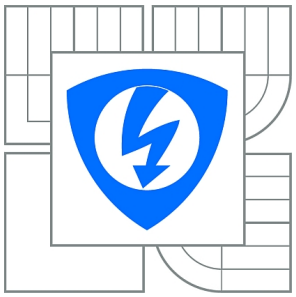
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

ÚSTAV ELEKTROTECHNOLOGIE

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

DEPARTMENT OF ELECTRICAL AND ELECTRONIC TECHNOLOGY

VYUŽITÍ MOBILNÍ PLATFORMY ANDROID K OVLÁDÁNÍ PŘÍSTROJOVÉ TECHNIKY

USE OF MOBILE PLATFORMS ANDROID FOR INSTRUMENTS CONTROL

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

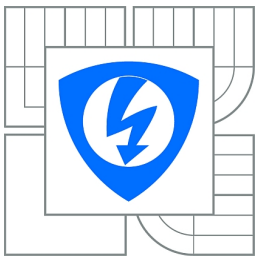
AUTOR PRÁCE — MICHAL MACEK

AUTHOR

VEDOUCÍ PRÁCE — Ing. MARTIN FRK, Ph.D.

SUPERVISOR

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky a komunikačních technologií

Ústav elektrotechnologie

Bakalářská práce

bakalářský studijní obor
Mikroelektronika a technologie

Student:	Michal Macek	***ID:***	147441
Ročník:	3	***Akademický rok:***	2013/2014

NÁZEV TÉMATU:

Využití mobilní platformy Android k ovládání přístrojové techniky

POKYNY PRO VYPRACOVÁNÍ:

Analyzujte historický vývoj a aktuální možnosti využití vzdáleného přístupu a ovládání přístrojové techniky prostřednictvím sítě Internet. Seznamte se s libovolným vývojovým prostředím a osvojte si programovací jazyk určený pro tvorbu mobilních aplikací založených na platformě Android.
V laboratoři realizujte připojení vybraných měřicích přístrojů, vybavených různými druhy komunikačních rozhraní, do sítě Internetu a naprogramujte univerzální aplikaci pro nastavení jejich parametrů a pro ovládání prostřednictvím SCPI příkazů pomoci příkazového řádku. Současně se soustřeďte i na vytvoření další mobilní aplikace, která by umožňovala ovládání funkčního generátoru a osciloskopu a využívala uživatelsky přívětivějšího obslužného prostředí. K příslušným aplikacím vytvořte návod, včetně detailního popisu jednotlivých programátorských kroků. Praktickou funkčnost mobilních aplikací demonstrujte na mobilním telefonu.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího bakalářské práce.

Termín zadání:	10.2.2014	***Termín odevzdání:***	5.6.2014

Vedoucí práce: Ing. Martin Frk, Ph.D.
Konzultanti bakalářské práce:

doc. Ing. Jiří Háze, Ph.D.
Předseda oborové rady

UPOZORNĚNÍ:

ABSTRAKT:

Předkládaná práce popisuje aktuální trendy a moderní metody využívající mobilní zařízení k ovládání laboratorních přístrojů a následnému měření. Hlavním cílem je vytvoření aplikace pro mobilní zařízení s operačním systémem Android, díky které bude možné vzdáleně ovládat a měřit s přístroji v laboratoři. Praktická činnost je zaměřena na naprogramování mobilní aplikace, která dokáže laboratorní přístroj na dálku kontaktovat, nastavit pro různé druhy měření podporované přístrojem a následné čtení naměřených dat z přístroje do mobilního zařízení.

ABSTRACT:

The present work describes current trends and modern methods using mobile devices to operate laboratory equipment and subsequent measurements. The main objective is create application for mobile devices running on Android system, which will be remotely controlled and measured with instruments in the laboratory. Practical activities are focused on programming mobile application that can remotely laboratory device contact, set for different types of measurements supported device and then read the measurement data from the device to a mobile device.

KLÍČOVÁ SLOVA:

Android, aplikace, měřící přístroj, vzdálené připojení, vzdálené ovládání, měření, programování, čtení dat.

KEYWORDS:

Android, applications, measuring instrument, remote access, remote control, measurement, programming, reading data

BIBLIOGRAFICKÁ CITACE DÍLA:

MACEK, M. *Využití mobilní platformy Android k ovládání přístrojové techniky*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 58 s. Vedoucí bakalářské práce Ing. Martin Frk, Ph.D., FEKT VUT v Brně

PROHLÁŠENÍ AUTORA O PŮVODNOSTI DÍLA:

Prohlašuji, že svou bakalářskou práci na téma Využití mobilní platformy Android k ovládání přístrojové techniky jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této bakalářské práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a~jsem si plně vědom následků porušení ustanovení § 11 a následujících zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne ....
(podpis autora)

PODĚKOVÁNÍ:

Děkuji vedoucímu bakalářské práce Ing. Martinu Frkovi Ph.D. za metodické a cíleně orientované vedení při plnění úkolů realizovaných v návaznosti na bakalářskou práci. Dále děkuji ústavu Elektrotechnologie za poskytnutí prostoru k realizaci experimentálních prací, zapůjčení měřících přístrojů a za vstřícnost při řešení nastalých problémů.

V Brně dne ....
(podpis autora)

OBSAH

SEZNAM OBRÁZKŮ

ÚVOD

V posledních letech se setkáváme s rozšířením chytrých mobilních zařízení a s tím jde ruku v ruce vývoj aplikací pro tato zařízení. Otevřený zdrojový kód operačního systému a možnost použití zdarma je jasnou výhrou na trhu s mobilními zařízeními. Tento chytrý tah má na svědomí společnost Google, která vydala operační systém Android.

Android byl veřejností velice dobře přijat, to hlavně díky otevřenosti systému a rychle přibývajícímu počtu aplikací různého druhu. Také jeho funkčnost a podpora různých zařízení napomohla k masovému rozšíření.

Aplikace pro Android může vyvíjet téměř kdokoliv, kdo umí alespoň základy programování. Výhoda aplikací u tohoto operačního systému je ve funkčnosti mezi různými verzemi systému a jejich rozmanitost. Také díky rychlému vývoji elektroniky a s tím souvisejícím rostoucím výkonem zařízení můžeme na mobilních zařízeních provozovat aplikace téměř stejné jako na klasických počítačích.

Využití mobilní platformy Android k ovládání přístrojové techniky je krokem kupředu. Tento krok otevírá velké možnosti a prostor pro improvizaci a zaplnění určitého místa na trhu s aplikacemi. Aplikace vytvořená na míru vybraného přístroje je mocnou zbraní a pomocníkem zároveň.

1. OPERAČNÍ SYSTÉM ANDROID

1.1. Co je to Android?

Operační systém Android je open source platforma, kterou vytvořila společnost Google, jedná se tedy o počítačový software. Díky otevřenému zdrojovému kódu je licenčně a technicky snadno dostupný. Uživatel může využívat systém zadarmo a má přístup ke zdrojovým kódům, pokud splní určité podmínky použití softwaru. Operační systém běží na jádru Linuxu různých verzí, který zabezpečuje funkčnost systému jako celku, správu paměti, správu procesů, přístup ke službám, ovladačům a různých komponent. Jednotlivé aplikace k funkcím jádra nepřistupují přímo, ale pomocí Android API. Android byl hlavně vyvíjen jako progresivní operační systém pro PDA, tablety a mobilní telefony. Systém umožňuje vývojářům vytvářet různorodé aplikace. Díky použití otevřeného jádra Linux jako vlastního virtuálního stroje, dokáže systém optimalizovat paměť a hardwarové prostředky. Vzhledem ke spolupráci vývojářské komunity má tento systém veliké předpoklady k dalšímu rozvoji. Od první verze bylo vydáno několik aktualizací, které opravují chyby a přidávají nové funkce. Od verze 1.5 má každá verze své vlastní jméno podle zákusků (Cupcake, Donut, Eclair atd.) [1]

1.2. Historie verzí

1.2.1. Android 1.0

Přišel na svět 23. září 2008 a jako první zařízení na světě ho měl HTC Dream (T-Mobile G1). Běžel na jádru Linuxu [1] 2.6.25. Do zařízení bylo možné stáhnout pomocí Android Marketu různé aplikace, kterých v té době bylo ještě málo. Jeho základní funkce byly:

- Webový prohlížeč pro zobrazování, posouvání a přibližování HTML [2] i XHTML[3] stránek s podporou zobrazení stránek v tzv. kartách.
- Fotoaparát bez rozšiřujících funkcí.
- Android Market - on-line obchod pro stažení a nákup aplikací.
- Emailový klient.
- Schopnost synchronizovat kontakty a kalendář pomocí Google Sync.
- Google mapy s funkcí určení polohy pomocí GPS[4].
- Media player pro přehrávání video soborů.
- Youtube přehrávač.
- Budík, kalkulačka, galerie obrázků, hlasové vytáčení.

[1] Linux - jedná se o volně šiřitelný operační systém s otevřeným kódem.

[2] HTML - jedná se o programovací jazyk hypertext, který je hlavním jazykem pro vytváření web. stránek.

[3] XHTML - je rozšiřitelný hypertextový programovací jazyk pro tvorbu web. stránek.

[4] GPS - Globální Polohový Systém, díky družicím je možno určit polohu a přesný čas kdekoliv na zemi.

1.2.2. Android 1.1

Byl oficiálně vydán 9. února 2009. Jednalo se o verzi, která měla za úkol opravit a doplnit pár funkcí jen do zařízení T-Mobile G1, ostatních zařízení se netýkala.

1.2.3. Android 1.5 (Cupcake)

Přišel 30. dubna 2009, byl postaven na Linuxovém jádře 2.6.27 a přinesl několik nových funkcí:

- Možnost nahrávání videa pomocí vestavěného fotoaparátu.
- Animace při přechodu mezi obrazovkami.
- Bluetooth - podpora.
- Nová softwarová klávesnice s automatickým dokončováním slov.
- Rychlejší získání polohy uživatele.
- Rotace aplikací při rotaci zařízení.
- Přehrávání videa ve formátu MPEG-4 a 3GP.
- Nové rozhraní API a Manifest prvků pro vývoj aplikací.

1.2.4. Android 1.6 (Donut)

Vychází nová verze Linuxového jádra 2.6.29 a s ní přichází nová verze Androidu 1.6. Byla uvolněna 15. září 2009 a vylepšena opět o nové funkce:

- Vylepšené grafické rozhraní pro fotoaparát.
- Přidání pole pro rychlé vyhledávání.
- Přidána podpora pro WVGA[5] rozlišení obrazovky.
- Nová verze Android Marketu.
- Vylepšené vyhledávání hlasem.
- Indikátor využití baterie, který umožňuje uživateli zobrazit, které aplikace a služby spotřebovávají nejvíce energie.

1.2.5. Android 2.0/2.1 (Eclair)

Uvolněna ne dlouho po verzi 1.6, ale postavena na stejné verzi jádra. Opět přišlo mnoho změn:

- Inovace kontaktů.
- Podpora více emailových účtů.
- Vylepšená softwarová klávesnice.
- Podpora více velikostí a rozlišení displeje.
- Podpora Bluetooth 2.1.
- Animovaná úvodní tapeta.

[5] WVGA - Wide Video Graphic Array je označení pro různá rozlišení displejů, které mají výšku 480 pixelů. Na šířku mají více jak 640 pixelů. Příklad: 752x480, 800x480 atd.

- Digitální zoom pro foroaparát.
- Podpora HTML5 ve webovém prohlížeči.

1.2.6. Android 2.2 (Froyo)

Na konferenci Google/IO představen 20. května 2010. Tato verze běží na Linuxovém jádře 2.6.32. Byla přidána podpora nových funkcí a nových technologií:

- Vylepšené nastavení fotoaparátu a kamery.
- Možnost vytvoření WiFi hotspotu.
- Sdílení internetového připojení ze zařízení přes USB.
- Vylepšená správa paměti RAM[6].
- Možnost instalace aplikací na paměťovou kartu.

1.2.7. Android 2.3/2.4 (Gingerbread)

Vydán 6. prosince 2010 a běží na Linuxovém jádru 2.6.35. Používá jej většina zařízení nižší třídy. I tato verze přinesla velké množství oprav a funkcí:

- Podpora video formátu WebM[7] a HTML5.
- Nové Google Maps s 3D přístupem.
- Podpora internetové telefonie vrámci protokolu SIP[8].
- Zlepšená funkce kopírování a vložení.

1.2.8. Android 3.0/3.1/3.2 (Honeycomb)

Tato verze je určena výhradně pro Tablety, změny se týkají hlavně úpravy grafického rozhraní a samozřejmě došlo in a několik vylepšení.

- Ovládání bylo přepracováno pro Tablety.
- Není potřeba fyzických tlačítek, vše obstará systém.
- 3D efekty pro přechod mezi obrazovkami.
- Nové Google Maps.
- Vylepšení aplikace YouTube a Gmail.
- Vylepšené widgety[9].

1.2.9. Android 4.0/4.0.1/4.0.2 (Ice Cream Sendwich)

Nová verze pro chytré telefony představena 19. října 2011. Jako u všech verzí i tato přináší novinky jak pro vývojáře, tak i pro uživatele.

---

[6] RAM - paměť s přímým přístupem, jedná se o polovodičové paměti s adresním přístupem.

[7] WebM - otevřený video formát umožňující kompresi pro použití HTML5 videem.

[8] SIP - internetový protokol určený pro přenos signálu v internetové telefonii.

[9] Widget - je základní element pro interakci programu s uživatelem.

- Synchronizace kontaktů se sociálními sítěmi.
- Odemčení telefonu obličejem.
- Vylepšená práce s fotoaparátem.
- Dokonalejší webový prohlížeč.
- Podpora NFC[10] čipů pro rychlý přenos dat mezi zařízeními pomocí funkce Android Beam.
- WiFi Direct pro propojení dvou zařízení.

1.2.10. Android 4.1/4.2/4.3 (Jelly Bean)

Představen na konferenci Google/IO 2012 na zařízení Google Nexus 7. Na podzim roku 2012 vyšla aktualizace na verzi 4.2, další verze byla ohlášena 24. července 2013. Momentálně použita na většině zařízení střední a vyšší třídy.

- Rozdvojená notifikační lišta.
- Nižší požadavky na paměť RAM.
- Možnost tvorby uživatelských účtu pro Tablety.

1.2.11. Android 4.4 (KitKat)

Poslední vydaná verze Androidu, která je momentálně nasazena pouze v Nexus 5. Postupem času se samozřejmě rozšíří mezi ostatní nová zařízení.

Tato verze přinesla změny především v rychlosti a plynulosti. Opět přišlo na lepší využití paměti RAM a lepšího využití potenciálu vícejádrových procesorů. [1]

[10] NFC - bezdrátová komunikace mezi zařízeními, například mezi telefony, pro platby atd.

2. INSTALACE A NASTAVENÍ VÝVOJOVÉHO PROSTŘEDÍ

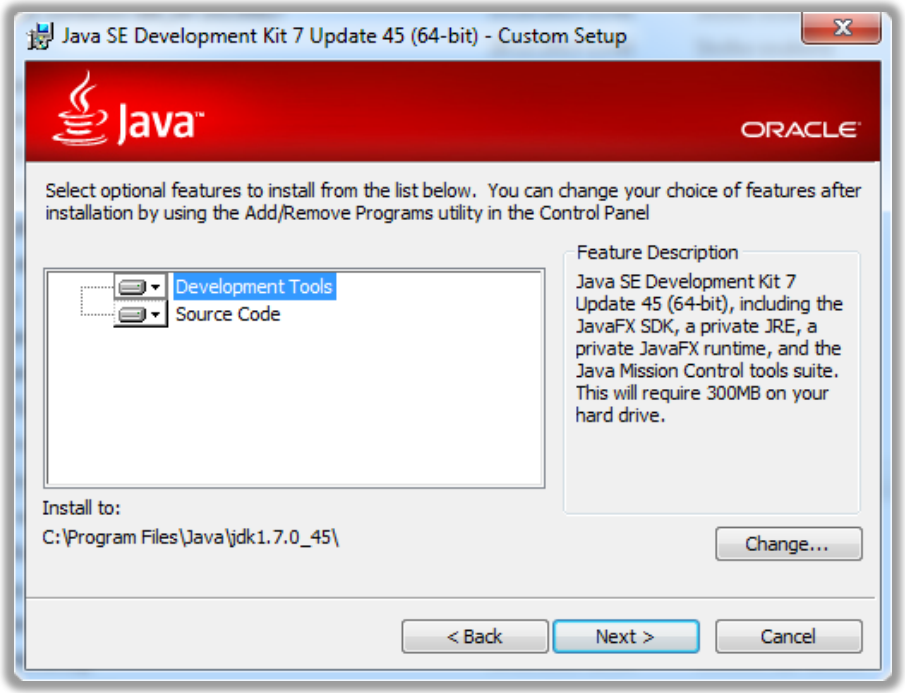
Pro tvoru aplikací je k dispozice velké množství vývojových nástrojů a prostředí. Většina z nich má integrovaný nástroj pro emulaci systému Android. Díky této emulaci má programátor možnost testovat a ladit svoji aplikaci přímo při programování. Odpadá tím zbytečné zdržován s exportováním aplikace z počítače do přístroje a zároveň umožňuje vyvíjet aplikace, i když programátor nevlastní Android zařízení.

2.1. Java Development Kit (JDK)

Jedná se o soubor základních knihoven a nástrojů pro vývoj Java aplikací. Jako hlavní součásti obsahuje Java Runtime Environment (JRE), který slouží pro spuštění aplikací i nástrojů, překladač a debugger. Bez balíčku JDK není vývoj aplikací možný [2].

Aktuální verze je dostupná na adrese:
http://www.oracle.com/technetwork/java/javase/downloads/index.html

Po stažení stačí soubor otevřít a následovat instalačního průvodce (Obr. 2.1).

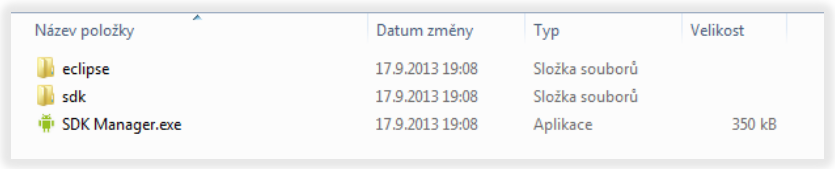


Obr. 2.1: Výběr součástí a cílové složky instalace Java JDK.

2.2. Android SDK (Software Development Kit)

Android SDK obsahuje balíček komponent a program Eclipse IDE s integrovaným ADT [11] (Android Developer Tools). Stažením tohoto balíčku dostane uživatel vše potřebné pro vývoj aplikací [3]. Baliček není potřeba instalovat, stačí jen rozbalit stažený archiv a začít používat (Obr. 2.2).

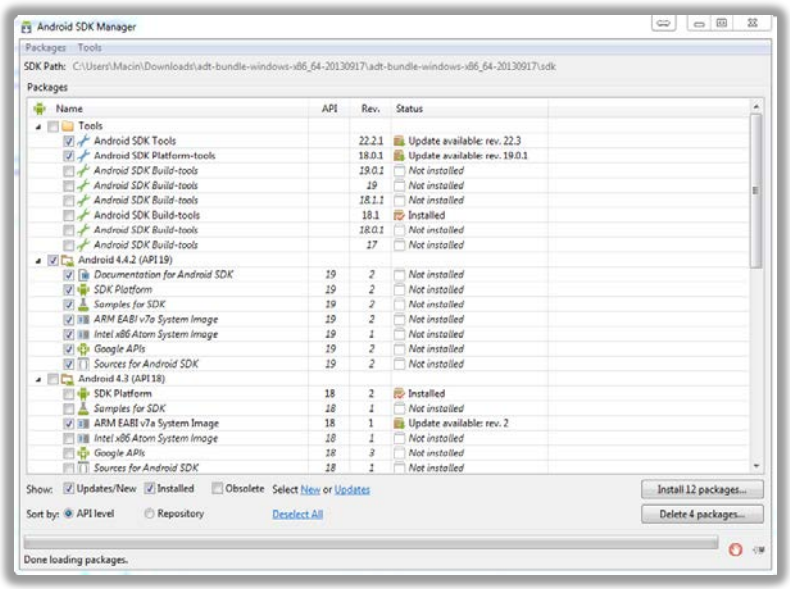
Android SDK je ke stažení na adrese: http://developer.android.com/sdk/index.html



Obr. 2.2: Obsah archivu Android SDK.

Po rozbalení archivu je vhodné jeho obsah zkopírovat do složky, kde bude po celou dobu používání programu. Pokud by byl později celý Android SDK přesunut do jiné složky, mohly by nastat problémy s funkčností některých doinstalovaných částí.

Po přesunutí Android SDK do příslušné složky, spustíme soubor *SDK Manager.exe*, který se nachází v kořenovém adresáři balíčku.

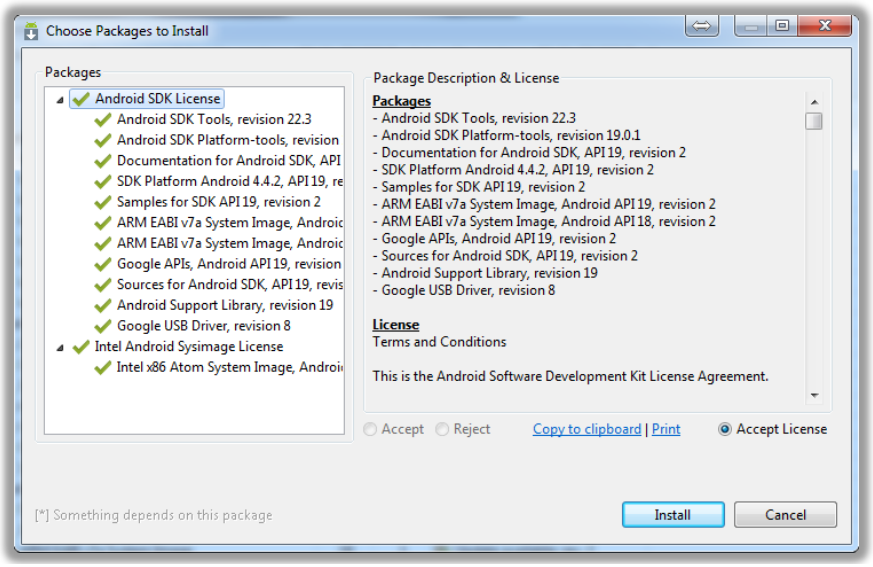


Obr. 2.3: SDK Manager - výběrem platforem.

[11] ADT - je výkonné integrované prostředí s editorem vizuálních aplikací, ladícími panely atd.

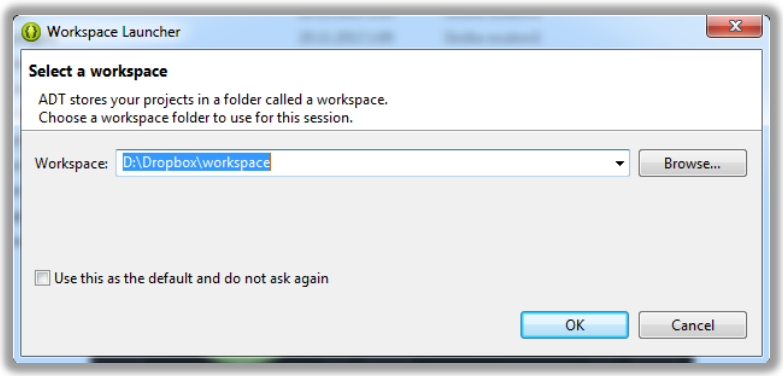
Otevře se okno Manageru (viz Obr. 2.3), kde má uživatel na výběr z různých platforem Androidu. Aktuální stažený SDK podporuje samozřejmě vždy poslední uveřejněnou verzi Android, v tomto případě se jedná o Android 4.4.2, stažení této verze není samozřejmě povinné, ale je vhodné. Žádný programátor nechce vyvíjet aplikaci, která nebude optimalizována na nejnovější Android. Nejnovější verze je vždy automaticky předem vybrána. Zůstane tedy označena a dále má uživatel možnost označit ostatní balíčky a součásti, které případně potřebuje. Důležité balíčky a součásti jsou již automaticky označené ke stažení. Uživatel může samozřejmě označit i více platforem najednou nebo je v případě potřeby později doinstalovat.

Jakmile je vybrána platforma, nebo platformy ve kterých chce uživatel vyvíjet aplikaci, stačí přejít k instalaci. V následujícím dialogovém okně je uživatel vybídnut k souhlasení s licenčními podmínkami jednotlivých stahovaných balíčků (Obr. 2.4).



Obr. 2.4: SDK Manager - odsouhlasení licenčních podmínek.

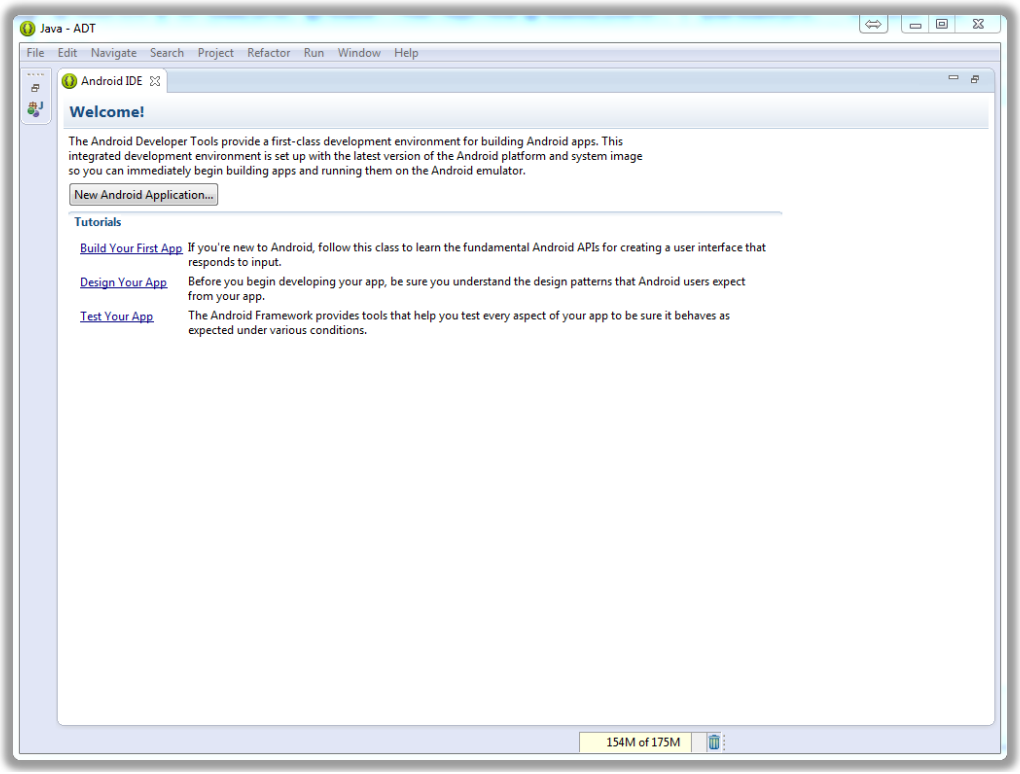
Nyní jsou nainstalovány platformy na kterých bude aplikace vyvíjena a následujícím krokem je spuštění vývojového prostředí Eclipse. Spouštěcí soubor *eclipse.exe* se nachází ve složce *eclipse*.



Obr. 2.5: Eclipse - nastavení pracovní složky.

Po spuštění je uživatel vybídnut k výběru pracovní složky, do které budou následně ukládány veškeré vyvíjené aplikace (Obr. 2.5). Pokud uživatel potřebuje složku nastavit na trvalo, stačí zatrhnout políčko *Use this as the default and do not ask again*, nebo-li *Použít jako primární a dále se neptat.*

Po stisknutí tlačítka *OK* se objeví úvodní okno vývojového prostředí Eclipse (viz Obr. 2.6).

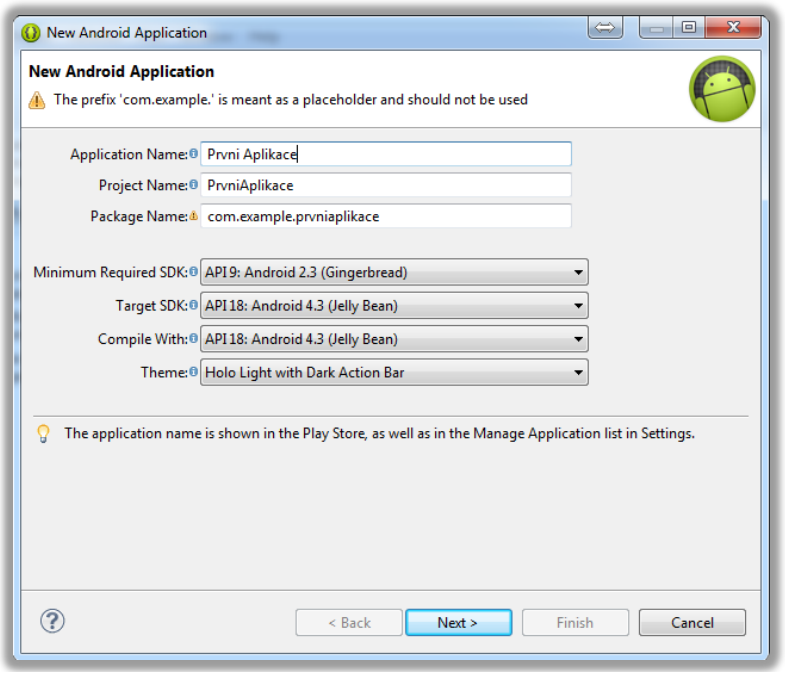


Obr. 2.6: Eclipse - úvodní obrazovka.

2.3. Vytvoření nové aplikace

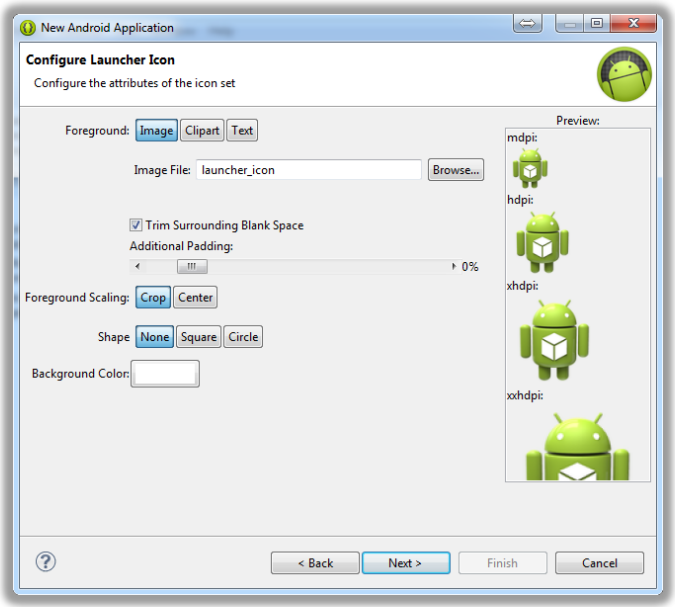
Pro vytvoření nové aplikace stačí stisknout tlačítko *New Android Application*, které se nachází na uvítací kartě, nebo otevřít menu *File* a zvolit *New Project*. Tím se otevře okno průvodce tvorby aplikace a jejího základního nastavení uvedeno na Obr. 2.7.

Do pole *Application Name* uživatel vloží název aplikace a do *Project Name* název projektu. Následně v poli *Minimum Required SDK* vybere nejnižší verzi Android, kterou bude aplikace vyžadovat pro chod. *Target SDK* je cílová verze Android, pro kterou bude aplikace primárně optimalizovaná. V poli *Compile With* je vybrána verze, na které bude aplikace testována a v položce *Theme* je na výběr ze základních stylů aplikace.



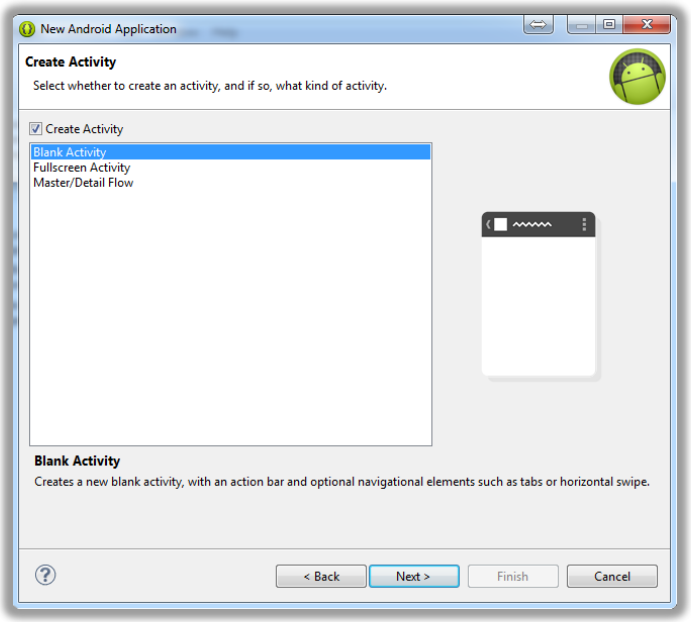
Obr. 2.7: Eclipse - základní parametry aplikace.

V dalším dialogovém okně uživatel stiskne tlačítko další a tím se dostane k nastavení vzhledu spouštěcí ikony aplikace (Obr. 2.8). Zde je možnost ikonu upravit podle libosti, případně může uživatel vložit vlastní vytvořené ikony pomocí tlačítka *Browse*.



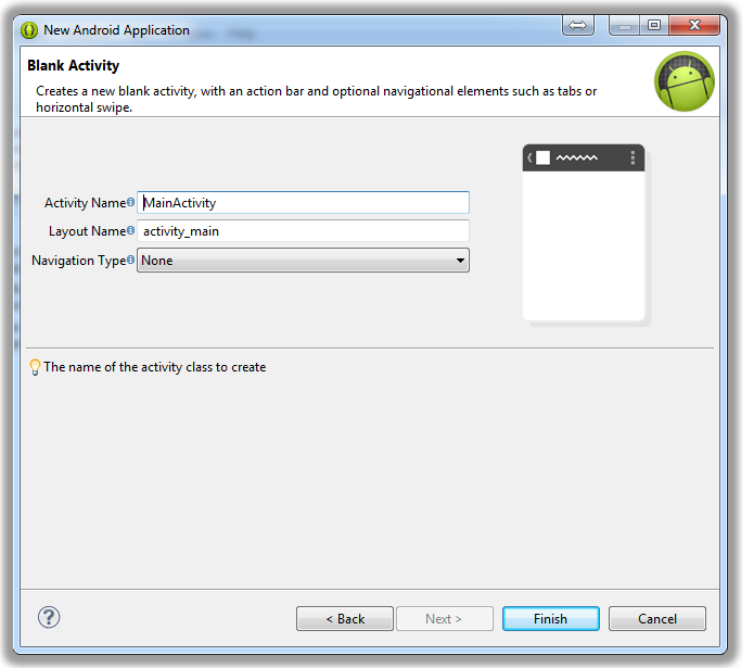
Obr. 2.8: Eclipse - nastavení ikony aplikace.

Po nastavení ikony aplikace následuje volba aktivity (viz Obr. 2.9). Pro začátek doporučuji zvolit prázdnou aktivitu *Blanc Activity*. V následujícím okně je ještě nutné aktivitu pojmenovat s zvolit typ navigace, pro začátek doporučuji zvolit možnost *None* (Obr. 2.10).

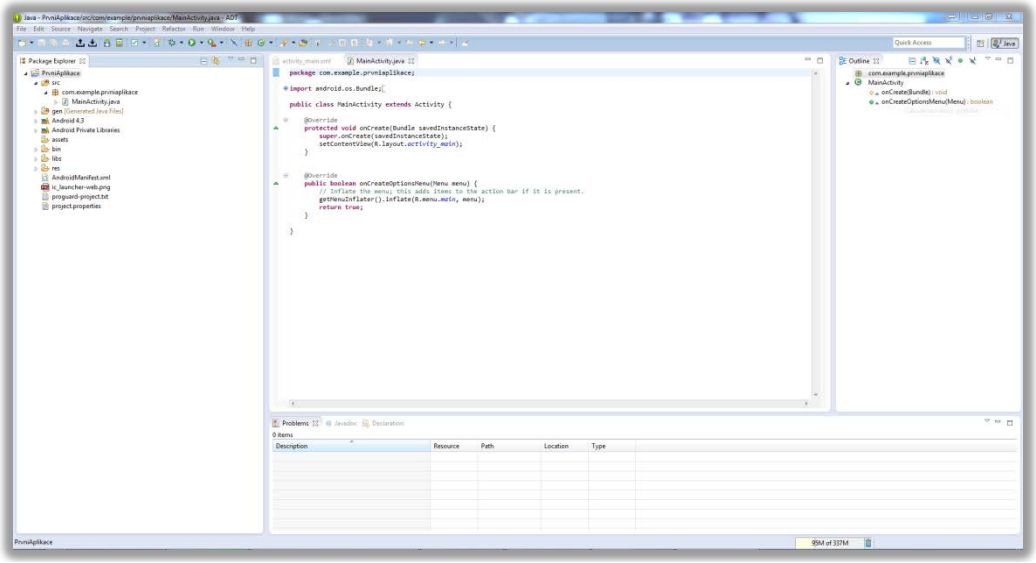


Obr. 2.9: Eclipse - výběr aktivity.

K vytvoření první aplikace už zbývá jen stisknout tlačítko *Finish* a otevře se vývojové prostředí Eclipse se zdrojovým kódem aplikace (Obr. 2.11). Tento kód obsahuje jen základní prvky jako je zdrojový kód *Java*, *XML* kód vzhledu aplikace, soubor *AndroidManifest* s informacemi které uživatel nastavil při tvorbě. Aplikace je naprogramována pro zobrazení textu „Hello World”.

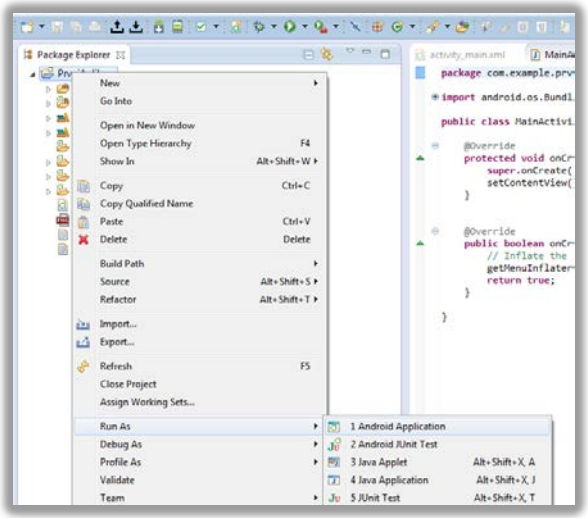


Obr. 2.10: Eclipse - pojmenování aktivity.



Obr. 2.11: Eclipse - zdrojový kód Hello World ve vývojovém prostředí.

Pro spuštění emulátoru vytvořené aplikace stačí kliknout pravým tlačítkem myši na název projektu a v položce *Run As* vybrat *Android Application*, tím uživatel nastartuje emulátor prostředí Android (viz Obr. 2.12).



Obr. 2.12: Eclipse - spuštění emulátoru s aplikací.

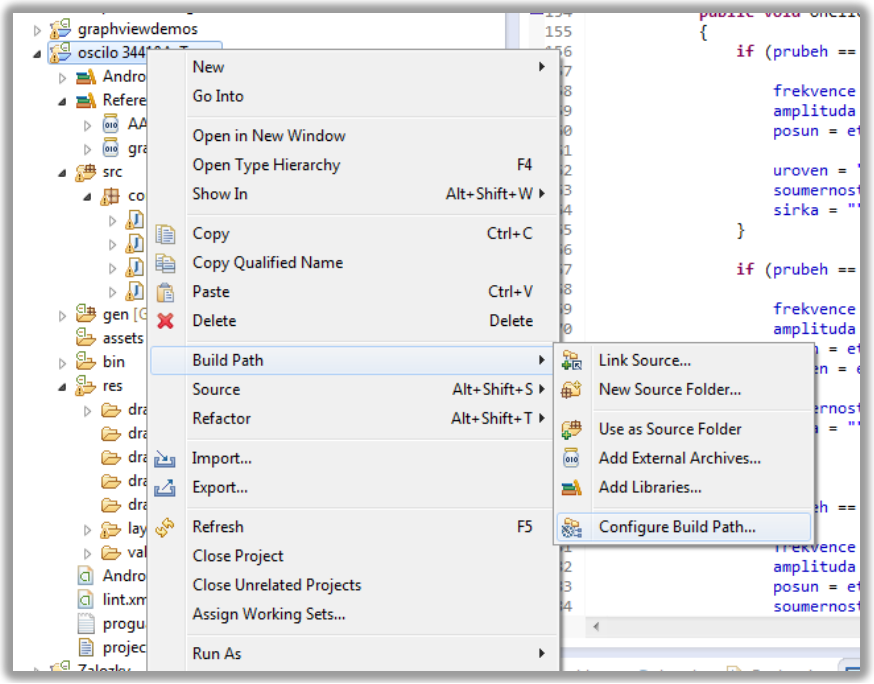
Spuštěný emulátor zobrazí po nastartování zamčený displej stejně jako je tomu u mobilních telefonů s OS Android. S emulátorem lze pracovat stejným způsobem jako s mobilním telefonem, jen mechanická tlačítka jsou zobrazena vpravo, aby byla zajištěna plná ovladatelnost. Po odemčení displeje se otevře uživatelem vytvořená aplikace „Hello World". Emulátor se ukončuje zavřením okna emulátoru (Obr. 2.13).

Obr. 2.13: Eclipse - okno emulátoru.

2.4. Importování externích knihoven

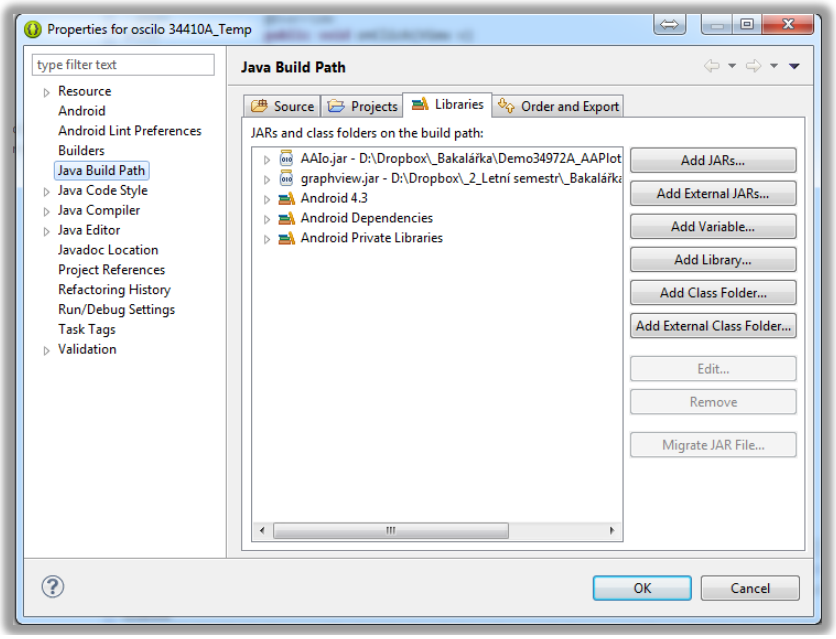
Pro tvorbu jednoduchých aplikací stačí programátorovi základní knihovny, ale v případě rozsáhlejšího projektu musí programátor použít externí knihovny ať už v podobě *jar* knihoven nebo *class* knihoven. Tyto externí knihovny se musí do projektu vložit a bývají často zdarma ke stažení na internetových stránkách příslušeného projektu.

Kliknutím pravého tlačítka myši na projekt se rozbalí nabídka ve které se nachází položka *Build Path*. Po najetí kurzoru na tuto nabídku se zobrazí podnabídka, kde uživatel klikne na položku *Configure Build Path* tak, jak je uvedeno na Obr. 2.14.



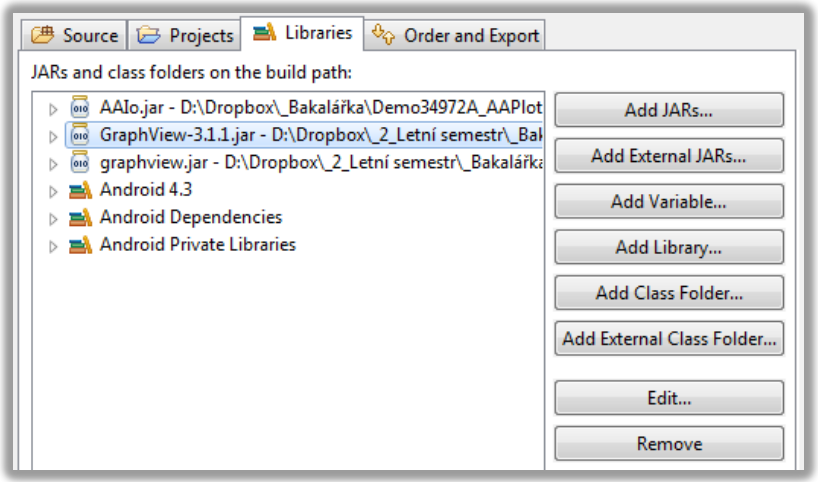
Obr. 2.14: Eclipse - otevření konfigurace balíčku.

Otevře se okno vlastností vybraného projektu, zde uživatel klikne na *Java Build Path* a jako kartu otevře *Libraries.* V okně uprostřed se nacházejí veškeré knihovny a balíčky zahrnuté do projektu (viz Obr. 2.15). Dále uživatel pokračuje podle toho, zda potřebuje přidat *jar* nebo *class* knihovnu. Pro import *jar* knihovny slouží tlačítko *Add External JARs* , kde uživatel vybere potřebný soubor a pro *class* knihovny slouží tlačítko *Add External Class Folder*, kde uživatel vybere složku obsahující *class* soubory.



Obr. 2.15: Eclipse - vlastnosti balíčku.

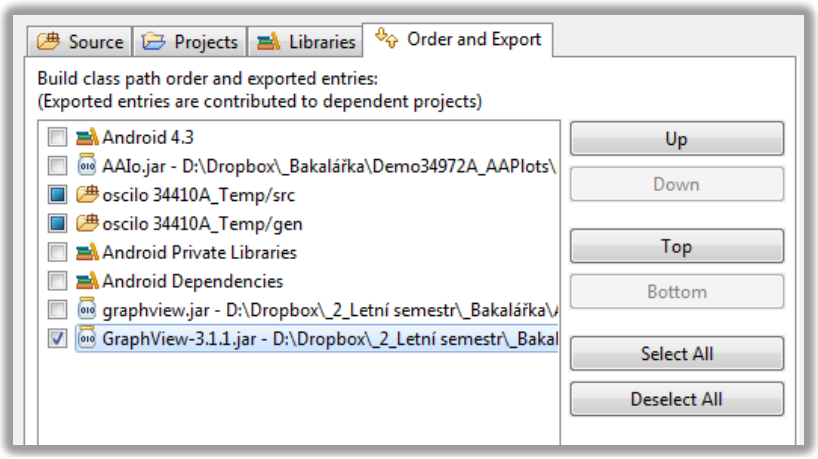
Vložení *jar* knihovny je uvedeno na Obr. 2.16, v tomto případě byla vložena knihovna *GraphView-3.1.1.jar*.



Obr. 2.16: Eclipse - vložení nové knihovny do balíčku.

Knihovna je nyní vložena do balíčku, ale stále není aktivní. Konečné propojení uživatel provede v kartě *Order and Export*, kde přidanou knihovnu či balíček propojí zaškrtnutím příslušné kolonky (viz Obr. 2.17).

Nyní je knihovna svázána s projektem. V případě souborů *class* je postup stejný až na rozdíl, kde se místo souboru vybírá celá složka se soubory.



Obr. 2.17: Eclipse - propojení nové knihovny s balíčkem.

3. PŘÍSTROJE AGILENT

3.1. Vlastnosti a specifikace přístroje 34410A

Multimetr Agilent 34410A je všestranný měřící přístroj na měření střídavých a stejnosměrných veličin. Je vyvinut pro rychlá měření a přesné trigerování. Rozlišení displeje je $6_{1/2}$ místa, duální zobrazení, přenesení 10 000 měření/s při rozlišení $5_{1/2}$ místa, logování dat. Dále dokáže zaznamenat do paměti 50 000 měření. Při připojení k počítači umí udělat 50 000 měření za sekundu při rozlišení $4_{1/2}$ míst nebo 10 000 měření za sekundu při rozlišení $5_{1/2}$ míst a 1000 měření za sekundu při rozlišení $6_{1/2}$ míst. Také ovládá 14 měřících funkcí:

- Měření napětí a proudu.
- Měření odporu.
- Zkoušky propojení a diod.
- Měření kmitočtu a periody.
- Měření kapacity.
- Měření teploty.
- Manuální i automatické nastavení rozsahu.
- Záznam dat.
- Matematické funkce.
- Ukládání nastavení přístroje.
- Dálkové ovládání přes GPIB[12], USB a LAN.
- Internetové rozhraní.
- Slučitelnost s SCPI.
- Možnost interního a externího spouštění.

Agilent 34410A má celkem 3 možnosti dálkového ovládání. První možností je GPIB, jedná se o paralelní sběrnici pro připojení maximálně 15 přístrojů. Má 24 vodičů a maximální délka sběrnice je 20m, ale maximální délka mezi dvěma jednotkami jsou 2m. Druhou možností je USB, kde je maximální délka vodiče bez oživení 15m. Poslední možností je připojení pomocí LAN, zde může být kabel dlouhý 100m a lze takto zapojit neomezené množství přístrojů do sítě [5].

[12] GPIB - je rozhraní pro měřící a zkušební přístroje, které umožňuje přenos dat mezi dvěma přístroji.

3.2. Vlastnosti a specifikace přístroje DSO-X2012A

Osciloskop Agilent DSO-X 2012A je digitální univerzální přístroj pro měření průběhů signálů různých druhů. Jeho předností je vysoký výkon, veliký 8,5 palcový WVGA[13] displej a 8 digitálních kanálů s 1 GS/s vzorkovací rychlostí. Dále nabídne velikost paměti 100kB pro každý kanál, pokročilé matematické funkce, FFT analýzu, digitální nastavitelné filtry a v neposlední řadě vestavěný generátor. Dále disponuje rychlostí obnovení průběhu 50 000/s, USB portem pro ukládání snímků obrazovky a dat a také integrovanou nápovědou.

Díky možnosti vyměnitelných modulů pro připojení osciloskopu do sítě, může zákazník zvolit mezi modulem LAN/VGA, ve kterém se nachází konektor pro připojení externího monitoru a konektor pro připojení LAN kabelu nebo může zvolit modul se sběrnicí GPIB. Samozřejmě nechybí propojení s počítačem pomocí USB.[6].

Ovládání osciloskopu je velice jednoduché, čelní panel je přehledný a ovládací prvky jsou vhodně rozmístěny. Pokud uživatel potřebuje ovládat osciloskop vzdáleně, stačí si do počítače nainstalovat software Agilent Connection Expert a s jeho pomocí osciloskop nastavovat. V případě, že je osciloskop připojen do sítě internet pomocí modulu LAN, stačí zadat do internetového prohlížeče IP adresu přístroje, která otevře webové rozhraní uložené přímo v přístroji. Toto rozhraní využívá Java Plugin prohlížeče k zobrazení vytvořeného modelu čelního panelu, který přesně odpovídá skutečnému panelu a je i plně funkční.

[13] WVGA - udává rozlišení displeje - 800 x 480 bodů

3.3. Ovládání přístrojů pomocí SCPI

Veškeré přístroje značky Agilent s komunikačním rozhraním USB, GPIB, LAN je možné ovládat vzdáleně pomocí příkazů SCPI[14]. Jedná se o standard, který shrnuje příkazy a pravidla pro komunikaci mezi řídící jednotkou a přístrojem v automatizovaném měřícím systému. Je nezávislý na technickém řešení i na protokolu přenosu dat [7]. SCPI má vlastní síťový port 5025 pro přenos dat mezi aplikacemi a přístrojem.

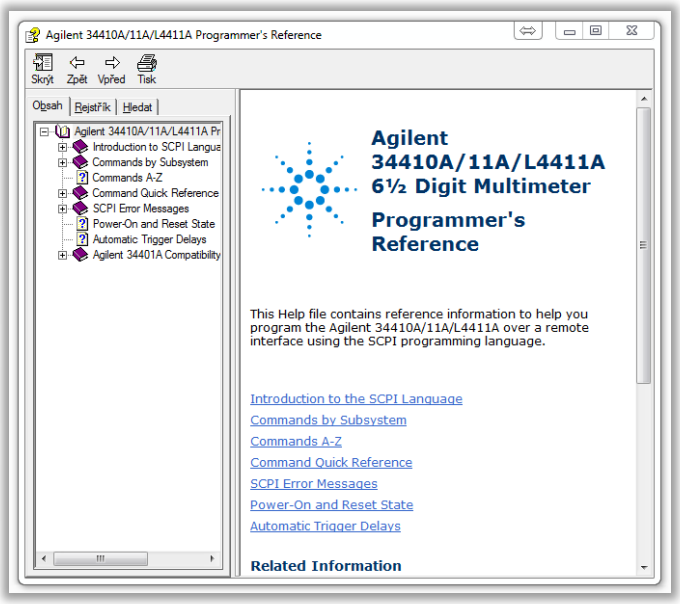
Vzhledem k tomu, že každý přístroj obsahuje jiné funkce a tím pádem i jiné ovládací příkazy, je kompletní přehled příkazů i s nápovědou, syntaxí a příklady použití dostupný v podobě klasické Windows nápovědy, kterou lze nalézt na dodaném CD nebo stáhnout na stránkách výrobce.

SCPI příručka programátora pro Agilent 34410A je dostupná na adrese:

http://www.home.agilent.com/agilent/redirector.jspx?action=ref&cname=AGILENT_EDITORIAL&ckey=768249&lc=eng&cc=CZ&nfr=-536902435.536908384.00

Pro Osciloskop Agilent DSO-X 2012A je programátorská příručka dostupná na:

http://www.home.agilent.com/agilent/redirector.jspx?action=ref&cname=AGILENT_EDITORIAL&ckey=2015143&lc=eng&cc=US&nfr=-33575.970741.00



Obr. 3.1: Příklad programátorské nápovědy pro SCPI.

[14] SCPI - (Standard Commands for Programmable Instruments) jedná se o standardní příkazy pro programovatelné přístroje.

3.4. Syntaxe příkazů pro Agilent 34410A

Jako příklad syntaxe příkazu je možné uvést zjištění, kterou verzi jazyka SCPI přístroj podporuje. Pomocí rozhraní dálkového ovládání (například Android aplikace) odešleme příkaz ve formě řetězce textu:

```
SYSTem:VERSion?;   nebo  SYST:VERS?;
```

Přístroj nám odpoví ve formě `RRRR.V`, zde `RRRR` označuje rok a `V` číslo verze pro příslušný rok.

Příklad syntaxe nastavení pro měření teploty. Do pole typ termistoru se uvádí `FTHermistor` nebo `THERmistor`. Do pole velikost odporu se uvádí hodnota odporu čidla a to: `2252, 5000` nebo `10000.`

```
CONFigure:TEMPerature <typ termistoru>, <velikost     odporu>;
```

Výsledný zápis může vypadat následovně:

```
CONF:TEMP THER, 5000;
```

Předchozí příkaz musí být použit s příkazem `INITiate.` Tento příkaz provede měření s nastavenými parametry z příkazu `CONF:`

```
INITiate; nebo INIT;
```

Naměřenou hodnotu načteme z přístroje pomocí příkazu:

```
FETCh?; nebo FETC?;
```

Typická odpověď přístroje: `+25.27150000`.

Veškeré příkazy jsou uvedeny v příslušné programátorské příručce přístroje, která je uvedena v kapitole č.3.3.

3.5. Syntaxe příkazů pro Agilent DSO-X 2012A

Pro tento osciloskop se syntaxe příkazů liší od přístroje Agilent 34410A. Důvodem je větší počet funkcí které osciloskop nabízí oproti multimetru. Odesílání příkazů a přijímání dat má u obou přístrojů stejný formát a to v podobě řetězce textu.

Ukázka syntaxe na příkazu, který po odeslání do přístroje vrací odesílateli informace o výrobci přístroje, modelu, sériovém číslu a revizi softwaru nainstalovaného v přístroji:

```
*IDN?
```

Přístroj odpoví ve formě:

```
AGILENT TECHNOLOGIES,<model>,<serial number>,X.XX.XX <NL>
```

Kde první pozice označuje výrobce, následující pozice `<model>` značí model přístroje, například DOS-X 2012A, `<serial number>` značí sériové číslo přístroje a `X.XX.XX` označuje revizi softwaru přístroje.

Příklad syntaxe pro nastavení frekvence generátoru funkcí.:

```
:WGEN:FREQuency <frequency>
```

kde <frequency> značí pozici pro číselný údaj v Hz pro frekvenci. Výsledný zápis například pro 1kHz vypadá následovně:

```
:WGEN:FREQuency 1000
```

nebo kratší forma:

```
:WGEN:FREQ 1000
```

Veškeré příkazy jsou uvedeny v příslušné programátorské příručce přístroje, která je uvedena v kapitole č.3.3.

4. APLIKACE 34410A TEMP

Jedná se o aplikaci, která slouží k nastavení a měření teploty pomocí multimetru Agilent 34410A přes síť. Naměřená hodnota je následně zobrazena přímo v aplikaci.

4.1. Popis funkce aplikace

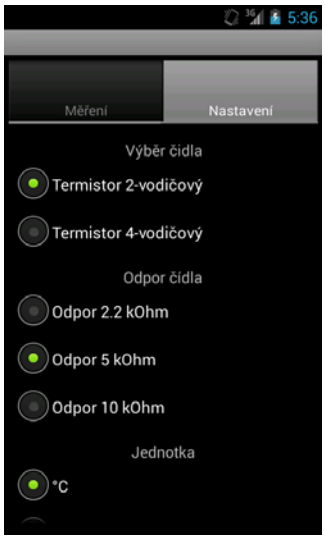
Aplikace se skládá ze záložek *Měření* a *Nastavení*. Po otevření aplikace se zobrazí záložka Měření, kde se nachází pole pro vyplnění IP adresy, tlačítko Připojit a tlačítko změřit, které je prozatím deaktivováno. Uživatel vyplní IP adresu přístroje v síti (Obr. 4.1).

Obr. 4.1: Aplikace 34410A Temp - záložka Měření.

Po vyplnění adresy může uživatel stisknout tlačítko *Připojit,* čímž naváže spojení mezi aplikací a přístrojem. Pokud se spojení naváže, je o tom informován, zároveň je uživateli znepřístupněno tlačítko pro připojení a zpřístupněno tlačítko pro měření (Obr. 4.2). Nyní může uživatel změřit teplotu nebo přejít do nastavení aplikace.

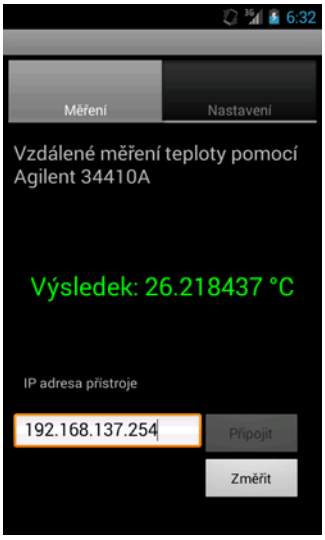
Obr. 4.2: Aplikace 34410A Temp - úspěšné připojení k přístroji.

Ve druhé záložce *Nastavení* má uživatel možnost nastavit aplikaci pro měření různými metodami (Obr. 4.3). Jako první nastavení je možnost výběru měřícího čidla, kde uživatel zvolí *Termistor 2-vodičový* nebo *Termistor 4-vodičový*. Druhá možnost je volba velikosti odporu čidla, uživatel volí mezi čidlem s odporem *2.2kOhm*, *5kOhm* nebo *10kOhm*. Jako poslední je volba jednotek, zde je na výběr mezi *°C*, *°F* a *K*.



Obr. 4.3: Aplikace 34410A Temp - záložka Nastavení.

Při každém startu aplikace je měření přednastaveno na *Termistor 2-vodičový*, odpor čidla *5kOhm* a jednotka měření nastavena na *°C*. Pokud uživatel vybere požadované nastavení, přepne se na kartu *Měření*, kde stisknutím tlačítka *Změřit* provede odeslání nastavení a následné odpovědi přístroje v podobě naměřené hodnoty (Obr. 4.4).

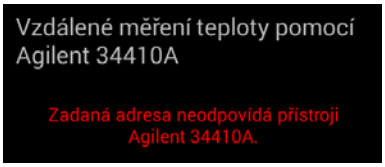


Obr. 4.4: Aplikace 34410A Temp - změření a zobrazení teploty.

4.2. Ošetření chybových stavů

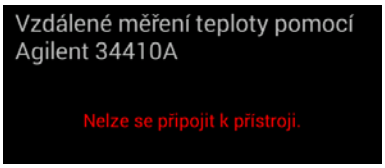
Pokud se připojujeme k přístroji pomocí sítě je možné, že nastanou nějaké komplikace při navazování spojení. Proto je aplikace naprogramována tak, aby rozpoznala úspěšnost připojení a informovala o ní uživatele. V případě, že spojení proběhne v pořádku je aplikace připravena k měření a uživatel informován o stavu v podobě zprávy na displeji, (viz. Obr. 4.2).

Jestliže se na zadané IP adrese nenachází přístroj Agilent 34410A, ale jiný přístroj, aplikace tuto skutečnost rozpozná a informuje uživatele zprávou uvedenou v Obr. 4.5. Uživatel následně může IP adresu změnit na správnou.



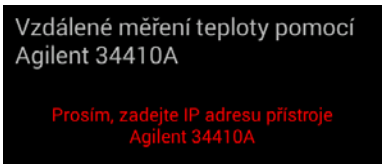
Obr. 4.5: Zobrazení informace o nesprávné IP adrese.

Dalším případem je možnost, kdy nelze spojení úplně navázat. I v takovém případě aplikace opět informuje uživatele o problému se spojením (viz Obr. 4.6).



Obr. 4.6: Zobrazení informace o chybě spojení.

Poslední možností je chyba uživatele. Pokud uživatel zadá neplatnou IP adresu je zobrazena stejná informace jako v případě špatného spojení (Obr. 4.6). Pokud uživatel zanechá pole IP adresy prázdné a stiskne tlačítko *Připojit*, je opět informován příslušným varováním (Obr. 4.7).



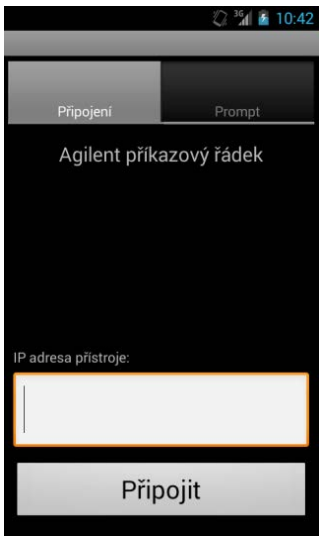
Obr. 4.7: Zobrazení informace o nevyplněné IP adrese.

5. APLIKACE AGILENT PROMPT

Aplikace vytvořená pro univerzální použití napříč přístroji firmy Agilent využívajících příkazy SCPI. Jedná se o všestrannou, jednoduchou aplikaci umožňující uživateli připojit se na jakýkoliv přístroj zapojený do sítě Internet a ovládat jej pomocí SCPI příkazů daných pro příslušný přístroj.

5.1. Popis funkce aplikace

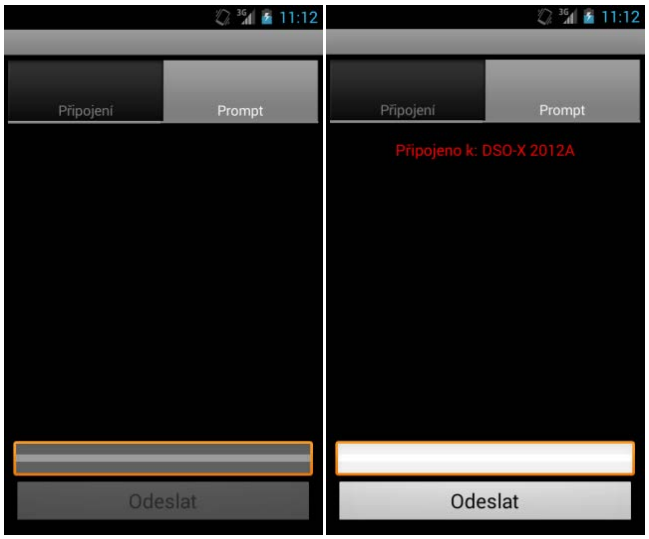
Po spuštění aplikace je uživateli zobrazena úvodní obrazovka s kartou *Prompt* a kartou *Připojení,* která je v popředí jak je vidět na Obr. 5.1. V kartě *Připojení* se nachází pole pro vyplnění IP adresy přístroje a tlačítko pro připojení. Ve druhé kartě *Prompt* uživatel nalezne příkazový řádek a tlačítko pro odeslání příkazu. Jak příkazový řádek, tak i tlačítko pro odeslání je deaktivováno do té doby, než proběhne úspěšné propojení s přístrojem.



Obr. 5.1: Aplikace Agilent Prompt - úvodní obrazovka.

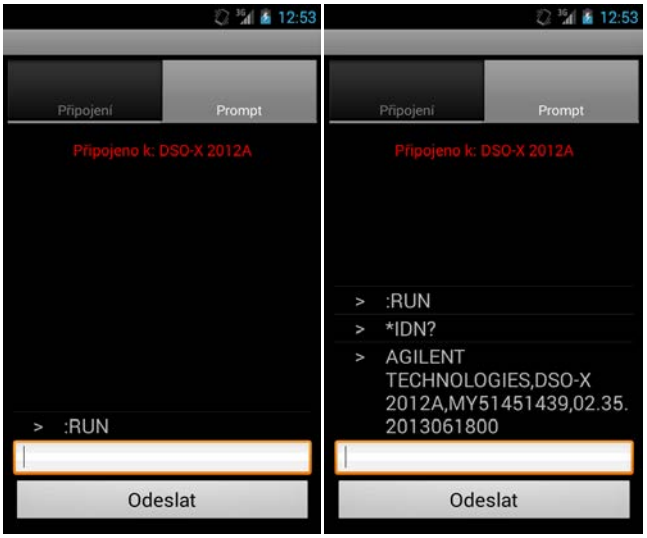
Zadáním adresy a stisknutím tlačítka *Připojit*, se aplikace spojí s přístrojem na dané adrese. Po připojení se zobrazí informace o jaký model přístroje se jedná, výrobce a sériové číslo (Obr. 5.2). Zároveň je zpřístupněna možnost zadávat a odesílat příkazy přístroji v kartě *Prompt* uvedeno na Obr. 5.3 (vpravo), kde je taktéž zobrazen model přístroje pro lepší informovanost uživatele. Pokud připojení doposud neproběhlo, je příkazový řádek a tlačítko pro odeslání příkazu deaktivováno (viz Obr. 5.3 vlevo).

Obr. 5.2: Aplikace Agilent Prompt - připojení přístroje a zobrazení informací.



Obr. 5.3: Aplikace Agilent Prompt - karta Prompt, nepřipojeno (vlevo), po připojení (vpravo).

V kartě *Prompt* se nachází pole pro zadání SCPI příkazu ve formátu určeného výrobcem a tlačítko odeslat. Po stisknutí tlačítka odeslat je příkaz zaslán a do přístroje a následně zobrazen v seznamu posledních příkazů. Tento seznam je automaticky doplňován odeslanými příkazy pro lepší přehled měření a v případě příkazu pro získání dat z přístroje tato data zobrazí. Na Obr. 5.4(vlevo) je zobrazeno odeslání příkazu *:RUN* pro spuštění měření. Zobrazení seznamu posledních příkazů, kde byl zaslán příkaz **IDN?* pro získání identifikačních údajů přístroje a jejich následné zobrazení pod odeslaným příkazem v seznamu je uveden na Obr. 5.4(vpravo).



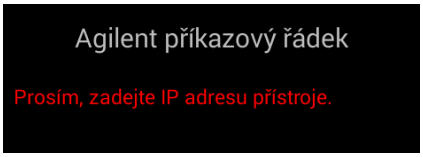
Obr. 5.4: Aplikace Agilent Prompt - odeslání příkazu :RUN (vlevo), odeslání příkazu *IDN? (vpravo).

Pokud uživatel potřebuje, může se z karty příkazového řádku přepnout na kartu *Připojení* a zadat IP adresu jiného přístroje který chce ovládat. Po připojení se opět může vrátit na kartu příkazového řádku a zadávat další příkazy.

5.2. Ošetření chybových stavů

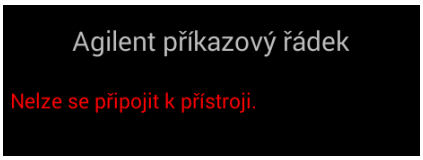
Aplikace je naprogramována tak, aby se zamezilo vzniku nepředpokládaného chování aplikace jak vlivem uživatele, tak i vlivem sítě. Uživatel je o veškerém dění, chybách a stavech informován.

Jestliže uživatel nevyplní IP adresu, je o tom informován výstražnou zprávou (viz Obr. 5.5) a zároveň je zamezeno zadávání a odesílání příkazů z karty *Prompt.*



Obr. 5.5: Zobrazení informace o nevyplněné IP adrese.

Uživatel také může zadat neplatnou nebo špatnou IP adresu na které se nenachází přístroj Agilent s podporou SCPI příkazů, případně mohou nastat problémy s připojením. I v tomto případě je uživatel informován zprávou o stavu komunikace (Obr. 5.6).

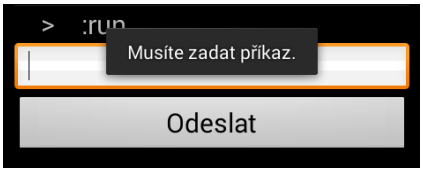


Obr. 5.6: Zobrazení informace o chybě spojení.

Dalším případem je možnost přerušení spojení během odesílání příkazů z příkazové řádky. Pokud taková situace nastane, aplikace ji vyhodnotí a zamezí uživateli další odesílání příkazů, dokud není opětovně navázáno spojení. Opět je zobrazena příslušná zpráva o stavu připojení (Obr. 5.7).

Obr. 5.7: Zobrazení informace o výpadku spojení.

Pokud uživatel nevyplní příkaz a odešle prázdnou příkazovou řádku, je informován pomocí speciální informační zprávy Toast [15]. Tato zpráva je zobrazena jako „vyskakovací“ okno s textem a po definovaném čase zmizí (Obr. 5.8) [8].



Obr. 5.8: Zobrazení informace o nevyplnění příkazového řádku.

[15] Toast - poskytuje jednoduchou zpětnou vazbu o operaci v malém „vyskakovacím“ okně.

6. APLIKACE DSO-X 2012A REMOTE

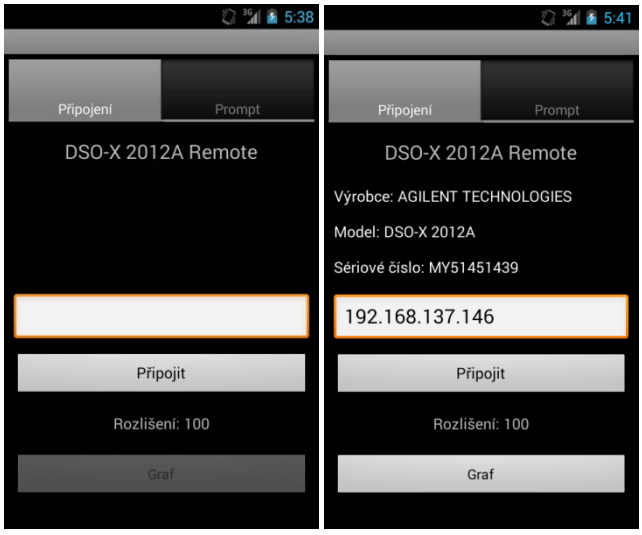
Tato aplikace byla vytvořena k ovládání dvoukanálového, digitálního osciloskopu Agilent DSO-X 2012A s vestavěným generátorem signálu. Aplikace obsluhuje nejdůležitější funkce osciloskopu jako je zobrazení průběhů signálů, zobrazení naměřených hodnot a nastavení generátoru. Dále je integrován příkazový řádek pro případné nastavení dalších funkcí. Opět je využito komunikace pomocí SCPI příkazů.

6.1. Popis funkce aplikace

Základ aplikace je podobný aplikaci Agilent Prompt a to hlavně díky integraci ovládání pomocí příkazové řádky. Na úvodní obrazovce se nacházejí dvě karty *Připojení* a *Prompt*.

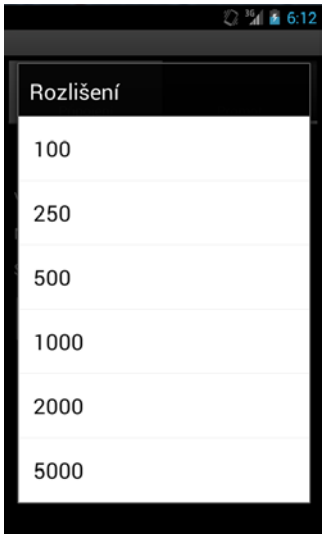
Druhá karta *Prompt* je obsahově i funkčně totožná s aplikací Agilent Prompt. Je tedy využitelná k ovládání funkcí nezahrnutých do aplikace, případně i k ovládání jiných přístrojů pomocí SCPI příkazů. Pokud bude takto aplikace využívána, nelze použít funkce grafu a generátoru. Tyto funkce jsou kompatibilní pouze s přístroji DSO-X 2012A a přístroji jejichž funkce jsou podobné.

První karta obsahuje pole pro vložení IP adresy přístroje (Obr. 6.1 vlevo). Po úspěšném navázání spojení je zobrazen model, výrobce a sériové číslo pro ověření totožnosti přístroje (Obr. 6.1 vpravo). Dále je zpřístupněno tlačítko pro spuštění grafu.



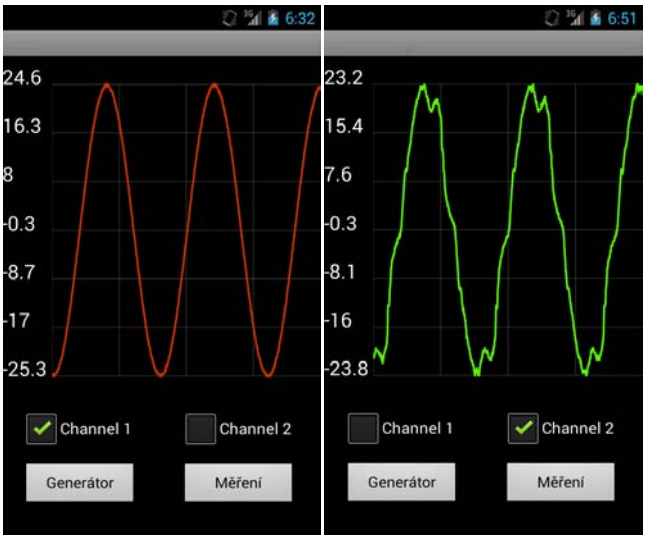
Obr. 6.1: Aplikace DSO-X 2012A Remote - úvodní obrazovka (vlevo), připojení k přístroji (vpravo).

Na první kartě se také nachází přepínač pro nastavení rozlišení grafu. Jedná se o počet bodů načtených ze zobrazeného průběhu signálu na displeji osciloskopu. Malé rozlišení přinese rychlejší obnovování grafu v aplikaci na úkor nekvalitního a méně přesného průběhu. Vysoké rozlišení zaručí jemnost grafu, ale pomalejší obnovování dat. Po kliknutí na text *Rozlišení:* se zobrazí nabídka s možnostmi, kde uživatel kliknutím na příslušnou číslici zvolí požadované rozlišení v rozsahu 100 bodů až 5000bodů (Obr. 6.2). Standardní hodnota je přednastavena na 1000 bodů, jedná se o kompromis mezi rychlostí a kvalitou zobrazeného grafu.



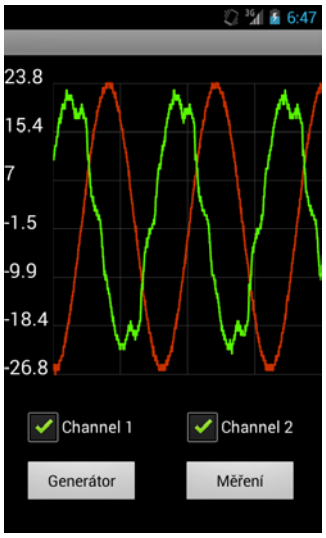
Obr. 6.2: Aplikace DSO-X 2012A Remote - nabídka rozlišení.

Po stisknutí tlačítka Graf se spustí nové okno s vykresleným průběhem signálu. Data signálu jsou automaticky obnovována a vykreslována do grafu (Obr. 6.3).



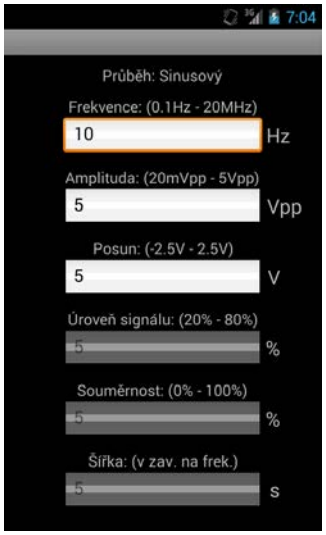
Obr. 6.3: Aplikace DSO-X 2012A Remote - vykreslený signál Channel 1(vlevo) a Channel 2 (vpravo).

V okně grafu se kromě samotného vykresleného průběhu signálu nachází dvě výběrová políčka pro nastavení zobrazení kanálů *Channel 1* a *Channel 2* (Obr. 6.4). Tato výběrová políčka zapínají nebo vypínají zobrazení a načítání dat příslušných kanálů. Dále se zde nacházejí tlačítka *Generátor* a *Měření*. Jestliže je zvoleno zobrazení obou průběhů najednou, rozlišení načítaných dat se dělí na polovinu. Pokud je tedy rozlišení nastaveno na 1000 bodů, výsledné rozlišení pro jeden průběh signálu je 500bodů. V případě potřeby změny IP adresy, nebo odeslání SCPI příkazů pomocí příkazové řádky se může uživatel stisknutím tlačítka *Zpět* na mobilním zařízení vrátit na úvodní obrazovku aplikace.



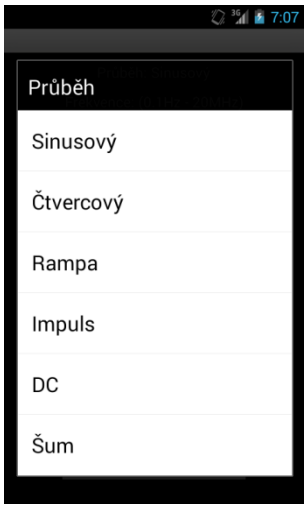
Obr. 6.4: Aplikace DSO-X 2012A Remote - zobrazení dvou průběhů signálů.

Vzhledem k tomu, že je přístroj vybaven generátorem, disponuje aplikace nastavením příslušného generátoru. Toto nastavení se spouští tlačítkem *Generátor* pod grafem. Otevře se nové okno se šesti vstupními poli pro zadání potřebných parametrů generovaného průběhu (Obr. 6.5).



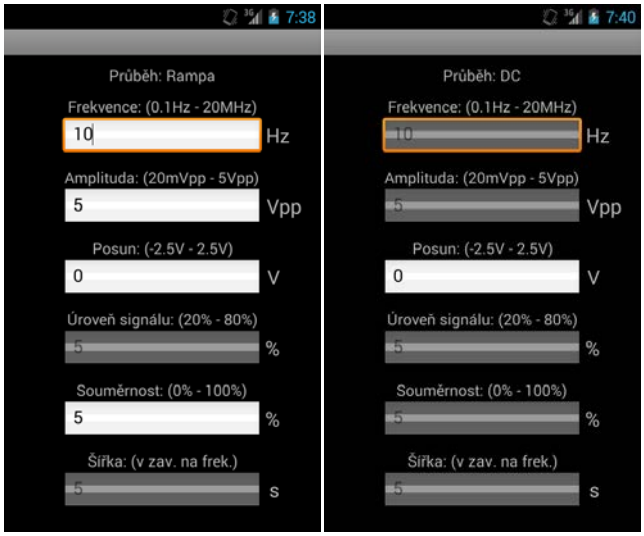
Obr. 6.5: Aplikace DSO-X 2012A Remote - nastavení generátoru.

Uživatel vybere požadovaný průběh signálu kliknutím na text *Průběh:*, čímž se otevře nabídka dostupných průběhů (Obr. 6.6). V nabídce se nachází veškeré dostupné průběhy generátoru osciloskopu, jimiž jsou: sinusový průběh, čtvercový, rampa, impuls, DC a šum.



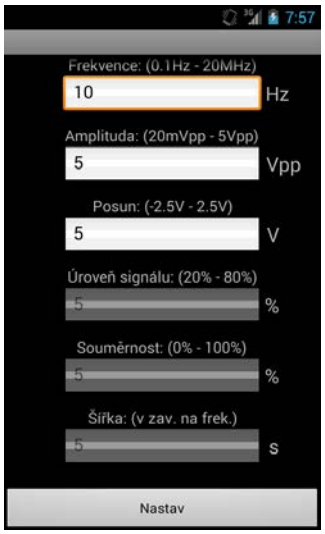
Obr. 6.6: Aplikace DSO-X 2012A Remote - nastavení generátoru.

Zvolením určitého průběhu se uživateli zpřístupní vstupní pole pro parametry podle příslušného průběhu (Obr. 6.7). Ostatní parametry jsou deaktivovány, aby se předešlo omylům ze strany uživatel v podobě zadávání hodnot do polí parametrů jiného průběhu. Nad příslušným polem je vždy uveden název parametru a rozsah hodnot ve kterých je generátor schopen pracovat. Na konci řádku se nachází jednotka zadávaného parametru. Hodnoty musí být vždy zadány podle rozsahu a v jednotkách uvedených u příslušného pole. Desetinná čísla se zadávají s tečkou místo čárky. V případě zadání nesprávných hodnot přístroj data příjme, parametr s chybnou hodnotou nenastaví a ponechá nastavenou původní hodnotu.



Obr. 6.7: Aplikace DSO-X 2012A Remote - ukázka nastavení generátoru pro průběh Rampa (vlevo) a DC (vpravo).

Pokud uživatel nastavý veškeré požadované parametry, stačí posunout nabídku směrem dolů, kde se nachází tlačítko *Nastav* pro odeslání a nastavení dat v generátoru (Obr. 6.8). Následně stisknutím tlačítka *Zpět* na mobilním zařízení se uživatel vrátí na graf nebo může pokračovat v dalším nastavování generátoru.



Obr. 6.8: Aplikace DSO-X 2012A Remote - tlačítko Nastav pro odeslání dat.

6.2. Ošetření chybových stavů

Vzhledem k integraci upraveného funkčního modelu z aplikace Agilent Prompt je ošetření chybových stavů stejné jako v původní aplikaci.

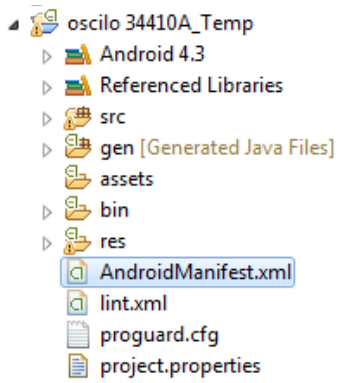
Přidáno bylo ošetření chybových stavů v nastavení generátoru, kde jsou vstupní pole jednotlivých parametrů aktivovány a deaktivovány podle vybraného průběhu. Tím je zamezeno chybnému zadávání hodnot pro parametry, které nenáleží příslušnému průběhu a také je zamezeno zmatení uživatele. Názorná ukázka je uvedena na Obr. 6.7.

7. ROZBOR ZDROJOVÉHO KÓDU

7.1. Základní kameny každé aplikace

Aplikace jako taková je složitý celek a pro svůj chod využívá velké množství souborů a knihoven se zdrojovým kódem. Pro aplikace vytvořené v souvislosti s touto bakalářskou prácí byl vždy využit balíček vygenerovaný vývojovým prostředí Eclipse při založení nového projektu přesně tak, jak je popsáno v kapitole č.2.3. V takto vytvořeném projektu byly využívány tři základní zdrojové soubory, které obsahuje každá aplikace: *AndroidManifest.xml*, *Název_vlastního_zdrojového_kódu.java* a *Název_vlastního_stylu.xml*. Dále byly využity knihovny AAIo a GraphView, které byly do projektu importovány.

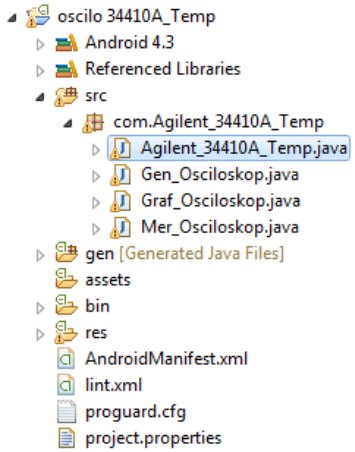
AndroidManifest.xml obsahuje informace jaké komponenty jsou v aplikaci k dispozici. Je to XML[16] soubor uložený v kořenovém adresáři a popisuje parametry aplikace jako je název, verze, požadovaná systémová práva, propojení dalších souborů Java, minimální a cílovou verzi systému android a další. Umístění souboru je přímo v kořenovém adresáři aplikace (viz Obr. 7.1).



Obr. 7.1: Adresářová struktura aplikace - soubor AndroidManifest.xml.

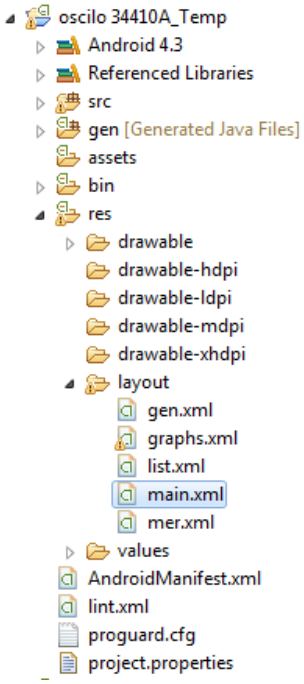
[16] XML - je obecný značkovací jazyk vyvinutý konsorciem W3C

Název_vlastního_zdrojového_kódu.java, jedná se o soubor tříd a jejich metod s hlavním zdrojovým kódem aplikace v jazyce Java. V případě jednoduché aplikace, může být naprogramována jen v tomto jednom souboru. Pokud se jedná o rozsáhlejší aplikaci, může být rozložena do více souborů s příponou *.java*. Tento způsob je pro programátora přehlednější a napomáhá lepšímu chodu aplikace. Soubory toho typu jsou uloženy ve složce *src* v kořenovém adresáři aplikace (viz Obr. 7.2).



Obr. 7.2: Adresářová struktura aplikace - soubory Java.

Název_vlastního_stylu.xml je XML soubor designu aplikace. Každý zobrazovaný prvek v něm musí být zaveden, např.: tlačítka, textová pole, obrázky atd. Zároveň při zavedení prvku můžeme upravovat jeho vlastnosti, pozici na displeji, funkci a další. Je provázaný s hlavním Java souborem a nachází se ve složce *res* a její pod složce *layout* kořenového adresáři aplikace (viz Obr. 7.3).

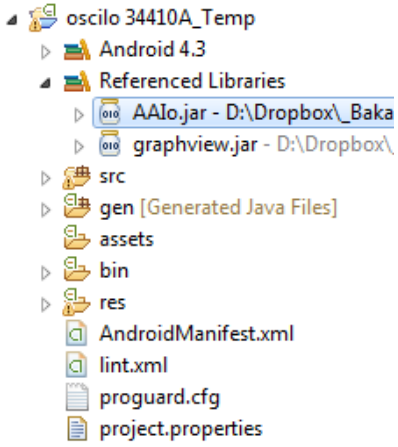


Obr. 7.3: Adresářová struktura aplikace - soubory stylu XML.

Soubory AAIo a GraphView jsou knihovny funkcí a procedur nebo objektů. Tyto soubory mají většinou příponu typu *.jar* nebo *.class* a jedná se o soubory zdrojového kódu plnící určité funkce, jako například funkce pro komunikaci s přístroji Agilent. Takovéto knihovny ulehčují programátorovi práci tím, že nemusí psát kód pro použité funkce sám, ale pomoci názvu funkce příslušnou funkci vyvolá z knihovny. Importování knihoven do projektu je popsáno v kapitole č.2.3. Po importování se vložené knihovny nachází ve složce Referenced Libraries v kořenovém adresáři aplikace tak jak je vidět na Obr. 7.4.

Knihovna AAIo je dostupná na adrese: http://www.home.agilent.com/agilent/redirector.jspx?action=ref&ckey=2086803&cname=AGILENT_EDITORIAL&lc=eng&cc=CZ&sd=Y

Knihovna GraphView je dostupná na adrese: http://android-graphview.org/#download



Obr. 7.4: Adresářová struktura aplikace - soubory knihovny.

7.2. Hlavní bloky zdrojového kódu vzhledu

Jedná se o bloky kódu ze souborů s příponou *.xml*, které definují vzhled aplikace. Kód každého zobrazovaného prvku začíná symbolem `<` s příslušným pojmenováním prvku a je zakončen symbolem `/>` , například `<TextView  />`. Mezi pojmenováním a zakončovacím symbolem jsou uvedeny parametry upravující zobrazení daného prvku. Komentáře se uvádějí mezi znaky `<!-- komentář -->`. Tato syntaxe je jednotná pro veškeré zobrazované prvky v soubor XML.

Základní zobrazovaná třída je text. Její zápis s parametry je proveden následovně [9]:

```
<TextView
            <!--TextView - zavedení textového pole. -->
        android:id="@+id/txtNadpis_2"
            <!--id - nastavíme prvku jedinečné jméno(např.
            txtNadpis_2), díky kterému jej lze jednoduše propojovat s
            ostatními komponentami. -->
        android:layout_width="wrap_content"
            <!--width - šířka prvku, parametr wrap_content přizpůsobí
            šířku prvku podle velikosti jeho obsahu.-->
        android:layout_height="fill_parent"
            <!--height - výška prvku, parametr fill_parent nastaví
            šířku prvku tak, aby vplnil jeho nadřazený prvek. -->
        android:layout_below="@id/txtNadpis_1"
            <!--below - pod prvkem, určuje zarovnání prvku Nadpis_2 pod
            prvkem Nadpis_1. -->
        android:text="Vzdálené měření teploty pomocí Agilent 34410A"
            <!--text - nastaví text pro zobrazení. -->
        android:textSize = "20sp"
            <!--textSize - velikost textu v jednotce obrazový pixel.-->
/>
```

Pole pro třídu zápisu textu uživatelem je například zapsáno [10]:

```
<EditText
            <!--EditText - zavedení upravitelného textového pole. -->
        android:id="@+id/etxtIP"
            <!--id - opět jedinčené pojmenování prvku. -->
        android:layout_alignBaseline="@+id/btnPripoj"
            <!--alignBaseline - zarovnání etxtIP na vodorovnou
            středovou osu prvku btnPripoj. -->
        android:layout_toLeftOf="@+id/btnPripoj"
            <!--toLeftOf - zarovnání etxtIP vlevo od prvku brnPripoj. -
            ->
```

```
        android:text="" />
            <!--text - nastavení prázdného pole pro zápis uživatelem. -
            ->
```

Zavedení prvku tlačítka [11]:

```
<Button
        android:id="@+id/btnPripoj"
        android:layout_width="100dp"
        android:layout_height="45dp"
        android:layout_below="@id/txtIP"
        android:layout_marginRight="10dp"
            <!--marginRight - velikost pravého odsazení od okraje na 10
            obrazových bodů. -->
        android:text="Připojit"
/>
```

Zavedení prvku pro skupinu přepínačů [12][13]:

```
<RadioGroup
            <!--RadioGroup - zavedení skupiny přepínačů. -->
        android:id="@+id/radio_cidlo"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@+id/txtVyberCidla"
        android:layout_marginTop="0dp">


  <RadioButton
            <!--RadioButton - zavedení prvního přepínače. -->
             android:id="@+id/termistor_2v"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
             android:text="Termistor 2-vodičový"

  />
        <RadioButton
            <!--RadioButton - zavedení druhého přepínače. -->
             android:id="@+id/termistor_4v"
             android:layout_width="wrap_content"
             android:layout_height="wrap_content"
             android:text="Termistor 4-vodičový"
        />
 </RadioGroup>
```

Jak je vidět z uvedených částí zdrojového kódu, sestavení stylu zobrazení není nikterak obtížné. Používají se jen prvky s parametry, které určují jak a kde se daný prvek bude zobrazovat. Důležité je *id* prvku, na které budeme navazovat v souborech s příponou *.java*.

7.3. Hlaví kód aplikace

Jedná se o kód ze souborů s příponou *.java* a jak již bylo popsáno, tento soubor obsahuje hlavní třídy a jejich metody samotné aplikace. Jednořádkové komentáře se v tomto souboru zapisují za symbol // , blokové nebo-li víceřádkové komentáře se zapisují mezi symboly `<** komentář **>`.

7.3.1. Import tříd a dalších částí

V první řadě musí programátor importovat do projektu třídy pomocí příkazu `import cesta.ke.třídě`. Pokud aplikace bude používat funkce nad rámec základních knihoven, je třeba importovat příslušné externí knihovny tak, jak je popsáno v kapitole č.2.3., aby programátor mohl potřebné třídy z knihovny importovat do programu. Důležitý je také import tříd prvků z XML souboru.

```
import android.app.Activity;
import android.graphics.Color;
import android.os.Bundle;
            //Activity, graphics, Bundle - import základních tříd.

import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
            <**Button, EditText atd. - import tříd prvků použitých v
            XML souboru vzhledu. **>

import com.Agilent_34410A_Temp.R;
import com.agilent.android.io.AAIoException;
import com.agilent.android.io.IO;
            //Import knihoven a balíčků.
```

7.3.2. Vytvoření hlavní třídy

V dalším kroku je vytvořena metoda, ve které je celá aplikace zapsána. Tedy veškerý zdrojový kód uvedený dále v dokumentu je zapsán přímo do této metody [14].

```
public class Agilent_34410A_Temp extends Activity
  {

              // zde se nachází tělo třídy

              // public - označuje modifikátor přístupu ke třídě

              <** Agilent_34410A_Temp - označuje identifikátor třídy, je
              to vlastní pojmenování třídy **>

  }
```

7.3.3. Zavedení proměnných

V hlavní třídě musí programátor na začátku zavést proměnné v příslušném formátu a také proměnné například pro tlačítka a texty se kterými bude v kódu pracovat. V jazyku Java mají proměnné stejné datové typy jako v ostatních programovacích jazycích.

```
private Button btnPripoj ;
private Button btnZmer;
            //Zavedení proměnných hodnot tlačítek.
private TextView txtInfo;
private TextView txtVysledek;
            //Zavedení proměnných textových polí.

private EditText etxtIP ;
            //Proměnná editovatelného pole.
private IO pripojeni = null;
boolean not34410A = false;
private String ipAddresa ="";
private String jednotka ="";
private String termistor ="";
private String odpor ="";
private String nast_cidla ="";
            //Ostatní proměnné.


private TabHost MyTabHost;
private TabHost.TabSpec MyTabSpec;
            //Proměnné záložek.


private RadioGroup radio_jednotkaGroup;
private RadioButton radio_jednotkaButton;
            //Proměnné přepínačů.
```

7.3.4. Vyvolání metody onCreate

V dalším kroku je vyvolána metoda *onCreate*, která je volána jednou a to při spuštění aplikace. V této metodě mohou být vnořeny další metody jako například metoda *onClick*, která obsluhuje tlačítko a reaguje na jeho stisknutí. Důležité je propojení externího XML stylu s Java dokumentem, které se provádí příkazem `setContentView(R.layout.main)`, `kde main` označuje název příslušného XML souboru ve složce *layout*. Neméně důležité je následné propojení prvků z XML souboru pomocí identifikátoru ID k jejich funkčním metodám [14].

```
@Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        requestWindowFeature(Window.FEATURE_CUSTOM_TITLE);
            setContentView(R.layout.main);
             //vyvolání metody onCreate a také propojení se stylem.
```

```
RadioButton ter2v = (RadioButton)
findViewById(R.id.termistor_2v);
RadioButton Odpor_5k = (RadioButton)
findViewById(R.id.odpor_5k);
RadioButton CEL = (RadioButton) findViewById(R.id.Cel);
     //Propojení přepínačů s jejich prvky v xml souboru
```

Pomocí příkazu setChecked(true) je možné nastavit určitý přepínač jako předem vybraný. Výběr příslušeného přepínače se provede zapsáním jména před nastavovací příkaz. Za pomocí hodnoty true = pravda a false = nepravda můžeme tyto prvky přepínat.

```
Odpor_5k.setChecked(true);
CEL.setChecked(true);
     //Nastavení přepínačů na předdefinovaný výběr.

etxtIP = (EditText)findViewById(R.id.etxtIP);
txtVysledek = (TextView)findViewById(R.id.txtVysledek);
txtInfo = (TextView)findViewById(R.id.txtInfo);
     //Propojení s textovými poli.
```

Příkazem setVisibility můžeme ovládat viditelnost textu. Parametrem 0 nebo false nastavíme text jako neviditelný a parametrem 1 nebo true jako viditelný. Tato funkce se dá využít stejným způsobem u všech zobrazovaných prvků, například u tlačítek.

```
txtInfo.setVisibility(0);
     //Nastavení viditelnosti txtInfo na 0.

btnZmer = (Button) findViewById(R.id.btnZmer);
btnZmer.setEnabled(false);
     /**Propojení s tlačítkem Změřit a jeho nastavení na
     neaktivní. **/
```

7.3.5. Metoda onClick pro tlačítko

Metoda reagující na stisknutí zobrazeného tlačítka. Pokud je tlačítko stisknuto, provedou se příkazy v těle metody.

```
btnPripoj.setOnClickListener(new View.OnClickListener()
     {
          @Override
          public void onClick(View v)
          {
               ipAddresa = etxtIP.getText().toString();
          //Načtení zadaného textu do proměnné ipAddresa.

               txtInfo.setText("");
               txtInfo.setTextColor(Color.RED);
          //Nastavení Infotextu na prázdno a červenou.
```

Ve zdrojovém kódu uvedeném níže je ukázka cyklu řídícího zobrazení informačního textu v případě různých chybových stavů. Tyto stavy jsou popsány v kapitole č.4.2.

Jako první cyklus testuje proměnnou *ipAddresa*, která načítá řetězec znaků z textového pole pro zadání IP adresy. Pokud není zadaná IP adresa, cyklus vykoná příkazy uvnitř těla: nastaví zadaný text do textového pole *txtInfo* a poté toto pole nastaví viditelné.

Jestliže je zadána IP adresa cyklus testuje zda proběhlo připojení. Pokud ano, deaktivuje tlačítko *btnPripoj*, aktivuje tlačítko *btnZmer*, nastaví a zobrazí text v textovém poli *txtInfo*.

V případě neúspěšného připojení cyklus otestuje zda je proměnná *not34410A* pravdivá, pokud je pravdivá, vykoná opět příkazy uvnitř cyklu a pokud není pravdivá, vykoná příkazy v části *else*.

```
        if(ipAddresa.equals(""))
        {
              txtInfo.setText("Prosím, zadejte IP adresu
              přístroje\nAgilent 34410A.");
              txtInfo.setVisibility(1);
/**Když ipAddresa není zadána, nastav txtInfo na uvedený
text a nastav jej viditelným. **/
        }
        else
        {
              if(Connect())
              {

                    btnPripoj.setEnabled(false);
                    btnZmer.setEnabled(true);
                    txtInfo.setText("Přístroj byl
                    připojen.");
                    txtInfo.setVisibility(1);
/**Pokud připojení proběhne v pořádku, nastav tlačítko
Připojit na neklikatelné a tlačítko Změřit na klikatelné,
zároveň vypiš text. **/
              }
              else
              {
                    if(not34410A){
                          txtInfo.setText("Zadaná adresa
                          neodpovídá přístroji Agilent
                          34410A.");
                          txtInfo.setVisibility(1);
 /**Pokud na zadané adrese není 34410A, vypiš text.**/
                    }
```

```
                                else
                                      txtInfo.setText("Nelze se
                                      k přístroji.");
                                      txtInfo.setVisibility(1);
                  /**Pokud spojení selže, vypiš text. **/
                                }
                        }
                  }
            });
```

7.3.6. Metoda Connect()

Tato metoda již není vnořená do *onCreate*. Jedná se o samostatnou metodu, která zajišťuje propojení aplikace s přístrojem.

Po zavolání této metody je vytvořena instance nové připojení, následně je otevřeno spojení se zadanou IP adresou na portu 5025. Pokračuje vyčištění portu zařízení a odeslání příkazu *IDN?. Dále je vytvořen zásobník pro odpověď a ta je do něj následně zachycena. Nakonec je odpověď otestována zda se v ní nachází řetězec 34410A a vyhodnocena. Pokud odpověď obsahovala hledaný řetězec, jsou data připravena k vyzvednutí pro další zpracování v dalších částech aplikace.

```
private boolean Connect()
{
      StringBuffer odpoved;
      try
      {
            pripojeni = new IO();
            pripojeni.openWithAddress(ipAddresa, 5025);
           /**Otevření spojení se zadanou IP adresou na portu pro SCPI
           5025. **/
            pripojeni.queryDeviceClearPort("");
            pripojeni.deviceClear();
            pripojeni.print("*IDN?");
           /**Odeslání příkazu *IDN?, který zpětně odešlě informace o
           přístroji. **/
            odpoved = new StringBuffer();
            pripojeni.scan(response);
           /**Nastavení proměnné pro informace a zachycení příchozích
           informací. **/
            if(odpoved.toString().lastIndexOf("34410A") == -1)
            {
                  not34410A = true;
           /**Pokud se příchozí informace neshoduje s 34410A, vrátí
           pravdu. **/
                  return false; }}}
           /**V jiném případě vrátí nepravdu. **/
```

7.3.7. Vytvoření dialogu

Tato metoda vyvolává dialogové menu a je použita v aplikaci DSO-X 2012A Remote. Vyvolání je nastaveno po stisknutí textu *Rozlišení* jak je vidět na Obr. 6.2. stisknutím tlačítka *Změřit* [15].

```
txtPoints.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v) {
      final String[] colors_array =
                  {"100","250","500","1000","2000","5000"};
      AlertDialog.Builder builder = newAlertDialog.Builder
                  (Agilent_34410A_Temp.this);
      builder.setTitle("Rozlišení");
      builder.setItems(colors_array, new
                  DialogInterface.OnClickListener()
      {
public void onClick(DialogInterface dialog, int which)
      {
        txtPoints.setText("Rozlišení: " + colors_array[which]);
        txtPoints.setVisibility(1);
        points = colors_array[which];
      }
}).show();
}
});
```

7.3.8. Spuštění nové aktivity

Tato metoda vyvolává novou aktivitu a je použita v aplikaci DSO-X 2012A Remote. Vyvolání je nastaveno po stisknutí tlačítka *Graf* jak je vidět na Obr. 6.3. První aktivita se automaticky přepne do režimu spánku a čeká na případné obnovení, druhá aktivita se spustí do popředí. Nová aktivita je zapsána v souboru *Graf_Osciloskop*, ze kterého se spustí.

Vzhledem k tomu, že lze s proměnnými pracovat jen v dané aktivitě kde byla proměnná vytvořena, je nutné mezi aktivitami přenášet data. Tento problém řeší příkaz `intent.putExtra()`, který data nachystá pro předání do druhé aktivity [16].

```
btnGraf.setOnClickListener(new View.OnClickListener()
{
@Override
public void onClick(View v)
      {
      Intent intent = new
            Intent(Agilent_34410A_Temp.this,Graf_Osciloskop.class);
      /**Vytvoření nové aktivity, v závorce je zapsána původní
      aktivita a   volaná aktivita.**/

      intent.putExtra("ipAdresa", ipAddresa);
      if (points.equals("")){
```

```
                intent.putExtra("points", "100");
                //zde jsou data připravena pro předání do spouštěné
                aktivity
                }
                else{
                      intent.putExtra("points", points);
                }
                startActivity(intent);
                }
        });
```

7.3.9. Načtení dat do nové aktivity

Data v původní aktivitě ze které voláme novou aktivitu jsou nachystaná k předání, jak je popsáno v kapitole č.7.3.8. Tato data do nové aktivity načteme pomocí níže uvedeného kódu.

```
Bundle bundle = getIntent().getExtras();
String ipAdresa = bundle.getString("ipAdresa");
     //načtení dat z proměnné ipAdresa v původní aktivitě

String points = bundle.getString("points");
     //načtení dat z proměnné points v původní aktivitě
```

7.3.10. Vytvoření grafu

V aplikaci Agilent DSO-X 2012A je vykreslení grafu naprogramováno následujícím způsobem [17]:

Níže popsaný kód vytvoří graf s nulovými hodnotami a tím jej připraví pro data načtená z osciloskopu. Tento kód je proveden pouze jednou při spuštění aplikace.

```
exampleSeries1 = new GraphViewSeries(new GraphViewData[] {
              new GraphViewData(0, 0d)
             ,new GraphViewData(50, 0d)
             ,new GraphViewData(100, 0d)
});
      //vytvoření a načtení nulových bodů série

graphView = new LineGraphView(this, "Graf");
      //vytvoření nového grafu

graphView.addSeries(exampleSeries1);
      //vykreslení dat do grafu

LinearLayout layout = (LinearLayout) findViewById(R.id.graph1);
      //vyhledání prvku pro zobrazení grafu v XML souboru podle ID

layout.addView(graphView);
      //vykreslení grafu na obrazovku
```

Obnovování a vykreslování nových dat zajišťuje metoda run(), ve které nepřetržitě probíhá spouštění příkazů v jejím těle. V první fázi je odstraněna původní vykreslená křivka z grafu, aby ji mohla nahradit nová. Jsou načtena data průběhu signálu z přístroje ve kterých se na začátku nachází 23znaková hlavička po níž následují jednotlivé hodnoty úrovně signálu oddělené čárkou. Následně je zjištěna délka řetězce načtených dat, díky tomu můžeme přesně odfiltrovat čistá data grafu od hlavičky. Ta jsou poté načtena a rozdělena podle čárky do pole řetězců. Poté je zjištěna délka vytvořeného pole dat a podle této délky je přizpůsobena osa grafu. Data jsou jednotlivě zapsána do pole grafu a vykreslena.

```
graphView.removeSeries(exampleSeries1);
            //odstraní původní zobrazená data z grafu

channel_1 = Channel_1().toString();
            //zavolání metody pro načtení dat z přístroje

int length_1 = channel_1.length();
            //zjištění délky řetězce příjmutých dat

String druhy_1 = channel_1.substring(24,length_1);
            //načtení čistých dat za hlavičkou

wave_array_1 = druhy_1.split(",");
            //rozdělení řetězce dat na pole dat podle čárky

size_array_1 = wave_array_1.length;
            //načtení délky upraveného pole

GraphViewData[] datos_1= new GraphViewData[size_array_1];
            // vytvoření pole dat grafu


for (int i=0; i<size_array_1; i++)
{
datos_1[i] = new GraphViewData(i, Double.valueOf(wave_array_1[i]));
            //přenesení pole dat do pole grafu
}
exampleSeries1 = new GraphViewSeries("Channel_1", new
GraphViewSeriesStyle(Color.rgb(200, 50, 00), 3), datos_1);
            //zapsání pole dat grafu do nové křivky

graphView.addSeries(exampleSeries1);
            //vykreslení grafu
```

8. ZÁVĚR

Z dostupné literatury byl vytvořen přehled historie operačního systému Android, který v dnešní době můžeme nalézt na většině mobilních zařízeních. Jeho další rozvoj a rozšíření do ostatních platforem jako jsou palubní počítače automobilů, bílá technika, ovládání domácnosti je velice pravděpodobné a to hlavně díky jednoduchosti a otevřenosti systému.

Dále bylo z dostupné literatury prostudováno připojení laboratorních přístrojů do sítě Internet a jejich možnosti vzdáleného ovládání. Připojení přístrojů do sítě internet je možné mnoha způsoby a je velice jednoduché. Jejich ovládání je také jednoduché a to díky příkazům SCPI.

Úkolem práce bylo seznámení se s vývojovým prostředím pro mobilní zařízení s operačním systémem Android a osvojení programovacího jazyka pro tuto platformu.

Jako první byla vytvořena aplikace umožňující uživateli připojení k přístroji Agilent 34410A a jeho nastavení pro měření teploty. Uživatel má dále v aplikaci možnost výběru z více metod měření, čidel a jednotek. Výsledek měření je následně zobrazen na displeji. Aplikace byla pojmenována „34410A Temp".

Druhá naprogramovaná aplikace slouží k inicializaci a ovládání libovolného měřícího přístroje pomocí příkazového řádku prostřednictvím SCPI příkazů. Tato aplikace byla pojmenována „Agilent Prompt" a dokáže komunikovat s jakýmkoliv zařízením firmy Agilent, které je připojené do sítě pomocí LAN nebo GPIB. V případě odeslání příkazu s dotazem na data je aplikace schopna tato data zachytit a vypsat na obrazovku.

Hlavním úkolem bylo vytvoření aplikace, která by umožňovala ovládání funkčního generátoru a osciloskopu Agilent DSO-X 2012A. Aplikace umí vykreslovat průběh měřeného signálu na displej mobilního zařízení v reálném čase a zároveň dokáže ovládat veškeré funkce integrovaného generátoru v osciloskopu. Název aplikace je „DSO-X 2012A Remote" a je do ní integrován i příkazový řádek pro možnost případného ovládání jiných laboratorních přístrojů firmy Agilent.

V práci je popsána instalace a nastavení vývojového prostředí Eclipse, které bylo vybráno z mnoha prostředí na základě recenzí programátorů a vývojářů. Dále je popsáno vytvoření a nastavení první ukázkové aplikace a importování externích knihoven do projektu. Další část se zabývá možnostmi připojení měřících přístrojů a jejich vzdáleného ovládání. V poslední části je popsaná struktura aplikace a jednotlivé části zdrojového kódu s popisem funkčnosti jednotlivých části. Vzhledem k rozsáhlému popisu problematiky v této práci, může být použita jako návod pro začínající vývojáře a programátory. Na práci by mohlo být plynule navázáno další činností, která by aplikaci rozšířila o další funkce a možnosti využití.

9. SEZNAM POUŽITÝCH ZDROJŮ

[1] UJBÁNYAI, Miroslav. *Programujeme pro Android*. Vyd. 1. Praha: Grada, 2012, 187 s. Průvodce (Grada). ISBN 978-80-247-3995-3.

[2] ORACLE. *Java SE 7: Features and Enhancements* [online]. 2013 [cit. 2013-11-20]. Dostupné z: http://www.oracle.com/technetwork/java/javase/jdk7-relnotes-418459.html

[3] GOOGLE. *Get the Android SDK* [online]. 2013 [cit. 2013-11-23]. Dostupné z: http://developer.android.com/sdk/index.html

[4] H TEST A.S. *Návod k obsluze 34410A /11A*. Praha, 2005. Dostupné z: http://www.htest.cz/download/34410A.pdf

[5] Agilent. *34410A Digital Multimeter, 6½ Digit High Performance* [online]. 2013 [cit. 2013-11-23]. Dostupné z: http://www.home.agilent.com/en/pd-692834-pn-34410A/digital-multimeter-6-digit-high-performance?cc=CZ&lc=eng

[6] DSOX2012A Oscilloscope: 100 MHz, 2 Channels. AGILENT. *Agilent Technologies* [online]. 2014 [cit. 2014-06-04]. Dostupné z: http://www.home.agilent.com/en/pd-1945059-pn-DSOX2012A/oscilloscope-100-mhz-2-channels

[7] Ústav řízení systémů a spolehlivosti. *Programování GPIB* [online]. 2013 [cit. 2013-11-25]. Dostupné z: http://www.rss.tul.cz/ftppub/cms/04_%20GPIB_%20SW.pdf

[8] Toasts. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/guide/topics/ui/notifiers/toasts.html

[9] TextView. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/widget/TextView.html

[10] EditText. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/widget/EditText.html

[11] Button. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/widget/Button.html

[12] RadioGroup. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/widget/RadioGroup.html

[13] RadioButton. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/widget/RadioButton.html

[14] Activity. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/app/Activity.html

[15] Dialog. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/app/Dialog.html

[16] Intent. *Android Developers* [online]. 2014 [cit. 2014-06-01]. Dostupné z: http://developer.android.com/reference/android/content/Intent.html

[17] Graph View. *GitHub* [online]. 2014 [cit. 2014-06-01]. Dostupné z: https://github.com/jjoe64/GraphView