Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Aplikace Android - teorie a praxe

Jakub Dudáš

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Jakub Dudáš

Informatika

Název práce

Aplikace android – teorie a praxe

Název anglicky

Android application – theory and practice

Cíle práce

Cílem bakalářské práce je charakterizovat problematiku vývoje aplikací pro operační systém Android. Ten se skládá z následujících dílčích cílů: popis architektury platformy Android a vývojových nástrojů, tvorba aplikace v programovacím jazyce Java, distribuce výsledné aplikace.

Metodika

Metodika je založena na studiu a analýze odborných informačních zdrojů. Na základě těchto zdrojů bude vytvořena jednoduchá aplikace. Na základě syntézy teoretických a praktických poznatků bude formulován závěr bakalářské práce.

Doporučený rozsah práce

30-40 str.

Klíčová slova

Android, mobilní operační systém, Java

Doporučené zdroje informací

ALLEN G. Android 4., Průvodce programováním mobilních aplikací. Computer Press, a.s.. 2011; 369 s. ISBN 978-80-2513-782-6

Android http://developer.android.com/guide/index.htm

MEIER R. Professional Android Application Development, 2. vydání. Indianapolis: Wrox, 2010, 576 str., ISBN 0470565527.

MURPHY, Mark L.. Android 2, Průvodce programováním mobilních aplikací. Computer Press, a.s.. 2011; 369 s. ISBN: 978-80-251-3194-7

ROGERS, R., LOMBARDO, J. Android Application Development, 1st Edition. Cambridge: O'Reilly Media, Inc. 2009, 336s., ISBN 978-0-596-52147-9.

Windows – http://code.google.com/android/documentation.html

Předběžný termín obhajoby

2016/17 LS – PEF

Vedoucí práce

Ing. Čestmír Halbich, CSc.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 18. 10. 2016

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 24. 10. 2016

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 14. 03. 2017

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Aplikace Android - teorie a praxe" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne _____

Poděkování

Rád bych touto cestou poděkoval vedoucímu bakalářské práce panu Ing. Čestmíru Halbichovi, Csc. za odborné rady a konzultace během vedení této bakalářské práce.

Aplikace Android - teorie a praxe

Souhrn

Bakalářská práce je zaměřena na problematiku vývoje aplikací na operační systém Android. V teoretické části je nedříve je popsána historie Android a historie konkurenčních platforem, Apple iOS a Windows Mobile. Dále jsou popsány výhody a nevýhody vytváření aplikací pro tyto operační systémy. Dále už je podrobný popis pouze operačního systému Android. Součástí popisu je i popis jednotlivých vývojových nástrojů pro Android. Dále pak popis architektury operačního systému Android, krátký úvod do programování mobilních aplikací a seznámení s Google Play.

Dále pak v praktické části je přiblížena aplikace, která byla vytvořena. Jednotlivé části kódu jsou poté popsány a vysvětleny. Zakončením praktické části se seznámení s nahráváním aplikací na Google Play.

Klíčová slova: Android, mobilní operační systém, historie, konkurence, vývojové prostředí, architektura, aplikace, Google Play

Application Android - theory and practice

Summary

Bachelor thesis is focused on developing applications on the Android operating system. The theoretical part describes the history nedříve Android and history of competing platforms, Apple iOS and Windows Mobile. The following describes the advantages and disadvantages of creating applications for these operating systems. Furthermore, it is merely a detailed description of the Android operating system. Part of the description is the description of development tools for Android. Then the description of the Android operating system architecture, introduction to programming mobile applications and familiarity with Google Play.

Furthermore, the practical part is zoomed application that was created. Individual parts of the code are then described and explained. By completing the practical part of the familiarization with recording applications on Google Play.

Keywords: Android, the mobile operating system, history, competition, development environment, architecture, applications, Google Play

Obsah

Seznam obrázků

1 Úvod

Téma bakalářské práce jsem zvolil z důvodu rapidně se zvyšujícího zájmu o chytré mobilní telefony. Android se stal nejpopulárnějším a nejpoužívanějším operačním systémem pro tyto chytré telefony. Je to hlavně kvůli tomu, že se jedná o zdarma nabízený open-source, který si mohou výrobci a prodejci chytrých telefonů upravit, aby fungoval právě na jejich zařízení bezchybně. Práce se věnuje právě tomuto vývoji na Android, protože je možné zasáhnout co možná největší skupinu potenciálních uživatelů.

2 Cíle práce

Cílem bakalářské práce je charakterizovat problematiku vývoje aplikací pro operační systém Android. Ten se skládá z následujících dílčích cílů: popis architektury platformy Android a vývojových nástrojů, tvorba aplikace v programovacím jazyce Java, distribuce výsledné aplikace.

3 Metodika

Metodika bakalářské práce je založena na studiu a analýze odborných informačních zdrojů a jejich zobecnění. Jako hlavní zdroj byl použit internet, kde je možné dohledat nejaktuálnější informace na toto téma. V praktické části je uveden samotný vývoj aplikace „Snake“. Vývoj této aplikace je podrobně popsán pro pochopení aplikační logiky. Závěr práce bude formulován na základě syntézy teoretických poznatků a výsledků vlastního řešení.

4 Mobilní operační systémy

4.1 Historie operačního systému Android

Historie verzí mobilního operačního systému Android začala s vydáním Android alpha v 5. 11. 2007. První komerční verze Android 1.0 byla vydána v září 2008. Android je neustále vyvíjen společností Google a Open Handset Alliance. Verze 1.0 a 1.1 nebyly uvolněny pod konkrétními kódovými názvy, ale od roku 2009, kdy byl představen Android 1.5 Cupcake. Jednotlivé Android verze měli tato tematická krycí jména. (4)

4.1.1 Android 1.5 Cupcake

Cupcake byl představen 30. dubna 2009. S ním přibylo i mnoho nových funkcí. Např. Možnost nahrávaní videí a jejich možnost následně sledovat, nebo nahrát na Youtube přímo z telefonu. Nová klávesnice, která dokončovala slova a rozšířená funkce kopírováni textu. Také přibyla možnost připojení přes Bluetooth A2DP.) (4)

4.1.2 Android 1.6 Donut

Donut byl představen 15. září 2009. V této verzi byl vylepšen Android Market na prostředí fotoaparátu. Bylo také vylepšeno vyhledávání hlasem a přidán Quick Search box, pro rychlé vyhledávání. Nově také podporoval rozlišení displeje WVGA. (4)

4.1.3 Android 2.0 / 2.1 Eclair

Eclair 2.0 byl představen 26. října 2009, posléze bylo vydáno doplnění Eclair 2.0.1 a to 3. prosince 2009 a Eclair 2.1 12. ledna 2010. Eclair přinesl podporu všech velikostí displejů a zoptimalizoval rychlost hardwaru. Byla přidána i možnost digitálního přiblížení u fotoaparátu a podpora světelné diody. Poté bylo vylepšeno uživatelské rozhraní a některé aplikace z minulých verzí operačního systému. (4)

4.1.4 Android 2.2 Froyo

Froyo byl představen 20. května 2010. V této verzi bylo umožněno instalovat aplikace na paměťovou kartu a vytvoření z mobilu tzv. WiFi hotspot, nebo sdílení internetového

připojení přes USB kabel. Bylo vylepšeno uživatelské rozhraní a i práce s hardwarem, např. vylepšená správa paměti RAM a přidání kompilátoru JIT (Just In Time), díky kterému se podařila zrychlit rychlost systému. Také byla přidána podpora pro OpenGL ES 2.0 a další vrstva vývojářského API. (4)

4.1.5 Android 2.3 / 2.4 Gingerbread

Gingerbread byl představen 6. prosince. Tato verze se hlavně zaměřila na vývoj internetových služeb a síťového propojení zařízení. Byla přidána podpora formátu WebM pro HTML5 video a pro Near Field Communication. Také bylo vylepšeno uživatelské rozhraní a ovládání kamer. (4)

4.1.6 Android 3.0 / 3.1 / 3.2 Honeycomb

Honeycomb byl představen 22. února 2011. Jeho největší předností je vylepšení multitaskingu. Honeycomb se hlavně zaměřoval na uživatelské rozhraní a to včetně optimalizace pro velké obrazovky. (4)

4.1.7 Anroid 4.0 – 4.0.4 Ice Cream Sandwich

Ice Cream Sandwich byl představen 19. října 2011. Největší novinkou byl Android Beam. Jedná se o funkci, která umožňuje posílat data na krátkou vzdálenost pomocí technologie NFC (Near field communication). (4)

4.1.8 Android 4.1 / 4.2 / 4.3 Jelly Bean

Jelly Bean byl představen 9. července 2012. V této verzi se Android zaměřil hlavně na funkčnost operačního systému. Toto zlepšení pojmenoval Project Butter. Kromě dalších vylepšení bylo přidáno i více uživatelských účtů a podpora OpenGL ES 3.0. (4)

Obrázek 1 - Android 1.5 - 4.2

Zdroj: http://i-cdn.phonearena.com/images/articles/96766-image/Android-2.2-Froyo.jpg

4.1.9 Android 4.4 KitKat

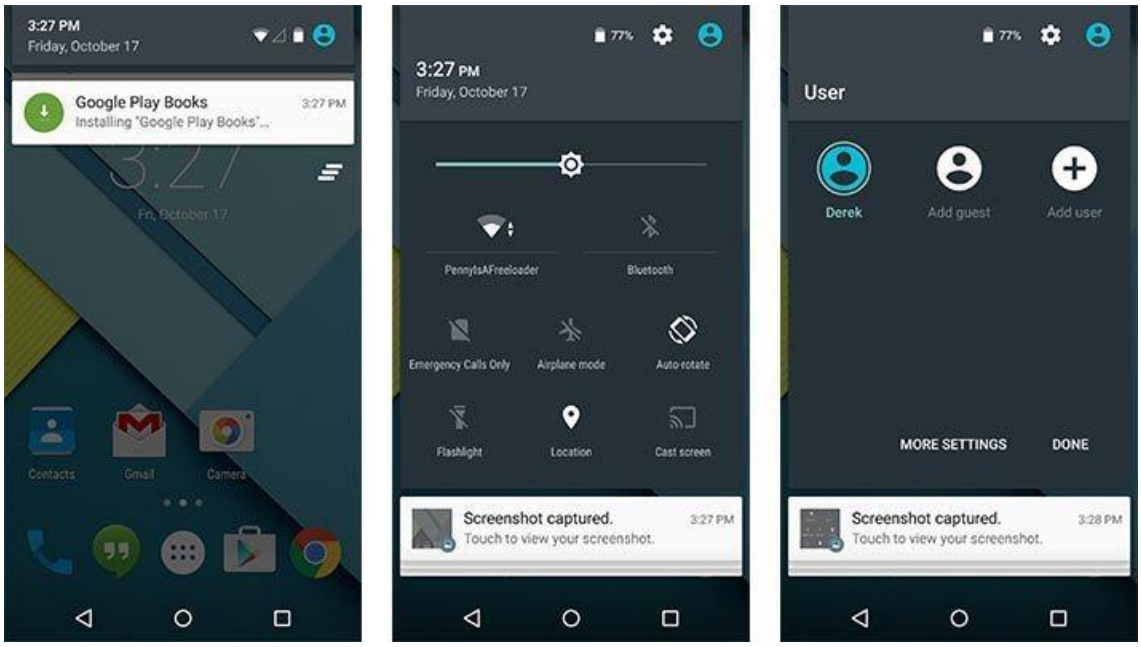
KitKat byl představen 25. června 2014. Největším zlepšením byla optimalizace operačního systému pro zařízení s menší RAM pamětí. Také bylo vylepšeno ovládání notifikační lišty tak, že je možnost ji úplně schovat a zobrazit z kterékoli jiné aplikace. Dále byla přidána možnost tisku přímo z mobilního telefonu. (4)

Obrázek 2 - Android 4.4 Kitkat

Zdroj: http://2.bp.blogspot.com/-GAbwewJeqGo/VB7aLYGgeoI/AAAAAAAAATc/Pg8d7RTTdXw/s1600/Download-Android-4.4-kitkat-apps-apk.jpg

4.1.10 Android 5.0 / 5.1 Lollipop

Lollipop byl představen 25. června. Tato verze přinesla zatím nejvíce vylepšení, ať už po grafické stránce tak i výkonnostní. Byla také přidána spousta nových funkcí, např. Nový notifikační systém, Project Volta, který snižuje energetickou náročnost systému. Dále pak funkce Smart Lock, funkce Tap & go, pohotovostní režim. Byla také poskytnuta možnost mít 64 bitový operační systém. Podpora více SIM karet najednou. (4)

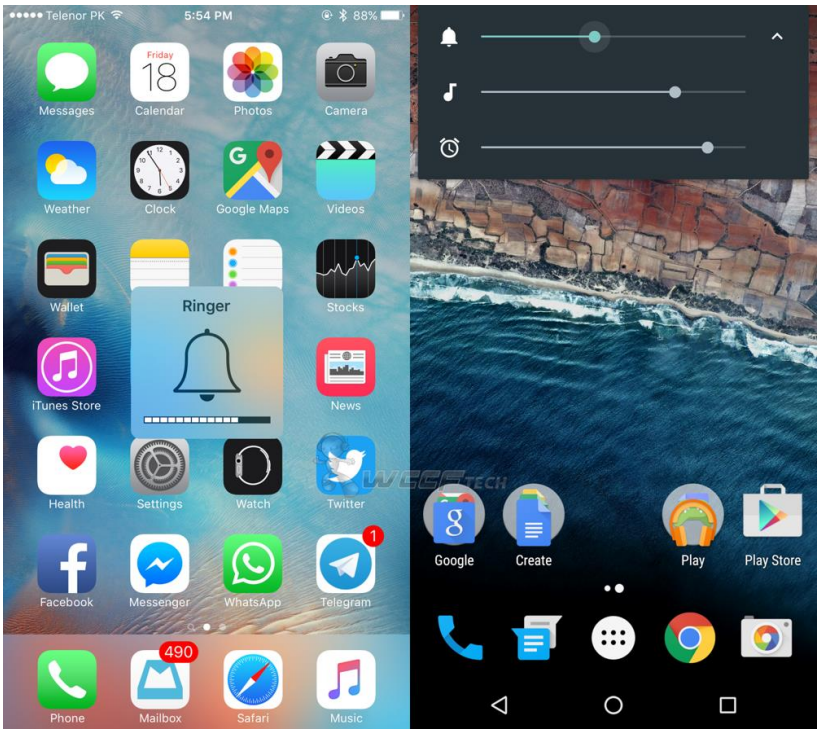


Obrázek 3 - Android 5.1 Lollipop

Zdroj: http://getpcsoft.wikisend.com/img_howto/0/358/Android-Lollipop1.jpg

4.1.11 Android 6.0 Marshmallow

Marshmallow byl představen 5. října 2015. V této verzi se vývojáři zaměřili na uživatelské a vývojářské potřeby. Byla zefektivněna výdrž baterie až na dvojnásobek a přidána podpora USB typu C. Uživatel může definovat skupiny a přidělit jim určitá oprávnění. Verze podporuje také rozeznávání otisků prstů. (4)

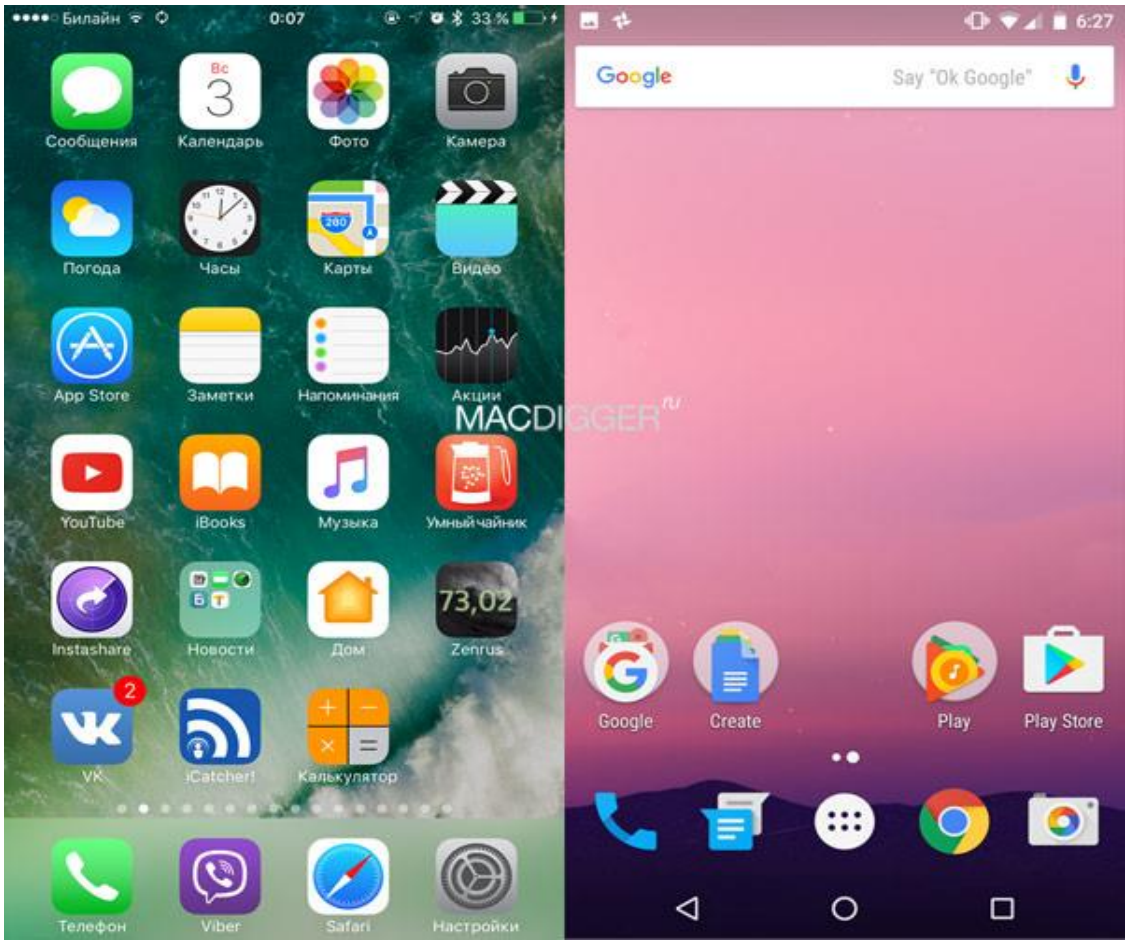


Obrázek 4 - Android 6.0 Marshmallow

Zdroj: http://www.likeagain.com/data/userfiles/image/Android%206_0%20Marshmallow%20vs%20iOS%209-1.png

4.1.12 Android 7.0 Nougat

Nougat byl představen 22. srpna 2016. Největší novinkou je práce s více okny naráz. Tedy je možné rozdělit obrazovku na dvě části a mít spuštěné dvě na sobě nezávislé aplikace. Je možné si vybrat které aplikaci bude přidělen větší výkon. Přidána také nová API pro virtuální realitu a podpora vykreslovacího API Vulkan 3D. (13)



Obrázek 5 - Android 7.0 Nougat

Zdroj: http://actualapple.com/wp-content/uploads/2016/07/1f0ef0eb8ca310c0761d56e217d1e3eb.jpg

4.2 Historie operačního systému Apple iOS

iPhone je řada chytrých telefonů, které vyrábí firma Apple Inc. Používají Apple iOS mobilní operační systém. První generace iPhone se začala prodávat 29. června 2007. V červnu 2016, Apple App Store obsahoval více než 2 miliony aplikací dostupných pro iPhone. Apple vydal dohromady deset generací modelů iPhone a každý z nich doprovázel jeden z desíti hlavních vydaných operačních systému iOS. Původní první generace iPhone byl GSM telefon se zavedeným designem iPhonů, například rozmístění ovládacích prvků, které přetrvávalo v průběhu všech verzí a velikostí obrazovky, které byly zachovány po dobu dalších čtyř iterací. (6)

4.2.1 iPhone 3G

iPhone 3G přidal podporu sítě 3G. Následně přidal i připojení přes 3GS s vylepšeným hardwarem. (6)

4.2.2 iPhone 4

Do verze iPhone 4 s kovovým krytem, přidal Apple vyšší rozlišení displeje a čelní fotoaparát. iPhone 4S následně přidal hlasové vyhledávání a asistent Siri. (6)

4.2.3 iPhone 5

iPhone 5 představoval 4 palcový displej a nově zavedený Apple Lightning connector. V roce 2013 Apple představil iPhone 5S s vylepšeným hardwarem a čtečkou otisků prstů. (6)

4.2.4 iPhone 6

Následoval větší iPhone 6 u kterého bylo možné zvolit ze dvou velikostí displeje, 4,7 a 5,5 palce. iPhone 6S byl představen následující rok, který uváděl upgrade hardwaru a podporu pro dotykové vstupy citlivých na tlak. (6)

4.2.5 iPhone 7

V roce 2016 Apple představil iPhone 7 a 7 Plus, které přidávají vodotěsnost, nové barevné zobrazení uživatelského rozhraní a grafický výkon. Nové zadní nastavení dvojité kamery na modelu Plus a odstranění sluchátkového konektoru 3,5 mm. Obchodní úspěch iPhonu s přebudováním smartphone průmyslu, přispěl k tomu, že se Apple stal jednou z nejcennějších veřejně obchodovaných společností na světě do roku 2011. Ve Spojených státech drží iPhone největší podíl na trhu se smartphony. Od konce roku 2015 měl iPhone podíl na trhu 43,6%, následuje Samsung (27,6%), LG (9,4%) a Motorola (4,8%). (6)

Obrázek 6 - iPhone evoluce

Zdroj: http://2.bp.blogspot.com/-sEEHDOG9Jjs/VB7dswdIYcI/AAAAAAAACzg/Lqa4dvFAHH0/s1600/ios_homescreen.jpg

4.3 Windows phone historie

Dne 11. února 2011 na tiskové konferenci v Londýně generální ředitel společnosti Microsoft Steve Ballmer a generální ředitel společnosti Nokia Stephen Elop oznámili partnerství mezi svými společnostmi, v nichž by se Windows Phone stal hlavním operačním systémem pro telefony Nokia a nahradil tím Symbian. (5)

4.3.1 Windows Phone 7

Windows Phone 7 byl veřejně představen 8. listopadu 2010 ve Spojených státech. V roce 2011 pak společnost Microsoft vydala aktualizovaný operační systém. (5)

4.3.2 Windows Phone 7.5 Mango

Windows Phone 7.5 Mango. Aktualizace obsahovala mobilní verzi Internet Explorer 9, který podporuje stejné webové standardy a grafické schopnosti jako desktopové verze. Multi-tasking aplikací třetích stran, Twitter integrace pro lidi Hub, a Windows live SkyDrive připojení k internetu. Menší aktualizace byla vydána v roce 2012 známá jako "Tango". (5)

4.3.3 Windows Phone 8

Windows Phone 8 je druhá generace mobilního operačního systému Windows Phone od Microsoftu. Ten byl představen 29. října 2012. Windows Phone 8 nahradí systém Windows CE, který byl použit v architektuře Windows Phone 7 a to jádrem Windows NT. Ale bohužel není možnost upgradovat z Windows Phone 7 na Windows Phone 8. (5)

4.3.4 Windows Phone 8.1

Windows Phone 8.1 byl představen dne 2. dubna 2014. Nová verze obsahuje systém oznámení, internet Explorer 11 s okrajem pro synchronizaci mezi Windows 8.1 zařízení a zařízení WP. Počínaje tímto vydáním Microsoft upustil od požadavku, aby všechny Windows Phone měli tlačítko fotoaparátu a ostatních fyzických tlačítek umístěných na zadní straně zařízení. Windows Phone 8.1 také přidává Cortanu, hlasový asistent podobný Siri u iPhone. Cortana nahrazuje předchozí vyhledávací funkce od Bing. (5)

4.3.5 Windows Mobile 10

Windows Mobile 10 byl představen 21. ledna 2015 jako mobilní operační systém pro chytré telefony a tablety se systémem postaveným na ARM architektuře. Jeho hlavním cílem je sjednocení s operačním systémem Windows 10, jeho protějšek PC, tedy hlavně s jeho softwarem a službami. Microsoft rovněž vyvíjí middleware známý jako Windows Bridge, který umožní iOS objective C a Android C ++ nebo Java software, běžet na Windows mobile 10 s minimálními změnami v kódu. (14)

Obrázek 7 - Windows Mobile 10

Zdroj:

https://www.windowscentral.com/sites/wpcentral.com/files/styles/larger/public/field/image/2015/06/w10mobile-10149-leak1.jpg?itok=R6dErpos

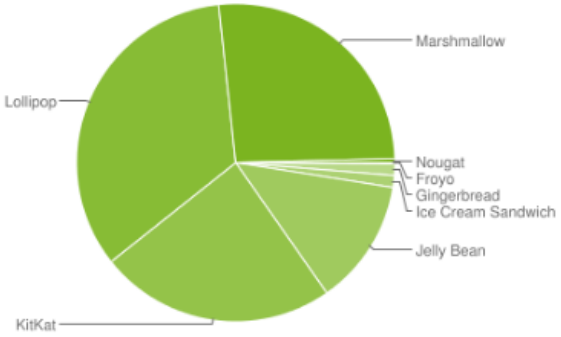
4.4 Podíl jednotlivých verzí operačních systémů na trhu

4.4.1 Android

Android je nejrozšířenější operační systém pro mobilní telefony. Není však povinná jeho aktualizace, a proto stále najdeme téměř všechny druhy operačního systému Android. Avšak Google play je podporován jen od verze 2.2 Froyo, jehož podíl byl v posledním roce pouhé 0,1 %. Největší zastoupení mají verze KitKat, 24 % Lollipop 34 % a Marshmallow 26 %. Zajímavé je, že poslední verze Androidu Nougat, má pouze 0,4 % mobilních telefonů. Je to způsobeno hlavně tím, že se výrobci mobilních zařízení bojí na svých telefonech spustit zatím málo otestovanou verzi nového operačního systému. Můžeme ale počítat s tím, že zastoupení operačního systému vzroste, jako tomu bylo i u předchozích verzí. (15)

Version	Codename	API	Distribution
2.2	Froyo	8	0.1%
2.3.3 - 2.3.7	Gingerbread	10	1.2%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	1.2%
4.1.x	Jelly Bean	16	4.5%
4.2.x		17	6.4%
4.3		18	1.9%
4.4	KitKat	19	24.0%
5.0	Lollipop	21	10.8%
5.1		22	23.2%
6.0	Marshmallow	23	26.3%
7.0	Nougat	24	0.4%

Data collected during a 7-day period ending on December 5, 2016.
Any versions with less than 0.1% distribution are not shown.

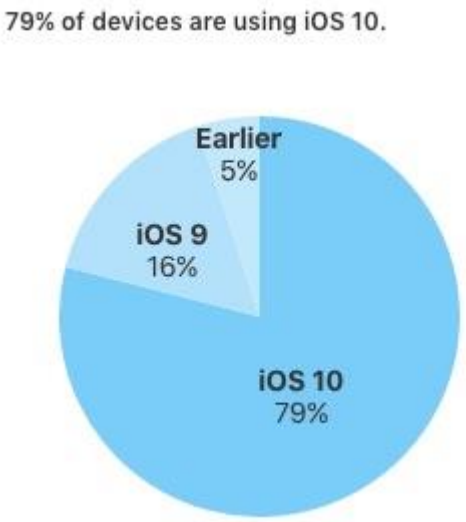


Obrázek 8 - Podíl verzí Android

Zdroj: https://www.svetandroida.cz/android-na-zacatku-prosince-201612

4.4.2 Apple iOS

iOS je druhý nejrozšířenější operační systém pro mobilní telefony. Protože je vyžadována aktualizace systému, tak jsou podporované verze operačních systémů velmi jednotné. Nejrozšířenější verzí je zároveň ta nejnovější, a to je iOS 10, kterou má nainstalovanou 80 % zařízení. Jenom 5 % má starší verzi než je iOS 9. (8)

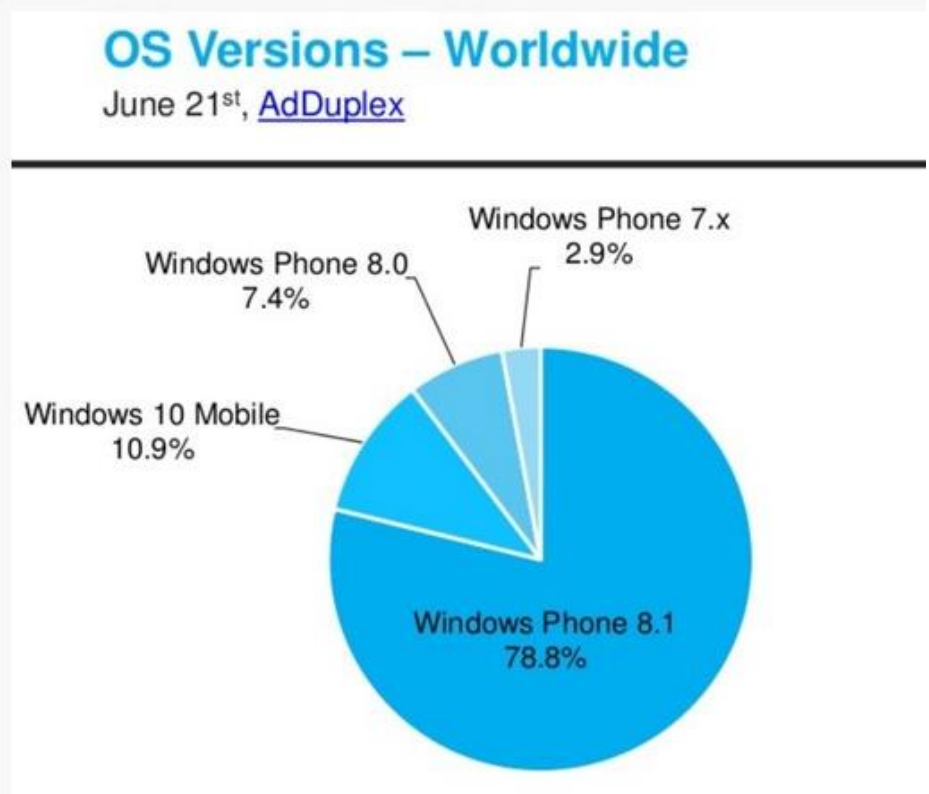


Obrázek 9 - Podíl verzí iOS

Zdroj: https://www.letemsvetemapplem.eu/wp-content/uploads/2017/02/ios-10-adoption-february-20.jpg

4.4.3 Windows Phone/Mobile

Windows Phone/Mobile je až třetí nejrozšířenější mobilní operační systém. Nemá povinné aktualizace operačního systému, a proto má podobně jako Android více aktivních verzí na trhu. Opět stejně jako u Android je nejpopulárnější předposlední verze Window Phone 8.1, která je nainstalovaná na 79 % zařízení. Nejnovější verze Windows Mobile 10 má 11 % zastoupení na trhu. Nejstarší verze s pouhými 3% zastoupení je Windows Phone 7. (16)



Obrázek 10 - Podíl verzí Windows

Zdroj: https://mobilizujeme.cz/clanky/windows-mobile-10-je-na-11-windows-mobilnich-zarizenich

4.5 Vývoj pro jednotlivé operační systémy

4.5.1 Android vývoj

K programování aplikací pro Android se nejčastěji používá jazyk Java. Také je většina nástrojů na programování zcela zdarma. Je možné na Android instalovat aplikace i z internetových stránek třetích stran a jiných online obchodů nejen z oficiálního Android marketu. To je velký rozdíl oproti IOS a Windows, kteří podporují pouze své oficiální aplikace. K tomu, aby mohl vývojář publikovat svoje aplikace je nutné uhradit jednorázový poplatek v hodnotě 25 dolarů. Není zde žádný schvalovací proces jako u jiných platforem. Koncem roku ale Google zavedl bezpečností nástroj, který se jmenuje Bouncer a ten zpětně aplikace kontroluje. Je to ochrana hlavně kvůli škodlivému softwaru, spywaru a také malwaru a různých virů. (17)

Velkou výhodou programování aplikací na Android je největší komunita vývojářů, která zajišťuje poměrně vysokou kvalitu a rychlí pokrok aplikací. Také je možné aplikace publikovat ihned bez čekání a pouze za jednorázový poplatek. Výhodou, ale i zároveň nevýhodou je velké množství zařízení na trhu. Výhoda je to kvůli velkému zájmu o nejrůznější aplikace a jejich masivní stahování. Nevýhoda je ale, že je vysoký počet operačních systémů Android, a ještě vyšší počet zařízení, které se liší jak velikostí, tak kvalitou hardwaru. (17)

4.5.2 Apple iOS vývoj

Nejpoužívanější jazyk na programování aplikací na IOS je jazyk objective-C. Vývoj aplikací pro IOS je možný pouze v aplikaci XCode, což je aplikace přímo od Apple, kterou je možné používat pouze na operačních systémech Mac OS X, tedy nejde použít jak Windows tak Linux. Aplikaci je poté možné naistalovat jedině pokud je podepsána certifikátem, který je vydán přímo od Apple. Certifikát je možné získat pouze pokud si vývojář zařídí účet u Apple, který musí být schválen a poté za něj platit 99 dolarů na rok. Bez tohoto účtu není možné propagovat své aplikace na AppStore. (18)

Výhodou programování na IOS je, že neexistuje mnoho verzí samotného operačního systému a ani mnoho různých typů zařízení. Je tedy jen málo odlišných velikostí displeje a odlišných operačních systémů. Také díky programovacímu jazyku C, je snažší portace jednotlivých aplikací. Také aplikace prodané na IOS mají největší celkové příjmy v porovnání s ostatními firmami. (18)

4.5.3 Windows Phone/Mobile vývoj

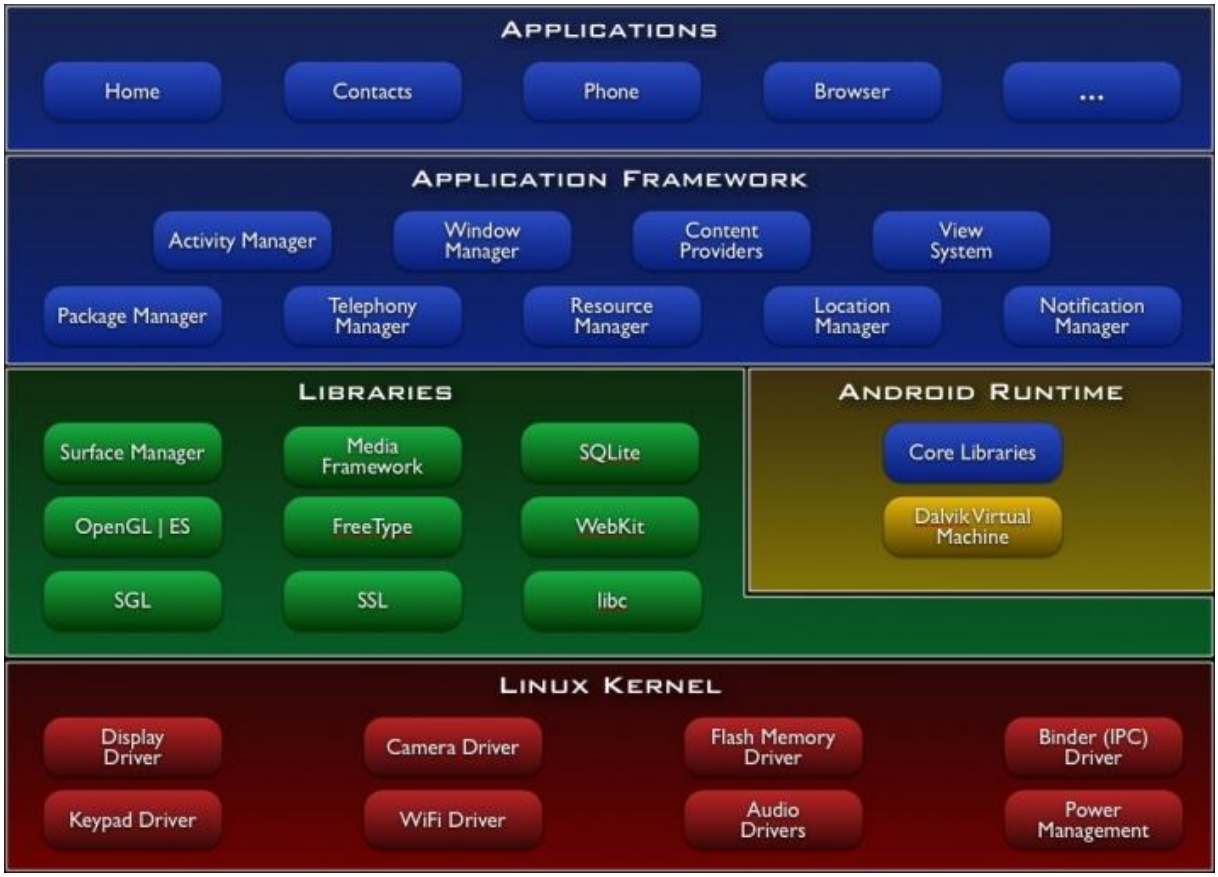
Jazyk používaný na programování aplikací je C#. Často používaným nástrojem pro programování je Visual studio, který je dostupný v základní verzi zdarma. Jedinou podmínkou obdobně jako u Apple je, že musí být použit operační systém Windows. Pro možnost publikovat aplikace na Marketplace, je nutné mít založený účet u Microsoftu jako vývojář a zaplatit poplatek 19 dolarů ročně. Pokud se jedná o společnost, tak poplatek činí 99 dolarů. Microsoft ale nabízí registraci zdarma a to studentům, kteří se účastní programu DreamSpark. Vytvořenou aplikaci je třeba nechat schválit Microsoftem a on poté aplikaci přidělí platný certifikát, který

je potřebný pro publikování aplikace. Aplikace tedy musí splňovat všechny podmínky od Microsoftu. Je možné tyto požadavky zkontrolovat pomocí nástroje Marketplace Test Kit. (19)

Východy programování na Windows tedy jsou, že studenti mohou své aplikace publikovat zdarma. Je také poměrně snadné v porovnání s ostatními operačními systémy, vytvářet výkonné 2D a 3D hry pomocí XNA. Další velkou výhodou je, že díky .NET technologii je možné aplikaci použít také např. na konzoli Xbox 360. Hlavní velká nevýhoda programování na Windows je, že má poměrně malé celkové zastoupení na trhu v porovnání s Androidem nebo IOS. (19)

4.6 Architektura operačního systému Android

Operační systém Android je postaven na linuxovém jádře. Většinou zřízení obsahuje ARM procesor. Aplikace na Androidu nekomunikují přímo s jádrem, ale pomocí Android API. Samotný běh aplikací pak zařizuje Dalvik Virtual Machine, který překládá kód a spouští aplikace. Operační systém Android se skládá z pěti vrstev, které však nemusí být od sebe oddělené. (3)



Obrázek 11 - Architektura Android

Zdroj: http://elinux.org/images/c/c2/Android-system-architecture.jpg

4.6.1 Linuxové jádro (Linux Kernel)

Nejdůležitější je samotné jádro operačního systému, které zajišťuje komunikaci mezi softwarem a hardwarem. Další funkce jádra jsou správa sítí, procesů a paměti. Linuxové jádro se používá hlavně kvůli jeho vysoké přenositelnosti. (3)

4.6.2 Knihovny

Knihovny jsou důležité hlavně pro programátory. Jedná se o druhou vrstvu, která je napsána v jazyce C/C++.

SSL – zajišťuje šifrování a ochranu dat

Open GL – pracuje s 3D grafikou

SQLite – práce s daty

FreeType – zajišťuje vykreslování písma

WebKid – Vykreslovací jádro pro webové prohlížeče

Surface manager – Zobrazuje aplikace

Media Framework - práce s mediálními soubory (3)

4.6.3 Android runtime

Obsahuje virtuální stroj Dalvik Virtual Machine. Překládá kód aplikací do nativního kódu. Využívá základní vlastnosti linuxového jádra. Pracuje tedy s operační pamětí a s vlákny. Také se v něm uchovávají základní knihovny Java. Zajišťuje aby každá aplikace byla spuštěna na vlastním procesu (3)

4.6.4 Application Framework (Aplikační rámec)

Další důležitá vrstva hlavně pro vývojáře. Jejím obsahem jsou Java knihovny. Zajišťuje velké množství služeb použitých přímo v aplikacích. Jedná se hlavně o zpřístupnění dat

v ostatních aplikacích a o API, které pracuje s notifikacemi. Zajišťuje také práci více aplikací naráz a přístup k grafickým prvkům. Je to poslední vrstva před aplikací. (3)

4.6.5 Aplikace

Aplikací na Android je ohromné množství. Dělí se na systémové aplikace, které jsou do zařízení naistalovány předem a aplikace dodatečně nainstalované uživatelem, které jsou staženy z Google play. (3)

4.6.6 Paměť ROM

Nejedná se sice o vrstvu operačního systému, ale velmi úzce s ní souvisí. Jedná se o část paměti, ve které jsou uloženy soubory operačního systému Android. Údaje o operátorovi a základní ovladače hardwaru jsou v Radio Rom, která je součástí ROM paměti. (3)

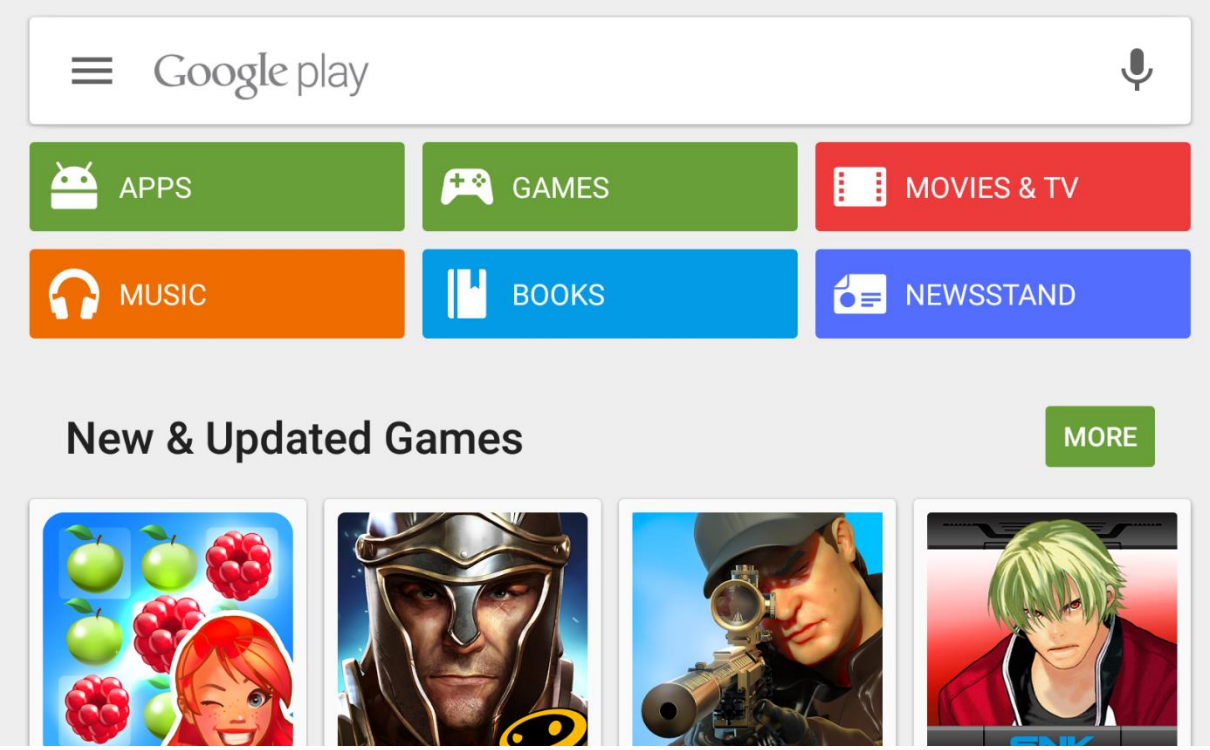
4.6.7 Zavaděč (bootholder)

Jedná se o samostatný program sloužící k nahrání jádra operačního systému do paměti ROM. Využívá se také pro nahrání nové ROM, nebo při jejím přemazání. Je možné se do zavaděče dostat pomocí různých kombinací kláves, které se liší u každého zařízení. (3)

4.7 Google play

Google play je aplikace, která slouží ke stahování aplikací, hudby, filmů a knih na mobilní zařízení s operačním systémem Android. Je potřeba se do aplikace přihlásit přes Google účet. Všechny nabízené produkty jsou zatříděné do jednotlivých kategorií a je možné používat i klasické vyhledávání skrz celý Google play. V aplikaci se rozdělují produkty na ty co jsou zdarma a na ty, za které uživatel musí zaplatit, aby si je mohl stáhnout. Je ale možné do určité doby zakoupení daného produktu nechat si vrátit peníze a produkt vymazat ze zařízení. Je tím ošetřena nespokojenost zákazníků s různými aplikacemi, které jim buď nefungují vůbec anebo nefungují podle jejich představ. To, že aplikace je zdarma ke stažení ještě nutně neznamená, že vývojář aplikace nebude mít žádný zisk ze stažení jeho aplikace. Je zde zřízen systém reklam přímo v aplikacích za jejíž shlédnutí vývojář dostane předem určenou odměnu. Je vidět, že se zájem o programování aplikací zvyšuje, protože v roce 2009 bylo na Google play k dispozici pouze 2300 aplikací a teď v roce 2017 jich je k dispozici 2 757 181. (20)

S prvním mobilním telefonem, který používal operační systém Android byl uveden místo Google play, původní Google market. Bylo umožněno si z něho také stahovat aplikace, ale nebyl zrovna uživatelsky přívětivý. V tu dobu také nebyla možnost instalovat aplikace přes internetový prohlížeč. Dalším nedostatkem bylo, že na všechny druhy produktů byla vlastní aplikace, tedy na hudbu byl Google Music a na e-knihy byl Google eBookstore. To zapříčiňovalo značnou nepřehlednost a matení zákazníků, proto se Google rozhodl vytvořit aplikaci Google play, která všechny spojí dohromady. (20)



Obrázek 12 - Google Play

Zdroj: http://img.talkandroid.com/uploads/2015/03/Google-Play-Store-e1426096224255.png

4.8 Nástroje pro tvorbu aplikací

Vývoj aplikací na Android je stále jednodušší a snadnější kvůli velké rozmanitosti nástrojů, které umožňují vývojářům dělat téměř cokoliv. Google má samozřejmě svou vlastní aplikaci známou jako Android Studio, která zahrnuje celou sadu užitečných nástrojů, stejně jako celé webové stránky s množstvím informací o vývoji Android. Třetí strany se také snaží zapojit s podobnými nástroji jako jsou GameMaker Genymotion a yoyo Games. (10)

4.8.1 Android Device Monitor

Jedná se o velmi zajímavou funkci, která je součástí Android Studio. Android Device Monitor umožňuje vývojářům sledovat jejich zařízení, když je zapojeno do počítače nebo sledovat virtuální zařízení. Zobrazuje kolik procesů běží a na jakém vláknu. Také sleduje síťové statistiky a umožňuje podívat se do Logcat. (10)

4.8.2 AVD Manažer

Je kvalitní nástroj podobný Android Studiu, který umožňuje vývojářům testovat jejich aplikaci na virtuálním zařízení. Je možné nasimulovat jakýkoli aspekt, včetně instrukční sady, velikosti RAM, velikosti obrazovky a jejího rozlišení. (10)

4.8.3 Unity3D

Je program, který je zaměřený na 3D programování. Slavné hry jako Crossy Road a Monument Valley jsou vytvořeny právě v Unity3D. Unity3D nabízí mnoho výukových programů a ukázky, které pomůžou začít s programováním. Unity3D využívá programovací jazyk C#. Je nutné použít externí software k vytvoření 3D modelů, které se pak nahrají do Unity3D. Unity3D není určen pro začátečníky, protože obecně 3D programování je složitá věc, která potřebuje hodně času a zkušeností. (10)

4.8.4 Corona SDK

Corona je známá ve vývojářské komunitě například kvůli hrám "Pop the lock" a "Fun Run 2". Přináší jiný způsob vývoje aplikací na Android ve srovnání s Android studiem. Corona používá programovací jazyk Lua, který není tak známý jako ostatní programovací jazyky. Lua je rychlý skriptovací jazyk, který je jednodušší než Java nebo C ++ a je velmi podobný jazyku GameMaker, který je popsán v další části. (10)

4.8.5 GameMaker: Studio

Jeden z nejjednodušších, nicméně ne horší nástroj na programování aplikací. GameMaker: Studio je určen pro 2D hry a také v této kategorii vyniká. Uživatelské rozhraní je velmi jednoduché s možností drag and drop. GameMaker používá svůj jazyk

GameMaker language. GML je mnohem snadnější na naučení než Java nebo C ++, neboť nemá metody (ani funkce v C++) a syntaxe obvykle mnohem kratší a jednodušší než u Javy a ostatních. Jediná nevýhoda je cena, která je $150 dolarů za GameMaker: Studio Professional plus $299 dolarů za Android export modul. (10)

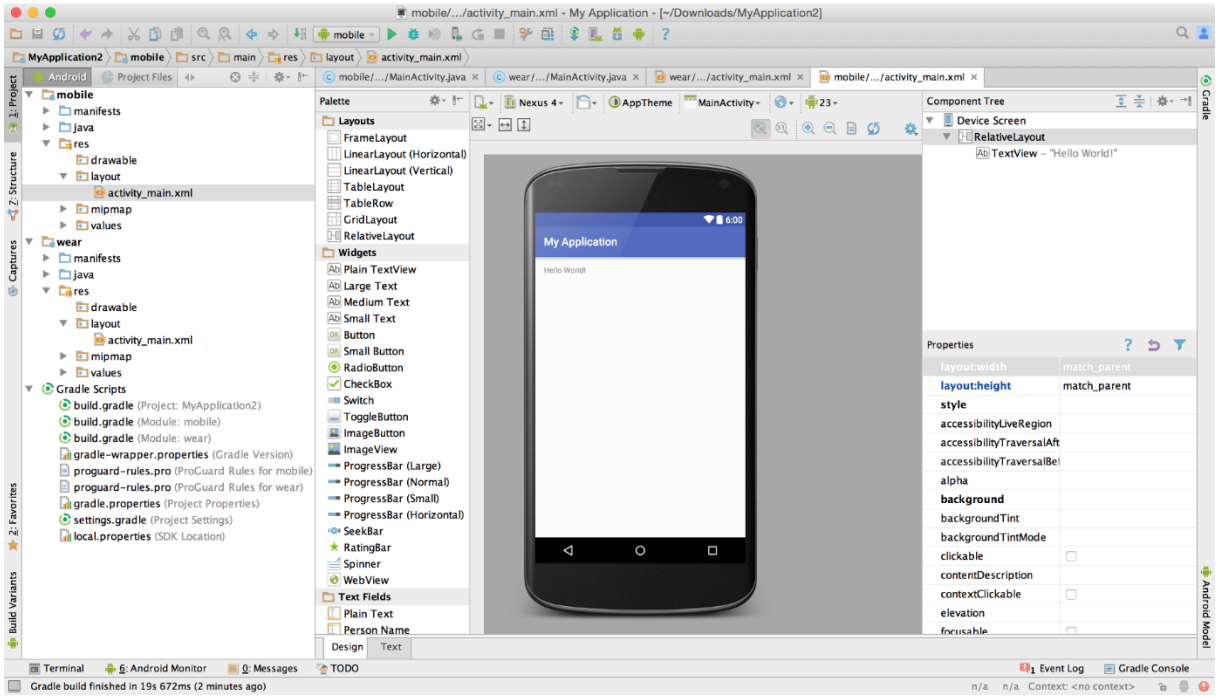
4.8.6 Eclipse

Jedná se o IDE, které slouží k vývoji aplikací pomocí programovacích jazyků. Používá například jazyky Java, C/C++, Python, Ruby. Dá se říci, že se jedná o předchůdce Android studia, protože před jeho vznikem byl Eclipse jedním z nejvhodnějších nástrojů pro programování na Android. Na programování aplikace na Android je potřeba si do Eclipse naistalovat Android SDK (software development kit). (12)

4.8.7 Android Studio

Vzhledem k tomu, že Google vytvořil Android, je příznačné, že se také zaměřil na vývoj nástroje IDE pro tvorbu aplikací na něj, aby zvýšil co nejvíce úroveň možných produktů. Programování v prostředí Android Studia se chová stejně jako jakýkoli jiný Java IDE, pokud jde o kontrolu chyb a souborové hierarchie. Kromě schopností programovat aplikace, Android Studio také nabízí možnost je testovat pomocí Android Správce virtuálních zařízení a zapojením fyzických zařízení přímo do Android Studia. Při tvorbě aplikací je možné si nechat generovat graf využití procesoru a paměti, takže je lze optimalizovat, aby využívaly co nejméně prostředků mobilního zařízení. Android studio nám také dává možnost logování, kdy si můžeme do kódu napsat log, který zobrazí jeho obsah v momentě kdy se k němu aplikace dostane. Tím se dají ověřovat všechny možné cesty aplikace a zkontrolovat například hodnotu proměnných v danou chvíli. (11)

Android studio má k dispozici editor překladů, který nám ukazuje, jak se jednotlivé texty na obrazovce budou překládat do jiných jazyků. Pro začínající programátory Google zveřejnil ukázky jednotlivých kódu na GitHub. Android studio je k dispozici zdarma. (11)



Obrázek 13 - Android studio

Zdroj: https://developer.android.com/studio/images/new-project-wizard-final-results_2-1_2x.png

4.9 Základy programování pro Android

4.9.1 Základní prvky Android aplikace

Před samotným návrhem aplikace je nutné znát několik zásad a informací potřebných k samotné realizaci návrhu. U aplikací Android se nevyskytuje žádná funkce typy main (), která by představovala hlavní funkci aplikace, jako tomu bývá u desktopových aplikací.

Základní prvky Android aplikace jsou Aktivity, které reprezentují obrazovku a Služby, které provádějí akci na pozadí. Dále existují ještě dvě komponenty a to · Content providers (Poskytovatelé obsahu) a Broadcast receivers. Všechny tyto komponenty jsou definovány v AndroidManifest.xml, což je soubor, který se nachází v kořenovém adresáři souboru. (1) (2) (21)

4.9.2 Aktivita

Aktivitu si lze představit jako stránku u webových aplikací, nebo dialogové okno u desktopových aplikací. Je to zpravidla jedna obrazovka, která zabírá celou plochu displeje. Je to tedy uživatelské rozhraní aplikace, které slouží k interakci s uživatelem. Aplikace se většinou skládá z více aktivit, mezi kterými uživatel může přepínat. Vytvoření aplikace probíhá tak, že se jako první vytvoří nový proces, který si alokuje paměť pro objekty GUI a pak se objekty rozloží do layoutu obrazovky, kde se vykreslí. K práci s aktivitami slouží metody, které zařizují jejich spuštění v určitý čas a za určitých podmínek. Tyto metody spolu se stavy aktivit, tvoří životní cyklus aktivit. (1) (2) (21)

Aktivita se může nacházet v pěti různých stavech, a to jako vytvořená, spuštěná, obnovená, pozastavená a zastavená. (1) (2) (21)

4.9.2.1 Příklad jednotlivých metod životního cyklu

OnCreate() – metoda, která je zavolána po spuštění aktivity a zajišťuje jaké GUI se má zobrazit a jak se má zobrazit na obrazovce.

OnPause() – v případě, že je aktivita překryta jinou aktivitou, ale není žádoucí původní aktivitu ukončit

OnStop() – slouží k zastavení aktivity. Aktivita již nadále není viditelná pro uživatele, ale může jí znovu zobrazit za použití metody onRestart.

Dále existují metody například OnStar(), OnDestroy(), OnResume().

Součástí aktivity mohou být Fragmenty. Reprezentují určitou část uživatelského rozhraní v Aktivitě. V Aktivitě může být i více fragmentů, což je využitelné hlavně u tabletů, které mají velký displej. Fragmenty mají svůj vlastní životní cyklus a jsou různě používány za běhu Aktivity. Jsou ale závislé na stavu aktivity, tedy fragment nemůže běžet pokud je aktivita pozastavená. (1) (2) (21)

4.9.3 Služby

Jak bylo zmíněno výše, služby provádějí určitou operaci na pozadí, tedy nepotřebují žádné uživatelské rozhraní. Z pravidla se používají pro přístup ke vzdáleným zdrojům jako je

například připojení serveru, či stahování různých souborů z internetu. Je také možné použít služby i pokud je aplikace již ukončená, například přehrávat hudbu na pozadí, zatímco běží už jiná aktivita. Služby jsou tedy násilně ukončeny systémem pouze ve výjimečných případech. Pokud ale přece jen dojde k takovémuto ukončení, tak ji systém po získání zdrojů obnoví. Služba tedy může být korektně ukončena sama sebou, jinou komponentou, nebo systémem. (1) (2) (21)

4.9.4 Intent (záměr)

Jedná se o asynchronní zprávu, jejíž prostřednictvím lze aktivovat komponenty a zároveň jim poskytovat potřebné informace. Záměr se používá v případech, kdy aplikace potřebuje využít ve svém běhu nějakou operaci, která je součástí jiné aktivity. Je také možné pomocí něho spustit aktivitu, která je součástí jiné aplikace. Záměr tedy funguje tak, že požádá systém o splnění potřebné aktivity a systém sám rozhodne, které aktivity nabídne uživateli. (1) (2) (21)

4.9.5 Content provider

Rozhraní, které zajišťuje sdílení dat. Je možné si vybrat, kam se budou data z aplikace ukládat a díky Content provider k těmto datům můžou také přistupovat ostatní aplikace, nebo je dokonce měnit. Je také možné ho využít pro čtení a zápis dat pouze pro jednu aplikaci. Data můžeme ukládat do souborového systému, do SQLite databáze nebo na web. (1) (2) (21)

4.9.6 Broadcast receiver

Broadcast receiver se stará o přístup k systémovým hlášením. Například je použit pro příchozí SMS nebo k upozornění na slabou baterii. Je možné vytvořit vlastní hlášení a je tak tedy možné předat informace ostatním aplikacím. Samotný Broadcast receiver nemá uživatelské rozhraní, ale je možné pomocí něho vytvořit upozornění pro systémovou lištu. (1) (2) (21)

5 Praktická část

Součástí bakalářské práce je i vytvoření vlastní aplikace. Vlastní aplikace je hra s názvem Had neboli Snake. Hlavním cílem této hry je demonstrace vytvoření jednoduché hry na Android. Aplikace má ukázat, jak se dá vytvořit jednoduchá hra na operační systém Android. Hra spočívá v tom, že uživatel ovládá vykresleného hada, který sbírá body po hrací ploše. Nesmí však u toho narazit do vyznačeného konce hrací plochy, ani sám do sebe. V tu chvíli hra skončí a uživateli se ukáže počet získaných bodů.

5.1 Třídy použité v aplikaci

V aplikaci jsou použité následující třídy.

5.1.1 Coordinate

Je třída, ve které jsou určené souřadnice celé hrací plochy. Dále je použitá funkce pro porovnávání těchto souřadnic, aby hra fungovala podle již zmíněných pravidel.

```
public class Coordinate {

    private int x;

    private int y;

    public Coordinate(int x, int y) {

        this.x = x;

        this.y = y;

    }

    public int getX() {

        return x;

    }

public void setX(int x) {

        this.x = x;
```

```
    }
    public int getY() {
        return y;
    }
    public void setY(int y) {
        this.y = y;
    }
    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return
false;
        Coordinate that = (Coordinate) o;
        if (getX() != that.getX()) return false;
        return getY() == that.getY();
    }
}
```

5.1.2 EndScreen

Jedná se o obrazovku, která je zobrazena po prohrání samotné hry, tedy po nárazu hada buď do stěny nebo do sebe. Na obrazovce jsou zobrazeny tři tlačítka pro další interakci se hrou a počet získaných bodů.

```
RESTART = (Button) findViewById(R.id.restart);
MENU = (Button) findViewById(R.id.menu);
```

```
EXIT = (Button) findViewById(R.id.exit);

points = (TextView) findViewById(R.id.points);
int Points = GameEngine.Points;
```

Tlačítko restart obnoví samotnou hru. Tlačítko Menu, přesměruje uživatele na počáteční obrazovku. Tlačítko Exit vypne hru.

```
RESTART.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View view) {
        startActivity(new Intent(getApplicationContext(),
MainActivity.class));
    }
});
MENU.setOnClickListener(new View.OnClickListener(){
    @Override
    public void onClick(View view) {
        startActivity(new Intent(getApplicationContext(),
StartScreen.class));
    }
});
EXIT.setOnClickListener(new View.OnClickListener() {

 @Override
    public void onClick(View view) {
        startActivity(new Intent(getApplicationContext(),
StartScreen.class));
```

```
            Intent intent = new Intent(Intent.ACTION_MAIN);
            intent.addCategory(Intent.CATEGORY_HOME);
            intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
            startActivity(intent);
        }
});
```

5.1.3 GameEngine

Tato třída slouží k oživení celé hry, zajišťuje tedy pohyb hada po hrací ploše. Je zde definována velikost hrací plochy, která je statická.

```
public int GameWidth = 26;
public int GameHeight = 36;
public static int Points = 0;
```

Dále jsou zde definovaná dvojrozměrná pole, pro možnost určení polohy na hrací ploše. Jedná se o pole hada, sbíraných bodů a stěn.

```
private List<Coordinate> walls = new ArrayList<>();
private List<Coordinate> snake = new ArrayList<>();
private List<Coordinate> apple = new ArrayList<>();
```

Je zde také určený počáteční směr a stav hry.

```
private Direction currentDirection = Direction.East;
```

```
private GameState currentGameState = GameState.Running;
```

Dále je tu funkce, která zajišťuje spuštění hry do jejího počátečního stavu a funkce, která vypočítává, jestli se změnil směr pohybu hada.

```
public void initGame(){
    AddSnake();
    AddWalls();
    AddApple();
}
public void UpdateDirection(Direction newDirection){
    if(Math.abs(newDirection.ordinal() - currentDirection.ordinal()) % 2 == 1){
        currentDirection = newDirection;
    }
}
```

Teď přichází na řadu jedna z nejdůležitějších funkcí v celém kódu a to funkce, která spravuje aktivní část aplikace. V této funkci se určuje směr jakým se had pohybuje a vyhodnocují se jednotlivé situace, ve kterých hra končí. Tedy, pokud had narazí do sebe nebo do stěny. Také je zde kontrola sbírání jednotlivých bodů, jejich znovuobjevení na hrací ploše a samotné přidání kousku těla k hadovi.

```
public void Update(){
    int AppleCounter = 1;
    switch (currentDirection) {
```

```
        case North:
            UpdateSnake(0,-1);
            break;
        case East:
            UpdateSnake(1,0);
            break;
        case South:
            UpdateSnake(0,1);
            break;
        case West:
            UpdateSnake(-1,0);
            break;
    }
    for (Coordinate w : walls){
        if(snake.get(0).equals(w)){
            currentGameState = GameState.Lost;
        }
    }
    Coordinate snakeHead = snake.get(0);
    for (int i = 1; i < snake.size(); i++) {
        Coordinate snaketail = snake.get(i);
        if (snakeHead.equals(snaketail)) {
            currentGameState = GameState.Lost;
            return;
```

```
        }
    }
for (Coordinate a : apple){
        if(snake.get(0).equals(a)){
            snake.add(new Coordinate(a.getX(),a.getY()));
            apple.clear();
            AppleCounter = 0;
            Points = Points + 10;
        }
    }
    if(AppleCounter < 1){
      AddApple();}}
```

V této funkci je jednoznačný identifikátor všech položek v poli. Je zde pět možností o co se může jednat a to: Hlava hada, tělo hada, stěna, body a prázdné pole.

```
public TileType[][] getMap(){
    TileType[][] map = new TileType[GameWidth][GameHeight];
    for(int x = 0; x < GameWidth; x++){
        for(int y = 0; y < GameHeight; y++){
            map[x][y] = TileType.Nothing;
        }
    }

    for( Coordinate s : snake) {
```

```
            map[s.getX()][s.getY()] = TileType.SnakeTail;
        }
        map[snake.get(0).getX()][snake.get(0).getY()] = SnakeHead;

        for(Coordinate wall: walls){
            map[wall.getX()][wall.getY()] = TileType.Wall;
        }
        for(Coordinate a: apple){
            map[a.getX()][a.getY()] = TileType.Apple;
        }
        return map;
    }
```

Dále jsou tu funkce pro pohyb hada po mapě, přesněji jeho pohyb ve dvojrozměrném poli, přidání kousku těla po úspěšném sebrání bodu a přidání samotných bodů poté co jsou sebrány hadem. Jejich objevení je náhodné. A jako poslední co je potřeba je přidání stěn, tedy vymezení hrací plochy.

```
private void UpdateSnake(int x, int y) {
    for (int i = snake.size() - 1; i > 0; i--) {
        snake.get(i).setX(snake.get(i - 1).getX());
        snake.get(i).setY(snake.get(i - 1).getY());
    }
    snake.get(0).setX(snake.get(0).getX() + x);
    snake.get(0).setY(snake.get(0).getY() + y);
```

```
}
private void AddSnake() {
    snake.clear();
    snake.add(new Coordinate(7,7));
}
private void AddApple(){
    int AppleX,AppleY;
    AppleX = (int) (Math.random() * (GameWidth - 2) + 1);
    AppleY = (int) (Math.random() * (GameHeight - 2) + 1);
    Coordinate appleCo = new Coordinate(AppleX,AppleY);
    for (int i = 0; i < snake.size(); i++) {
        Coordinate snakeCo = snake.get(i);
     if (appleCo.equals(snakeCo)) {
            AppleX = (int) (Math.random() * (GameWidth - 2) +
1);
            AppleY = (int) (Math.random() * (GameHeight - 2) +
1);
        }
    }

    apple.add(new Coordinate(AppleX,AppleY));
    return;
}
private void AddWalls() {
    Points = 0;
```

```
        for (int x = 0; x < GameWidth; x++){
            walls.add(new Coordinate(x,0));
            walls.add(new Coordinate(x,GameHeight-1));
        }
        for (int y = 1; y < GameHeight; y++){
            walls.add(new Coordinate(0,y));
            walls.add(new Coordinate(GameWidth-1,y));
        }
    }
```

5.1.4 MainActivity

Je třída ve které se odehrává samotná hra. Popisuje základní stav hrací obrazovky. Jsou zde uvedeny funkce jednotlivých tlačítek na obrazovce, které ovládají hru. Je možné hru ovládat dvěma způsoby a to buď pomocí již zmíněných tlačítek nebo posunem prstu po obrazovce. Druhá možnost je příjemnější u zařízení s velkým displejem Je zde také určena rychlost hry. Je zajištěno, aby hra byla přes celou obrazovku a mohla být zobrazena jenom vertikálně.

```
private final long updateDelay = 150;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

setRequestedOrientation(ActivityInfo.SCREEN_ORIENTATION_PORTRAIT);
    setContentView(R.layout.activity_main);
    getWindow().getDecorView().setSystemUiVisibility(
            View.SYSTEM_UI_FLAG_LAYOUT_STABLE
```

```
                        |
View.SYSTEM_UI_FLAG_LAYOUT_HIDE_NAVIGATION
                        | View.SYSTEM_UI_FLAG_LAYOUT_FULLSCREEN
                        | View.SYSTEM_UI_FLAG_HIDE_NAVIGATION
                        | View.SYSTEM_UI_FLAG_FULLSCREEN
                        | View.SYSTEM_UI_FLAG_IMMERSIVE_STICKY);
    gameEngine = new GameEngine();
    gameEngine.initGame();
    snakeView = (SnakeView) findViewById(R.id.snakeView);
    snakeView.setOnTouchListener(this);
    UP = (Button) findViewById(R.id.UP);
    RIGHT = (Button) findViewById(R.id.RIGHT);
    LEFT = (Button) findViewById(R.id.LEFT);
    DOWN = (Button) findViewById(R.id.DOWN);
    Typeface myFontButton =
Typeface.createFromAsset(getAssets(), "fonts/Mona Shark.otf");
    UP.setTypeface(myFontButton);
    RIGHT.setTypeface(myFontButton);
    LEFT.setTypeface(myFontButton);
    DOWN.setTypeface(myFontButton);
    UP.setOnClickListener(new View.OnClickListener() {

        @Override
        public void onClick(View view) {
            gameEngine.UpdateDirection(Direction.North);
```

```
        }
    });
    RIGHT.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            gameEngine.UpdateDirection(Direction.East);
        }
    });
    LEFT.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            gameEngine.UpdateDirection(Direction.West);
        }
    });
    DOWN.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            gameEngine.UpdateDirection(Direction.South);
        }
    });
    startUpdateHandler();
}
private void startUpdateHandler(){
    handler.postDelayed(new Runnable() {
```

```
        @Override
        public void run() {
            gameEngine.Update();
            if(gameEngine.getCurrentGameState() ==
GameState.Running){
                handler.postDelayed(this, updateDelay);
            }
            snakeView.setSnakeViewMap(gameEngine.getMap());
            snakeView.invalidate();
        }
    }, updateDelay);
}

@Override
public boolean onTouch(View view, MotionEvent motionEvent) {
    switch (motionEvent.getAction()){
        case MotionEvent.ACTION_DOWN:
            prevX = motionEvent.getX();
            prevY = motionEvent.getY();
            break;
        case MotionEvent.ACTION_UP:
            float newX = motionEvent.getX();
            float newY = motionEvent.getY();
            if (Math.abs(newX - prevX) > Math.abs(newY -
prevY)){
```

```
                if(newX > prevX){

gameEngine.UpdateDirection(Direction.East);

                }else{

gameEngine.UpdateDirection(Direction.West);

                }

            }else{

                if(newY > prevY){

gameEngine.UpdateDirection(Direction.South);

                }else{

gameEngine.UpdateDirection(Direction.North);

                }  }

        break;

    }

  return true;}
```

5.1.5 SnakeView

Tato třída zajišťuje samotné vykreslení hrací plochy. Je zde metoda, která zjistí šířku displeje, aby se hrací plocha mohla vykreslit správně pro každý displej.

```
public static float getScreenWidth() {

    return
Resources.getSystem().getDisplayMetrics().widthPixels;

}
```

Co se týče hrací plochy, tak je použita funkce onDraw() ve které se vykreslí všechny potřebné položky.

```
@Override
protected void onDraw(Canvas canvas){
    if( snakeViewMap != null){
        float tileSizeX = getScreenWidth() /
snakeViewMap.length;
     for(int x = 0; x < snakeViewMap.length; x++){
            for(int y = 0; y < snakeViewMap[x].length; y++){
                switch (snakeViewMap[x][y]) {
                    case Nothing:
                        mPaint.setColor(Color.rgb(199, 199,
199));
                        break
                    case Wall:
                        mPaint.setColor(Color.rgb(105, 105,
105));
                        break;
                    case SnakeHead:

                        mPaint.setColor(Color.rgb(249, 133,
0));
                        break;
                    case SnakeTail
                        mPaint.setColor(Color.BLACK);
                        break;
```

```
                    case Apple:
                        mPaint.setColor(Color.rgb(40, 130,
0));
                        break;
                }
                canvas.drawRect(x * tileSizeX, y * tileSizeX,
x * tileSizeX + tileSizeX, y * tileSizeX + tileSizeX, mPaint);
            }
        }
    }
}
```

5.2 Enumerace

Dále jsou součástí kódu tři enumerace, které v sobě mají stavy hry, které můžou nastat. Dále pak typ daného bodu v poli a vyjmenované směry na které se může had pohybovat.

```
public enum Direction {
    North,
    East,
    South,
    West
}
public enum GameState {
    Running,
    Lost
}
```

```
public enum TileType {
    Nothing,
    Wall,
    SnakeHead,
    SnakeTail,
    Apple
}
```

5.3 Layout aplikace

Aplikace používá celkem tři různé layouty, které definují vzhled jednotlivých obrazovek, na které se může uživatel v rámci aplikace dostat. Pro představu je uveden pouze jeden druh layoutu.

5.3.1 Startscreen

Layout, který určuje vzhled úvodní obrazovky aplikace.

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@color/Background">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:id="@+id/header"
        android:textSize="100dp"
        android:layout_centerHorizontal="true"
        android:layout_above="@+id/start"
```

```
        android:layout_marginBottom="25dp"
        android:textColor="@color/Text"/>

    <Button
        android:text="Exit"
        android:layout_below="@+id/start"
        android:layout_alignStart="@+id/start"
        android:layout_marginTop="25dp"
        android:id="@+id/exit"
        android:textColor="@color/Text"
        android:textSize="50sp"
        android:layout_width="270dp"
        android:layout_height="100dp"
        android:background="@drawable/buttonbig"
        android:shadowColor="#A8A8A8"
        android:shadowDx="0"
        android:shadowDy="0"
        android:shadowRadius="5"/>

    <Button
        android:text="Start"
        android:id="@+id/start"
        android:textColor="@color/Text"
        android:textSize="50sp"
        android:layout_width="270dp"
        android:layout_height="100dp"
        android:background="@drawable/buttonbig"
        android:shadowColor="#A8A8A8"
        android:shadowDx="0"
        android:shadowDy="0"
        android:shadowRadius="5"
        android:layout_centerVertical="true"
```

```
        android:layout_centerHorizontal="true" />
</RelativeLayout>
```

5.4 Xml soubory drawable

Grafickou stránku hry tvoří následující xml soubory. Pro představu je uveden pouze jeden druh souboru.

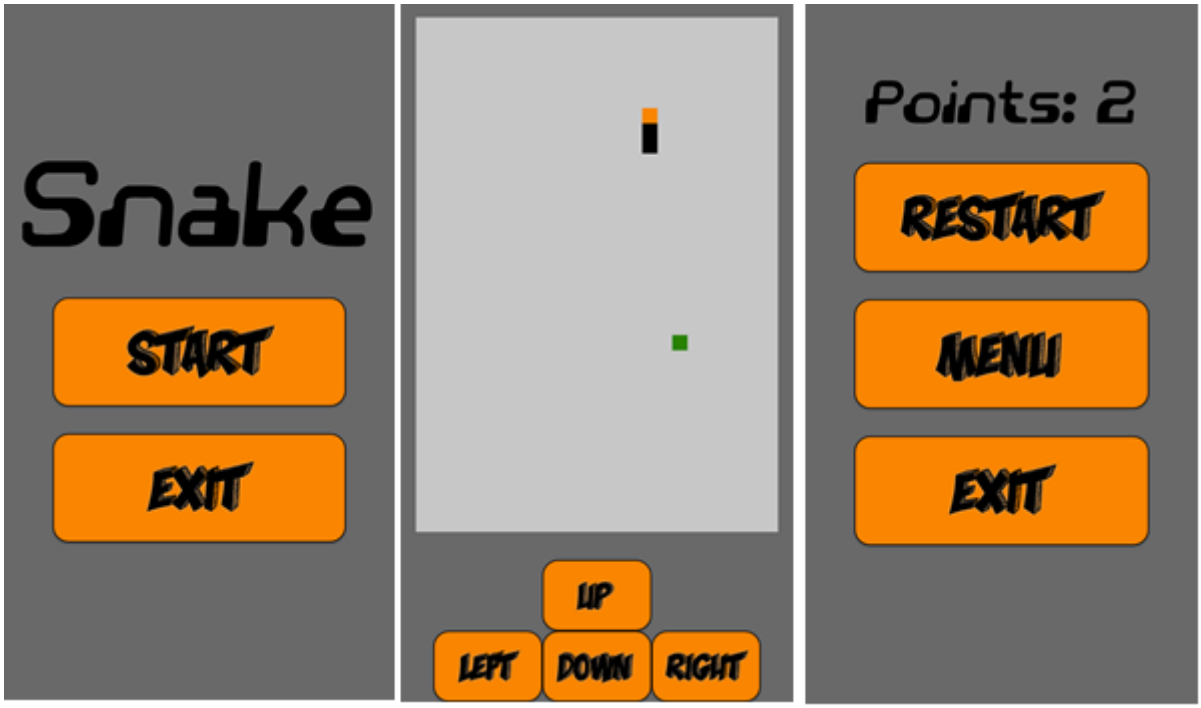
5.4.1 Buttonbig

Soubor buttonbig určuje vzhled velkých tlačítek v menu aplikace

```
<?xml version="1.0" encoding="utf-8"?>
<shape
xmlns:android="http://schemas.android.com/apk/res/android"
android:shape="rectangle" >
    <corners
        android:radius="14dp"
        />
    <solid
        android:color="#F98500"
        />
    <size
        android:width="270dp"
        android:height="100dp"
        />
    <stroke
        android:width="1dp"
        android:color="#000000"
        />
</shape>
```

5.5 Distribuce aplikace

K tomu, aby vývojář mohl umístit svoji aplikaci na Google play, je potřeba aktivovat vývojářský účet a zaplatit poplatek za registraci v hodnotě 25 dolarů. Je také potřeba vyplnit osobní údaje a souhlasit s podmínkami Google o distribuci aplikací. Jako první se musí nahrát APK soubor dané aplikace, která je určená k distribuci. Soubor může mít maximálně 50 MB. Poté je nutné vyplnit doplňující informace o aplikaci do záložky „Záznam v obchodu“, kde se dá přidat i překlad aplikace. Je nutné vyplnit u aplikace název, krátký popis, kompletní popis, snímky obrazovky z různě velkých zařízení, ikonu aplikace, kategorii, hodnocení obsahu, email a zásady ochrany soukromí. Další položka, která je potřeba vyplnit je „Cena a distribuce“. Je možné vybrat, jestli aplikace bude zdarma, nebo pro její stáhnutí bude muset zákazník zaplatit. Je možné také vybrat země ve kterých bude aplikace dostupná a jestli obsahuje nějaké funkce pro Android TV, nebo Android Wear. Jako poslední věc je možné vyplnit použité produkty v aplikaci, rozhraní API a služeb, nebo různých typů k optimalizaci aplikace. Po nahrání aplikace už jen stačí počkat z pravidla jeden až dva dny, než se zobrazí na Google play. (7)



Obrázek 14 - Vlastní aplikace

6 Závěr

V teoretické části byl podrobně charakterizován operační systém Android ze všech možných hledisek. Byla popsána jeho historie, vývojová prostředí a vývoj jednotlivých verzí. V práci jsou charakterizovány srovnatelným způsobem i konkurenční operační systémy. Tím byl splněn primární cíl bakalářské práce. Praktická část byla zaměřena na vývoj jednoduché hry pro operační system Android. Jako vývojové prostředí bylo vybráno Android studio, které se na tuto aplikaci hodilo nejlepé hlavně kvůli jednoduchosti instalaci a uživatelsky přívětivému prostředí. Jako další výhoda Android studia, byla jednoduchá distribuce hry na Google play, vzhledem k tomu, že tento nástroj je přímo od společnosti Google. Na konci praktické části byl popsán postup nahrávání vytvořené aplikace na službu Google Play. Tím byl tedy splněn i sekundární cíl, který měl co nejlépe charakterizovat problematiku vývoje aplikací pro operační systém Android.

7 Zdroje

1. DIMARZIO, J. *Programujeme hry pro Android 4*. 1. vyd. Brno: Computer Press, 2012. ISBN 978-80-251-3754-3.

2. LACKO, Ľuboslav. *Vývoj aplikací pro Android*. 1. vyd. Brno: Computer Press, 2015, 472 s. ISBN 978-80-251-4347-6.

3. Android - Architecture. *Tutorialspoint* [online]. [cit. 2015-11-08]. Dostupné z: http://www.tutorialspoint.com/android/android_architecture.htm

4. Android. *Android History* [online]. Dostupné z: https://www.android.com/history/

5. ALLISON, Michael *A History of Windows Phone* [online]. [cit. 2015-10-04]. Dostupné z: https://mspoweruser.com/a-history-of-windows-phone-the-road-to-threshold/

6. ORF, Darren. *A Brief History of iOS* [online]. [cit. 2016-07-06]. Dostupné z: http://gizmodo.com/a-brief-history-of-ios-1780790760

7. Android developer. *Launch Checklist* [online]. Dostupné z: https://developer.android.com/distribute/tools/launch-checklist.html

8. ZUNA, Dominik. *iOS 10 už se nachází na téměř 80 % všech aktivních Apple zařízeních* [online]. [cit. 2017-02-22]. Dostupné z: https://www.letemsvetemapplem.eu/2017/02/22/ios-10-uz-se-nachazi-na-temer-80-vsech-aktivnich-apple-zarizenich/

9. AppBrain. *Number of Android applications* [online]. [cit. 2017-04-06]. Dostupné z: https://www.appbrain.com/stats/number-of-android-apps

10. MULLIS, Alex. *Best Android developer tools* [online]. [cit. 2016-02-25]. Dostupné z: http://www.androidauthority.com/best-android-developer-tools-671650/

11. Android developer. *Meet Android Studio* [online]. Dostupné z: https://developer.android.com/studio/intro/index.html

12. Electronics Weekly. *What is…Android Eclipse?* [online]. [cit. 2012-01-12]. Dostupné z: http://www.electronicsweekly.com/blogs/eyes-on-android/what-is/android-eclipse-2012-01/

13. SWIDER, Matt. PECKHAM, James. *Android Nougat release date: when you'll get it and everything you need to know* [online]. [cit. 2017-03-06]. Dostupné z: http://www.techradar.com/news/phone-and-communications/mobile-phones/android-7-what-we-want-to-see-1311290

14. Microsoft. *What's new in Windows 10 Mobile* [online]. Dostupné z: https://support.microsoft.com/en-us/help/17284/windows-10-mobile-whats-new

15. KILIÁN, Karel. *Android na začátku prosince: KitKat konečně svržen, Nougat lehce stoupá* [online]. [cit. 2016-12-06]. Dostupné z: https://www.svetandroida.cz/android-na-zacatku-prosince-201612

16. KRATOCHVÍL, Lumír. *Windows Mobile 10 je na 11 % Windows mobilních zařízeních* [online]. [cit. 2016-06-24]. Dostupné z: https://mobilizujeme.cz/clanky/windows-mobile-10-je-na-11-windows-mobilnich-zarizenich

17. Android. *Get Started with Publishing* [online]. Dostupné z: https://developer.android.com/distribute/googleplay/start.html

18. Apple. *About App Distribution Workflows* [online]. Dostupné z: https://developer.apple.com/library/content/documentation/IDEs/Conceptual/AppDistributionGuide/Introduction/Introduction.html

19. Microsoft. *Publish Windows apps* [online]. Dostupné z: https://developer.microsoft.com/en-us/store/publish-apps

20. SERRE, Matthew. *A Brief History of the Google Play Store and Impressions on Newest Version 4.0.16* [online]. Dostupné z: http://droidlessons.com/a-brief-history-of-the-google-play-store-and-impressions-on-newest-version-4-0-16/

21. Android. *Getting Started* [online]. Dostupné z: https://developer.android.com/training/index.html