

Sem vložte zadání Vaší práce.

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
KATEDRA SOFTWAROVÉHO INŽENÝRSTVÍ



Diplomová práce

Hra pro OS Android s využitím rozšířené reality

Bc. Daniel Pína

Vedoucí práce: Ing. Martin Kačer, Ph.D

8. května 2017

Poděkování

Rád bych poděkoval panu Ing. Martinu Kačerovi, Ph.D za jeho ochodu vést toto téma, vstřícný přístup a odhodlání dotáhnout tuto diplomovou práci do konce, i přes nedostatek zbývajícího času.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 8. května 2017

.....

České vysoké učení technické v Praze

Fakulta informačních technologií

© 2017 Daniel Pína. Všechna práva vyhrazena.

Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí, je nezbytný souhlas autora.

Odkaz na tuto práci

Pína, Daniel. *Hra pro OS Android s využitím rozšířené reality*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Tato práce si klade za cíl provést analýzu technologie rozšířené reality, na- vrhnout a následně implementovat mobilní hru pro operační systém Android. Hra bude podporovat hru více hráčů a bude využívat rozšířené reality. Hru budou doplňovat fyzické herní karty reprezentující virtuální modely, které budou hráči ovládat a pomocí nich soupeřit mezi sebou.

Klíčová slova Android, hra, mobilní aplikace, multiplayer, rozšířená realita, Java, Bluetooth, 3D animace

Abstract

The goal of my work is to analyze a technology of augmented reality, propose and implement a mobile game for an Android operating system. The game will support a multiplayer game and it will use an augmented reality. The real game cards which represent virtual models, will supplement the game. The players will control the virtual models and compete against each other with the help of the models.

Keywords Android, game, mobile app, multiplayer, augmented reality, Java, Bluetooth, 3D animation

Obsah

Úvod	1
1 Analýza	3
1.1 Rozšířená realita	3
1.2 Platforma Android	8
1.3 Rešerše obdobných aplikací	10
1.4 Rešerše vývojářských nástrojů pro AR	12
2 Návrh	15
2.1 Popis hry	15
2.2 Požadavky	17
2.3 Případy užití	18
2.4 Architektura aplikace	19
2.5 Uživatelské rozhraní	20
2.6 Pohyb modelu	22
2.7 Objekty ve hře	23
2.8 Použité nástroje a technologie	25
2.9 Komunikace mezi zařízeními	27
3 Realizace	29
3.1 Struktura projektu	29
3.2 Práva aplikace	31
3.3 Rozšířená realita	31
3.4 Vykreslování rozšířené reality	32
3.5 Aktualizace modelu	32
3.6 Pohyb modelu	33
3.7 Kolize modelů	35
3.8 Komunikace mezi zařízeními	36
3.9 Server	38
3.10 Herní karty	40

4 Testování	43
4.1 Heuristická analýza	43
4.2 Testování použitelnosti	44
4.3 Výsledky testování	44
5 Rozšíření	47
5.1 Server	47
5.2 Aplikace	47
5.3 Design mobilní aplikace	47
5.4 3D modely	48
5.5 Komunikace zařízení	48
5.6 Hra jednoho hráče	48
5.7 Audio	48
Závěr	49
Literatura	51
A Seznam použitých zkratek	55
B Testovací scénář	57
C Obrázky ze hry	59
D Instalační příručka	61
E Obsah přiloženého CD	63

Seznam obrázků

1.1	Rozšířená realita ve sportovních přenosech [1]	6
1.2	Marker s histogramem [2]	8
1.3	Poměr OS v prodaných zařízeních [3]	10
1.4	Poměr verzí OS Android [4]	11
1.5	Rozpoznání markeru: Vuforia [5]	13
2.1	Diagram případů užití	19
2.2	Model-View-Controller vs. Model-View-Presenter	21
2.3	Návrh herní obrazovky	22
2.4	2 varianty reakce modelu na vstup joysticku	23
2.5	Kostra 3D modelu	24
2.6	Mezní pozice chůze člověka	25
3.1	Diagram: Vykreslování rozšířené reality	33
3.2	Diagram: Aktualizace modelu podle vstupu	34
3.3	Herní karta s rozpoznávanými body	41
C.1	Hra	59
C.2	Hra	60
C.3	Přihlášení a úprava profilu	60

Seznam tabulek

1.1 Poměr verzí OS Android	11
--------------------------------------	----

Úvod

V dnešní době jsou mobilní zařízení nejrozšířenější hardwarovou platformou. Kromě toho, že majitelé těchto zařízení instalují miliony aplikací denně, hry mezi nimi jsou převládajícím typem. Výsledkem toho je, že obrovský počet lidí interaguje s mobilními hrami každou sekundu. Navíc herní průmysl v současné době produkuje větší příjmy než filmový a hudební dohromady. Tím všim se herní průmysl stává velkou příležitostí pro vývojářské firmy, vymyslet nový koncept a na tomto trhu prorazit. Hry jsou vyvíjeny především pro zábavu hráče, ale vytvářejí se i hry beroucí si za úkol vychovávat a vzdělávat.

Smartphony jsou široce používány v našem každodenním životě ať pro práci nebo zábavu. Obvykle tyto zařízení mívají obrazovky s vysokým rozlišením, videokamery, webové prohlížeče, GPS a vysokorychlostní přenos dat s přístupem přes Wi-Fi a mobilní síť. Ohledně operačního systému, je Android uveden jako nejpopulárnější operační smartphone systém na světě s 85% trhu se smartphony v roce 2017. Každý den se také na Google Play nahraje přes 2000 nových aplikací což s kombinací toho, že lidé mohou používat smartphony téměř kdykoli a kdekoli, vytváří obrovský trh s mobilními aplikacemi.

Rozšířená realita je nová technologie, která míchá dohromady virtuální objekty a reálný svět. Technologie využívající rozšířenou realitu může být použita ve všech oblastech života, ale je velmi zajímavé, pokud se využívá pro herní účely. Lidé rychle přijali tuto technologii a začali využívat její výhody, které nabízejí úžasný zážitek nejen z hraní her, ale především z pozorování upraveného reálného světa. Nyní již existuje celá řada počítačových a mobilních her, které využívají možnosti rozšířené reality, a proto jsem se chtěl této technologii více věnovat.

Tato práce se zabývá návrhem a realizací hry pro jednoho nebo dva hráče ve světě rozšířené reality pro mobilní operační systém Android. Oba hráči používají svá zařízení jako ovladače pro ovládání svého modelu, který bude bojovat s modelem soupeře. Hra je výjimečná tím, že herní svět je zasazen do reálného prostředí a souboje tak mohou probíhat na pracovním stole nebo na chodníku před školou. Hra by měla být uživatelsky přívětivá a měla by

ÚVOD

hráči dát zážitek, který u jiných her mít nebude, o což by se měla zasloužit rozšířená realita.

Práce je rozdělena do několika částí. První část se zabývá analýzou problému. V další tvořím návrh, následně ho implementuji a v poslední části vytvořenou aplikaci testuji a diskutuji o budoucí práci na hře.

Analýza

1.1 Rozšířená realita

V této kapitole se budu věnovat rozšířené realitě z obecného hlediska. Vysvětlím, co je rozšířená realita, popíši ve zkratce její historii, uvedu příklady jejího použití v dnešní době a popíši, jak tato technologie funguje.

1.1.1 Definice

Rozšířená realita integruje digitální informace do uživatelského prostředí v reálném čase. Na rozdíl od virtuální reality, která vytvoří zcela umělé prostředí, rozšířená realita využívá prostředí okolo a přidává do něj překryvy, nové objekty nebo například zvuk. Rozšířená realita tedy kompletně nenahrazuje pohled na svět, ale spíše ho doplňuje. Rozšířená i virtuální realita spadají do oboru zprostředkování reality. To je místo, kde počítačový systém modifikuje naše vnímání reality světa kolem nás. V ideálním případě dovolí uživateli vidět předměty z reálného a virtuálního světa dohromady a umožní jim spolu interagovat[6].

Rozšířená realita stírá rozdíly mezi skutečným světem a uživatelským rozhraním tím, že je kombinuje přirozeným způsobem a umožňuje vytvoření jednoduchého a intuitivního uživatelského rozhraní i pro komplexní aplikace[7].

Aby se dalo mluvit o rozšířené realitě, musejí být splněny následující podmínky.

1. Kombinuje skutečnou i virtuální realitu
2. Pracuje v reálném čase, což znamená, že musí reagovat na uživatele, což rozlišuje rozšířenou realitu například od počítačových efektů ve filmech.
3. Virtuální obsah musí být registrován ve 3D v reálném světě. Nejedná se tedy o překryvné 2D vrstvy, které se zobrazují na displeji v kombinaci s reálným světem.

1. ANALÝZA

Rozšířená realita často využívá celou řadu senzorů (včetně fotoaparátu, akcelerometru a gyroskopu), počítačových komponent a zobrazovací zařízení pro vytvoření iluze virtuálních objektů v reálném světě.

1.1.2 Historie rozšířené reality

V roce 1966 profesor Ivan Sutherland z Harvardské univerzity vynalezl první model jednoho z nejdůležitějších zařízení používaného v oboru rozšířené a virtuální reality. Byl to display, který se nasazoval na hlavu. Byl ale tak velký, že bylo pro člověka nemožné ho nosit samotný na hlavě, takže musel být přimontovaný ke stropu místnosti a z něj se spouštěl a nasazoval na hlavu pracovníků. Byl to nicméně první krok k tomu, aby mohla být rozšířená realita použitelná.

V roce 1990 výzkumník společnosti Boeing Tom Caudell jako první použil výraz rozšířená realita. Při hledání způsobu, jak zjednodušit a zrychlit výrobu letadel začal využívat technologii rozšířené reality. Vytvořil displej, který pomocí softwaru překrýval pozice kabelů v místech kudy měli být vedeny. Mechanici se tedy nemusel ptát nebo čist příručky, aby mohli kably montovat. Byl to tedy první přístroj, který propojoval virtuální realitu s fyzickou.

O 2 roce později, tedy v roce 1992, udělali další dva týmy velké kroky v technologii rozšířené reality. L.B. Rosenberg vytvořil první rozšířenou realitu pro US Force, která rozšiřovala smyslové informace o pracovním prostoru a tím zlepšovala pracovní výkon uživatele, který jí používal. Druhý tým zasadil rozšířenou realitu do nositelných brýlí, ve které se zobrazovali návody na použití a opravu zařízení, aniž by se uživatel musel koukat do návodu. V tom samém roce byla na konferenci "Hawaii International Conference on Systems Science" představena definice rozšířené reality. Tom Caudell a David Mizell představili termín rozšířená realita a ukázali její podstatu tím, že vkládali počítačem vygenerované obrazy do reálného světa na obrazovce[8].

Od nového tisíciletí se rozmohly samotné aplikace s rozšířenou realitou. Bylo to také tím, že „Nara Institute of Science and Technology“ zpřístupnil ARToolKit jako open source, takže i když zatím nebyly vynalezeny chytré telefony, bylo možné zobrazovat rozšířenou realitu na mobilních zařízeních s kamerou a jednoduchým operačním systémem. Tím se rozšířená realita stala přístupnou velkému počtu lidí a byla tak vytvořena první venkovní mobilní hra s rozšířenou realitou, která se jmenovala ARQuake. Aby bylo možné hrát tuto hru, musel mít hráč na zádech batoh s výkonným počítačem té doby, na hlavě brýle, kde se zobrazovaly příšery, po kterých střílel z ovladačů, které měl v rukou. Od té doby vzniklo mnoho aplikací založených na rozšířené realitě[9, 10].

V současné době se díky popularitě smartphonů, které mají všechny potřebné komponenty, dostala do povědomí většiny světa a není to tedy již nic futuristického, co bychom vídali pouze ve filmech nebo počítačových hrách.

1.1.3 Využití

Rozšířená realita má širokou škálu aplikací v různých průmyslových odvětvích, a to díky vzestupu inteligentních přístrojů a celkovému vývoji výpočetní techniky. V současné době má velký potenciál i u běžné populace díky snadnému přístupu k zařízením, na kterých ji mohou provozovat. Jedním z hlavních a určitě nejrozšířenějších zařízení jsou chytré telefony, které má již většina populace na Zemi (v Evropě přes 70% celkové populace [11]). Současné aplikace se hlavně zaměřují na přidávání objektů do reálného prostředí. Ale objekty je také možno přesouvat a jinak s nimi interagovat.

1.1.3.1 Herní průmysl

Hry s rozšířenou realitou se obvykle hrají na zařízení, jako jsou chytré telefony, tablety a přenosné herní systémy. Je to kvůli možnosti využít kamery, gyroskop, GPS a jine jejich funkce, které většina jiných zařízení nemá. Hra sama o sobě může být velmi jednoduchá, jako je třeba virtuální dáma, která se hraje na povrchu normální hrací desky. Náročnější AR hry mohou vytvářet nové prostředí v rámci toho reálného. Například stavění mostů mezi jednotlivými kusy nábytku v místnosti. Hry zakomponované do reálného prostředí dostavání nový náboj pro hráče a činí je to velmi zajímavými. Hra Pokémon GO, je považována za průlomovou aplikaci v odvětví rozšířené reality. Hra využívá různé možnosti, které nabízí mobilní zařízení. Této hře se budu více věnovat v sekci „Rešerše“.

1.1.3.2 Medicína

Klinické postupy jsou pro začínající lékaře často obtížné zvládnout. Nikdo by pravděpodobně nechtěl chirurga bez zkušeností, aby ho operoval. Proto v Kanadě kliničtí lékaři z „University of Montreal“ vyvinuli užitečný software založený na virtuální a rozšířené realitě, který simuluje různé typy operací a mladí chirurgové se tak snadněji naučí a zdokonalí postupy a úkony spojené s operací. Momentálně pracují na podobné technologii pro neurochirurgii. [12]

1.1.3.3 Vojenské využití

Rozšířená realita přilákala zájem armády, které umožňuje provádět výcvik v simulovaném prostředí s vysokou mírou realismu. Armáda trénuje taktické operace díky nasazování virtuální osob (avatarů) a modelů vojenské techniky do reálného prostředí, které sledují přes brýle s technologií umožňující použití rozšířené reality. Vložené objekty se musejí v reálném světě pohybovat a chovat přirozeně. To umožňuje kvalitnější výcvik jednotek před vojenským zásahem a šetří tím lidské životy i materiální škody. Takový virtuální výcvik se používá i při nacvičování konkrétních akcí, například pro osvobození rukojmí. Tato

1. ANALÝZA



Obrázek 1.1: Rozšířená realita ve sportovních přenosech [1]

technologií se rozšiřuje i do vzdělávání police, hasičů a dalších bezpečnostních složek. [13]

1.1.3.4 Sportovní přenosy

Rozšířená realita se rozšířila do mnoha sportovních přenosů. Možnost přidat do obrazu porovnání rychlosti, času nebo přesnosti jednotlivých sportovců v reálném čase dává divákovi nový pohled na sportovní klání. Může se jednat například o porovnání závodníků se světovým rekordem nebo zobrazení národností, které se závodů účastní viz obrázek 1.1.

1.1.3.5 Reklama a marketing

Výrobci reklam a marketingová oddělení mohou díky rozšířené realitě oslovit své potencionální zákazníky dalším zajímavým způsobem. Reklamy interagují s okolím a mohou být tedy umístěny téměř všude. Takováto reklama je sice dražší nebo náročnější na zavedení než tradiční tištěné nebo vysílané reklamy, ale jsou originální a na zákazníky zapůsobí daleko více. Firma IKEA má svou vlastní aplikaci s názvem IKEA Catalog, která dokáže zobrazit pomocí rozšířené reality nábytek přímo v místnosti, kde bude stát a může si lépe vybrat typ nebo barvu. Více o této aplikaci bude v podkapitole „Reserse“.

1.1.4 Vykreslení modelu

Tato aplikace, kterou vytvářím v rámci diplomové práce bude vykreslovat 3D modely na určitém místě (na takzvaném markeru neboli značce). V této

podkapitole se budu věnovat tomuto typu vykreslování.

1.1.4.1 Nalezení značky

Aby mohl být 3D model vykreslen, musí být nejprve nalezen marker (značka), nad který se bude daný model vykreslovat. Marker může být vyobrazen různými způsoby.

Jedním z jeho hlavních typů, je QR-kód. Jeho výhodami je, že má jasné kontrastní zobrazení je tak jednoduše rozpoznatelný. Dále má výhodu jednoduché modifikace. Aplikace tak může rozpoznávat mnoho jeho variant, a tím uživateli zobrazovat větší množství virtuálních modelů. Tento typ se často využívá k business aplikacím, kde tolik nezáleží na vzhledu, ale jsou spíše účelné.

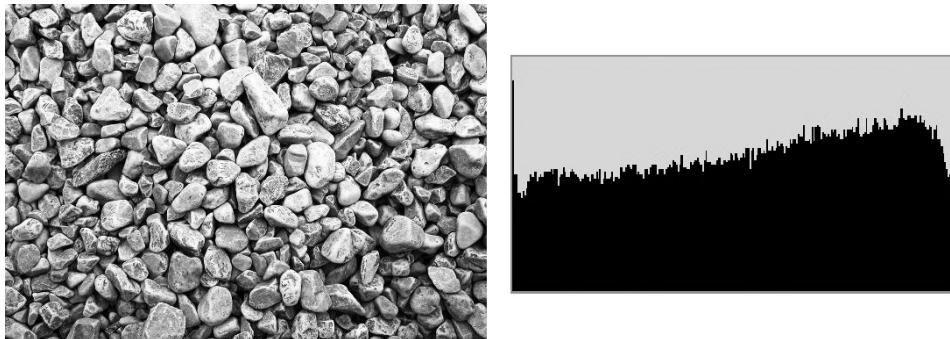
Druhým typem je jakákoli obrázek splňující pravidla rozpoznávajícího algoritmu. Takový obrázek musí být dostatečně velký, aby rozpoznávací algoritmus v něm dokázal najít rozpoznávané body a byl také rozpoznatelný z větší vzdálenosti. Dále by měl být obrázek bohatý na detaily, měl by mít dobrý kontrast a neměly by se na něm opakovat stejné vzory stále dokola, jako je například šachovnice. S jednoduchostí vzoru obecně platí, že čím je obsah markeru jednodušší, tím může být dál od kamery a stále bude značka detektovatelná. Všechny toto vlastnosti ovlivňují konečnou úspěšnost rozpoznání markeru, a to nejen při dobrých světelných podmínkách, ale také při horším osvětlení nebo například velkém náklonu snímacího zařízení vůči markeru. Na obrázku 1.2 je vidět, že histogram markeru je široký a neobsahuje žádné výrazné špičky. To znamená, že marker bude mít velkou šanci na rozeznání rozpoznávacím algoritmem. [14]

Musí být také zjištěno odkud zařízení značku zabírá, tedy kde se v reálném světě nachází zařízené vůči značce. Výpočet polohy zařízení vůči značce se provádí v reálném čase, protože model musí být sladěn s reálním světem. Kamera tedy zachytí obraz a pošle ho na zpracování. Algoritmus prohledá zachycený obraz a zjistí, jestli se na něm nevyskytuje značka. Pokud algoritmus nalezne předem definovanou značku, kterou hledá, vypočte vzdálenost a orientaci snímajícího zařízení vůči značce.

1.1.4.2 Vykreslení modelu

Proces vykreslování 3D modelů vyžaduje několik kroků. Nejprve je potřeba takový 3D model získat a následně se musejí vytvořit animace pro jednotlivé akce. Za akci se může považovat chůze, útok, obrana nebo třeba smrt. Takový animovaný model vykreslí knihovna k tomu určená, které je potřeba předat souřadnice a úhel, na kterých má model vykreslit. Knihovnu pro vykreslení 3D objektů v rámci své diplomové práce dělat nebudu. V kapitole návrhu budu porovnávat a volit mezi jednotlivými knihovnami, které jsou k tomuto účelu určeny.

1. ANALÝZA



Obrázek 1.2: Marker s histogramem [2]

1.2 Platforma Android

1.2.1 Architektura OS

Architektura OS Android je rozdělena do pěti vrstev: jádro, knihovny, aplikační framework, běhové prostředí a aplikace. Každá vrstva má svoji funkci. Vrstvy zároveň nemusí být přímo odděleny a mohou se částečně prolínat s ostatními vrstvami.

Nejnižší vrstvou je kernel (jádro), který je základem operačního systému a zajišťuje komunikaci mezi používaným hardwarem se zbytkem softwaru ve vyšších vrstvách. Jádro OS Androidu je postaveno na Linuxu a využívá mnoho jeho vlastností, jako například správu paměti, správu sítí nebo správu procesů, která se používá, pokud běží více aplikací najednou. Výběr systému Linux jako jádro OS Android byl odůvodněn tím, že se Linux snadno sestaví na různých zařízeních a je tedy zaručena přenositelnost mezi zařízeními.[15]

Další vrstvou jsou knihovny. Tyto knihovny jsou psány v jazyce C a C++. Funkce těchto knihoven jsou poskytovány pomocí Android Application Framework. Knihovny se starají například o zobrazování aplikací, jejich vrstvení, operaci s grafikou nebo mediálními soubory.[15]

Třetí vrstvou je takzvaná Android Runtime vrstva, která obsahuje aplikační virtuální stroj. Tato vrstva slouží primárně pro běh aplikací. Dalvik Virtual Machine (DVM) je registrově orientovaná architektura, která využívá výše uvedených vlastností linuxového jádra. DVM také pomáhá při překladu aplikace, která probíhá zkompilováním zdrojového kódu v jazyce Java do Java byte kódu a následně se Java byte kód pomocí Dalvik kompilátoru znova překompiluje a výsledný Dalvik byte kód je spuštěn na DVM.

Aplikační vrstva poskytuje přístup ke službám, které mohou být následně použity přímo v aplikacích. Mezi nejdůležitější služby patří například Sada prvků View, které se používají pro sestavení uživatelského rozhraní. Další službou je např. Content providers, která umožňuje přístup k datům jiných aplikací a mnohé další užitečné služby. [16]

Nejvyšší vrstvou je samotná aplikace, která je používána běžným uživatelem. Aplikace na OS Android se stahují z oficiálního obchodu Google Play Store. [17]

1.2.2 Vývoj aplikací pro OS Android

Aplikace pro OS Android jsou psány v jazyce Java. Pro vývoj se používají 2 hlavní vývojové prostředí, a to Android Studio nebo Eclipse. Já jsem vybral Android Studio, jelikož je to oficiální vývojové prostředí od firmy Google[18]. Android Studio je rychlé, má menší nároky na výkon a obsahuje lepší naštěvač.

Java je objektově orientovaný programovací jazyk od firmy Sun Microsystems z roku 1995. Jeho výhodou je přenositelnost mezi různými systémy, tedy i na mobilní zařízení.[19]

Dále je potřeba k vývoji Android aplikací Android Software Development Kit (SDK) a Java Development Kit (JDK).

SDK obsahuje nástroje pro vývoj aplikací pro OS Android, jako jsou například USB ovladače, emulátory, dokumentace a hlavně samotný Android. JDK je soubor základních nástrojů a knihoven pro vývoj aplikací. Součástí JDK je Java Runtime Environment (JRE), které slouží pro spouštění aplikací a vývojových nástrojů. Dále JDK obsahuje překladač, debugger a další. V aplikacích pro OS Android jsou tři hlavní typy souborů, a to sources, resources a soubor Manifest.

Sources jsou zdrojové kódy s příponou .java, kde se píše programová logika, stejně jako při programování v Javě.

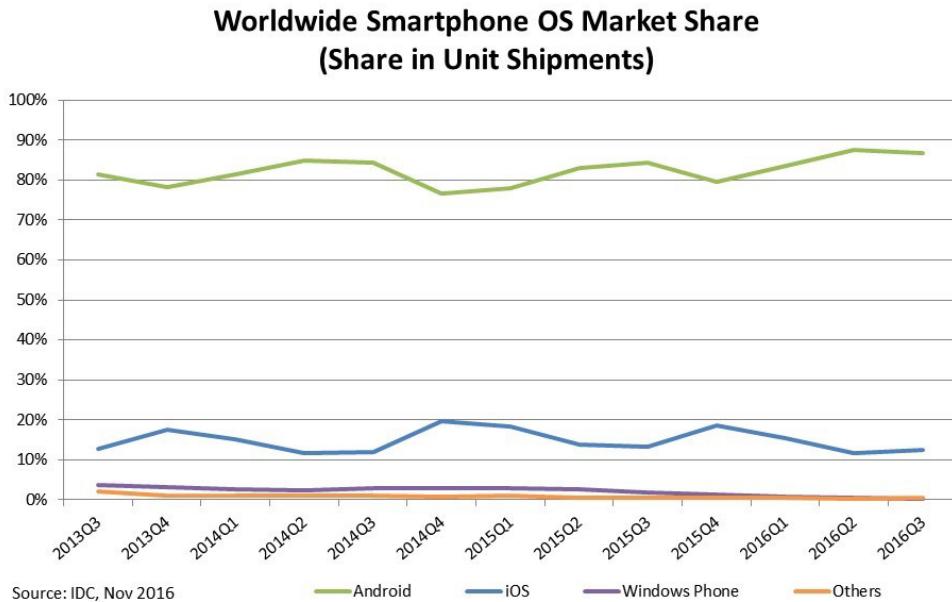
Resources jsou prostředky, které podporují aplikaci. Patří mezi ně například obrázky, definice barev a další prostředky, které představují, jak bude aplikace vypadat. Hlavní součástí zdrojových kódů jsou takzvané layout soubory, které se píší jazykem XML a představují, jak budou vizuálně vypadat jednotlivé obrazovky aplikace.

Manifest je hlavní soubor, který musí mít každá Android aplikace. Soubor je uložen v kořenovém adresáři, je psaný stejně jako layout pomocí XML a dává systému základní informace o aplikaci, které musí systém mít před jejím spuštěním. Jsou zde definované všechny použité prvky a oprávnění, která aplikace potřebuje.[20]

1.2.3 Verze OS Android

V říjnu roku 2008 vyšla první verze OS Android. Od té doby každý rok přibyla minimálně jedna verze. Nové verze opravují chyby předchozích verzí a zároveň přidávají nové funkce. Na obrázku 1.3 je vidět, že procentuální poměr užívaných mobilních zařízení s OS Andorid se stále zvyšuje. Na grafu je vidět, jak jsou na sobě operační systémy Android a iOS závislé. Kdykoliv je vydána nová verze iPhonu, prodej zařízení s iOS stoupá a s OS Android klesá. V současné

1. ANALÝZA



Obrázek 1.3: Poměr OS v prodaných zařízeních [3]

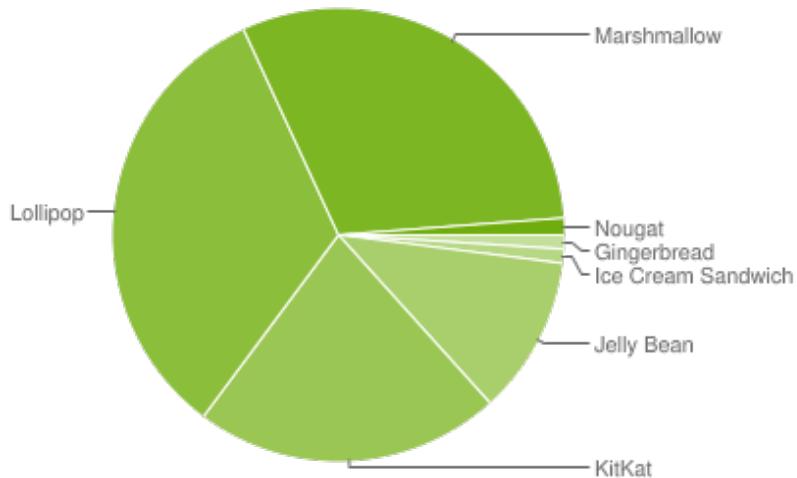
době má OS Android asi 85% uživatelů mobilních zařízení. Od 3. verze se každá nová verze pojmenovává podle některé sladkosti začínající na následující písmeno z abecedy oproti předchozí verzi. Tedy od nejstarší po nejnovější se verze jmenují: Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, KitKat, Lollipop, Marshmallow a Nugat. [21]

Tolik verzí samozřejmě způsobilo, že vývojáři aplikací pro OS Android musí své aplikace přizpůsobovat jak novým, tak starým verzím, a to bývá občas trochu problém. Toto částečně řeší Google, který vytvořil support knihovny, které převádějí nové funkce do starších verzí. Na obrázku 1.4 a v tabulce 1.1[4] je vidět v jakém procentuálním množství jsou jednotlivé verze zastoupeny. Android podporuje dopřednou kompatibilitu, což znamená, že aplikace bude fungovat i na verzích, které teprve vyjdou. [22]

Moje aplikace bude podporovat většinu stále používaných verzí, tedy od verze Jelly Bean, a díky dopředné kompatibilitě bude fungovat i na všech novějších verzích.

1.3 Rešerše obdobných aplikací

Aplikace pro rešeršní zpracování, které fungují na podobném principu a mají podobné využití, jsem hledal na Google Play[23]. Google Play je oficiální online distribuční služba pro Android aplikace. Porovnával jsem pouze bezplatné



Obrázek 1.4: Poměr verzí OS Android [4]

Verze	Název	API	Distribuce
2.3	Gingerbread	10	1.0%
4.0	Ice Cream Sandwich	15	1.0%
4.1 - 4.3	Jelly Bean	16 - 18	11.3%
4.4	KitKat	19	21.9%
5.0 - 5.1	Lollipop	20 - 22	32.9%
6.0	Marshmallow	23	30.7%
7.0 - 7.1	Nugat	24 - 25	1.2%

Tabulka 1.1: Poměr verzí OS Android

aplikace, které mají společné či podobné prvky s mojí aplikací. Každá nalezená a porovnávaná aplikace má vlastní sekci, ve které ji popisuji a hodnotí. V mé aplikaci jsem se snažil poučit z nedostatků těchto aplikací, které jsem porovnával a naopak jsem se nechal inspirovat mnohými dobrými nápady.

1.3.1 IKEA Catalog

Aplikace od firmy IKEA pracuje na podobném principu jako aplikace, kterou vytvářím v rámci této diplomové práce. Aplikace IKEA Catalog dokáže zakomponovat 3D modely do reálného prostředí pomocí takzvaných markérů, které označují místo, kde se má model vykreslit. Aplikace vykresluje pouze statické 3D modely nábytku. Uživatelé pomocí této aplikace získají lepší představu, jak zvolený kus nábytku bude vypadat v prostředí, kde bude používán.

1. ANALÝZA

1.3.2 Pokémon GO

Hra Pokémon GO využívá mimo jiné i technologii rozšířené reality tím, že zobrazuje pokémony v reálném světě. Hráč prozkoumává reálný svět přes fotoaparát svého zařízení a hledá pokémony, které se v něm pomocí rozšířené reality zobrazují. Algoritmus pro zobrazení pokémonů vyhledá rovnou plochu a vypočte, jestli je vodorovná. Pokud ano, tak na ní může zobrazit model pokémona. Tento způsob vylepšuje typ zobrazení a pozicování virtuálních objektů pomocí markerů.

1.3.3 Google Překladač

Google Překladač není striktně AR aplikace, ale má jednu užitečnou funkčnost s AR spojenou. Je to funkce na překládání textů a nápisů. Jednoduše stačí namířit fotoaparát telefonu na text, kterému uživatel nerozumí a aplikace ho v reálném čase přeloží a zobrazí překlad přes text, který chtěl uživatel přeložit.

1.4 Rešerše vývojářských nástrojů pro AR

Jedním z nejobtížnějších částí vývoje aplikací s rozšířenou realitou je přesný výpočet pohledu uživatele v reálném čase. Virtuální objekty totiž musejí být přesně sladěny s reálnými objekty, což je výpočetně náročná operace. Musí se tedy vypočítat skutečná pozice kamery a její orientace vzhledem k obrazu, který používáme jako značku (marker). Tato značka je následně překryta virtuálním objektem a samotné zobrazení animovaného modelu je taktéž výpočetně náročné. Rychlý, přesný tracking umožňuje nový způsob, jak takovéto aplikace využít. Proto je velmi důležité vybrat správné nástroje pro tyto výpočetní operace. Na začátek také musím napsat, že jsem se rozhodl si tyto složité operace neprogramovat sám, a to proto, že na trhu existují již kvalitní řešení, které bych nebyl schopen překonat.

1.4.1 Rozpoznání markeru

V této podkapitole popíši a porovnám dva nástroje pro rozpoznání markeru, se kterými jsem pracoval.

1.4.1.1 ARToolkit

ARToolKit je multiplatformní open source knihovna pro rozšířenou realitu, která je, kvůli rychlosti, psaná v jazyce C a C++. Mimo jiné ARToolKit knihovna podporuje rozpoznávání značek. Výhodami této knihovny je, že je multiplatformní (funguje na Android, Windows Phone, iOS, macOS i Windows). Dalšími výhodami je skutečnost, že tato knihovna je velmi rychlá a zároveň nezabírá mnoho místa (to se hodí například na mobilních zařízeních).



Obrázek 1.5: Rozpoznání markeru: Vuforia [5]

ARToolKit podporuje Unity - herní engine, o kterém se zmiňuji níže a dokáže zároveň sledovat více značek najednou.

Hlavní nevýhodou této knihovny je, že dokáže rozeznávat pouze čtvercové QR kódy a ne jakékoliv obrazy. ARToolKit má také problémy s rozeznáváním značek při větším náklonu kamery. [24]

1.4.1.2 Vuforia

Vuforia je open source knihovna od společnosti Qualcomm, která může běžet na OS Android, iOS nebo Unity3D platformách. Je to freeware, který má své placené části. Vuforia se liší od ARToolkitu tím, že dokáže rozeznávat jenom čtvercové QR kódy, ale také téměř jakékoliv obrazy. Databáze těchto značek může být uložena v zařízení nebo v cloudovém uložišti, kde se počet rozpoznání a počet značek platí. Cena se odvíjí od počtu značek v cloudovém uložišti a počtu jejich rozpoznání. Pro tuto hru bude však stačit lokální uložiště, a proto je tato knihovna ideální volbou. Vuforia má jednoduchou instalaci a implementování do projektu. [25]

Knihovna pracuje s maticemi, které určují polohu kamery vůči značce. Samotné rozpoznávání funguje tak, že porovnávaný marker z databáze se převede na černobílé a následně se na něm určí kontrastní vzory viz obrázek 1.5. Ty se poté vyhledávají v obrazu zachyceného pomocí snímacího zařízení. [5]

1. ANALÝZA

1.4.2 Vykreslení 3D modelu

Níže v této podkapitole popíši a porovnám knihovny pro vykreslení 3D modelu.

1.4.2.1 JPCT

JPCT je malá open source knihovna založená na OpenGL, která je velmi snadno implementovatelná do projektu. Knihovna má jednoduchou a jasnou dokumentaci. Knihovna je dobrá pro zobrazování statických 3D modelů, bohužel není dělaná pro práci s animacemi. Grafické nástroje pro tvorbu animací vkládají údaje o animaci přímo do 3D objektu, tato knihovna je bohužel neumí použít a pro animaci je potřeba vygenerovat celou řadu statických modelů a ty zobrazovat za sebou tak, aby celá scéna tvořila animaci. Tuto knihovnu jsem tedy kvůli nehezkým animacím a paměťové náročnosti s tím spojenou, zamítl

1.4.2.2 LibGDX

Po delším hledání jsem našel open-source knihovnu LibGDX založenou také na OpenGL. Knihovna obsahuje C a C++ komponenty pro lepší výkon a je vyvíjena komunitou vývojářů, což zaručuje dobrou podporu při vývoji a pravidelné aktualizace. Knihovna LibGDX umí oproti JPCT pracovat s animacemi a také umí přehrávat zvuk a zvukové stopy ve formátech WAV, MP3 a OGG. Mezi další výhody patří podpora uživatelského vstupu jako je například dotek nebo různá gesta, dál podpora měřiče zrychlení (akcelerometr) a kompasu.

Tato knihovna je sice násobně větší než JPCT, ale i tak je pro uživatele její velikost zanedbatelná.

1.4.2.3 Unity

Unity je profesionální nástroj pro tvorbu virtuálního prostředí, animací a her. Tato knihovna je používaná v mnoha hrách a je obecně hojně používaná ve velkých projektech se složitějším prostředím a animacemi. Tento nástroj je z počátku vývoje zdarma, ale bohužel následné výdaje dosahují částky až 125\$[26] měsíčně. Proto jsem se rozhodl tento nástroj nevyužít.

KAPITOLA **2**

Návrh

Tato kapitola se zabývá návrhem aplikace. Do návrhu aplikace patří návrh vlastní hry, architektury a uživatelského rozhraní. Dále se kapitola zabývá návrhem algoritmů pro virtuální objekty v reálném světě a jejich vzájemnou interakcí.

2.1 Popis hry

Hru vytvářenou v rámci této diplomové práce je inspirována původní japonskou hrou Pokémon. Každý hráč představuje jednoho trenéra, který vlastní jednotlivé pokémony a vzájemné souboje probíhají v reálném světě pomocí rozšířené reality. Mnoho podobných aplikací pro OS Android na trhu není a proto věřím, že rozšířená realita je inovace, která dokáže zaujmout. V této části budou popsány herní mechanismy tak, jak byly měly ve výsledné aplikaci vypadat. Inspirace pro hru pochází z již zmiňované japonské hry, kde mezi sebou soupeří jednotliví trenéři se svými pokémony. Samozřejmě některé mechanismy jsou upraveny, aby je bylo možno použít v aplikaci s rozšířenou realitou.

2.1.1 Princip hry

Herní princip bude jednoduchý, každý hráč bude vlastnit fyzické karty, které budou reprezentovat jednotlivé typy pokémonů. Každý pokémon bude mít různé vlastnosti (počet životů, síla útoku, síla obrany, rychlosť, různé útoky a další), které budou ve hře zohledněny. Souboj trenérů bude probíhat tak, že oba vyloží jednu kartu a tím zvolí svého pokémona pro tento souboj. Oba pokémoni se zobrazí nad kartami, které byly vyloženy a souboj započne. Oba hráči budou mít možnost se svým pokémonem útočit, bránit nebo se s ním pohybovat. V závislosti na volbě pokémon provede akci, které volbě odpovídá. Souboj končí ve chvíli, kdy jeden pokémon ztratí všechny své životy.

2. NÁVRH

Každý hráč bude mít svůj účet, kde budou základní informace o něm a ty budou viditelné i ostatním hráčům. Mezi další principy hry bude patřit vývoj pokémonů i vlastního trenéra (hráče). Pokémoni budou souboji získávat zkušenosti a po dosažení určité hranice budou vyvinuti na vyšší úroveň. Hráči svými výhrami budou sbírat body do žebříčku, a tím se budou moci poměřovat vzájemně mezi sebou. Hra by měla být svižná, souboje krátké (max. 2 minuty) a měla by obsahovat i modul pro trénování, tedy hru jednoho hráče.

2.1.2 Vlastnosti pokémona

Každý pokémon má 5 základních vlastností, které se v souboji projevují. Jedná se o následující vlastnosti.

- **Počet životů:** životy, které pokémon ztrácí při souboji, když ho zasáhne druhý pokémon
- **Síla útoku:** jak velké poškození dá jinému pokémonovi během útoku - počet životů, které ubere nepříteli
- **Síla obrany:** jak moc dokáže pokémon odolat při zásahu druhým pokémonem, pokud se brání
- **Rychlosť:** Jak rychle se pokémon dokáže pohybovat
- **Typ pokémona:** typ pokémona inspirován původní hrou. Typ má vliv na souboj tím, že některé typy pokémonů mají na sebe vzájemný vliv, například větší účinnost při zásahu. Typy pokémonů jsou normální, ohnivý, vodní, elektrický a travní

2.1.3 Ovládání

Hráč ovládá svého pokémona pomocí tlačítka na obrazovce. Na obrazovce budou tlačítka pro obranu a útok. Dále bude moci uživatel ovládat pohyb pokémona pomocí virtuálního joysticku na obrazovce a celá hra bude také možná pozastavit pomocí tlačítka „PAUSE“.

2.1.4 Herní svět

Hra se odehrává kdekoli na rovné ploše a hráči tak mohou hrát na zemi, stole nebo kdekoli jinde a nejsou místem témař omezováni. Hra se odehrává v blízkém okolí jedné (pro hru jednoho hráče) nebo dvou (pro hru dvou hráčů) karet, které hráči vyložili. Herní prostředí zasazené do reálného světa vidím jako jeden z inovativních prvků této aplikace.

Hráči budou omezení určitou vzdáleností od vyložených karet, která bude značit herní prostor. Za tuto hranici se nebudou moci pohybem dostat. Toto omezení musí být nastaveno tak, aby hráči měli dostatečně velké herní pole

pro pohyb a manévrování, ale zároveň ne moc velké, aby se pokémoni mohli vzdálit natolik, že by snímač kamery již nerozeznal marker. Pokud by nastala tato situace, aplikace by přestala vykreslovat model a hráč by musel své herní zařízení vrátit opět do blízkosti markeru. Až poté by mohl pokračovat.

2.2 Požadavky

2.2.1 Funkční požadavky

F1 Registrace uživatele

F2 Přihlášení uživatele ke svémú účtu

F3 Zobrazení a úprava profilu uživatele

F4 Hra jednoho hráče

F5 Hra dvou hráčů

F6 Zobrazení statistik

F1 Registrace uživatele: Pro registraci uživatele bude potřeba jen emailový účet a heslo, které se vyplní do jednoduchého formuláře a tlačítkem „Registrovat“ se vytvoří uživatelský účet na serveru se službou Firebase od Googlu.

F2 Přihlášení uživatele ke svémú účtu: Pokud má uživatel již vytvořený účet, může se k němu přihlásit. Zde bude opět stačit jen emailový účet a heslo, které zadával při registraci. Po přihlášení se stáhnou všechna data o uživateli, jako je například jméno a umístění v žebříčku.

F3 Zobrazení a úprava profilu uživatele: Pokud je uživatel přihlášen, může si zobrazit svůj účet, kde bude moci upravovat informace nebo si změnit heslo.

F4 Hra jednoho hráče: Hra jednoho hráče bude obsahovat pokémona, kterého si uživatel vybere ze svých karet a pokémona, se kterým bude trénovat. Při hře bude moci použít všechny možnosti souboje popsané v sekci „Princip hry“.

F5 Hra dvou hráčů: Hráči nejprve propojí svá zařízení pomocí Wi-Fi neb bluetooth, následně každý hráč vyloží kartu se svým pokémon a hra se odstartuje. Oba hráči budou hrát současně, přičemž prováděné akce se budou navzájem sdílet a oba hráči tak budou moci sledovat herní scénu na vlastním zařízení.

F6 Zobrazení statistik: Pokud bude uživatel připojen na internet bude si moci zobrazit statistiky, které budou obsahovat pořadí jednotlivých hráčů.

2.2.2 Nefunkční požadavky

N1 Data uložena v aplikaci

N2 Synchronizace dat o uživateli se serverem

N3 Intuitivní ovládání

N4 Přehlednost zobrazených informací

N5 Uživatelsky přívětivé rozhraní

N6 Jednoduchý herní systém

N1 Data uložena v aplikaci: Všechna potřebná data o uživateli jsou uložena v aplikaci, aby byla aplikace funkční, i když zařízení není připojeno k internetu. V aplikaci budou uložena jen data o uživateli a pokud je bude chtít upravit nebo provést jinou úpravu, která by ovlivnila data na serveru, bude muset být uživatel připojen k internetu.

N2 Synchronizace dat o uživateli se serverem: Při zapnutí aplikace se zjistí, zda je zařízení připojeno k internetu a případně aplikace aktualizuje data ze serveru. Toto vše se děje na pozadí aplikace, tedy uživateli to nijak neovlivní.

N3 Intuitivní ovládání: Uživatel by se neměl v aplikaci ztratit a vždy by měl vědět na co kliknout v průchodu aplikací.

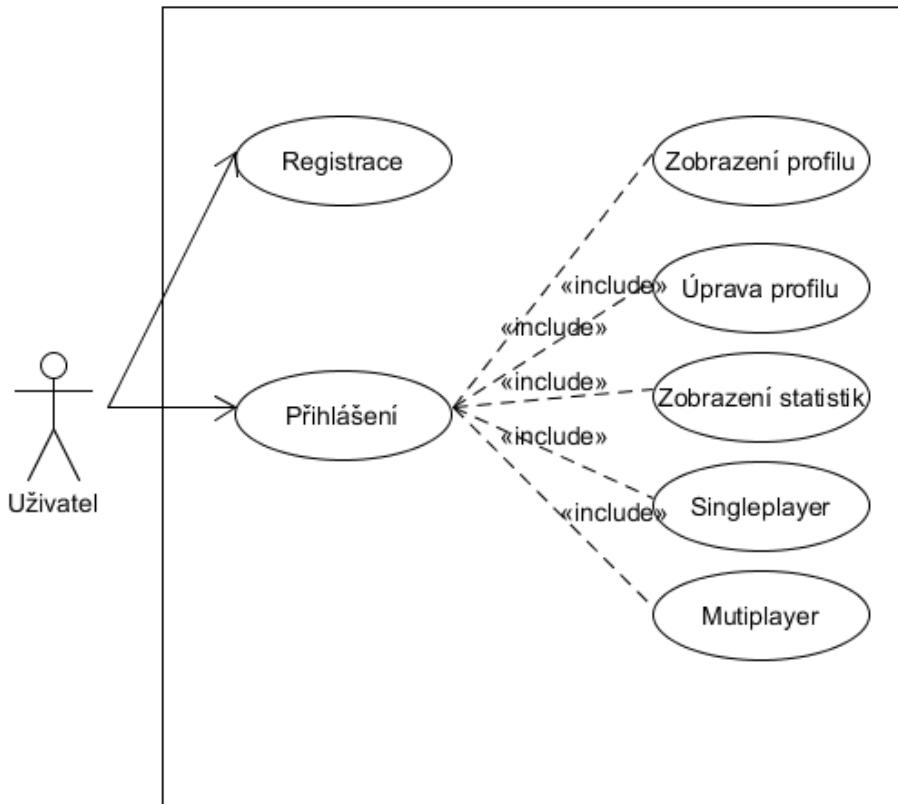
N4 Přehlednost zobrazených informací: Aplikace nezobrazuje mnoho informací, i přesto je důležité, aby se v nich uživatel dobře vyznal.

N5 Uživatelsky přívětivé rozhraní: Uživatelské rozhraní se snaží dodržovat zásady a doporučení stanovené na oficiálních stránkách pro vývoj Android aplikací. Na stránkách se popisují konstrukční zásady, užití barev, stylů a rozměrů.

N6 Jednoduchý herní systém: Herní systém je zásadní věcí na celé aplikaci. Proto, aby byla úspěšná musí být jednoduchý, rychle pochopitelný a zábavný. Kombinace herních karet, aplikace a rozšířené reality je zábavná. Na hlavní herní obrazovce bude jenom minimum tlačítek, a to zaručí jednoduchost a rychlé pochopení.

2.3 Případy užití

Výše uvedené funkční požadavky se přímo promítají do případů užití. Tento diagram slouží pro návrh UI i pro tvorbu scénářů pro usability testing, který ověří, zda je daný návrh pro uživatele dostatečně jednoduchý, srozumitelný



Obrázek 2.1: Diagram případů užití

a zda je uživatel schopen v aplikaci dosáhnout všech daných cílů. Dále slouží ke zpřesnění odhadů pracnosti implementace.

Model případů užití se skládá ze seznamu účastníků a diagramů případů užití. V seznamu účastníků je jen jedna osoba, a to uživatel.

2.4 Architektura aplikace

Aplikace je založena na klasickém principu vrstev. Jako nejvhodnější architekturu pro Android aplikace jsem zvolil Model-View-Presenter (MVP). Jednotlivé vrstvy a jejich spolupráce jsou znázorněny na obrázku 2.2 a jsou popsány dále v textu. MVP je podobný jako Model-View-Controller (MVC). MVP je softwarový architektonický vzor, který rozděluje danou aplikaci do tří vzájemně propojených částí. Hlavním rozdílem od MVC je to, že všechny uživatelské vstupy a akce se do modelu dostávají přes střední vrstvu (presenter). To zaručuje oddělenost vrstev, a tím například lepší testování nebo nahrazení

jedné vrstvy jinou.

2.4.1 View

View je prezentováno layouty, Aktivitami, Fragmenty, Dialogy a dalšími prvky, které se zobrazují uživateli na obrazovce mobilního zařízení. Jediná práce View je dát vědět Presenteru, co uživatel napsal nebo udělal.

2.4.2 Presenter

Presenter působý jako „muž uprostřed“. protože je zodpovědný za předávání informací mezi View a Modelem. Načítá data z Modelu a zformátovaná je předá View, ale na rozdíl od MVC rozhoduje, co se bude dít s daty od uživatele.

2.4.3 Model

Model je pouze vstupní branou pro modely, databázi a business logiku.

2.4.4 Důvody zvolení MVP

Ve většině moderních aplikací pro Android stačí použít View model architekturu, ale to většinou skončí tak, že vše je propojeno se vším. Programování pro Android je k tomu velmi náchylné, protože u Aktivity nebo Fragmenty je jednoduché použít k vykonávání kódu na pozadí. To samozřejmě není správně ať už kvůli přehlednosti, tak třeba kvůli testování.

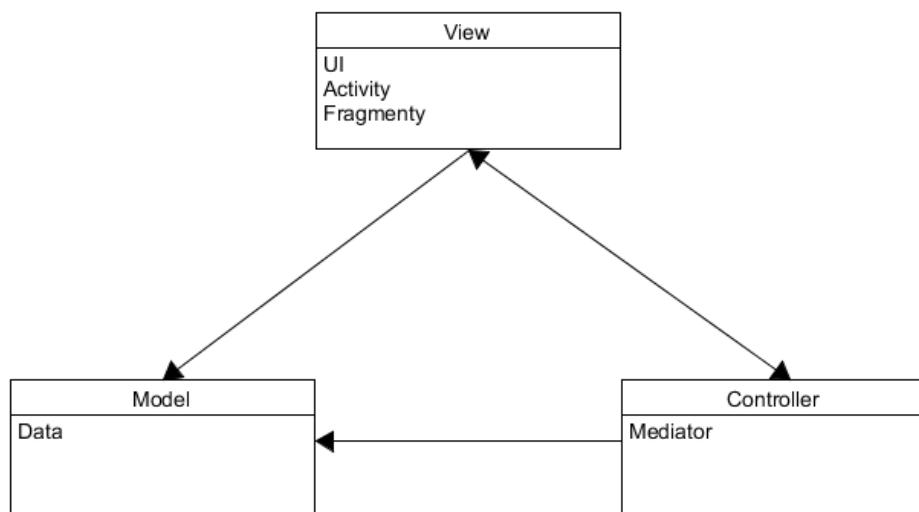
MVP je způsob, jak na Androidu oddělit procesy na pozadí od aktivit, views a fragmentů tak, aby byly nezávislé na většině událostí životního cyklu. Tento způsob aplikace se stává jednodušším, spolehlivějším a zkracuje kód jednotlivých tříd až 10krát. Navíc je kód lepší pro udržitelnost a testování. [27]

2.5 Uživatelské rozhraní

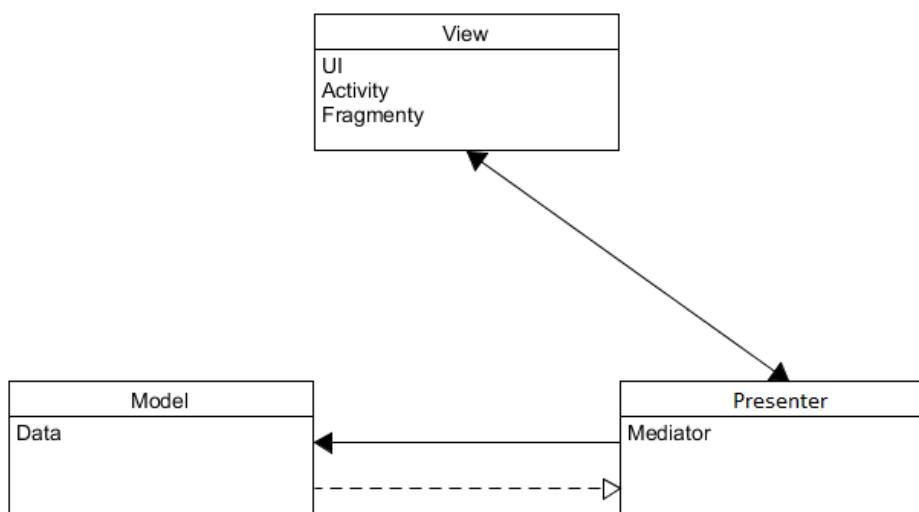
U mobilních aplikací je velice důležité uživatelské rozhraní a samotný design. Vše by mělo být jednoduché, přehledné a rychle pochopitelné. Proto jsem při návrhu pracoval s Material Design specifikacemi[28] přímo od společnosti Google. Díky těmto informacím a vlastním zkušenostem jsem navrhl jednoduché a příjemné uživatelské rozhraní.

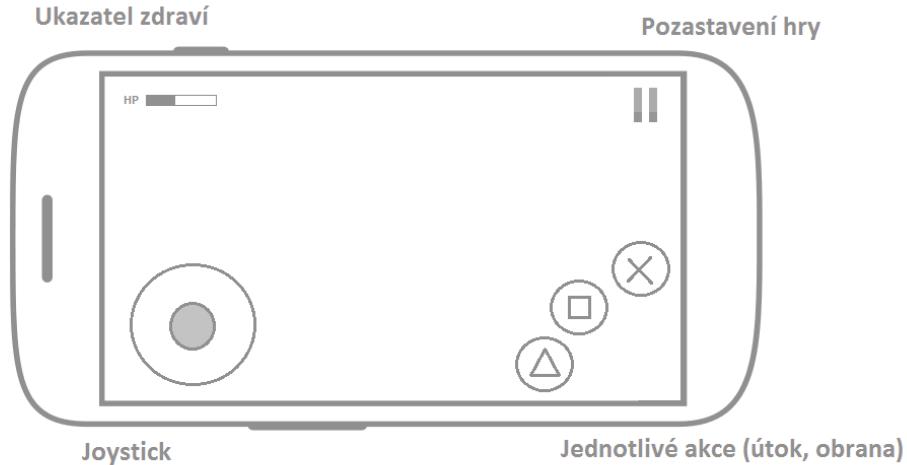
Design samotné hry je nejpodstatnější. Na této obrazovce se bude uživatel nacházet většinu času, který s aplikací bude trávit. Tato obrazovka musí zobrazovat samotnou hru snímanou fotoaparátem a zobrazovanou na obrazovce a zároveň obsahuje ovládací prvky hry. Důležité je tedy tyto prvky zakomponovat na obrazovku tak, aby uživateli nevadily při hraní hry (aby například nezakrývala příliš velkou část obrazovky) a zároveň byly dobře dostupné s jednoduchým ovládáním. Na Obrázku 2.3 je vidět, jak jsou jednotlivé

MVC



MVP





Obrázek 2.3: Návrh herní obrazovky

komponenty herní obrazovky uspořádány, aby měl uživatel co nejlepší pocit ze hry. Pro tento design jsem byl inspirován ovladači ke konzolím, které mají stejně jako v aplikaci v levé části ovládání pohybu a v pravé vyvolávání akcí.

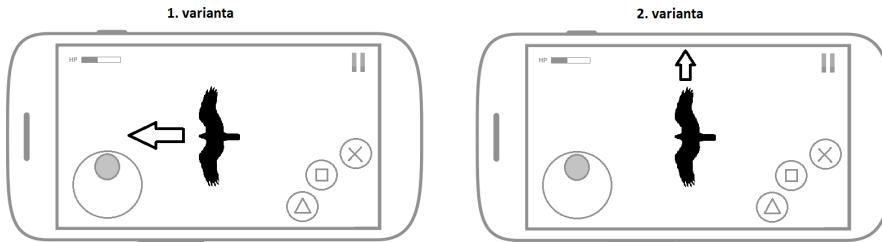
Ostatní obrazovky jsou navrhnuty velmi jednoduše. Jsou to proklikávací menu, popřípadě vyplňovací formuláře pro přihlašování, registraci nebo například úpravu profilu. Kombinace barev jsou voleny tak, aby tvořily příjemné prostředí. Kombinace jsou voleny pomocí nástroje „Adobe Kuler“[29], který pro zadanou barvu najde odstíny stejné barvy nebo kontrastní barvu, která s původní zadanou barvou tvoří dobře použitelnou kombinaci.

2.6 Pohyb modelu

Pohyb modelu je ve hře velmi důležitý a proto jsem dlouho rozmýšlel, jak ho realizovat, aby byl pro uživatele co nejjednodušší. Měl jsem dvě varianty, nad kterými jsem uvažoval. První varianta byla pohyb pomocí klikání na obrazovku a model by se pohyboval na místo, kde bylo zaznamenáno kliknutí. Druhou variantou bylo použití joysticku v levé nebo pravé straně obrazovky.

Rozhodl jsem udělat testování na vzorku tří lidí, kterým jsem dal vyzkoušet obě varianty. Výsledek jasně určil jako ideálnější způsob použití joysticku. Bylo to především proto, že uživatel v průběhu hry drží telefon oběma rukama, což znesnadňovalo ovládání pomocí klikání na obrazovku.

Po rozhodnutí o tom, že joystick bude hlavním ovládacím prvkem, jsem musel rozmyslet, jak budu implementovat reakce modelu na vstup joysticku.



Obrázek 2.4: 2 varianty reakce modelu na vstup joysticku

Byly zde dva přístupy, které jsem zvažoval. První přístup je jednoduší pro implementaci a funguje tak, že pokud hráč drží joystick směrem nahoru, model jde vždy dopředu (směrem, kterým je otočen). Natočení joysticku na jednu nebo na druhou stranu od směru vzhůru znamená pootočení modelu na zvolenou stranu.

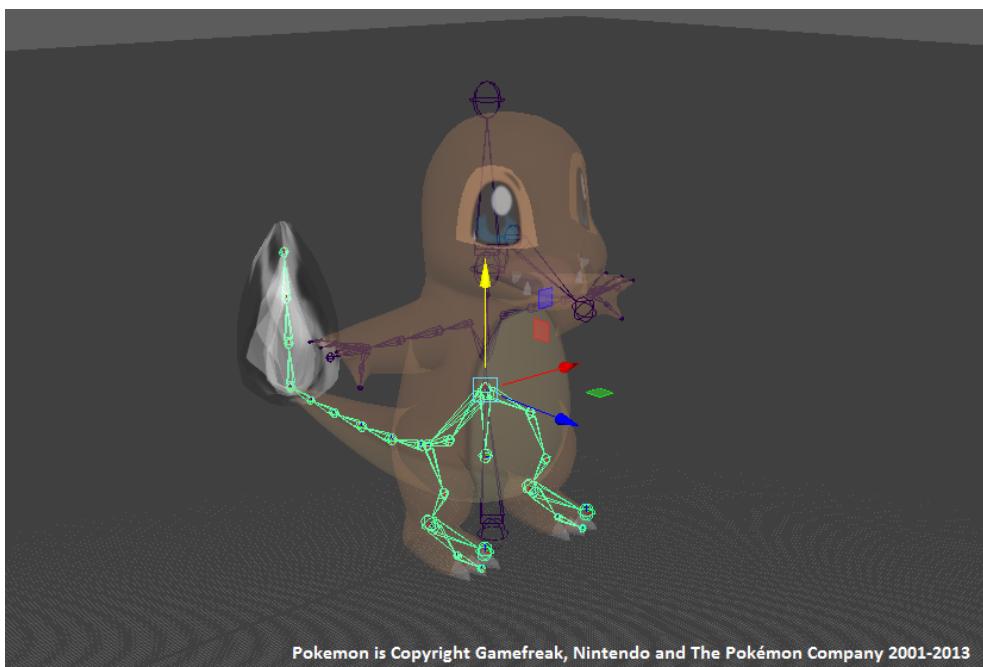
Druhým o něco složitějším přístupem je implementovat algoritmus, který podle vstupu hráče pomocí joysticku směruje model vždy směrem, kterým joystick směruje. Tento přístup je složitější v tom, že musí zohledňovat pozici kamery vůči modelu a z ní vypočítat, kterým směrem model otočit. Obě varianty jsou znázorněny na obrázku 2.4. Tento obrázek popisuje směr, kterým bude model směrovat, pokud je joystick směruje nahoru.

Pro rozhodnutí jsem opět využil testery, kteří testovali rozložení ovládacích prvků na herní obrazovce. Všichni tři testeři se přiklonili k druhé variantě, ale po konzultaci s vedoucím této diplomové práce jsem se rozhodl pro použití obou typů směrování modelu. Jako primární ovládání bude zvolena varianta č. 2 a do nastavení aplikace bude přidána možnost toto ovládání změnit na první variantu.

2.7 Objekty ve hře

Statické modely, které v této hře používám byly vytvořeny společnostmi Nintendo a GameFreak a jsou držiteli práv k jejich používání. Objekty se mohou používat dle vlastních potřeb, ale použití musí odpovídat US Copyright (USC) 17 §107 a nesmí porušovat mezinárodní autorské právo. To znamená, že model nesmí být použitý žádným způsobem, který generuje zisk pro sebe nebo někoho jiného. Ve hře dále bude muset být uveden zdroj modelů a u všech modelů musí být textově uvedeno, že se jedná o modely vytvořené společnostmi Nintendo a GameFreak.

Použitím těchto modelů jsem nemusel řešit jejich vytváření a mapování textur na ně. Co jsem ale musel řešit, byly animace, které jsou v celé aplikaci



Pokémon is Copyright Gamefreak, Nintendo and The Pokémon Company 2001-2013

Obrázek 2.5: Kostra 3D modelu

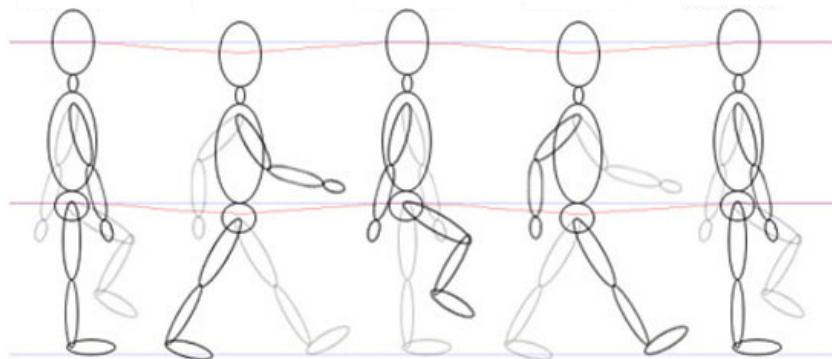
klíčové.

2.7.1 Animace

Pro tvorbu animací jsem vybíral mezi dvěma nástroji, a to Blender a Autodesk Maya. Nakonec jsem zvolil program Autodesk Maya pro rychlejší pochopení základních principů animace a přehlednost uživatelského rozhraní.

2.7.1.1 Kostra

Pro animaci se používá kostra modelu. Kostra fyziologicky kopíruje kostru lidskou nebo kostru jiných živočichů. Kosti a klouby tvoří hierarchii. Aby byla animace realistická a správně fungovala, musí být všechny kosti a kluby napojeny. Tím se docílí takového efektu, že s pohybem jedné kosti se pohybují i kosti k ní přiléhající. Hlavní kost, od které se odvíjí ostatní, je umístěna v oblasti kostrče nebo pánve tam, kde má mít model těžiště. Ostatní kosti jsou s touto hlavní kostí propojeny přímo nebo přes jiné kosti, a tím je tvořena jejich hierarchie. Každá kost má minimálně jeden kloub, kterým je připojena k jiné kosti a platí že kostra by měla být symetrická. Hotová základní kostra je vidět na Obrázku 2.5.



Obrázek 2.6: Mezní pozice chůze člověka

2.7.1.2 Tvorba animace

Při samotné animaci se model pohybuje podle toho, jak se pohybují kosti, stejně jako v lidském těle. Animace závisí na znalosti anatomie těla a je tvořena metodou zvanou „keyframing“, která funguje na definování mezních pozic kostí a program následně dopočítá animaci mezi těmito mezními pozicemi. Je zřejmé, že čím více se vytvoří mezních pozic, tím je animace modelu více pod kontrolou. Hezký se to dá ilustrovat na animaci chůze, která má čtyři mezní pozice viz obrázek 2.6. Pro vytvoření animace stačí animovaný 3D model nastavit, pomocí směrování a posouvání kostí, do těchto pozic, nastavit v jakém časovém odstupu mají pozice být a zbylé snímky animace dopočítá program. Pokud animátor není s dopočítanou animací spokojen, stačí přidat další kontrolovanou pozici a spustit výpočet znova. Tím by se animace měla přiblížit výsledku, který animátor očekává.

2.7.1.3 Typy animací

Pro tento program je potřeba vytvořit několik typů animací, které budou vytvořeny každý použitý model. Mezi tyto typy patří chůze, postávání na místě, běh, útok, obrana nebo obdržení zásahu. Každá z těchto animací musí být vytvořena zvlášť a pojmenována unikátním jménem pro následné použití v programu.

2.8 Použité nástroje a technologie

Jako vývojové prostředí (IDE) jsem zvolil Android Studio (AS), protože je to oficiální IDE poskytované společností Google pro vývoj Android aplikací. AS má inteligentní doplňování a mnoho funkcí zaměřených přímo pro vývoj pro Android (např. emulátory).

2. NÁVRH

Pro rozpoznání markeru jsou použil knihovnu Vuforia a pro zobrazování 3D modelů a animací knihovnu LibGDX, které jsou popsány v kapitole „Analýza“

2.8.1 Verzovací nástroj

Do budoucna bych rád dále v této práci pokračoval a předpokládám, že se zapojí další vývojáři, a proto je již při návrhu nutné přemýšlet o tom, jak se bude kód verzovat a ukládat. I v případě, že bych na této práci pracoval sám, je dobré používat kvalitní verzovací nástroj, a to kvůli přehlednosti verzí. Dva nejpoužívanější nástroje jsou Git a Subversion (SVN). Pro verzování této aplikace jsem nakonec vybral Git, který svými výhodami převyšuje SVN.

2.8.1.1 Rozdíly mezi SVN a Git

Oba nástroje pro správu verzí jsou různé, takže výběr optimálního závisí na potřebách vývojářů a projektu. Oba nástroje jsou zdarma a cena proto nebyla rozhodujícím faktorem. Ve srovnávání jsem se zaměřoval pouze na části nástrojů, které budou použity při vývoji, a proto jsem například neřešil problematiku práce s velkými binárními soubory.

Hlavním rozdílem mezi Git a SVN je, že Git je distribuovaný systém pro správu verzí a SVN je centralizovaný systém. Rozdíl je v tom, že v distribuovaném systému pro správu verzí má každý vývojář svou vlastní lokální kopii se všemi změnami, které v ní byly provedeny, zatímco v centralizovaných systémů jsou všechny změny uloženy na serveru. V tomto ohledu jsem při zvažování výběru volil Git, protože pracuji i na cestách, kde nemám stálé a rychlé připojení k internetu.

Nástroj Git umožňuje jednodušší větvení projektu, které mezi sebou mohou být spojovány nebo být dále větveny. Jednotlivé větve projektu se používají například pro různé nové funkce a přitom původní funkční verze může zůstávat stále ve vlastní větvi nedotčená. Nástroj SVN dovoluje ukládat rozdělanou práci v různých verzích, ale není to optimální nástroj pro projekty, na kterých spolupracují více lidí.

Dalším rozdílem je přístup k projektu. Ve výchozím nastavení, Git předpokládá, že všichni přispěvatelé mají stejná oprávnění. SVN, na druhé straně umožňuje specifikovat práva čist a zapisovat pro jednotlivé soubory a adresáře. To je určitě výhoda u některých projektů, které pracují s tajnými informacemi, které by neměli být pro všechny vývojáře dostupné, ale takové informace v tomto projektu zatím nejsou. [30]

2.8.2 Serverová část

Aby bylo možné porovnávat výsledky jednotlivých hráčů, je nutné přidat do aplikace přihlašování. Nejjednodušší a nejrychlejší způsob, jak přidat tuto možnost a mít na serveru i malou databázi, je s použitím Firebase. Firebase se skládá ze tří produktů: databáze Firebase, statického hostingu a autorizační

infrastruktury pro přístup k databázi. Výhodou je, že základní funkcionalita je dostupná zdarma. Základní funkcionalita zahrnuje uložiště o velikosti 100 MB a přenos 5 GB měsíčně. To pro tuto aplikaci naprosto stačí, protože data, která budou na serveru uchovávána budou jen textové informace o uživateli.[31]

2.9 Komunikace mezi zařízeními

Pro hru dvou hráčů je potřeba navrhnout a implementovat řešení, které umožní zařízením obou hráčům spolu komunikovat. Díky komunikaci mohou mezi sebou sdílet akce, které provádějí, a tím aktualizovat svůj model na soupeřově zařízení. Pro tento účel se nejvíce hodí dvě řešení. První řešení je použití technologie Bluetooth a druhé řešení je použití služby Nearby Connections od společnosti Google. Implementovat herní systém přes internet jsem zavrhl, protože jsem nechtěl omezovat uživatele nutností být připojen k internetu.

Technologie Bluetooth je výhodný způsob propojení dvou zařízení, která jsou v blízkosti. Vzdálenost obou zařízení bude vždy minimální, protože hra se bude odehrávat na jednom místě a pokud by se uživatel se svým zařízením vzdálil dál od markeru, modely by se přestaly vykreslovat. Díky tomu mohu předpokládat, že obě zařízení zůstanou v blízkosti a připojení by tím mělo být stabilní. Problémy s bluetooth mohou nastat v latenci komunikace, a tím související synchronizace akcí obou hráčů. Tento problém se pokusím vyřešit implementováním algoritmu, který bude minimalizovat problémy s latencí tím, že bude předvídat následující kroky modelu. Výhodou použití Bluetooth je hlavně to, že není nutné implementovat žádný centrální server, a tím se zjednoduší implementace a údržba systému. Další velkou výhodou je fakt, že bluetooth je multiplatformní technologie a pokud bych se rozhodl implementovat aplikace pro iOS, neměl by být s komunikací větší problém.

V roce 2015 Google představil novou službu pro vývojáře jménem Nearby Connections, která umožňuje objevovat zařízení připojená na stejně Wi-Fi síti. Následná komunikace mezi zařízeními je rychlejší než přes Bluetooth a pro samotný zážitek ze hry je tento způsob propojení ideálnější. Pokud se však budou uživatelé nacházet na místě, kde se nemohou připojit na stejnou Wi-Fi síť, jsou odkázáni na použití Bluetooth.

2.9.1 Topologie propojení

Pro komunikaci mezi zařízeními je možné použít různé druhy topologií. Protože tato hra uvažuje pouze hru jednoho nebo dvou hráčů, zvolil jsem topologii Peer-to-Peer (P2P), která je pro tento případ nejjednodušší a nejhodnější. Topologie Client-server při připojení dvou hráčů, je velmi podobná Peer-to-Peer, ale pro nutnost implementovat dva rozdílné způsoby komunikace jsem ho nezvolil. Navíc topologie Peer-to-Peer má tu výhodu, že oba uživatelé mají všechna potřebná data, protože obě zařízení fungují zároveň jako klient i server.

KAPITOLA 3

Realizace

V této kapitole se budu věnovat částem aplikace, které byly obtížně implementovatelné nebo jsou z hlediska implementace a samotné realizace něčím zajímavé. Pokusím se také nastínit, jak jsem postupoval při realizaci těchto částí. Nejprve se zaměřím na strukturu celého projektu a následně na jeho implementaci. V rámci realizace se taky budu věnovat tvorbě herních karet.

3.1 Struktura projektu

Zjednodušeně se dá projekt Android aplikace rozdělit na 2 části. Část obsahující implementaci logiky aplikace, která je tvořena Java soubory, a část obsahující zdroje potřebné pro část obsahující logiku aplikace.

3.1.1 Logická část

Tato část obsahuje zdrojové kódy psané v jazyce Java, které jsem rozdělil do jednotlivých balíků. Každý balík obstarává jinou logiku aplikace. Zvolil jsem následující rozdělení, kterým jsem dosáhl oddělení částí, které spolu nesouvisí a kód je tím strukturován a je přehlednější:

ar Rozšířená realita: Zde jsou třídy pro rozpoznání markeru a vykreslení 3D modelu.

interactors Práce s pamětí a komunikace se serverem: Interaktory jsou třídy, které komunikují nebo pracují vždy s jedním zdrojem. V této aplikaci jsou tedy interaktory pro správu paměti a pro komunikaci se serverem. Je důležité, aby se funkčnost jednotlivých interaktorů neprolínala. Tím umožníme jednoduchou nahradu a jednoduší testování. Každému interaktoru se při testování mohou předkládat různá data a aplikace tak zůstává snadno testovatelná.

3. REALIZACE

model **Datový model aplikace:** Zde se nacházejí objekty, se kterými se v aplikaci pracuje. V tomto balíku se nachází jsou objekty nesoucí informace o modelu, které udržují synchronizovanost obou zařízení při hře dvou hráčů.

mvp **Model-View-Presenter architektura:** Tento balík je rozdělen na dva. Jeden obsahuje presentery a druhý rozhraní, přes které presentery komunikují s views. Presentery jsou třídy, které se starají o běh aplikace a s uživatelem komunikují přes view, které je zrovna na obrazovce. V části view je definováno rozhraní, přes které komunikují presentery s fragmenty (a tím s uživatelem) a naopak.

services **Služby provádějící se na pozadí aplikace:** Service (služba) se provádí na pozadí aplikace a používá se pokud je potřeba vykonávat velké množství práce. Může se také provádět, i když uživatel uspal zařízení nebo je v jiné aplikaci. V této aplikaci je service pro práci s bluetooth pro komunikaci mezi zařízeními.

ui **User Interface:** Zde jsou Activity a Fragmenty, které slouží pro komunikaci s uživatelem

utils **Pomocné třídy:** Zde pomocné třídy s obecnými metodami dělené podle použití. Například je zde třída na práci s texturami nebo na práci s obrazovkou.

Závislosti jednotlivých balíků se dají popsat následovně. Základní prvek je Activita, ta obsahuje jeden nebo více Fragmentů. Každý Fragment má k sobě napárovaný Presenter, kterému jsou přes Fragment předávány všechny uživatelské vstupy. Fragment pouze zobrazuje prvky uživatelského rozhraní a pomocí něj probíhá veškerá interakce s uživatelem.

Presentery obsahuje instance všech Interactorů, které potřebuje. Pokud se chce uživatel například přihlásit, tak Fragment dá vědět Presenteru, Presenter pověří interactor, aby uživatele přihlásil a o výsledku informuje uživatele přes napárovaný Fragment. Dále presenter používá třídy z balíku „utils“ a třídy z balíku „model“ pro úkony, které jsou potřeba udělat.

3.1.2 Zdroje

Zdroje obsahují soubory, které jsou používány logickou částí. Nejčastěji se jedná o rozložení prvků na obrazovce nebo obrázky, které se aplikaci zobrazují. Následující rozdělení znázorňuje strukturu zdrojů v této aplikaci:

anim Složka s animacemi.

drawable Složka s grafikou využitou v aplikaci, například obrázky a ikony.

layout Složka s rozložením jednotlivých grafických prvků na obrazovce.

menu Definování menu, která jsou v aplikaci. Zpravidla se zobrazují v pravém horním rohu jednotlivých obrazovek.

values Konstanty využívané v aplikaci. Jedná se například o texty a barvy.

Balík „values“ obsahuje XML soubory stejně jako balíky „anim“, „layout“ a „menu“. Na rozdíl od nich zde nejsou důležité názvy souborů. Avšak držel jsem se při programování konvence a soubor s textovými konstantami jsem pojmenoval „strings.xml“, s barvami „colors.xml“ atd.. Tato aplikace je psaná v českém jazyce, ale pokud by bylo potřeba ji přeložit do dalších jazyků, vytvoří se v tomto balíku pouze soubor pro další jazyk. Následně se překládají pouze texty z původního „strings.xml“ souboru a nemusejí se jednotlivé konstanty hledat po celém projektu. Toto je sice všeobecný způsob překladů, je však dobré na to myslit a být v tom důsledný.

3.2 Práva aplikace

V této podkapitole popíši práva přidělená aplikaci, která jsou potřebná pro její plnou funkčnost. Všechna práva se zobrazují uživateli před samotnou instalací a ten se poté může rozhodnout, jestli aplikaci nainstaluje. Proto je potřeba nepřidělovat aplikace práva, která nevyužije, a tím zvyšovat šanci, že si někdo instalaci rozmyslí.

INTERNET Klíčové povolení, bez kterého by aplikace nemohla komunikovat přes internet. To by znemožnilo komunikaci s jinými zařízeními (multiplayer) anebo přihlášení ke svému účtu na serveru.

CAMERA Další klíčové povolení, bez kterého by samotná hra nemohla fungovat. Hra využívá fotoaparát a obraz z něj zobrazuje na obrazovce společně s rozšířenou realitou

AUTOFOCUS Toto povolení zajišťuje správné zaostření fotoaparátu. Pokud by fotoaparát nemohl správně zaostřit, marker by pravděpodobně nebyl rozeznán a modely rozšířené reality by se ve hře nezobrazily.

BLUETOOTH Povolení pro použití bluetooth, při které může probíhat hra dvou hráčů.

3.3 Rozšířená realita

Zde budu popisovat zajímavosti a problémy, které jsem řešil při vytváření rozšířené reality. V rámci tohoto problému byly použity dvě knihovny. První z nich je „Vuforia“, která v rámci této aplikace rozpoznává marker a druhá je „LibGDX“, která vykresluje samotný model.

3. REALIZACE

3.3.1 Rozpoznání markeru

O rozpoznání markeru na snímcích kamery se stará knihovna „Vuforia“, která nemá svůj kód volně ke stažení a prozkoumání. Proto jsem nemohl prozkoumat, jak přesně funguje, ale z dokumentace lze vyčíst, že si hledaný marker přetrasformuje a vyznačí si určité body, které následně hledá na snímku. Pokud tyto body nalezne, zjistí jejich orientaci vůči snímači, kterou následně předá do aplikace. Aplikace se tedy pouze táže knihovny, jestli, případně kde a v jaké jsou poloze vůči snímači se vyskytují nějaké hledané markery. Tato knihovna pracuje spolehlivě a rychle, proto jsem se jí rozhodl využít místo vlastního naprogramování.

3.4 Vykreslování rozšířené reality

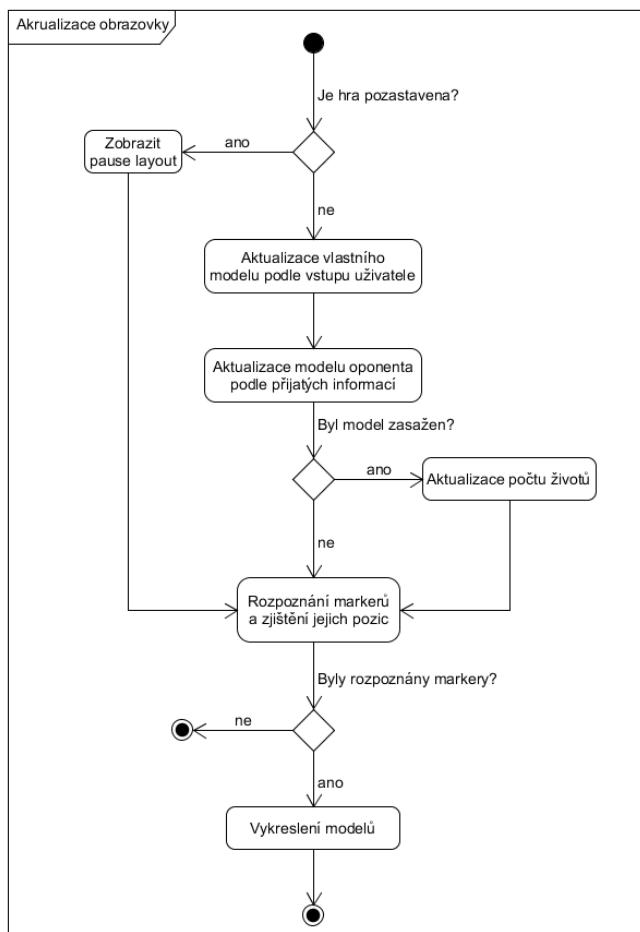
V rámci aplikace jsem implementoval automatickou aktualizaci modelu, při každém vykreslení obrazovky. Na obrázku 3.1 je diagram aktivit znázorňující aktualizaci virtuálních modelů při vykreslení obrazovky snímané kamerou. Na následujících řádcích tento diagram slovně popíši.

Nejprve algoritmus zkонтroluje, jestli jeden z hráčů nepozastavil hru a pokud ano, program vykreslí přes obrazovku překryvnou vrstvu, která značí pozastavenou hru. Pokud je hra pozastavena, není možno pohybovat s modely, nebude tedy probíhat přepočítávání jejich pozic. Pokud hra pozastavena není, program aktualizuje atributy modelu podle vstupu uživatele. Tato aktualizace vlastního modelu je znázorněna na obrázku 3.1, kde je algoritmus detailněji popsán v diagramu. Po aktualizaci vlastního modelu proběhne aktualizace modelu soupeře, podle přijatých informací, předávání informací mezi zařízeními bude popsáno níže v kapitole „Komunikace mezi zařízeními“. Pokud jsou oba modely aktualizovány proběhne výpočet, jestli vlastní model obdržel zásah a aktualizuje počet svých životů. Na závěr je potřeba zjistit, jestli na obrazovce jsou nějaké markery, které by sloužily jako orientační body pro virtuální modely.

3.5 Aktualizace modelu

V programu jsou dva způsoby, jak probíhá aktualizace 3D modelů. První způsob se řídí vstupem uživatele a druhý přijatými informacemi od druhého hráče. Druhý způsob pouze nasetuje přijaté atributy do instance soupeřova modelu. První způsob názorně popisují na obrázku 3.2, který znázorňuje diagram aktivit pro aktualizaci modelu podle vstupu uživatele. Dále tento diagram popíší slovně.

Aktualizaci modelu nejprve provádí podle akcí, které uživatel může zvolit (útok nebo obrana). Zde mohou nastat různé situace závisející na aktuálním vstupu od uživatele a jeho na předchozí volbě. Zde se kontrolují tři stavů



Obrázek 3.1: Diagram: Vykreslování rozšířené reality

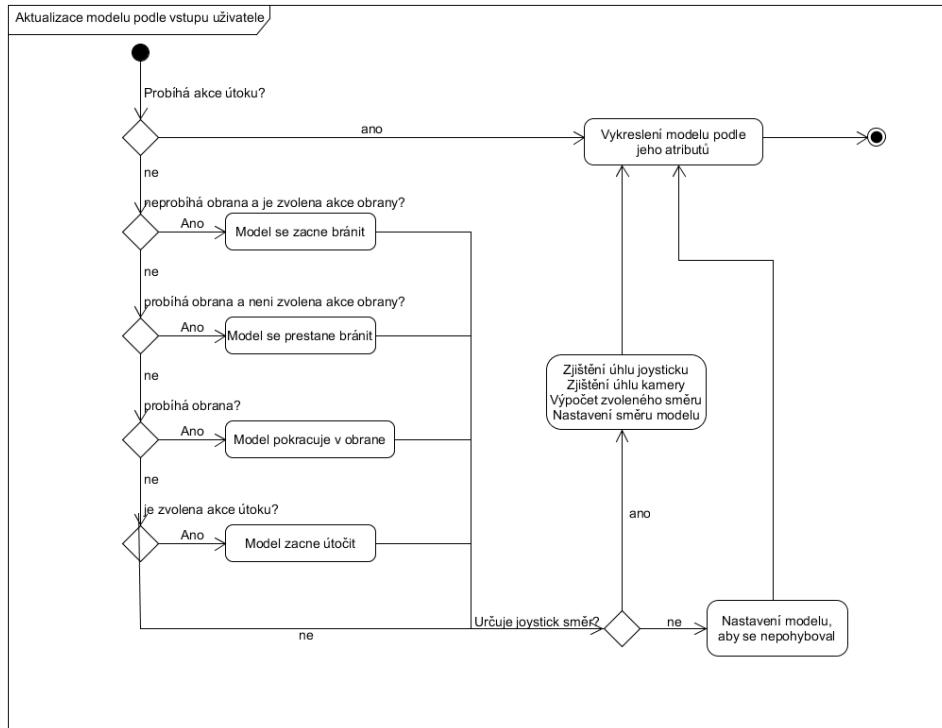
obrany (začátek, průběh a konec) a stav útoku. Tyto podmínky určují typ animace modelu a stav jeho atributů akcí.

Po aktualizaci stavu podle akcí se provádí aktualizace stavu podle joysticku. To obnáší vypočtu pozice a směru modelu. Tomu věnuji celou následující podkapitolu „Pohyb modelu“. Pokud joystick neurčuje žádný směr ani sílu pohybu, model se přestane pohybovat.

3.6 Pohyb modelu

V kapitole „Návrh“ jsem diskutoval možné přístupy k ovládání modelu pomocí joysticku. V této podkapitole popíši implementaci primární varianty ovládání modelu pomocí joysticku.

3. REALIZACE



Obrázek 3.2: Diagram: Aktualizace modelu podle vstupu

Joystick jsem nakonec umístil do levé dolní části, protože to bývá standardem u her obsahující tento typ ovládání. Joystick je naprogramován tak, že na dotaz vrátí úhel, který svírá s úhlem 0° na jednotkové kružnici. Vrací také údaj o síle, se kterou se tento úhel svírá, a ta se vypočítává vzdáleností „páčky“ joysticku od středu.

Protože pozice snímacího zařízení se vůči markeru stále mění, musí se tento úhel přepočítat, aby byl určen správný směr, kterým se má model směřovat. Pokaždé když aktualizuji pozici modelu, tak se zeptám instance joysticku, jestli a případně v jakém úhlu uživatel joystick drží. Z tohoto úhlu vytvořím 2D vektor svírající tento úhel na jednotkové kružnici pomocné vzorečku.

$$joystickVector = \begin{pmatrix} \cos \alpha & \sin \alpha \end{pmatrix}$$

Stejně se zeptám kamery v jakém úhlu je vůči markeru a tyto dva úhly sečtu. Tím získám směrový vektor, který ukazuje směr, kam bude směřovat model.

Rychlosť pohybu závisí na síle, kterou uživatel pomocí joysticku zadává. Maximální rychlosť proto násobíme silou, která je v intervalu $(0;1>$.

3.6.1 Predikce polohy modelu

Model druhého hráče a jeho pohyb je závislý na kvalitě propojení zařízení. Algoritmem pro predikci chování modelu se snažím minimalizovat problémy s částečným přerušením nebo pozdržením spojení. Tento algoritmus pracuje s předpočítáním předpokládaného cíle, kam se chce model dostat. Tento předpokládaný cíl se přepočítává při každé aktualizaci modelu, podle vstupu uživatele. To znamená, že pokud uživatel směruje joystick opačným směrem, než směruje model, algoritmus předpokládá, že se model bude otáčet a nastaví předpokládaný cíl za jeho zadní část.

Pokud se během přerušení spojení odchylí model od reálné pozice, při prvním přijetí nových informací se pozice modelu aktualizuje. Díky předpočítávání cíle „skoky“ modelu, zapříčiněné výpadkem spojení, měly být minimální.

3.7 Kolize modelů

Při plánování pohybů nebo při rozhodování o akcích útoku a obrany, je důležité zjistit, zda existují kolize s jinými objekty v herním světě. Je tedy potřeba se s kolizemi správně a v souladu s potřebami hry vypořádat.

Kolizi objektů jsem pro jednoduchost řešil pouze z 2D roviny. Každý objekt má svou pozici, směr a velikost. Díky těmto atributům je možné jednotlivé objekty vykreslit na obrazovce ve správné pozici vůči hracímu prostoru. Aby se nestalo, že modely budou přes sebe procházet, musí hra detekovat, jestli se modely nepřekrývají. To se kontroluje při aktualizaci pozic jednotlivých modelů. Při výpočtu posunu každého modelu program kontroluje, jestli se nedostala do okolí jiného modelu, které představuje jeho velikost. Pokud ano, tento posun modelu se nevykoná. Pokud se jedná o útok jednoho modelu proti druhému, tak se tato kolize zanese a aktualizuje se počet životů zasaženého modelu.

Kolize je počítána jednoduchým vzorcem pro vzdálenost mezi dvěma body v rovině podle pozic obou modelů na ose x a ose y.

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Tuto vypočtenou vzdálenost poměruji se součtem velikostí obou modelů a pokud je tento součet větší než vzdálenost mezi nimi, tak nastala kolize a program jí musí vyřešit.

Kolize nastávají ve čtyřech případech:

1. **Kolize pokémonů:** Pokud do sebe při souboji narazí samotní pokémoni pouze pohybem (bez útoku), vyřeší se kolize jednoduchým neprovedením aktualizace pozice, vedoucí ke kolizi.
2. **Kolize při útoku:** Pokud jeden pokémon zaútočí a při útoku nastane kolize, znamená to, že druhý pokémon je zasažen a musí aktualizovat svůj počet aktuálních životů.

3. REALIZACE

3. **Kolize s projektilem:** Někteří pokémoni mají možnost svým útokem vystřelit projektil. Pokud tento projektil zasáhne soupeře, dá soupeřovi zranění a ten si aktualizuje počet svých životů.
4. **Kolize s okrajem:** Aby jsem předešel situacím, že jeden hráč utíká před druhým co nejdál, ohraničil jsem herní plochu určitou vzdáleností, za kterou se modely nemohou dostat. Pokud tedy nastane kolize s okrajem hrací plochy, model bude nebude moci dále pokračovat stejně jako v bodě 1.

3.8 Komunikace mezi zařízeními

Hra dvou hráčů se neobejde bez propojení jejich zařízení. Propojit zařízení bude umožněno samotnou hrou, a to pomocí technologie Bluetooth, která umožňuje bezdrátovou výměnu dat mezi jednotlivými zařízeními. Tomuto způsobu se budu věnovat v rámci této kapitoly. Dále se budu věnovat samotné komunikaci, posílaným informacím a jejich struktuře.

3.8.1 Bluetooth

Platforma Android obsahuje podporu pro technologii Bluetooth, které umožňuje zařízení bezdrátově vyměňovat data s jinými zařízeními, která též mají Bluetooth. OS Android poskytuje API Bluetooth, které může provádět různé operace od vyhledávat ostatních zařízení, přes získání spárovaných zařízení, až po samotné posílání témeř jakýchkoliv zpráv. Bluetooth technologie má malé požadavky na výkon a OS Android od verze 4.3 navíc zavedl podporu pro „Bluetooth Low Energy“, což přidává i nízké požadavky na baterii zařízení.

Aby mohla zařízení přenášet data mezi sebou, musí se nejprve vytvořit komunikační kanál. Jedno zařízení musí být detekovatelné a být k dispozici pro příchozí požadavky na připojení. Druhé zařízení najde to zjistitelné, vymění si navzájem bezpečnostní klíče a poté jsou již oba přístroje připravené si vyměňovat informace. Po dokončení přenosu informací, přístroj, který inicioval žádost o spárování, uvolní datový kanál, čímž ukončí spojení.

Pokud uživatel vyhledává zařízení, se kterým není spárovaný, musí vyhledat všechna okolní zařízení a vybrat ze všech zjistitelných to, se kterým se chce propojit. Pokud je druhé zařízení v režimu viditelnosti a potvrdí propojení zařízení, tak zařízení, které vyvolalo žádost o propojení se díky informaci o jedinečné MAC adrese dokáže k tomuto zařízení připojit. Proces zjištování obvykle probíhá asi 12 sekund, po kterých se prozkoumává každé zařízení nalezené během předešlého objevování. Tímto zkoumáním se zjišťuje název a unikátní MAC adresa těchto zařízení.

Aby bylo možné vytvořit spojení mezi dvěma zařízeními, musí být implementován, jak mechanismus, co vyvolá spojení (server), tak mechanismus co

spojení přijme (klient), protože jedno zařízení musí otevřít socket a druhé musí iniciovat připojení pomocí MAC adresy druhého zařízení.

Zařízení jsou považována za vzájemně připojená, pokud obě mají otevřený BluetoothSocket na stejném kanálu. V tu chvíli mohou obě zařízení začít komunikovat.

Po propojení zařízení je již práce s Bluetooth velmi jednoduchá. Pomocí metod „getInputStream()“ a „getOutputStream()“ se získají kanály do kterých je možno zapisovat nebo z nich číst.

3.8.2 Problémy

Hlavním problémem her na více zařízeních je jejich synchronizace. Kvůli nespolehlivosti sítě nebo případné latenci komunikace je potřeba vymyslet takové řešení, které se bude s tímto problémem vypořádávat s co nejvyšší úspěšností a co nejrychleji, aby byla hra plynulá. Je tedy potřeba vyřešit všechny případy přerušení spojení, protože u mobilních připojení není jejich pravděpodobnost zanedbatelná. Zadruhé je důležité rozmyslet s jakou frekvencí a v jaké struktuře se budou data s informacemi posílat do druhého zařízení. Je dobré minimalizovat velikost posílaných dat, aby komunikační kanál nebyl přehlcen a mohl spolehlivě a rychle pracovat.

3.8.3 Realtime komunikace

Datová komunikace probíhá přímo mezi hráči, tedy využívá se Peer-to-peer (P2P) komunikační síť viz kapitola „Komunikace mezi zařízeními“ v části „Návrh“. Proto je potřeba aby oba hráči měli všechny potřebné informace pro zobrazení obou modelů a jejich interakce. Protože hra využívá P2P komunikaci, neobsahuje subjekt, který by kontroloval stav hry, každý hráč musí kontrolovat svůj vlastní a posílat veškeré změny druhému hráči. Díky tomu hráč vidí dvě linie najednou. Jeho model se pohybuje podle jeho vstupu a model druhého hráče podle dat, která od něj přijdou.

Pohyb a akce modelu hráče na aktuálním zařízení jsou vidět téměř okamžitě, zatímco pro pohyb a akce druhého modelu musí hráč obdržet síťovou zprávu s informacemi, kde se nachází tento model a v jakém je stavu. V dokonalém světě by nenastávala latence sítě, takže by se zprávy přijímaly a posílaly okamžitě a simulace by byly velmi přesné. Jak se zvyšuje latence, tak se hra zobrazuje více nepřesně. Pokud toto nastane, model se bude pohybovat přerušovaně a jeho animace nebude plynulá.

Aby se nestávalo, že při vynechání nebo opoždění zprávy, že bude soupeřův model nehybně stát na místě nebo se bude pohybovat trhaně, herní engine vypočítá podle poslední polohy, směru a rychlosti budoucí akci a model aktualizuje i bez obdržené zprávy z druhého zařízení. Při každé doručené zprávě se tento odhadnutý stav aktualizuje a stav modelů jsou tak synchronizované.

3. REALIZACE

Tomuto predikování budoucí akce jsem se věnoval již v podkapitole „Predikce polohy modelu“.

3.8.3.1 Struktura dat

Požadavky na strukturu posílaných dat jsou: musí obsahovat všechny nezbytné informace o stavu modelu, které jsou potřeba pro vykreslení v druhém zařízení a zároveň musí být velikost dat co nejmenší pro minimalizaci vytížení komunikačního kanálu. Struktura posílaných dat je následující:

Pozice modelu Pozice modelu vůči markeru hráče, který hru vytvořil. V rámci pozice se posílají souřadnice X a Y, které jsou před odesláním validovány na kolizi modelů a opuštění herní zóny.

Směr modelu Směr modelu vůči markeru hráče, který hru vytvořil. Tento směr je vyjádřen pomocí úhlu z intervalu <0 ; 360)

Akce Akce, kterou model provádí (útok, obrana nebo pohyb)

Počet životů Aktuální počet životů

3.8.3.2 Zpracování přijatých informací

Každý hráč má během probíhající hry v aplikaci instance všech modelů, které se ve hře vyskytují (pokémonů a projektilů). Po přijetí zprávy o soupeřově modelu nasetuje získané informace do již vytvořené instance a při vykreslování dalších snímků jsou modely již aktualizované.

3.8.3.3 Rozhodnutí o zásahu

Každý model si sám rozhoduje o tom, jestli byl protivníkem zasažen. Při každé komunikaci mezi zařízeními se posílá pozice a směr modelu a akce, kterou provádí. To jsou informace potřebné k tomu, aby model zjistil, zda byl nebo bude zasažen. To je kontrolované před každou aktualizací pozice modelu.

Při kontrole obdržení zásahu se zjistí akce, kterou druhý model provádí. Pokud se jedná o útok, kontroluje se, jestli je soupeřův model nasměrován směrem k mému modelu a jestli je ve vzdálenosti, kterou dosáhne jeho útok. Pokud jsou všechny tyto podmínky splněny, model dostal zásah a aktualizuje si počet životů, které mu zbývají. Tento aktualizovaný stav posílá následně druhému zařízení v rámci dalších aktualizací svého modelu.

3.9 Server

Firebase nedokáže nahradit plnohodnotný backend, ale pro potřeby této aplikace je dostačující a optimální. Interakce aplikace se serverem probíhá pouze, pokud se uživatel registruje, přihlašuje, mění osobní údaje nebo se aktualizuje

jeho pozice v žebříčku hráčů. Obecně se mi díky dobré dokumentaci pracovalo s online databází Firebase dobře a níže nastním zajímavé problémy, které jsem v rámci této sekce dělal.

3.9.1 Struktura dat

Byla potřeba navrhnout struktura databáze, se kterou by se dalo jednoduše pracovat, byla lehce rozšiřitelná a neobsahovala duplicity. V sekci jménem „Users“ jsou uchováváni všichni registrovaní uživatelé s informacemi o nich. Každý ze seznamu uživatelů v této sekci má nastaveno jméno, přezdívku, email, kterým se zároveň přihlašuje do aplikace a aktuální skóre, které určuje jeho pozici v žebříčku nejlepších hráčů. Základní údaje o uživateli jsou upravovány buď při samotné registraci nebo následně v sekci nastavení. Níže je vidět struktura dat obsahující informace o uživatelích. Každý uživatel má unikátní identifikátor, který mu je přidělen při registraci a je neměnný.

```

1  {
2      "users" : {
3          "8xloVEej6dXsCAzyZulXNX00Lj72" : {
4              "email" : "pinadaniel192@gmail.com",
5              "name" : "Daniel Pina",
6              "username" : "pinadaniel192"
7              "score" : 10
8          },
9          "Xsp6dSRWNyZYBww49BPzLBID07I2" : { ... },
10         "u4K5mma1XwfP7qrVhcz0aEpsnKV2" : { ... }
11     }
12 }
```

Skóre uživatele se aktualizuje po každé hře, při které byl přihlášen a byl připojen k internetu. Pro zobrazení žebříčku nejlepších hráčů provedu dotaz na uživatele, které zároveň seřadím podle atributu „score“ sestupně. Tím dostanu seznam všech registrovaných hráčů seřazených podle doposud nahraného skóre.

3.9.2 Práva zápisu

Jak zajistit, aby uživatel mohl upravovat pouze své záznamy? Co by se stalo, pokud by zjistil id jiného uživatele? I toto jsem v aplikaci řešil. Je potřeba omezit práva uživatele natolik, aby mohl zapisovat data pouze do své složky v sekci „users“. Pokud by tomu nebylo zabráněno, mohl by útočník upravovat údaje hráčů a tím je poškozovat nebo naopak přidávat skóre, které sami neahráli. Dále musejí být práva nastavena tak, aby nebylo možno data mazat, a to ani ve své vlastní složce. To by mohlo vést k nekonzistenci dat. Níže je implementace pravidla pro Firebase databázi, které uděluje přístup pro zápis do /users/<uid> pouze ověřeným uživatelům a kde <uid> odpovídá ID

3. REALIZACE

uživatele získané prostřednictvím firebase ověřování. To znamená, že pouze přihlášený uživatel může zapisovat, a to pouze do své vlastní složky.

```
{  
2   "rules": {  
3     "users": {  
4       "$uid": {  
5         ".write": "$uid === auth.uid"  
6       }  
7     }  
8   }  
}
```

3.10 Herní karty

Nejprve jsem musel rozhodnout jakou budou mít karty velikost a zvolil jsem klasické s poměrem 2,5 x 3,5. Design karet jsem se snažil přizpůsobit skutečnosti, že se jedná o hru spojenou s mobilní zařízením. Proto jsem zvolil „Material design“, který se vyskytuje v rámci aplikace. Karty jsem se snažil vytvářet co nejjednodušší, ale zároveň na nich zobrazit všechny důležité informace. Každá karta poskytuje informace o jménu pokémona, jeho typu, počtu jeho životů a síle útoků. Dále je na každé kartě vyobrazen samotný pokémon. Na obrázku 3.1 je vidět výsledný design karty společně s body pro rozpoznávání knihovnou „Vuforia“.

3.10. Herní karty



Obrázek 3.3: Herní karta s rozpoznávanými body

Testování

Tato kapitola se zabývá průběžným a celkovým závěrečným testováním. Pro usability testování jsem vytvořil scénáře, které testují kritické části aplikace. Pro testování UI jsem použil heuristickou analýzu.

Po celou dobu implementace byly jednotlivé části aplikace testovány průběžně převážně na mobilním telefonu Huawei Honor 7 s verzí OS Android 6.1 a úhlopříčkou 5,2 palce a pro zobrazení výstupů testů jsem používal třídu Log, která pracuje s několika typy výstupů.

4.1 Heuristická analýza

Heuristická analýza vyhodnocuje použitelnost počítačového softwaru a pomáhá identifikovat problémy v designu uživatelského rozhraní. Tato analýza spočívá ve srovnání softwaru s pravidly, které vytvořil Jakob Nielsen [32]. Desatero definovaných pravidel zní přibližně takto:

Kolize nastávají ve čtyřech případech:

1. **Viditelnost stavu systemu:** Uživatel musí být informován o tom, co systém dělá.
2. **Shoda mezi systémem a realitou:** Jazyk, kterým aplikace komunikuje s uživatelem by měl být uživateli známý a měl by působit reálně a logicky navazovat.
3. **Minimální zodpovědnost:** Uživatel by měl mít možnost vrátit se do předchozího stavu, ze kterého se omylem dostal stavu současného.
4. **Shoda s použitou platformou a obecnými standardy:** Při návrhu by se mělo postupovat podle konvence platformy. Je težké důležité dodržovat stejná slova a akce pro stejné prvky na různých místech.
5. **Prevence chyb:** Uživatel by neměl být možnost vyvolat v aplikaci chybu nebo do formuláře zadat špatnou hodnotu.

4. TESTOVÁNÍ

6. **Rozpoznání namísto vzpomínání:** Nezatěžovat uživatelovu paměť. Akce, které uživatel může momentálně provést by měly být viditelné a snadno dosažitelné.
7. **Flexibilita a efektivita:** Aplikace by měla počítat jak se zkušeným uživatelem tak i s nezkušeným. Pro zkušeného uživatele by měla nabízet rychlý průchod a pro nezkušeného by měla být dostatečně podrobná.
8. **Minimalita:** Zobrazovat pouze informace, která jsou aktuálně k něčemu dobré.
9. **Smysluplné chybové hlášky:** Všechny systémové hlášky pro uživatele by měli být srozumitelné a v běžném jazyce. Měly by popsat, co se stalo špatně, jak se to stalo a jak tomu příště předejít.
10. **Ná pověda a návody:** Dobrá aplikace by měla být použitelná i bez návodů. Případný návod by měl být přesný, a ne zbytečně dlouhý.

4.2 Testování použitelnosti

Tento typ testování se používá pro vyhodnocení použitelnosti softwaru, při sledování testera při používání aplikace. Testování probíhá na více uživatelích z cílové skupiny, testujících předložené scénáře průchodů a používání softwaru. V průběhu testu je tester sledován člověkem, který má za úkol sepsat všechny vzniklé problémy, nedostatky a případné nepochopení procesů v rámci aplikace. Cílem celého testování je tedy identifikovat problémy s použitelností a navrženým uživatelským rozhraním.

4.2.1 Průběh testování použitelnosti

Nejprve bylo nutné vybrat uživatele pro testování, kteří by měli splňovat podmínu, potenciálních uživatelů konečné aplikace. Vybral jsem pět testerů ve věkovém rozmezí 12–28 let. Dále jsem jim poskytl scénář, podle kterého by se při testování měli řídit. Tento testovací scénář se nalézá v příloze této diplomové práce. Při samotném testování jsem sledoval, jak si při plnění jednotlivých bodů ze scénáře vedou a zapisoval si případné problémy nebo zaváhání, které nastávaly. Po dokončení všech testování jsem provedl analýzu výsledků a sepsal jí do následující podkapitoly „Výsledky testování“.

4.3 Výsledky testování

Během průběžného i závěrečného testování se objevilo více problému, ze kterých jsem vybral ty závažnější a detailněji je zde popíši níže. Chyby, které byly nalezeny se týkaly jak technické stránky, tak použitelnosti.

Z výsledků testování použitelnosti, lze odvodit, že hra má potenciál. Všichni testerovi byli pozitivně překvapeni inovativním přístupem této hry a konstatovali, že si hru alespoň vyzkouší, až bude vydána.

4.3.1 Propojení zařízení pomocí bluetooth

Zařízení jednoho testera mělo problémy s udržením navázaného spojení přes bluetooth. Stávalo se to pravidelně na jednom zařízení s verzí operačního systému 4.4. Tento problém jsem bohužel nebyl schopný vyřešit, ale eviduji ho a v budoucím vývoji, se mu určitě budu věnovat podrobněji. Na jiných zařízeních se stejnou verzí operačního systému se tento problém nevyskytoval.

4.3.2 Vykreslení modelu

Při testování nastala situace, že zařízení přestalo vykreslovat modely pokémonů. Po podrobnějším zkoumání jsem došel k závěru, že se to děje díky pomalému zaostrování snímače kamery. Pokud hráč měnil vzdálenost zařízení od markeru, snímač nestačil zaostřit na marker, a protože algoritmus pro rozpoznávání markeru ho v tu chvíli nerozpoznal, všechny virtuální modely se přestaly vykreslovány. Pokud se nerozpoznal marker, modely nevědí, kde na obrazovce se mají vykreslit.

Tento problém jsem v rámci této práce nedokázal vyřešit. Možným řešením je, naimplementovat svůj vlastní nebo upravit stávající rozpoznávací algoritmus, aby dokázal rozpoznat i lehce rozmazané markery.

4.3.3 Internetové připojení

Pokud uživatel nebyl připojen k internetu a pokusil se změnit nastavení údajů, aplikace zůstala zaseklá v ve stavu načítání aktualizovaného stavu. V tu chvíli aplikace nereagovala na vstupy uživatele a ten ji musel restartovat.

Do balíku „utils“ jsem přidal novou komponentu, který kontroluje stav připojení k internetu. Zároveň s tím jsem do presenteru, který se stará o nastavení hry a aktualizaci hráčova profilu přidal kontrolu před aktualizací, jestli je uživatel připojen a pokud ne, tak uživatele na tuto skutečnost upozorní a dále v aktualizaci nepokračuje.

4.3.4 Odstartování hry

Jednou z častějších a zásadních připomínek byla absence kontrolovaného odstartování hry. Souboj začal vždy ve chvíli, kdy se načetla herní obrazovka a uživatel o tom nebyl informován. Stávalo se tedy, že hráči nezačali hru ve stejný čas, a tím pádem měl jeden hráč výhodu.

Abych tento problém vyřešil, přidal jsem do herní části stav, který představuje hru před samotným zahájením a hráč, který hru vytvářel jí také musí odstartovat. Ve chvíli kdy hráč odstartuje hru, začne na obrazovce odpočet

4. TESTOVÁNÍ

tří vteřin, který značí zbývající čas před začátkem hry. Do skončení odpočtu začíná hra stejně jako původně.

4.3.5 Hra jednoho hráče

Všichni testeri se shodli v názoru, že je potřeba vylepšit umělou inteligenci protivníka ve hře jednoho hráče. Současná umělá inteligence přiřazuje protivníkovi spíše roli cvičného terče pro trénink než důstojného soupeře.

Bohužel jsem do výsledného prototypu nestihl dopracovat lepsí algoritmus pro umělou inteligenci, a proto hra jednoho hráče slouží hlavně pro pochopení herního mechanismu.

Rozšíření

Zde bych rád sepsal možnosti pro rozšíření hry, na které budu určitě pokračovat až do vydání přes obchod „GooglePlay“. Aplikace zatím na publikaci není připravená, ale základní koncept je již vyřešen a jsou potřeba doprogramovat některé funkce, případně udělat designové úpravy. Do následujících podkapitol sepíši, co budu dodlávat před samotným publikováním a případně nastíním funkce, které se budou doprogramovávat po vydání a distribuovat v rámci aktualizací.

5.1 Server

Na serveru dopíši strukturu pro přidávání hráčů do seznamu přátel pro možnost sledovat jiné hráče a jejich pokroky. S tím souvisí možnost posílat mezi přáteli zprávy. Aby funkcionality psaní zpráv zbytečně nezdržovala vydání aplikace, bude dopsána později. K funkcionalitě zpráv bych rád přidal i možnost push notifikací, pro vyzvání jiného hráče ke hře na určeném místem a s časem souboje. Do struktury hráče bych rád přidal získávání levelů a případných odměn.

5.2 Aplikace

Po vydání aplikace a zjištění zájmu ze strany uživatelů, bych rád rozšířil Android aplikaci o aplikaci na iOS.

5.3 Design mobilní aplikace

Před vydáním aplikace bych rád konzultoval design mobilní aplikace s odborníkem, který by udělal návrh designu aplikace odpovídající jejímu typu, účelu a cílové skupině uživatelů. V současném stavu je design aplikace navržen velmi primitivně a pro konkurenční schopnost hry je nutno ho aktualizovat.

5. ROZŠÍŘENÍ

5.4 3D modely

Aktuální hra obsahuje pouze dva 3D modely. Rád bych před vydáním měl v aplikaci alespoň 15 typů 3D modelů.

5.5 Komunikace zařízení

Rád bych také ke komunikaci pomocí technologie bluetooth přidal i možnost komunikace zařízení přes Wi-Fi. To by měl umožnit koncept „Nearby Connections“ poskytovaný společností Google. Komunikace pře Wi-Fi, by měla za následek možnost rychlejší komunikace a snížení latence při posílání.

5.6 Hra jednoho hráče

Z testování vyplynulo, že umělá inteligence protivníka ve hře je potřeba vylepsit, aby hra jednoho hráče nebyla jen pro pochopení herních mechanismů, ale byla hratelnou součástí kompletní aplikace. Rád bych naprogramoval více úrovní obtížnosti, mezi kterými by si mohl uživatel vybírat.

5.7 Audio

Mezi možnosti rozšíření bych zařadil přidání zvukových stop k jednotlivým akcím (útok, prohra, výhra, atd.). Tato funkce by pravděpodobně vylepšila herní zážitek, proto by v možnostech rozšíření neměla chybět.

Závěr

Cílem této diplomové práce bylo navrhnut a implementovat aplikaci pro operační systém Android, která využívá technologie rozšířené reality. Aplikace měla kombinovat mobilní a karetní hru, kde ta mobilní závisí na fyzických kartách, se kterými se hraje. Výsledkem je funkční a stabilní prototyp aplikace, který bude dopracován a umístěn na oficiální obchod Google Play. Aplikace splňuje všechny požadavky, které na ní byly kladené. Tuto práci jsem zpracoval tak, aby mohla být do budoucna užitečná pro studenty, kteří se tomuto tématu budou chtít věnovat.

V první kapitole jsem se věnoval analýze technologie rozšířené reality, operačního systému Android a podobným aplikacím. Z této analýzy jsem určil požadavky a technologie, které jsem následně použil při realizaci aplikace, která byla tvořena v rámci této diplomové práce.

V další kapitole byl vytvořen návrh, který popisuje aplikaci, funkčnost jejích komponent a vzhled. V rámci této kapitoly jsem navrhl architekturu aplikace a učinil důležitá rozhodnutí spojená s následnou realizací.

Kapitola „Realizace“ se zabývá samotnou implementací prototypu aplikace a detailněji popisuje implementaci zajímavých částí a funkcí. Vytvořený prototyp byl následně podroben testování a chyby, které byly nalezeny, byly posléze opraveny.

Výsledná aplikace není v současné době hotová, aby mohla být publikována. Vytváří ale základ, na němž bude vystavěna velká herní aplikace, která bude mít ambice prorazit nejen na českém, ale i světovém herním trhu. Implementovaný prototyp nevykazuje žádné kritické chyby a byl testery pozitivně oceněn jako velmi inovativní a do budoucna perspektivní. Tento typ mobilní hry je novinkou v herním průmyslu a do budoucna bude mít určitě velký potenciál.

I když jsem s těmito technologiemi neměl předchozí zkušenosti a během implementace se vyskytovalo velké množství problému, tak jsem rád že jsem se mohl tomuto tématu věnovat a prozkoumat ho podrobněji. Tato diplomová práce byla pro mě velmi poučná a doufám, že aplikace, která v rámci ní vznikla

ZÁVĚR

dá základ vytvoření produktu, který bude mít úspěch a bude své uživatele bavit.

Literatura

- [1] Kao, X.: Virtual Lane Graphics for Swimming with CSS3 [online]. 2013, [Citováno 2017-03-04]. Dostupné z: <http://xy-kao.com/projects/virtual-graphics-for-swimming/>
- [2] Image evaluation [online]. 2017, [Citováno 2017-04-15]. Dostupné z: <https://library.vuforia.com/articles/Solution/Image-Target-Enhancement-Grayscale-Histogram-Quality-Indicator>
- [3] IDC: Poměr OS v prodaných zařízeních [online]. 2016, [Citováno 2017-03-03]. Dostupné z: <http://www.idc.com/promo/smartphone-market-share/os>
- [4] Poměr verzí OS Android [online]. [Citováno 2015-04-12]. Dostupné z: <http://developer.android.com/about/dashboards/index.html>
- [5] TECH. - Implementing augmented reality [online]. [Citováno 2017-03-22]. Dostupné z: <http://www.techinsight.io/review/mobile-app-development/implementing-augmented-reality/>
- [6] Azuma, R. T.: A Survey of Augmented Reality [online]. 1997, [Citováno 2016-11-11]. Dostupné z: <http://www.cs.unc.edu/~azuma/ARpresence.pdf>
- [7] Wagner, D. H.: Augmented Reality [online]. 2007, [Citováno 2016-11-10]. Dostupné z: http://studierstube.icg.tugraz.at/thesis/Wagner_PhDthesis_final.pdf
- [8] Sung, D.: The history of augmented reality [online]. 2011, [Citováno 2016-12-18]. Dostupné z: <http://www.pocket-lint.com/news/108888-thehistory-of-augmented-reality>
- [9] Anh, N. T. N.: Augmented reality [online]. 2014, [Citováno 2016-12-18]. Dostupné z: <http://ar-hcmut.weebly.com/history.html>

LITERATURA

- [10] Sung, D.: The history of augmented reality [online]. 2011, [Citováno 2016-12-18]. Dostupné z: <http://www.pocket-lint.com/news/108888-the-history-of-augmented-reality>
- [11] Mobile Phone, Smartphone Usage [online]. 2016, [Citováno 2017-04-15]. Dostupné z: <https://www.emarketer.com/Article/Mobile-Phone-Smartphone-Usage-Varies-Globally/1014738>
- [12] Gardner, J. A.: Augmented Reality Training Systems Destined for Marine Schoolhouses [online]. 2016, [Citováno 2017-03-04]. Dostupné z: <http://www.military.com/daily-news/2016/03/09/augmented-reality-training-systems-destined-marine-schoolhouses.html>
- [13] Gardner, J. A.: Augmented and virtual reality in medicine - Medtech Boston [online]. 2016, [Citováno 2017-03-04]. Dostupné z: <https://medtechboston.medstro.com/blog/2016/05/24/16045/>
- [14] Vince Bui, A. M. R. P., Sing Chen: Interactive Augmented Reality Card Game [online]. 2000, [Citováno 2016-11-23]. Dostupné z: <http://www.singjchen.com/InteractiveAugmentedRealityCardGame.pdf>
- [15] Matěj Konečný: Jak vypadá Android uvnitř - Android Market [online]. [Citováno 2017-03-03]. Dostupné z: <http://www.zdrojak.cz/clanky/vyvijime-pro-android-zaciname/>
- [16] Pína, D.: *Bakalářská práce - Daniel Pína: Průvodce programováním mobilních aplikací*. Praha, 2015.
- [17] tutorialspoint: Android Architecture [online]. [Citováno 2016-12-01]. Dostupné z: http://www.tutorialspoint.com/android/android_architecture.htm
- [18] Jamal Eason: Android Studio 2.3 | Android Developers Blog [online]. 2017, [Citováno 2017-03-03]. Dostupné z: <https://android-developers.googleblog.com/2017/03/android-studio-2-3.html>
- [19] TutorialsPoint: Java - Overview [online]. [Citováno 2017-03-04]. Dostupné z: http://www.tutorialspoint.com/java/java_overview.htm
- [20] Allen, G.: *Android 4: Průvodce programováním mobilních aplikací*. Brno: COMPUTER PRESS, 2013, ISBN 978-80-251-3782-6.
- [21] Android Developer: Verze [online]. [Citováno 2017-03-03]. Dostupné z: <http://developer.android.com/guide/topics/manifest/uses-sdk-element.html>
- [22] Matěj Konečný: Vyhýjme pro Android: Začínáme - Zdroják [online]. [Citováno 2015-03-16]. Dostupné z: <http://www.zdrojak.cz/clanky/vyvijime-pro-android-zaciname/>

- [23] Google play [online]. [Citováno 2017-03-04]. Dostupné z: <https://play.google.com/store>
- [24] ARToolkit - documentation [online]. [Citováno 2017-03-22]. Dostupné z: <https://artoolkit.org/documentation/>
- [25] Vuforia - Library [online]. [Citováno 2017-03-22]. Dostupné z: <https://library.vuforia.com/>
- [26] Unity - Pricing [online]. [Citováno 2017-03-23]. Dostupné z: <https://store.unity.com/>
- [27] Mikheev, K.: Introduction to Model View Presenter on Android [online]. 2015, [Citováno 2017-03-23]. Dostupné z: http://konmik.com/post/introduction_to_model_view_presenter_on_android/
- [28] Material design [online]. 2017, [Citováno 2017-04-15]. Dostupné z: <https://material.io/guidelines/>
- [29] Adobe color CC [online]. 2017, [Citováno 2017-04-15]. Dostupné z: <https://color.adobe.com/cs/create/color-wheel/>
- [30] Kelly, S.: Git vs SVN [online]. 2017, [Citováno 2017-03-23]. Dostupné z: <https://deveo.com/git-vs-svn/>
- [31] Firebase - Documentation [online]. [Citováno 2017-03-24]. Dostupné z: <https://firebase.google.com/docs/>
- [32] Usability Heuristics [online]. 2005, [Citováno 2017-04-19]. Dostupné z: <https://www.nngroup.com/articles/ten-usability-heuristics/>

Seznam použitých zkrátek

OS Operační systém

iOS iPhone Operation System

GPS Global Positioning System

DVM Dalvik Virtual Machine

SDK Software Development Kit

JDK Java Development Kit

JRE Java Runtime Environment

2D Two-dimensional space

3D Three-dimensional space

AR Augmented reality

XML Extensible markup language

AS Android Studio

IDE Integrated Development Environment

API Application Programming Interface

ID Identifikační kód

P2P Peer-to-peer

MVP Model View Presenter

UI User Interface

UX User Experience

A. SEZNAM POUŽITÝCH ZKRATEK

SVN Subversion

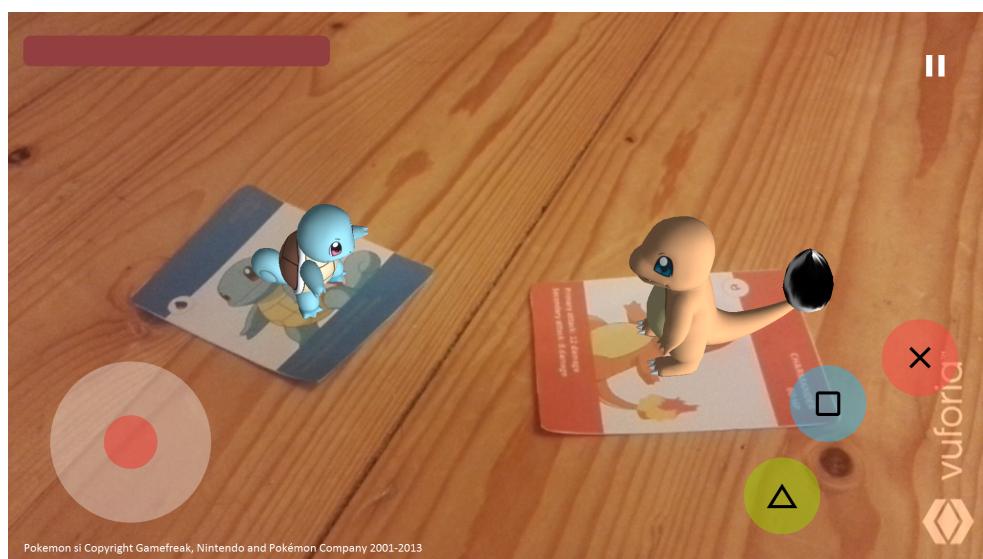
USC US Copyright

Testovací scénář

Testovací scénář slouží testerům jako pomocný materiál pro průchod aplikací a měl by testovat její základní funkčnost.

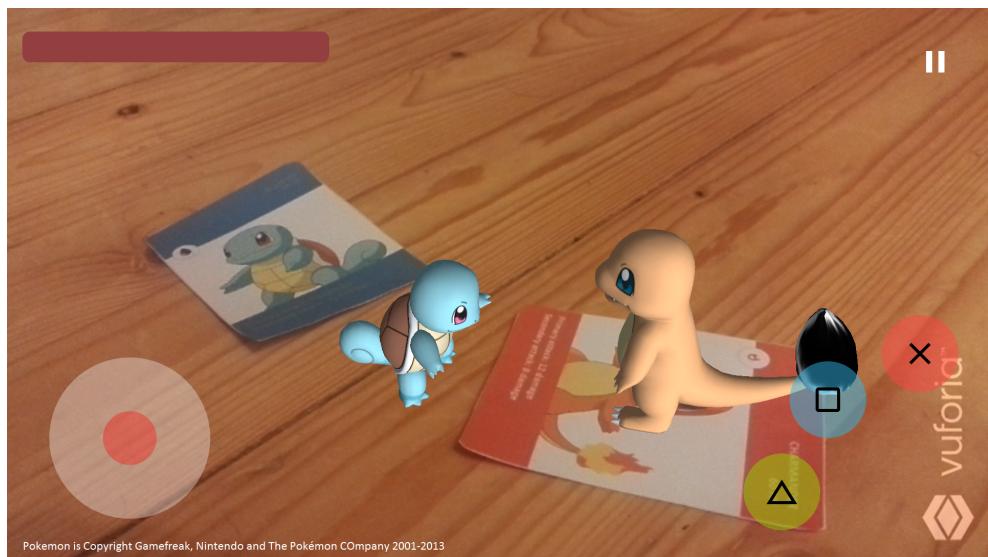
1. **Registrace do aplikace:** Registrujte se pomocí emailu a hesla do aplikace.
2. **Přihlášení a odhlášení:** Po registraci se z aplikace odhlašte a následně se přihlašte pomocí údajů, které jste zadali v předchozím bodě.
3. **Změna informací a hesla:** Zkuste změnit parametry v nastavení účtu.
4. **Hra pro jednoho hráče:** Vyzkoušejte hru jednoho hráče.
5. **Hra pro dva hráče:** Vyzkoušejte hru dvou hráčů. Zejména propojení zařízení a samotnou hratelnost.
6. **Zobrazení statistik:** Na závěr zjistěte své pořadí mezi hráči.

Obrázky ze hry

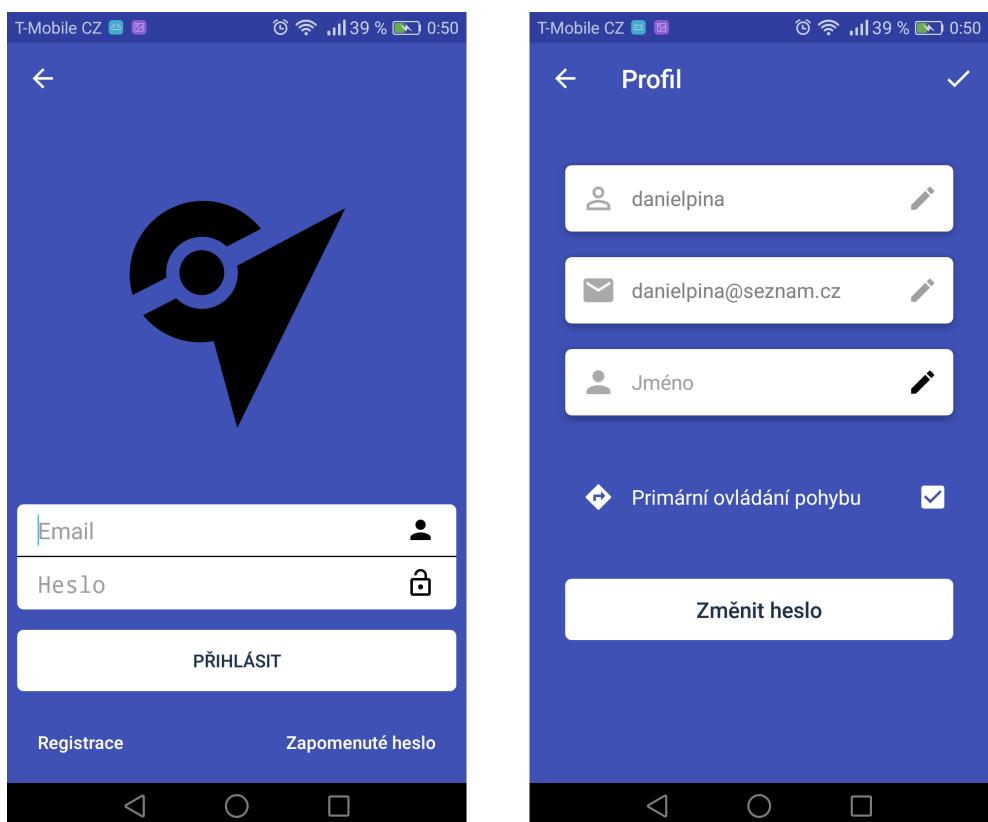


Obrázek C.1: Hra

C. OBRÁZKY ZE HRY



Obrázek C.2: Hra



Obrázek C.3: Přihlášení a úprava profilu

Instalační příručka

Aplikaci je možné nainstalovat na mobilní zařízení s operačním systémem Android 4.0 (Ice Cream) a vyšším. Na různých typech zařízení se díky nástavbám nad OS Android může následující postup mírně lišit. Aplikace zatím není publikována na oficiálním distribučním kanále (Google Play) a aplikace musí být nainstalována přímo z CD přiloženém k práci.

Pro instalaci aplikace z přiloženého CD zkopírujte instalační soubor s příponou „.apk“, který se nachází ve složce „apk“ v kořenovém adresáři, do paměti cílového zařízení. V cílovém zařízení spusťte správce souborů nebo jiný program pro instalaci aplikací z uložiště. Pomocí tohoto programu otevřete instalační soubor („apk“) a dále postupujte podle instrukcí na obrazovce až do dokončení instalace. Aplikace se bude po dokončení instalace nacházet v seznamu aplikací.

Pokud vám OS nedovolí naistalovat aplikaci z neznámého zdroje, bude potřeba tento typ instalace povolit. Tato možnost se většinou nachází v nastavení v sekci „Zabezpečení“. Aplikace je částečně závislá na serveru, a proto je možné, že aplikace na přiloženém CD nebude kvůli změnám na serveru plně funkční.

Obsah přiloženého CD

```
readme.txt.....stručný popis obsahu CD
├── apk.....adresář se spustitelnou formou aplikace
├── src
│   └── thesis ..... zdrojová forma práce ve formátu LATEX
└── text ..... text práce
    └── thesis.pdf ..... text práce ve formátu PDF
```