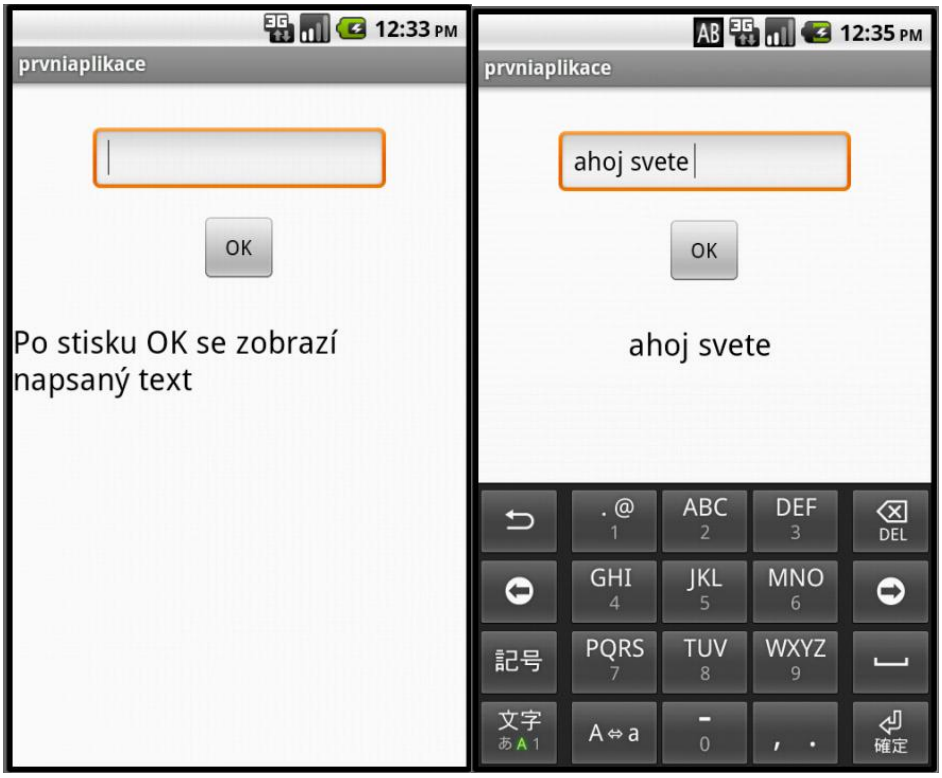
4.3 Vytvoření první aplikace

Teď už bychom měli být připraveni na napsání vlastní první aplikace. Pro začátek si vytvoříme aplikaci, která bude obsahovat textové pole (EditText), tlačítko (Button) a popisek (TextView). Tato aplikace bude dělat pouze toto: Po zadání textu do textového pole a stisknutí tlačítka se nám tento text zobrazí v popisku. Není to náročná aplikace, ale bude sloužit jako návod pro složitější programy.

Začneme založením nového projektu (kapitola 4.1) a nazveme ho prvniaplikace. Po vytvoření projektu se nám otevře layout projektu. Přidáme do něj jednotlivé prvky. Začneme s textovým polem (EditText). Pod něj vložíme tlačítko (Button) a popisek (TextView). To je naše veškerá práce v layoutu. Poté si otevřeme Java kód a začneme programovat. Neměl by to být žádný problém. Začneme s tím, že najdeme jednotlivé prvky z layoutu a přidáme jim jméno, které budeme používat v programu. Pak už stačí napsat jen pár řádků a aplikace bude funkční. Kód XML si neukážeme a není pro tuto aplikaci potřebný. Zde je uveden Java kód funkční aplikace:

```
package com.example.prvniaplikace;
import android.app.Activity;
import android.os.Bundle;
import android.view.Menu;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

public class MainActivity extends Activity {
	EditText TextovePole;
	TextView text;
	Button OK;
	@Override
	protected void onCreate(Bundle savedInstanceState) {
		super.onCreate(savedInstanceState);
		setContentView(R.layout.activity_main);
		TextovePole= (EditText) findViewById(R.id.editText1); //
Nalezení EditText v layoutu
		OK = (Button) findViewById(R.id.button1);// Nalezení tlačítka
v layoutu
		text= (TextView) findViewById(R.id.textView1); // Nalezení
popisku v layoutu

		text.setText("Po stisku OK se zobrazí napsaný text");//
Nastaveni textu popisku

		OK.setText("  OK  ");   // Nastaveni textu tlačítka
		OK.setOnClickListener(new View.OnClickListener() {

			public void onClick(View v) {       // Akce po stisknutí
tlačítka

				text.setText(TextovePole.getText().toString());
// Nastevení textu TextView získáným textem z EditText
			}
		});
	}
```

Vše uložíme a spustíme. Tato aplikace běžící na zařízení může vypadat například takto:



Obrázek 13 - První aplikace - testování

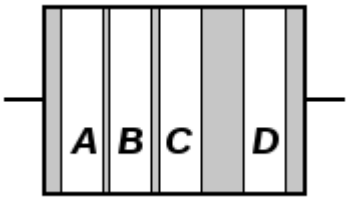
To, jak bude vypadat, záleží pouze na nás. Máme na výběr, jak jednotlivé komponenty rozmístíme. Můžeme změnit i barvu textu, pozadí prvků nebo samotné pozadí aplikace. Nic nám nebrání vše změnit.

5 Funkční aplikace

5.1 Barevné značení rezistorů

5.1.1 Jak se určuje hodnota

Při barevném značení rezistorů (i jiných elektronických součástek) se používá mezinárodní kód podle normy EIA-RS-279. Údaje o hodnotě rezistoru jsou zakódovány mezi barevné proužky. Počet proužků je čtyři nebo pět. Poslední proužek uvádí toleranci rezistoru. Ostatní určují jeho hodnotu.[9]



Obrázek 14 - Schéma pro určování proužků rezistoru [9]

- Pruh A – první platná číslice hodnoty rezistoru v ohmech
- Pruh B – druhá platná hodnota
- Pruh C – desítkový násobitel
- Pruh D – jedná se o toleranci, pokud není uveden, znamená to toleranci 20%

Obdobné je to i pro rezistory s pěti proužky. Ty jsou mnohem přesnější. Určování hodnoty je stejné jako u čtyř proužků. Pouze je přidán další proužek mezi druhý a násobitele.

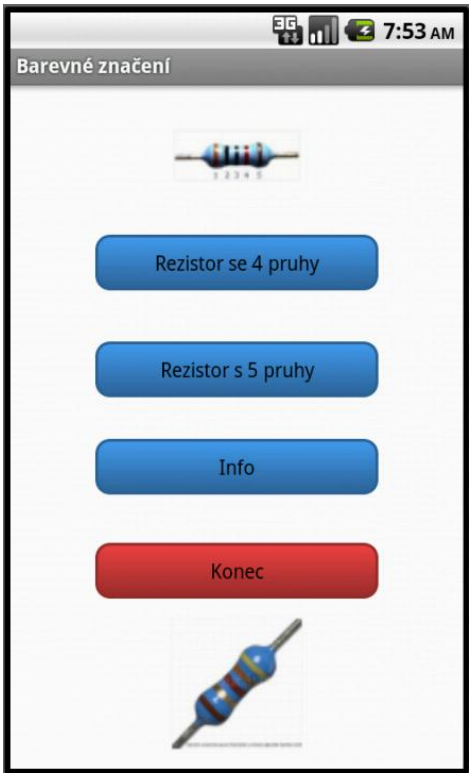
Tabulka 1 - Barevné značení - Tabulka

Barva	1. pruh	2. pruh	3. pruh	Násobitel	Tolerance
Černá	0	0	0	$x10^0$	
Hnědá	1	1	1	$x10^1$	±1% (F)
Červená	2	2	2	$x10^2$	±2% (G)
Oranžová	3	3	3	$x10^3$	
Žlutá	4	4	4	$x10^4$	
Zelená	5	5	5	$x10^5$	±0.5% (D)
Modrá	6	6	6	$x10^6$	±0.25% (C)
Fialová	7	7	7	$x10^7$	±0.1% (B)
Šedá	8	8	8	$x10^8$	±0.05% (A)
Bílá	9	9	9	$x10^9$	
Zlatá				x0.1	±5% (J)
Stříbrná				x0.01	±10% (K)

5.1.2 Aplikace na určování hodnoty

Ukážeme si, jak tato vytvořená aplikace pracuje. Nebudu uvádět zdrojové kódy. Ty budou přibaleny na přílohu DVD a odtud je možné je použít. Celá aplikace je tvořena ze tří aktivit (obrazovek). První aktivita je úvodní obrazovka, která slouží jako rozcestí ke zbylým dvěma aktivitám.

Aplikace běžící na zařízení potom vypadá takto:



Obrázek 15 - Barevné značení - úvodní obrazovka

První a druhé tlačítko nám spouští zbylé aktivity. A to sice podle počtu pruhů na rezistoru. Tlačítko Info nám zobrazí informace o aplikaci. Tlačítkem Konec se aplikace zavře.

Obrázek 16 - Barevné značení - úvodní obrazovka - Info

Na této obrazovce si vybereme, kolik proužků má náš rezistor. Poté se objeví další obrazovka s možností měnit barvy podle součástky, kterou vybereme ze seznamu. Obrazovky se od sebe moc neliší, pouze u výběru pěti proužků je na výběr o řádek více. V horní části obrazovky se nám ukazuje výsledný navolený odpor i s tolerancí, kterou zadáme. Je zde i možnost zvolit si jednotky, ve kterých se bude hodnota zobrazovat. Aktivity vypadají takto:

Obrázek 17 - Barevné značení - Rezistor se čtyřmi proužky

Obrázek 18 - Barevné značení - Rezistor s pěti proužky

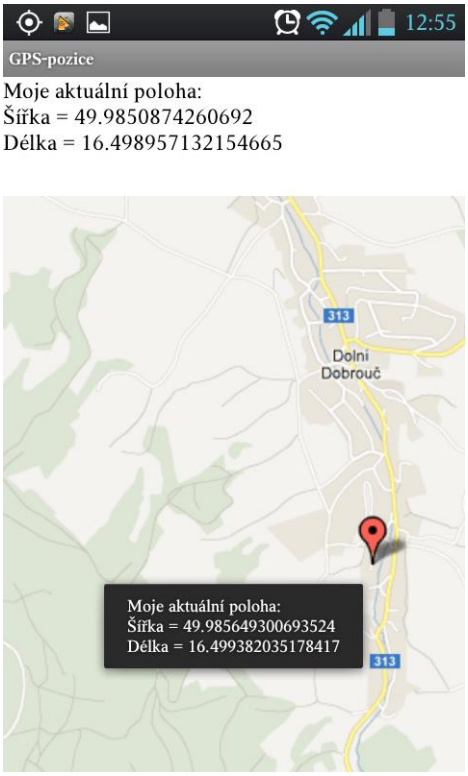
5.2 Aplikace na použití GPS

O systému GPS se zde nebudeme zmiňovat. Toto téma by bylo spíše vhodné jako samostatná práce. My si pouze ukážeme aplikaci na využití určování polohy. Zdrojové kódy budou opět přiloženy na DVD. Pouze upozorním na jednu důležitou věc, a to sice že je potřeba povolit využití GPS v manifestu androidu (viz 2.2.4). To se provede přidáním těchto řádků do manifestu:

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

První řádek povoluje využití systému GPS a druhý povoluje připojení na internet.

Aplikace na telefonu potom vypadá takto. Neobsahuje žádné interaktivní prvky, zobrazuje pouze informace s GPS a následnou mapu s polohou.



Obrázek 19 – GPS - Nalezení polohy a zobrazení na mapě

5.3 Aplikace na použití Bluetooth

Tato aplikace slouží k ukázce funkčností Bluetooth. Na obrazovce můžeme zvolit, zda chceme Bluetooth zapnout nebo vypnout. Poté je tu možnost hledání zařízení a následné spárování s jiným zařízením. Zároveň je tu možnost zviditelnit zařízení pro ostatní po určitou dobu. V tomto případě se jedná o 300s. Zdrojový kód bude opět přiložen na DVD. Pouze opět zmíním důležitou část, a to povolení Bluetooth v manifestu. Přidají se následující řádky:

```
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
<uses-permission android:name="android.permission.BLUETOOTH" />
```

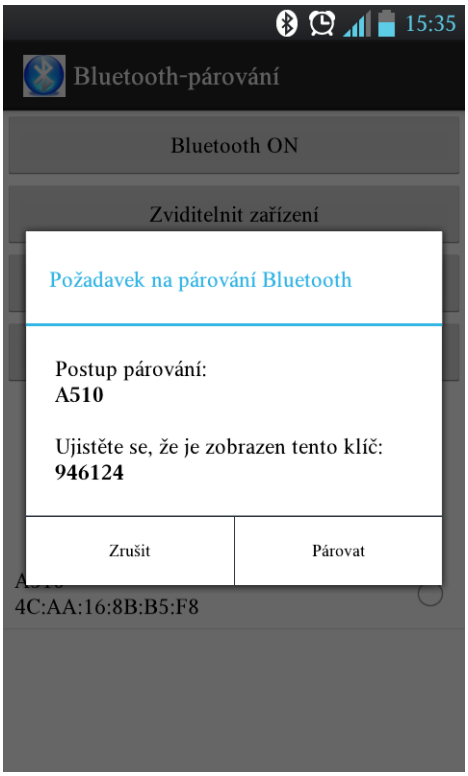
Běžící aplikace potom vypadá takto:

Obrázek 20 – Bluetooth - Hledání zařízení

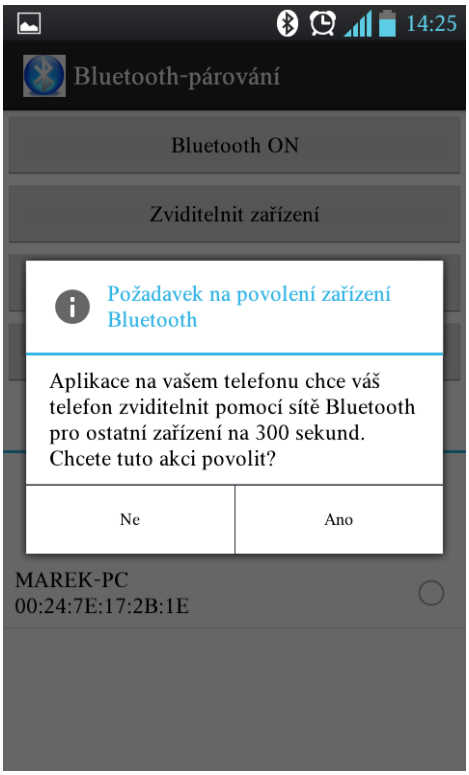
První a poslední tlačítko slouží k zapnutí a vypnutí Bluetooth. Třetí tlačítko používáme k hledání ostatních zařízení, které mají tuto funkci zapnutou. V dolní části obrazovky vidíme nalezená zařízení, se kterými se může telefon spárovat. V Java kódu je zapínání, vypínání a zviditelnění zařízení vcelku jednoduché. Slouží k tomu jen pár příkazů. Ty vypadají takto :

```
    private void onBluetooth() {
        if(!bluetoothAdapter.isEnabled())
    {
    bluetoothAdapter.enable();  // zapínání bluetooth, pokud není zapnuté
            Log.i("Log", "Bluetooth is Enabled");
     }
    }
    private void offBluetooth() {
        if(bluetoothAdapter.isEnabled()) // vypínání bluetooh, pokud je
zapnuté
        {
            bluetoothAdapter.disable();
        }
    }
    private void makeDiscoverable() { // povolení zviditelnění zařízení,
po dobu 300 vteřin.
   Intent discoverableIntent = new
Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
discoverableIntent.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION,
300);
        startActivity(discoverableIntent);
        Log.i("Log", "Discoverable ");
    }
```

Pokud klikneme na jméno zařízení, které chceme spárovat s telefonem, objeví se nám následující hláška s výběrem, zda je chceme spárovat, a nebo akci zrušit (obr. 21). Obdobná akce je i na stisk tlačítka pro zviditelnění zařízení. I zde máme na výběr, jestli to chceme povolit nebo ne (obr. 22).



Obrázek 21 – Bluetooth – Párování



Obrázek 22 – Bluetooth - Zviditelnit zařízení

Závěr

Tato práce se zabývala operačním systémem Android. I přes to, že se jedná o relativně mladý systém, jeho start by se dal považovat za raketový a to především na trhu s mobilními telefony. Společnost vznikla na začátku tohoto tisíciletí a dnes je již téměř každý druhý prodaný telefon právě s tímto systémem. Oblíbenost je celkem jasná. Je to ve své podstatě jednoduchý systém, snadný na používání a velmi intuitivní. První verze samozřejmě měly svoje vady, ale dalo by se říci, že od verze 2.2.x je to velice dobrý a stabilní systém, který se stále vyvíjí a vylepšuje. Asi největší výhodou je to, že se jedná o otevřený systém, a proto není problém na něj vyvíjet aplikace a jiný software. Díky tomu se dá objevit i mnoho zdrojových kódů a příkladů na to, jak začít.

Na úvodních stránkách nalezneme základní informace a hlavně je zde postup, jak správně nainstalovat vývojové prostředí. Dalo by se říci, že je to ta nejtěžší věc při vývoji aplikací na systém Android. Já osobně jsem musel zkoušet mnoho postupů jak na to, než-li jsem našel ten správný, který fungoval. Samotný vývoj pak není nijak složitý. Samozřejmostí je ovšem mít zkušenosti s programováním v Java, na jejíchž základech je systém postaven, ale i ti, co Javu nikdy neviděli, nebudou mít problém se do toho rychle dostat.

Výsledkem této práce jsou tři plně funkční aplikace. První a stěžejní je aplikace na určování hodnot rezistoru podle jejich barevného označení. Každý, kdo pracoval s těmito součástkami, určitě narazil na problém, že si rezistory zamíchal do sebe a potom nevěděl, jakou hodnotu má. Jediná možnost byla najít si tabulku a sám si spočítat velikost. To samozřejmě stojí nějaký čas. K tomu účelu je tu tato aplikace. Pouze si navolíme barvy proužků a hned vidíme výslednou hodnotu a svůj čas zaměříme na něco jiného. Druhá aplikace slouží ke zjištění polohy pomocí GPS. Jedná se spíše o ukázku, jak se s GPS pracuje. Aplikací je mnoho a samotný software od Googlu Google Map je dle mého názoru velmi povedený. Poslední třetí aplikace pracuje s rozhraním Bluetooth. Je zde možnost zapnutí, vypnutí a zviditelnění telefonu pro ostatní zařízení. Další funkce je spárování a připojení k jiným zařízením. To bylo testováno na Bluetooth handsfree. Řešení těchto aplikací mohlo být i jiné. Ať už jde o vzhled, nebo i funkčnost. Uvažuji o tom, že by se mohly objevit i na marketu ke stažení pro všechny uživatele. Nápady na vytvoření aplikací se těžko shánějí, ale v oboru informatiky a elektrotechniky je mnoho úkonů, které by šly tímto zjednodušit. Ať už jde o počítání sériových nebo paralelních zapojení součástek, počítání filtrů a nebo nastavení pracovního bodu tranzistoru a mnoho dalšího.

Vývoj aplikací pro operační systém Android je relativně lehký. A jelikož se dá vymyslet aplikace v podstatě na cokoliv, tak se budou vyvíjet stále další a další. Některé nahradí ty zastaralé a jiné budou naopak úplně nové. A i kdyby vyšly podobné, bude každá jinak graficky řešená a je už na uživateli, kterou vybere jako lepší. Android má před sebou ještě velkou budoucnost a dle mého názoru se poptávka bude ještě zvyšovat, stejně jako tomu bylo doposud.

Literatura

[1] Mark L. Murphy : Android 2 – Průvodce programováním mobilních aplikací; Computer Press, a.s 2011; ISBN 978-80-251-3194-7

[2] Android (operační systém) [online]. Dostupný z WWW: http://cs.wikipedia.org/wiki/Android_(operační_systém) (15.1.2013)

[3] Vyvíjíme pro Android –úvod [online]. Dostupný z WWW: http://www.abclinuxu.cz/clanky/vyvijime-pro-android-uvod (19.1.2013)

[4] Vyvíjíme pro android: Začínáme [online]. Dostupný z WWW: http://www.zdroják.cz/clanky/vyvijime-pro-android-zaciname/ (19.1.2013)

[5] Android developer (anglicky) [online]. Dostupný z WWW: http://developer.android.com/index.html (20.1.2013)

[6] Android development (anglicky) [online]. Dostupný z WWW: http://www.anddev.org/index.php (20.1.2013)

[7] Elitec Software – Historie verzí platformy Android [online]. Dostupný z WWW: http://www.elitecsoftware.cz/historie-verzi-platformy-android/ (14.2.2013)

[8] Mobil idnes – Naučíme vás programovat aplikace pro android [online]. Dostupný z WWW: http://mobil.idnes.cz/naucime-vas-programovat-aplikace-pro-android-zaciname-prave-dnes-phe-/aplikace.aspx?c=A120410_125436_aplikace_ham (14.2.2013)

[9] Barevné značení elektronických součástek [online]. Dostupný z WWW: http://cs.wikipedia.org/wiki/Barevné_značení_elektronických_součástek (20.2.2013)

Příloha A – DVD

1. Obsahuje celou práci ve formátu PDF.
2. Veškeré zdrojové kódy, na které se odkazuji (celková dokumentace ke všem naprogramovaným aplikacím).
3. Adresáře s projekty pro import do programu Eclipse

Příloha B – Zdrojový kód použití GPS

```
public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_gps);

            position = (WebView) findViewById(R.id.webView1);
            position.getSettings().setJavaScriptEnabled(true);

            textInfo = (TextView) findViewById(R.id.textView1);
            textInfo.setText("Pozice : ");

            LocationManager locManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
            //získání pozice
            LocationListener locListener = new MyLocationListener();

locManager.requestLocationUpdates(LocationManager.GPS_PROVIDER, 0, 0,
                        locListener);
      }

      public class MyLocationListener implements LocationListener {
            @Override
            public void onLocationChanged(Location location) {

                  // Načítání Latitude (šířky)
                  location.getLatitude();
                  // Načítání Longitude
                  location.getLongitude();

                  textInfo.setText("");
                  String text = "Moje aktuální poloha:\nŠířka = "
                              + location.getLatitude() + "\nDélka = "
                              + location.getLongitude();
                  textInfo.setText(text);
                  Toast.makeText(getApplicationContext(), text,
Toast.LENGTH_SHORT)
                              .show();

                  // nastavení Google Map na webview
                  String url = "http://maps.google.com/staticmap?center="
                              + location.getLatitude() + "," +
location.getLongitude() +"&zoom=14&size=512x512&maptype=mobile/&markers="
+ location.getLatitude() + "," + location.getLongitude();
                  position.loadUrl(url);
            }
            @Override
            public void onProviderDisabled(String provider) {
                  Toast.makeText(getApplicationContext(), "GPS vypnuto",
                              Toast.LENGTH_SHORT).show();
            }
            @Override
            public void onProviderEnabled(String provider) {
                  Toast.makeText(getApplicationContext(), "GPS zapnuto",
                              Toast.LENGTH_SHORT).show();
            }
```