Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Diplomová práce

Multiplatformní vývoj her pro mobilní zařízení

Plzeň, 2014 Bc. Jiří Zikmund

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 14. května 2014

Bc. Jiří Zikmund

Poděkování

Rád bych poděkoval vedoucímu práce Ing. Ladislavu Pešičkovi za odborné vedení a čas strávený při konzultacích. Dále chci poděkovat rodičům a přítelkyni za podporu.

Abstract

Subject of this diploma thesis is Cross-platform game development for mobile devices. In the first part I describe possibilities in cross-platform mobile development in theory, focusing on the game development and choosing a suitable framework. This part also compares the current mobile platforms in therms of market share and profitability. In the second part of the thesis I proceed to the development process itself. Product of this thesis is a mobile game that uses the game engine with a physical model and runs on a real iOS, Android and Windows Phone devices.

Abstrakt

Tématem této diplomové práce je Multiplatformní vývoj her pro mobilní zařízení. V první části teoreticky popisuji možnosti multiplatformního mobilního vývoje, zaměřuji se především na mobilní hry a výběr vhodného frameworku. Porovnávám zde také současné mobilní platformy z pohledu podílu na trhu a potenciálního zisku. Ve druhé části se věnuji samotnému vývoji. Výsledkem této práce je mobilní hra využívající herní engine s fyzikálním modelem, kterou lze spustit na reálných zařízeních se systémy iOS, Android a Windows Phone.

Obsah

1 Úvod

Tématem této diplomové práce je multiplatformní vývoj her pro mobilní zařízení. Mobilní hry byly mezi lidmi vždy velmi populární, ale jejich skutečný potenciál se ukázal až po příchodu současné generace dotykových telefonů. Ty s sebou přinesly nové technologie a pohodlný způsob distribuce aplikací. Vznikl tak obrovský trh, ve kterém působí nezávislí vývojáři i velká herní studia. Jeho problémem je ale nejednotnost. V současné době existuje několik významných platforem a má-li být produkt úspěšný, musí být dostupný na co nejvíce z nich. Při nativním přístupu se vývoj musí pro každou platformu opakovat stále dokola, což je neefektivní a časově náročné. Začaly proto vznikat nástroje, které tento proces sjednocují.

Cílem práce je prozkoumat možnosti multiplatformní vývoje, vybrat vhodný framework pro tvorbu her a použít jej pro implementaci vlastní hry, která bude spustitelná na více platformách. V první části se věnuji rozvoji mobilních platforem a jejich významu v oblasti her. Srovnávám je z několika hledisek a vybírám ty, které jsou v současnosti pro vývoj nejvhodnější. Pro tyto platformy se pak snažím najít vhodný herní framework. V druhé části diplomové práce navrhuji vlastní hru, která demonstruje možnosti vybraného nástroje a popisuji postup při její realizaci.

Výsledkem je mobilní hra využívající pokročilý herní engine včetně fyzikálního modelu, která funguje v identické podobě na telefonech a tabletech se systémy iOS, Android a Windows Phone.

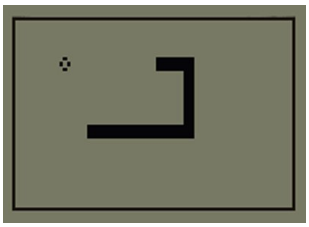
2 Historie mobilních her

V této kapitole se věnuji vzniku her na mobilních telefonech a popisuji zde mobilní platformy, které v minulosti měly největší podíl na jejich rozvoji.

2.1 Vestavěné hry

Když se v devadesátých letech minulého století poprvé objevily hry na mobilních telefonech, byly pevnou součástí systému a nebylo možné je žádným způsobem instalovat či modifikovat. Jejich herní složitost a grafické zpracování odpovídaly hardwaru, na kterém fungovaly a musely si tak vystačit pouze s malým monografickým displejem a velmi omezenou pamětí. Jako vůbec první se v roce 1994 objevila variace hry Tetris na telefonu Hagenuk MT-2000 a relativně dlouhou dobu neměla žádnou konkurenci. [2]

Až po třech letech v roce 1997 uvedla finská firma Nokia svůj model 6610 obsahující hry rovnou tři, a to Snake, Logica a Memory. Především první zmiňovaná byla pro herní mobilní průmysl zlomová. Hra Snake (viz obr. 2.1) si i přes svou jednoduchost, nebo možná právě díky ní, získala mezi hráči obrovskou popularitu a vnesla do širšího povědomí lidí, že telefon se dá využít i jinak, než jen k telefonování. Nabízela dokonce možnost multiplayeru, kdy se dvě zařízení spojila pomocí vestavěného infraportu. Tato hra se objevila v mnoha modelech telefonů značky Nokia v celkovém počtu asi 400 milionů kopií. [3]



Obrázek 2.1: Ukázka hry Snake na Nokii 6110 [4]

Jednoduché černobílé hry se v následujících několika letech staly standardem a mohli jsme je nalézt v telefonech většiny tehdejších výrobců. Sloužily jen jako doplněk k základní funkcionalitě a sami o sobě negenerovaly žádný zisk, jejich účelem bylo především zvýšení atraktivity zařízení. Na jednodušších telefonech

s uzavřeným systémem nalezneme tento typ her ještě dnes. Rozdílem je, že mohou využívat větší a barevné displeje.

2.2 WAP

Dalším milníkem mobilních her byla technologie WAP, standard vyvinutý pro připojení mobilů k internetu. Za jeho vznikem stojí organizace Wap Forum[1], založená v roce 1997, přičemž technologicky se na něm nejvíce podílela firma Unwired Planet. Ta také vytvořila první mikroweb, jenž mohl být zobrazen na mobilním telefonu. První veřejné wapové stránky byly spuštěny v říjnu roku 1999 nizozemským mobilním operátorem Telfort BV a stalo se tak společně s představením Nokie 7110, což byl první telefon podporující tuto technologii. [5]

Z uživatelského hlediska nabízel WAP, vzhledem k možnostem tehdejších zařízení, pouze několikařádkový textový výstup s jednoduchou černobílou grafikou. I tak ale poskytl dostatečný prostor pro nový typ mobilních her. Byl vhodný např. pro textové, tahové nebo deskové hry a svou podstatou vybízel k jednoduché tvorbě multiplayerových her. Naopak jakékoliv akční tituly nebyly technologicky dost dobře proveditelné. Příkladem úspěšných her této doby mohou být fantasy adventura Steve Jackson's Sorcery od Digital Bridgesfirmy (viz obr. 2.2a) nebo sociální multiplayerová tahová hra Lifestylers od firmy Picofun (viz obr. 2.2b). [6, 7]

(a) Steve Jackson's Sorcery [6]

(b) Lifestylers [7]

Obrázek 2.2: Ukázky WAP her [6, 7]

[1]Wap Froum tvořily firmy Unwired Planet, Ericsson, Nokia a Motorola.

I přes velký zájem hráčů bylo složité WAP hry zpoplatňovat, protože jediným subjektem generujícím zisk byl operátor za poskytování připojení. Původním záměrem bylo díky hrám zvyšovat používání wapu, přičemž vývojáři a distributoři her by byli placeni podle průměrného potenciálního navýšení zisku. Na to však operátoři nepřistoupili. Vznikly proto speciální portály, kde mohl uživatel ke hrám snadno přistupovat. Ty byly ale pod správou operátorů, se kterými si vývojáři museli dojednat podmínky. Pokud vývojáři provozovali hru na vlastních serverech, měli možnost zpoplatňovat hru poněkud nešikovně např. pomocí přístupových kódů zaslaných prémiovou SMS.

V následujících letech se sice objevily mobily s barevnými displeji podporující WAP a tato technologie se tak dále rozvíjela, ale kvůli výše zmíněným omezením byla brzy překonána.

2.3 Java

Zlomovým okamžikem pro mobilní hry byl rok 2001. Na konferenci JavaOne v San Franciscu byla představena nová platforma pro mobilní hry `J2ME`, tedy Java 2 Micro Edition. Tato mikro verze programového prostředí Java měla přinést na displeje mobilních telefonů nové grafické možnosti, což zde bylo prezentováno na prototypu zařízení od Motoroly. Kromě toho to také znamenalo otevřenost systému pro externí vývojáře a herní studia. V roce 2002 na trh skutečně vstoupily přístroje podporující tuto technologii. Jednalo se o telefony Nokia 3410 a Siemens M50. Oba dva však disponovaly stále jen monochromatickým displejem o rozlišení 96x64, resp. 101x64 pixelů a maximální možná velikost her byla pouhých 30 kB. [8]

Navzdory těmto omezením začalo vznikat mnoho kvalitních titulů a zájem o ně byl i ze strany hráčů. Dobře fungoval také platební model a začal se tak utvářet mobilní herní průmysl. Distribuce probíhala většinou stahováním přes WAP přímo do mobilu, čímž byla zároveň zajištěna i platba. Vývojáři inkasovali peníze od zákazníků prostřednictvím mobilních operátorů. Někteří distributoři prodávaly hry i v krabicových verzích, kde se nacházel odkaz a kód pro stažení, což bylo cílené na konzervativnější uživatele. Jednou z prvních Java her ještě na černobílých displejích byl Siberian Strike od firmy Gameloft (viz obr. 2.3a). Na rozdíl od doposud rozšířených statických WAP her se jednalo o akční titul, který tak trochu předpovídal směr, kterým se budou mobilní hry ubírat.

Mobilní telefony podporující Javu urazily v následujících letech dlouhou cestu. Přišly barevné displeje, větší paměť a výkonnější hardware. Kromě pokročilých 2D her nepřeberného množství žánrů se na trh dostaly i 3D tituly přinášející poměrně vydařené grafické zpracování. Příkladem mohou být arkádové závody Rally Master Pro z roku 2008 (viz obr. 2.3b). Java hry se staly na dlouhou dobu nejrozšířenější herní mobilní platformou dostupnou na opravdu mnoha zařízeních. Jejich rozmanitost byla ovšem kvůli jejich odlišným hardwarovým parametrům zároveň nevýhodou této platformy. Různý výkon a rozlišení displeje znamenaly problémy s kompatibilitou a často tak musely být hry vydávány v několika specifických verzích. I když je tato technologie dnes již překonána, stále se s ní v některých současných mobilních telefonech setkáváme.

(a) Siberian Strike [8]

(b) Rally Master Pro [9]

Obrázek 2.3: Ukázky Java her [8, 9]

2.4 Symbian a N-Gage

Symbian je operační systém od počátku navržený pro mobilní telefony, který vyvinul Symbian Ltd., joint venture společností Psion, Nokia, Ericsson a Motorola. Vznikl na základě operačního systému EPOC a byly na něm založeny softwarové platformy S60, UIQ a MOAP. Tou nejzásadnější a nejrozšířenější z nich byla S60 používaná na telefonech značek Nokia, Samsung a LG. Systém od začátku předpokládal velké barevné displeje, tlačítkové ovládání a byl otevřený pro aplikace třetích stran. To bylo samozřejmě pro hry vhodné prostředí. První telefon založený na platformě S60 přišel na trh už v roce 2002, jednalo se Nokii 7650. Systém se v následujících letech velmi rozšířil, do roku 2006 bylo prodáno 100 milionů

zařízení a jeho podíl na trhu chytrých telefonů v tu dobu činil 73%. S příchodem dotykových telefonů však platforma začala ztrácet své dominantní postavení a i přesto, že se v roce 2008 objevila dotyková Nokia 5800, za svou konkurencí zaostávala a už ji nikdy nedohnala. Posledním vyrobeným Symbianovým telefonem se stala v létě 2013 Nokia 808. [10]

Kromě klasických telefonů přinesla platforma i jeden velmi zajímavý produkt, především z herního pohledu. V roce 2003 totiž Nokia vyrobila zařízení N-Gage a o rok později mírně vylepšeného nástupce N-Gage QD (viz obr. 2.4). Jednalo se o plnohodnotný mobilní telefon fungující na platformě S60, ale svým zaměřením a konstrukcí připomínal spíše kapesní herní konzole. Nabízené hry byly velmi propracované a byly prodávány na paměťových kartách. Velkou výhodou bylo snadné ovládání díky herně přizpůsobeným tlačítkům. Pomocí technologie bluetooth některé tituly umožňovaly i hru pro více hráčů. Do roku 2005 bylo vydáno 50 herních titulů a celosvětově se prodalo více než dva miliony telefonů N-Gage. To však bylo pod očekáváním Nokie a v tomto roce byla výroba těchto zařízení ukončena. Platforma jako taková však fungovala dál a v roce 2007 se dočkala oživení, když Nokia zpřístupnila N-Gage tituly i pro některé své ostatní S60 telefony. [11]

Obrázek 2.4: Nokia N-Gage QD [12]

2.5 Windows Mobile

Nelze určitě opomenout Windows Mobile, operační systém od společnosti Microsoft. Poprvé byl uveden již v roce 2000 a během svých deseti let existence se stal

velmi rozšířeným. Ve svém nejúspěšnějším roce 2004 pokrýval 23% trhu se smartphony. Zaměřením se jednalo spíše o telefony určené pro organizaci a práci, velkou výhodou byla např. přítomnost kancelářského balíku Microsoft Office. Vývoj byl ukončen v roce 2010, kdy podobně jako Symbian přestal být systém jako produkt konkurenceschopný. [13]

Díky otevřenosti systému se samozřejmě objevilo mnoho herních titulů vytvořených nativně pro tuto platformu. Svou popularitou ale nikdy nepřekonaly např. Java hry dostupné pro mnohem více zařízení. Ty mimochodem bylo možné na Windows Mobile spustit, ale byla nutná manuální instalace Javy a problémem byla kompatibilita, jelikož se jednalo často o zařízení s dotykovým displejem bez hardwarových tlačítek, na což Java hry většinou nebyly přizpůsobené. Displeje byly omezením i pro nativní hry, protože u většiny telefonů s tímto systémem byly vyrobeny rezistivní technologií[2]. To bylo sice vhodné na přesné ovládání kancelářských aplikací, nebo psaní poznámek dotykovým perem, ale dokázaly rozpoznat pouze jeden dotyk, což je pro složitější ovládání her značně omezující.

2.6 Ostatní historické platformy

Kromě výše zmíněných mobilních platforem existovaly i další, které však nebyly z hlediska herního průmyslu tak důležité. Objevilo se např. několik mobilních platforem založených na Linuxu. Jedním z nich byl operační systém **Maemo** vyvinutý společností Nokia. Poprvé se objevil v roce 2005 na zařízení Nokia 770 a později i na dalších několika telefonech. V roce 2010 bylo oznámeno jeho spojení s mobilním systémem **Moblin**, vyvíjeným společností Intel od roku 2007, za účelem vytvoření systému **MeeGo**. Jeho vývoj byl však ukončen již v roce 2011.

Mohli jsme se také setkat se systémem **PalmOS**, vyvíjeným společností Palm a uvedeným již v roce 1996. Byl od začátku vytvářen pro dotykové displeje a původně zajišťoval chod pouze v zařízeních PDA, později přibyly i mobilní telefony. Oproti konkurenci však zaostával a v roce 2009 byl pak uveden nástupce **WebOS**, založený na linuxovém jádře a doplněný o moderní webové technologie. V roce 2010 pak celou firmu Palm včetně WebOS koupil Hawlet Packard, který však poslední telefon s tímto systémem vyrobil v roce 2011. V roce 2013 jej pak prodal společnosti LG, která ho dnes používá např. ve svých chytrých televizích.

[2]Rezistivní neboli odporové displeje rozpoznávají dotyk podle tlaku.

3 Výběr platformy pro vývoj

Ještě před průzkumem konkrétních frameworků je potřeba si ujasnit, na jaké platformy vývoj primárně cílit. Výroba mobilních her je dnes obrovský byznys, vývojáři či herní studia investují do vývoje čas a finanční prostředky s cílem co nejlépe je zpeněžit. Je proto důležité znát potenciál jednotlivých platforem. V této kapitole popisuji současné nejvýznamnější mobilní systémy, jejich možnosti vývoje a distribuce aplikací a v neposlední řadě také jejich rozšířenost a podíl na trhu.

3.1 Současné platformy

3.1.1 iOS

Za počátek současných mobilních platforem se dá jednoznačně považovat rok 2007, kdy společnost Apple uvedla na trh iPhone, mobilní telefon, který mnoho lidí označovalo za revoluční. Smartphony v té době byly primárně ovládané pomocí hardwarové klávesnice (jako např. nejrozšířenější Symbian, viz kap. 2.4) a v případě dotykových displejů sloužil jako ovládací prvek stylus. Doménou iPhonu byl především kvalitní dotykový displej, na svou dobu s poměrně velkou úhlopříčkou 3,5", který díky kapacitní technologii[1] a technologi multitouch nabízel komfortní ovládání pomocí prstů. Tomu byl přizpůsoben i operační systém zaměřený především na jednoduchost a uživatelskou přívětivost. Byl postaven na léty prověřeném jádře převzatém z desktopového operačního systému OS X, díky čemuž byl stabilní, spolehlivý a dával také společnosti znatelný náskok před konkurencí. Dnes jej kromě telefonu iPhone najdeme i v tabletu iPad a hudebním přehrávači iPod. Apple nedává systém k dispozici jiným výrobcům.

Pro mobilní hry a aplikace se stal opravdu zásadní až rok 2008, když Apple uvolnil SDK pro vývojáře a zároveň také otevřel Appstore, obchod s mobilními aplikacemi. Pro uživatele to znamenalo jejich opravdu jednoduchou dostupnost, kdy instalace probíhá na několik málo doteků. Na druhé straně se pro vývojáře otevřela velmi pohodlná cesta k distribuci svých produktů. Autor aplikace při publikaci na Appstore sám určuje koncovou cenu, z níž po prodeji získá 70% a zbytek si nechává Apple [14]. Tento model se ukázal jako skvěle fungující, což ukazuje několik statistik platných k prosinci 2013 [15]:

[1] Funguje na základě vodivosti lidského těla, pro ovládání není potřeba vyvíjet tlak.

- Více než 1.000.000 dostupných aplikací
- Denně staženo průměrně 645.000 aplikací
- Vývojářům vyplaceno celkem 13 miliard dolarů
- Na každém zařízení staženo průměrně 80 aplikací

Pro vývoj na tuto platformu je k dispozici prostředí Xcode, které Apple poskytuje zdarma, ale dostupné je výhradně na systému Mac. Jako programovací jazyk lze použít Objective-C, případně C, nebo C++. Je také potřeba mít vývojářský účet, který je zpoplatněný částkou 99 $ za rok, bez něj nelze aplikace distribuovat v Appstore a dokonce ani testovat na reálném zařízení [16]. Programátor tedy musí splňovat relativně hodně podmínek a poměrně přísný je i schvalovací proces společnosti při publikaci do jejich obchodu. Odměnou je ale velký trh potenciálně platících zákazníků (viz kap. 3.2.3).

3.1.2 Android

Operační systém Android od společnosti Google byl vydán na podzim roku 2008, tedy více než rok po iOS. Stal se ale nejrychleji rostoucí platformou a současnému trhu s chytrými telefony jasně dominuje (viz obr. 3.1). Google na rozdíl od Applu vyvíjí pouze systém, v mobilních telefonech a tabletech ho pak využívají výrobci jako např. Samsung, Sony, LG nebo Motorola [17]. V prodeji jsou i velmi levné telefony (cca 1500 Kč), což samozřejmě přispívá jejich popularitě a rozšířenosti. Tyto modely však v porovnání s těmi dražšími přinášejí omezení např. ve výkonu nebo rozlišení displeje, což je pro vývojáře problém z hlediska optimalizace a kompatibility. Systém je v porovnání s iOS více otevřený, aplikace je např. možné instalovat i z jiných zdrojů, než oficiálního obchodu, a je také přístupný souborový systém. Daní za to je výrazně vyšší výskyt škodlivého softwaru [18].

Aplikace pro Android se vyvíjí v jazyce Java s využitím Android SDK. Práci usnadňuje plugin do vývojového prostředí Eclipse a vyvíjet v něm lze bez problémů na systémech Windows, Mac i Linux. V tomto ohledu mají tedy vývojáři oproti iOS více volnosti. Výhodou je také to, že kromě emulátoru je možné testovat aplikace na fyzických zařízeních i bez vývojářského účtu, za který je účtován jednorázový poplatek 25 $ [19]. Ten je nutné si založit až ve chvíli publikace do obchodu s aplikacemi Google Play, který je založen na stejném principu jako Appstore od Apple. Google převzal i prodejní model, kdy vývojáři nechává 70% podílu ze zisku [20].

3.1.3 Windows Phone

Microsoft ztratil u svého Windows Mobile (viz kap. 2.5) krok s konkurenčními platformami iOS a Android a v roce 2010 proto představil zcela přepracovaný systém Windows Phone 7. Jednalo se o samostatný systém, který byl poskytován výrobcům, přičemž byly stanoveny minimální hardwarové požadavky na zařízení. Díky tomu byl systém rychlý, stejně jako konkurence nabízel kapacitní multidotykové ovládání a svými funkcemi a pojetím se moderním konkurentům vyrovnal. Několikaleté zpoždění ovšem znamenalo vstup do již obsazeného trhu. V roce 2012 byl pak představen nástupce, Windows Phone 8, který není zpětně kompatibilní se staršími přístroji. Poslední verze systému s označením 8.1 přišla v dubnu 2014 a dostupná je na všechna současná WP8 zařízení. Významným krokem pro platformu byla akvizice mobilní divize společnosti Nokia, která proběhla v září 2013 za sumu 5,44 miliard eur (cca 140 miliard korun) [21]. Microsoft tak bude podobně jako Apple vyrábět hardware i software svých zařízení.

Aplikace pro WP lze programovat v jazycích C#, Basic a C++. Primární vývojové prostředí je Visual Studio 2010 pro WP 7 a Visual Studio 2012 či 2013 pro WP 8. Do budoucna Microsoft plánuje sjednocení vývoje pro Windows Phone 8.1 a Windows 8.1 RT, který je využíván především v tabletech [22]. Pro distribuci aplikací je k dispozici obchod Windows Phone Store.

3.1.4 BlackBerry

Neopominutelnou mobilní platformou je BlackBerry od kanadské firmy Research In Motion Limited (RIM), která stejně jako Apple stojí jak za výrobou operačního systému, tak i samotných zařízení, na kterých fungují. Současným platformám se vymyká svou dlouhou historií. Prvním přístrojem byl obousměrný pager s označením RIM 850 již z roku 1999, v roce 2003 pak uvedli první mobilní telefon. Z počátku užívané černobílé displeje byly nahrazeny barevnými, nechyběla podpora Java her a později se platforma dokázala vyrovnat i s příchodem iPhonu a novou érou dotykových telefonů. BlackBerry telefony jsou typicky manažerské přístroje, jejichž doménou je především emailová a internetová komunikace, se kterou vždy souvisela kvalitní qwerty klávesnice nebo např. technologie push email[2]. Významné jsou tyto telefony z hlediska bezpečnosti, při komunikaci používají vlastní šifrování. Současná podoba platformy je otevřená pro vývojáře třetích stran a existuje také

[2]Technologie umožňující aktivní zaslání emailu do zařízení ihned po doručení.

obchod s aplikacemi BlackBerry World, nicméně kvůli svému manažerskému zaměření a z toho plynoucí cílové skupině uživatelů není pro vývoj her primárním cílem. [23]

3.1.5 Ostatní současné platformy

V současnosti existuje také několik dalších mobilních operačních systémů. Jedním z nich je **Sailfish OS** společnosti Jolla, vytvořený na základě výše zmiňovaného MeeGo (viz kap. 2.6). Dále je to **Tizen**, za nímž stojí Linux Foundation, **Ubuntu Touch OS** od Canonical Ltd., nebo **Firefox OS** od Mozilla Foundation. Dohromady ale svým podílem tvoří jen marginální část trhu (viz obr. 3.1).

3.2 Výběr platformy

3.2.1 Aktuální podíl na trhu

Při výběru nejvhodnější platformy pro vývoj mobilních aplikací hraje roli několik faktorů. Prvním z nich je aktuální podíl na trhu. Jednoznačně nejrozšířenější je v současnosti (únor 2014) Android, podle serveru statcounter.com běží téměř na 60% všech chytrých mobilních telefonech. Na druhém místě nalezneme iOS s 29% a ostatní platformy dosahují pouhých 5% nebo méně (viz tab. 3.1).

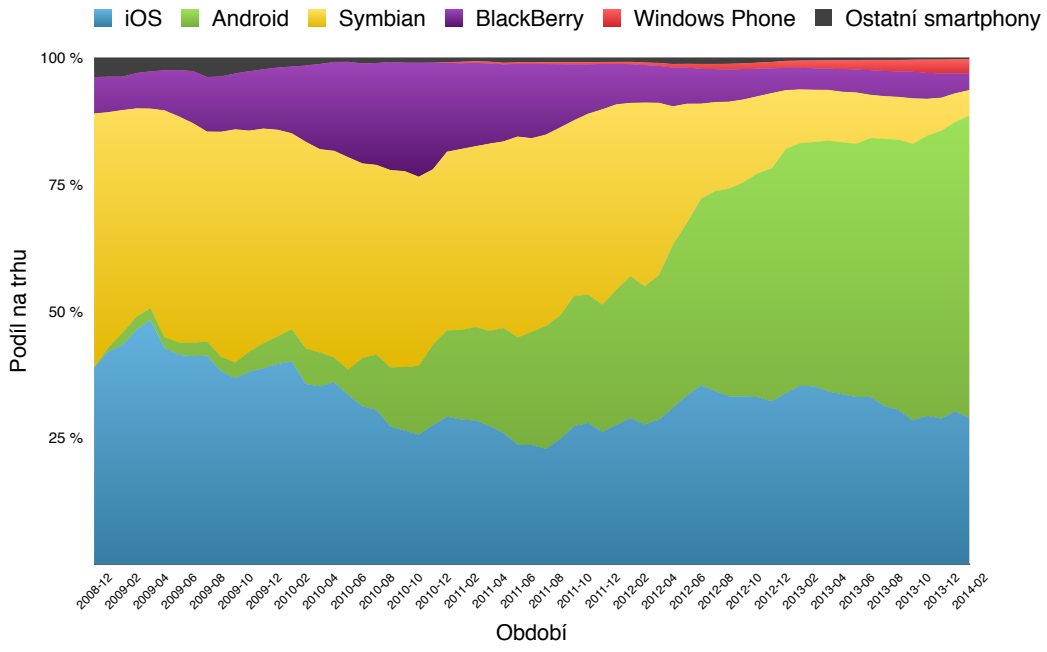
Platforma	**Podíl na trhu**
Android	59,67 %
iOS	28,89 %
Symbian	5,05 %
BlackBerry	3,29 %
Windows Phone	2,79 %
Ostaní smarphony	0,34 %

Tabulka 3.1: Podíl mobilních operačních systémů na trhu v únoru 2014 [24]

3.2.2 Vývoj podílu na trhu v čase

Překvapivá je aktuální pozice Symbianu, který se stále drží na třetím místě a to i přesto, že se již od poloviny roku 2013 telefony s tímto systémem nevyrábí (viz kap. 2.4). A to je přesně důvod, proč je potřeba se zaměřit i na další faktory. Podle dat dostupných ze serveru statscounter.com [24] jsem vytvořil graf zobrazený na obrázku 3.1. Ukazuje vývoj podílu jednotlivých platforem od prosince 2008 až do února 2014. Je patrné, že na začátku tohoto období byl Symbian ještě jasným lídrem, ovšem po příchodu systémů iOS a Androidu (viz kap. 3.1.1 a 3.1.2) svou pozici už jen permanentně ztrácel. Protože se dnes již nevyrábí (viz kap. 2.4) je jen otázka času, kdy z trhu vymizí úplně. Vývoj pro něj je proto již neperspektivní.

Systém iOS byl na konci roku 2008 na trhu již přes rok a půl (viz kap. 3.1.1) a za tu dobu si vybudoval poměrně silnou pozici. Nástup Androidu mu sice část uživatelů ubral, ale jeho podíl se dlouhodobě drží kolem hodnoty 30%. Android je v tomto ohledu jednoznačně nejsilnější a jeho postavení stále roste. Poslední dvě platformy, u kterých z dlouhodobého hlediska lze uvažovat o vývoji aplikací, jsou BlackBerry a Windows Phone. BlackBerry je na tom aktuálně sice o něco lépe, ale během posledních let je na ústupu. Naopak Windows Phone jako nejmladší platforma se dočkává velmi rychlého růstu a očekává se, že BlackBerry v budoucnu výrazně překoná. Z tohoto hlediska má tedy pro vývoj lepší předpoklady.



Obrázek 3.1: Vývoj podílu mobilních operačních systémů na trhu

3.2.3 Výnosnost

Důležitým aspektem je také výnosnost jednotlivých platforem. V tabulce 3.2 jsou k dispozici některé statistiky aktuální ke dni 2.3.2014. Nejvýnosnější je jednoznačně iOS, jehož uživatelé jsou za aplikace ochotni nejvíce platit. Celkem 55% z nich za aplikace utratilo více než jeden dolar. Nejmarkantněji se to projevuje na ziscích vývojářů. Ti si za rok 2013 v Appstore vydělali 6,4 miliardy dolarů, což je 5-krát více, než u Androidu (1,2 miliardy $), který má dvojnásobné množství uživatelů a v nabídce podobný počet aplikací. Dobře si stojí i Windows Phone, který se ziskem blíží Androidu, přičemž nabídka aplikací je oproti němu pouze čtvrtinová a podíl na trhu asi 20 krát menší. Nejhůře si zde pak stojí BlackBerry.

Uvedené částky zahrnují zisky pouze z jednorázového prodeje aplikací a nezohledňují příjem z in-app nákupů nebo zobrazovaných reklam. Jedná se o údaje z oficiálních obchodů daných platforem, tedy Apple App Store, Google Play, Windows Phone Store a BlackBerry World.

	iOS	**Android**	**Windows Phone**	**BlackBerry**
Celkový počet stažení aplikací	27 miliard	29 miliard	4,1 miliard	2,4 miliard
Uživatelé, kteří utratili více než 1 $	55 %	38 %	42 %	37%
Průměrný počet stažených aplikací	88	68	57	49
Celkový počet dostupných aplikací	1 000 000	850 000	220 000	130 000
Zisk vývojářů za rok 2013	6,4 miliard $	1,2 miliard $	950 milionů $	550 milionů $

Tabulka 3.2: Statistiky obchodů s mobilními aplikacemi [25]

3.2.4 Shrnutí

Z výše uvedených informací jsem určil prioritu operačních systémů při výběru multiplatformního frameworku následovně:

1. iOS
2. Android
3. Windows Phone
4. BlackBerry
5. Ostatní

Na prvním místě je jednoznačně iOS, který přináší nejvyšší potenciální zisk a má velmi stabilní uživatelskou základnu. Druhý Android se svým výrazně největším podílem na trhu sice nabízí nejvíce uživatelů, ti ale nejsou tak ochotní za aplikace platit. Často se proto přistupuje k modelu, kdy stejná aplikace je ve verzi pro iOS placená a na Android dostupná zdarma, ale s reklamou. Třetí volbou je Windows Phone, který je současně nejrychleji rostoucí platformou a telefony s tímto systémem se stávají stále oblíbenější. BlackBerry je zatím rozšířenější než Windows Phone, ale dlouhodobě je na ústupu a jeho budoucnost je velmi nejistá. Zařadil jsem ho proto až na čtvrté místo a při výběru frameworku by měl hrát jen minimální roli. Všechny ostatní systémy, např. Firefox OS, nebo Tizen, tvoří tak malou část trhu, že při výběru frameworku nemají vliv žádný, i když jejich podpora by byla samozřejmě plusem.

4 Nativní a multiplatformní vývoj

V předchozí kapitole jsem určil jako primární platformy pro vývoj mobilních aplikací iOS, Android a Windows Phone. Otázkou je nyní, jakým způsobem postupovat, pokud chceme na všechny z nich vytvořit totožnou aplikaci. Základní možností je vyvíjet nativně pro každou platformu zvlášť. Problémem ale je, že na všech se vyvíjí v jiném programovacím jazyce a vývojovém prostředí a celý proces programování, kompilace, sestavování a testování je nutné opakovat. Navíc v případě her využité herní a fyzikální enginy nemusí být na všech platformách dostupné a je potom velice těžké docílit shodného chování. Oddělený vývoj znamená také nutnost rozumět více technologiím, z čehož plyne potřeba více lidí nebo času, tedy vyšších finančních nákladů. V tabulce 4.1 jsou uvedeny možnosti nativního vývoje na jednotlivých platformách.

	iOS	**Android**
Programovací jazyk	Objective-C, C, C++	Java
Vývojové prostředí	Xcode	Eclipse, Android Studio
Operační systém	Mac	Windows, Mac, Linux

	Windows Phone	**BlackBerry**
Programovací jazyk	C#, Basic, C++	Java
Vývojové prostředí	Viusal Studio	Momentics IDE
Operační systém	Windows	Windows, Mac, Linux

Tabulka 4.1: Nativní vývoj pro mobilní platformy

Z výše zmíněných důvodů by tedy bylo dobré nějakým způsobem vývoj sjednotit. Existuje naštěstí několik možností, jak toho docílit. V této kapitole všechny mapuji, porovnám a pokouším se vybrat nejvhodnější variantu pro implementaci ukázkové hry.

4.1 Webové HTML5 aplikace

Nejjednodušším způsobem pro vytvoření multiplatformní aplikace na mobilní zařízení je použití HTML5, které je podporováno na všech moderních platformách. Aplikace jsou v podstatě webové stránky optimalizované pro malé rozměry displeje a vše běží v rámci internetového prohlížeče (viz obr. 4.1). Zdálo by se tak,

že aplikace nemá žádný přístup k hardwaru nebo uživatelským datům, ovšem mobilní prohlížeče již některé tyto informace poskytují. V tabulce 4.2 je přehled těch nejpodstatnějších. Data jsou platná k březnu 2014 a v budoucnu se očekává, že další funkce budou přibývat. V případě Androidu se informace týkají prohlížeče Google Chrome, u ostatních systémů pak prohlížečů základních.

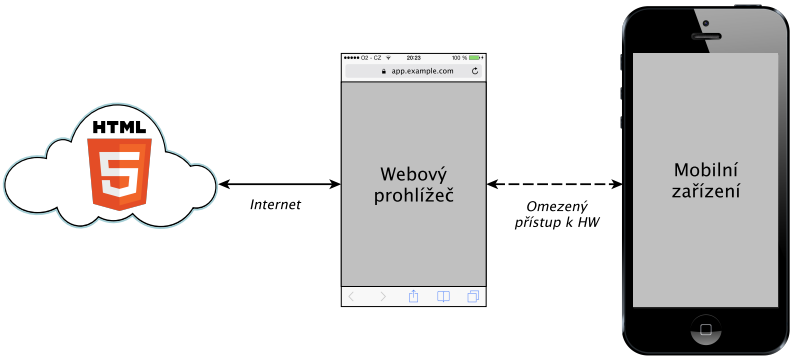
					Popis
Odeslání SMS	✓	✓	✓	✓	Vyvolání nabídky pro odeslání SMS s předdefinovaným textem a příjemcem
Uskutečnění hovoru	✓	✓	✓	✓	Uskutečnění hovoru na předdefinované telefonní číslo
Geolokace	✓	✓	✓	✓	Informace o aktuální poloze podle GPS, WiFi nebo operátora
Pohybová čidla	✓	✓		✓	Prostorová poloha zařízení podle gyroskopu, akcelerometru a kompasu
Multitouch	✓	✓	✓	✓	Rozpoznávání více doteků na obrazovce zařízení
Multimédia	✓	✓	✓	✓	Přehrávání hudby a videa, může probíhat i na pozadí
Fotoaparát	✓	✓		✓	Přístup k fotoaparátu a mikrofonu pro upload média
Video stream		✓		✓	Přímý přístup k fotoaparátu pro video a audio stream
Vibrace		✓		✓	Možnost ovládání vibrací telefonu
Celoobrazovkový režim		✓		✓	Možnost inicializace režimu celé obrazovky
File System API		✓		✓	Přístup k souborovému systému pro trvalé ukládání dat

Tabulka 4.2: Hardwarová podpora HTML5 na mobilních platformách [26, 27]

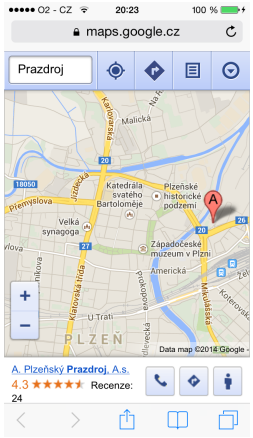
Jak je z tabulky vidět, některé funkce nejsou podporovány všemi prohlížeči a aplikace na takové případy musí být připraveny. Např. ale geolokační služby, hovory a SMS jsou již standardem, výborně tyto funkce využívá např. aplikace Google mapy, dostupná na `<http://maps.google.cz>`. Vykresluje mapu, kterou je možné pohybem prstu posunovat a pomocí gesta pinch-to-zoom[1] ji také přibližovat a oddalovat. Zobrazují rovněž aktuální polohu uživatele, umožňují vyhledávat místa na mapě a v případě, že lokace obsahují telefonní číslo, je možné na něj přímo zavolat (viz obr. 4.2a). Lze dokonce plánovat trasu a zajímavá je

[1]Současné přibližování nebo oddalování dvou prstů na displeji.

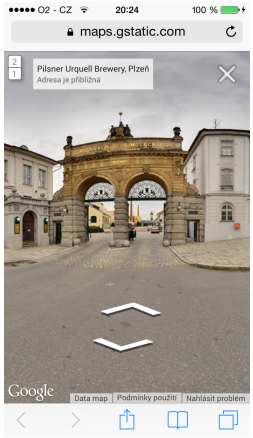
rovněž funkce Street View (viz obr. 4.2b), která zobrazuje interaktivní panoramatické snímky určeného místa. Aplikaci jsem testoval na telefonu iPhone 5 (systém iOS 7.1) a vše funguje velmi plynule.



Obrázek 4.1: Princip webových HTML5 aplikací



(a) Google mapy



(b) Google Street View

Obrázek 4.2: Ukázky webových HTML5 aplikací

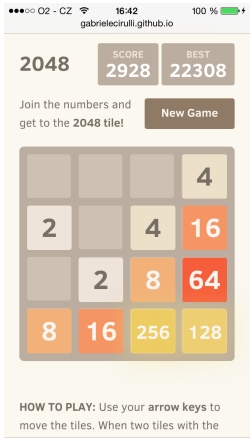
HTML5 se dá poměrně dobře využít i pro vývoj her. Veškerý výkonný kód se píše v JavaScriptu, který je v dnešních prohlížečích interpretován již velmi rychle. Pro vykreslování grafiky a animací existuje několik možností (viz tab. 4.3). Nejzákladnější je plátno, HTML5 canvas, které se velmi dobře hodí na zobrazování 2D grafiky a podobně je na tom i SVG. Pomocí CSS 3D a WebGL lze dokonce vytvářet trojrozměrné aplikace. WebGL je API založené na OpenGL ES 2.0 [28] a je ze všech nejvýkonnější, bohužel ale není dostupné na všech platformách. Podle serveru webglstats.com jej v březnu 2014 podporovalo pouze 17% mobilních zařízení [29]. Práci ulehčují grafické enginy, např. `three.js` abstrahuje všechny zmíněné

technologie. Programátor se tak může rozhodnout, kterou z nich použije až na základě toho, na jakém zařízení je aplikace spuštěna. Může tak např. použít WebGL pro rychlejší běh tam, kde je to možné, ale zároveň zachovat co největší kompatibilitu díky CSS 3D nebo Canvas API. Pro vývoj samotné hry je dále k dispozici mnoho herních a fyzikálních enginů, bez kterých se kvalitní hry dnes již neobejdou.

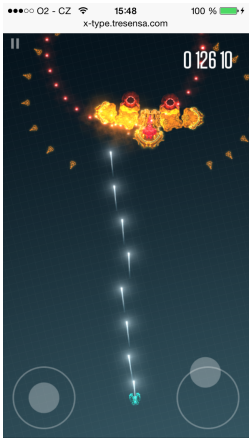
					Zobrazení
HTML5 Canvas	✓	✓	✓	✓	2D
SVG	✓	✓	✓	✓	2D
CSS 3D	✓	✓	✓	✓	3D
WebGL		✓		✓	2D, 3D

Tabulka 4.3: Podpora technologií pro vykreslování v HTML5

Pro ukázku jsem zvolil logickou hru 2048 (viz obr. 4.3a), dále X-Type (viz obr. 4.3b) demonstrující plynulost 2D animací i s poměrně velkým počtem objektů a jako poslední CubeSlam (viz obr. 4.3c), což je jedna z mála her, která v HTML5 u mobilních zařízení využívá 3D zobrazení.



(a) 2048 (2D)



(b) X-Type (2D)

(c) Cube Slam (3D)

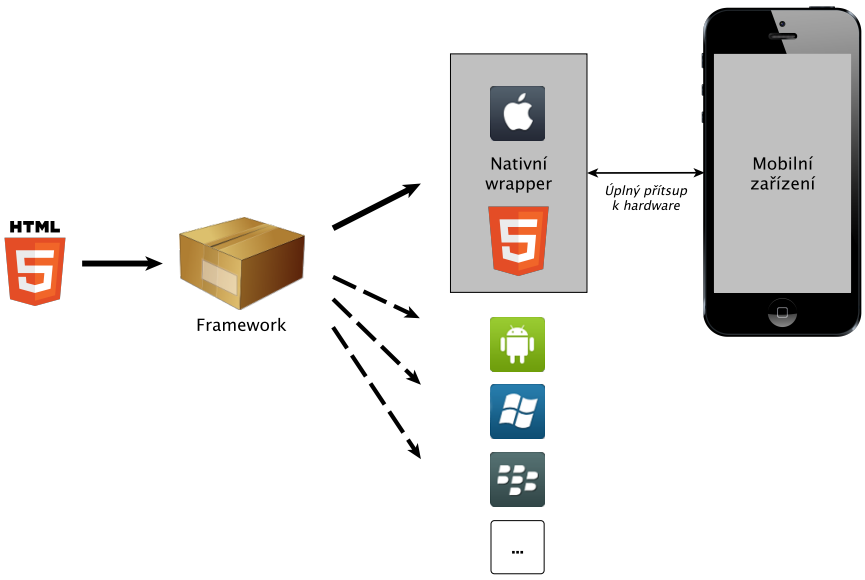
Obrázek 4.3: Ukázky webových HTML5 her

Výhody HTML5 jsou zřejmé, vývojář dokáže pomocí známého jazyka snadno vytvářet aplikace na všechny současné mobilní platformy. Přitom se nemusí učit pracovat s novými nástroji a distribuce je opravdu jednoduchá. Navíc vývoj pokročil natolik, že aplikace mohou využívat i některé hardwarové možnosti zařízení.

Stále je jich ale málo na to, aby se mohly rovnat s nativními aplikacemi, navíc výkonově a přívětivostí uživatelského prostředí také zůstávají pozadu. Vše se projevuje i na statistikách. Uživatelé mobilních zařízení tráví používáním webového prohlížeče pouhých 20% času [30]. Proto jsou cílem HTML5 aplikací ve většině případů v současnosti desktopové prohlížeče. Velkým problémem je také zpoplatňování. V podstatě jediným možným příjmem jsou zobrazované reklamy, neexistuje žádný centralizovaný obchod. Na druhou stranu se vývojáři vyhnou schvalovacím procesům a případný zisk připadne pouze jim. Nevýhodou je samozřejmě také závislost na internetovém připojení, bez něj aplikaci uživatel nespustí.

4.2 Hybridní HTML5 aplikace

Další možností multiplatformního vývoje jsou tzv. hybridní aplikace. V základu jsou to opět webové aplikace, proto jsou možnosti vývoje a použití enginů stejné, jako u čistého HTML5 popsaného v kapitole 4.1. Rozdíl je v tom, že neběží na vzdáleném serveru, ale jsou součástí wrapperu. Wrapper je nativní aplikace na cílovém zařízení, jakási schránka, která zaobaluje HTML5 a zpřístupňuje mu nativní API, tedy hardwarové a systémové prostředky zařízení. Vývojáři tak mohou jednoduše pomocí HTML5, CSS a Javascriptu vytvořit aplikaci chovající se v mnoha ohledech stejně jako aplikace nativní a využívat mnoho jejich výhod. Existuje široké množství frameworků, které tento proces automatizují (viz obr. 4.4). Distribuce, in-app nákupy a další pak probíhají standardní cestou jako u nativní aplikace.



Obrázek 4.4: Princip hybridních aplikací

Co se týká uživatelského prostředí, to hodně závisí na použitém frameworku. Některé z nich sice zprostředkovávají i nativní prvky systému, výsledek ovšem ani tak není stoprocentní. Frameworky se také liší v poskytovaném API, ale obecně oproti čistému HTML5 ve webových prohlížečích je jejich nabídka výrazně širší. Zásadní nevýhodou je, že nelze naplno využít výkonnostní a grafický potenciál zařízení a uživatelské prostředí nenabídne takový komfort. Spíše než na hry se proto hodí na informativní aplikace typu katalog či prezentace.

4.3 Multiplatformní nativní aplikace

Aplikace vytvořené technologiemi založenými na HTML5 mají ze své podstaty obrovskou nevýhodu. Jejich výkonný kód je interpretován internetovým prohlížečem, který jako mezivrstva spotřebuje značnou část výkonu telefonu. Naštěstí ale existuje i způsob skutečně nativního multiplatformního vývoje. K dispozici jsou frameworky, které jeden zdrojový kód překládají a sestavují přímo do nativních aplikací pro různé cílové platformy. Je to tedy technicky náročnější, od čehož se odvíjí i vyšší cena těchto nástrojů, nicméně tak můžeme získat plný přístup k systémovému API a především pak plně využít výkonnostní potenciál zařízení.

4.4 Srovnání a výběr technologie

Nelze jednoznačně říci, který ze způsobů multiplatformního vývoje pro mobilní zařízení je nejlepší, to závisí na potřebách byznysu, účelu aplikace a v neposlední řadě také na schopnosti vývojářů. V tabulce 4.4 je k dispozici srovnání nejpodstatnějších rozdílů. I přesto, že možnosti čistého HTML5 se stále rozšiřují, považuji ho jako vhodné především pro jednodušší aplikace, kde je kladen důraz na rychlou distribuci informací. U her se touto cestou vydají zřejmě spíše lidé, kteří na hře nepotřebují profitovat a nebo hledají tu opravdu nejjednodušší cestu distribuce.

Pokud je to ale myšleno s vývojem vážněji a od hry se očekává finanční návratnost, nabízí se možnost hybridní technologie. Samotný vývoj je stejně snadný, ovšem cena za to, že aplikace lze zpoplatňovat, je nutnost mít u každé cílové platformy zařízený vývojářský účet, což už vyžaduje finanční náklady a musí se také počítat s poměrně složitým schvalovacím procesem. Dle mých zkušeností programy vytvořené tímto způsobem narážejí na problém uživatelského prostředí. Většina z nich nenabízí při používání ani plynulost, ani pohodlí nativních aplikací. U her je

situace jiná. Vykreslování ve 3D naráží na výkonnostní limity, na druhou stranu 2D herní i fyzikální enginy poskytují už plynulou hratelnost i u poměrně složitějších scén. U jednodušších titulů tak hráč ani nemusí poznat, že se nejedná o nativní hru.

Nativní multiplatformní vývoj považuji za nejpokročilejší variantu. Využívá po všech stránkách naplno potenciál zařízení a poskytuje tak nástroj pro profesionální využití. Na druhou stranu je ale používání těchto nástrojů také nejsložitější a vyžaduje naučit se pracovat s novými nástroji a případně i s jazykem. Nástroje samotné jsou velmi sofistikované a od toho se odvíjí i jejich poměrně vysoká cena. Protože lze ale takto vytvořit jak aplikace s nativním UI, tak i pokročilé hry využívající plný grafický výkon, vidím zde velký potenciál do budoucna a rozhodl jsem se pro praktickou část práce využít tuto možnost.

	HTML5	**Hybridní**	**Nativní**
Vývoj pomocí známých technologií	✓	✓	
Okamžitá distribuce bez schvalovacího procesu	✓		
Distribuce v oficiálních obchodech		✓	✓
Zpoplatnění, in-app nákupy, reklamy		✓	✓
Nativní API	částečně	✓	✓
Nativní UI		částečně	✓
Práce offline		✓	✓
Plné využití výkonu zařízení			✓

Tabulka 4.4: Srovnání technologií multiplatformního vývoje

5 Multiplatformní frameworky

V této části práce popisuji některé konkrétní frameworky pro multiplatformní vývoj mobilních aplikací. Pokouším se zmínit nejvýznamnější zástupce všech tří způsobů z předešlé kapitoly, přičemž nejvíce se zaměřuji na nativní multiplatformní způsob, který jsem vybral pro praktickou část práce.

5.1 Mono

Mono je softwarová platforma navržená pro jednoduchý vývoj multiplatformních aplikací. Jedná se o open source implementaci .Net Frameworku od Microsoftu. Hlavními součástmi .Net jsou kompilátor do jazyka CIL[1], běhové prostředí CLI[2] a knihovny. Mono obsahuje kompilátor jazyka C# (Mono C# Compiler), v současnosti plně podporující C# až do verze 4.0. Implementací CLI je Mono Runtime, které poskytuje Just-in-Time (JIT) i Ahead-of-Time (AOT) kompilátory, zavaděč knihoven, garbage collector a systém pro využití vláken. Výhodou je, že dokáže pracovat s jakýmkoliv jazykem fungujícím v CLR (implementace CLI v .Net Frameworku). Mono funguje prakticky na všech používaných platformách, příkladem může být desktopový Linux, Windows, Mac OSX, herní konzole Wii, PlayStation 3 i platformy založené na mobilních ARM procesorech [31]. Na platformě Mono jsou dle mých informací postaveny dva mobilní multiplatformní frameworky Xamarin a Unity.

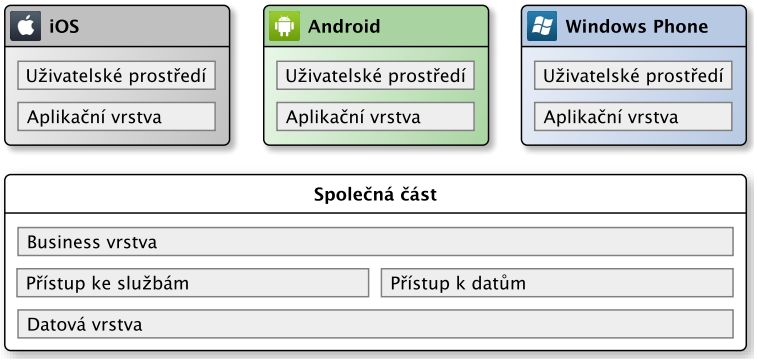
5.2 Xamarin

Za frameworkem Xamarin stojí stejnojmenná společnost, která zároveň sponzoruje Mono projekt. Založil ji v roce 2011 Miguel de Icaza, což je zároveň tvůrce a hlavní vývojář Mona. Stalo se tak poté, co se projektu vzdala společnost Novell, jenž ho do té doby zaštiťovala [32]. Xamarin umožňuje vývoj pro iOS, Android, Windows Phone, Mac OSX a Windows, nicméně já se zaměřím pouze na mobilní systémy.

Mnoho multiplatformních frameworků fungují tak, že zdrojový kód je zcela shodný pro všechna zařízení, což je sice výhodu z hlediska opravdu rychlého vý-

[1] Common Intermediate Language - jazyk v podobě bytekódu pro běhové prostředí CLI
[2] Common Langugae Infrastructure - ECMA specifikace pro spustitelný kód a běhové prostředí

voje, nicméně to má jeden zásadní problém. Tyto frameworky jednotně implementují i uživatelské prostředí, které je ale na každém systému odlišné. Výsledkem je tak totožný vzhled, ale výrazně horší ve srovnání s nativním prostředím. Xamarin proto přistupuje k problému jinak. Zdrojový kód, který se píše v jazyce C#, je společný pouze pro datovou, servisní a business vrstvu. Uživatelské prostředí a aplikační vrstva se pak řeší pro každou platformu zvlášť (viz obr. 5.1). Znamená to tedy práci navíc, ale výsledkem je perfektně vypadající aplikace na všech třech operačních systémech [33].



Obrázek 5.1: Vrstvy aplikace v Xamarinu

Jinak je to v případě her při použití herního frameworku a grafického enginu. K jednotlivým platformám není potřeba z pohledu uživatelského rozhraní přistupovat individuálně, protože jej tvoří grafický výstup. V takovém případě je společná velká většina kódu.

Vývoj pro Windows Phone je sám o sobě postaven na .Net Frameworku, jeho podpora je proto samozřejmostí. Pro iOS a Android nabízí Xamarin knihovny Xamarin.iOS a Xamarin.Android, které zpřístupňují veškeré API těchto systémů. Překlad do nativního kódu probíhá na obou jinak. U iOS musí být z důvodů restrikcí Applu prováděna Ahead-of-Time kompilace a výsledkem je binární soubor pro ARM. Pro Android je používána Just-in-Time kompilace za běhu aplikace. V obou případech se jedná o nativní přístupy.

Přestože samotné Mono je open-source projekt, výše zmíněné knihovny jsou zpoplatněny. Ceny za licence pro jejich používání se vztahují na jednoho vývojáře a jednu platformu za rok. Nejlevnější verze stojí 299 $ pro nezávislé vývojáře, u organizací je to pak 999 $ nebo 1899 $ s rozšířenou zákaznickou podporou (viz příloha A.4). Studentům je nabízena licence se speciální slevou za 99 $. [34]

5.3 Unity

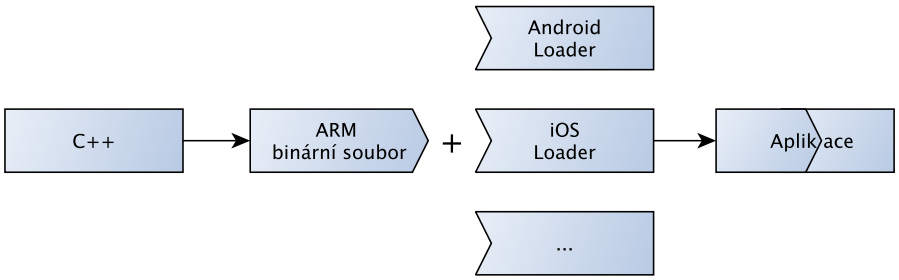
Druhým frameworkem založeným na technologii Mono je Unity, který je cílený čistě na tvorbu her. Jedná se o velmi sofistikovaný ekosystém, který v sobě zahrnuje herní, grafické i fyzikální enginy a poskytuje mnoho nástrojů pro zefektivnění vývoje. Herní scény lze vytvářet v editoru, nabízející živý náhled hry s možností úprav za běhu. Framework byl původně určen primárně pro 3D hry, na podzim 2013 byly ale přidány nástroje speciálně pro 2D. Unity se dá považovat za profesionální nástroj, ze známějších firem jej pro vývoj použila např. firma Rovio u svých Bad Piggies [35]. Výhodou je velké množství podporovaných platforem. U mobilních telefonů jsou to iOS, Android, Windows Phone a BlackBerry, u desktopových systémů Mac, Windows i Linux a z herních konzolí nechybí podpora PS Vita, PS 3, PS 4, Xbox 360, Xbox One a Wii U. Hry lze dokonce spustit ve webovém prohlížeči pomocí pluginu Unity Web Player.

Slabým místem Unity je přístup k nativnímu API a používání systémových prvků uživatelského prostředí. V základu není možné jednoduchým způsobem přistupovat např. k fotoaparátu, kontaktům nebo zobrazovat reklamy. Je to ale pochopitelné, jelikož u her je důležitější počet dostupných platforem. Naštěstí má ale Unity podporu pluginů ve formě knihoven napsaných v nativním kódu konkrétní platformy, např. tedy C, C++, Objective-C, nebo Java, z kterých lze k API přistoupit. Funkce z těchto knihoven je pak možné volat v kódu samotné hry. Existují již hotové knihovny třetích stran, které jsou ale pochopitelně placené. Díky podstatě Mona (viz kap. 5.1) není programování omezeno pouze na jeden jazyk a v Unity tak lze použít C#, UnityScript a Boo.

Součástí Unity je tzv. Asset Store, kde lze zakoupit či získat zdarma velké množství připravených herních prvků, postav, scén, animací, 3D modelů a dalších, což velmi usnadňuje a urychluje vývoj. K dispozici jsou také různé tutoriály a návody pro lidi, kteří s tímto nástrojem teprve začínají. Co se týká ceny samotného Unity, v základní verzi je zdarma, přičemž omezení vývojář pocítí až u komplexnějších her s pokročilou 3D grafikou. Hry lze i prodávat a pro nezávislé vývojáře je to tak velmi dobrá volba. Licence za verzi Pro stojí pro jednoho vývojáře 1500 $ a zahrnuje desktopové platformy, Windows Phone a BlackBerry. Verzi pro iOS a Android je pak nutné zakoupit zvlášť, každou za dalších 1500 $. Lze platit i měsíčně, 75 $ za každou zmíněnou licenci. [36]

5.4 Další nativní frameworky

Existují i další frameworky, jejichž výstupem jsou nativní aplikace a nejsou založeny na Monu. Např. **Marmelade** zdrojový kód C++ překládá do ARM binárního souboru (v případě mobilních procesorů), který následně spojí s loaderem specifickým pro konkrétní platformu, čímž vznikne aplikace připravená pro instalaci do zařízení (viz obr. 5.2). Bohužel není možné používat nativní UI prvky a hodí se tak jen pro tvorbu her. Podobně jsou na tom i další frameworky **Unreal Engine**, **UniGine** a **ShiVa3D**, které technicky fungují na podobném principu. To jsou už ale profesionální nástroje speciálně určené pro 3D hry a využívají je velká herní studia i pro vývoj na desktopy a herní konzole. Integrují v sobě herní, grafické i fyzikální enginy a nástroje pro tvorbu scén, skriptování levelů a další.



Obrázek 5.2: Princip frameworku Marmelade

5.5 Frameworky pro hybridní a HTML5 aplikace

Frameworků pro hybridní aplikace existuje veliké množství, jsou to např. **PhoneGap**, **Telerik AppBuilder** nebo **Apache Cordova**. Všechny fungují na stejném principu, kdy vývoj probíhá v JavaScriptu, HTML5 a CSS, framework poskytuje JavaScriptové SDK pro přístup k nativním funkcím systému a vše je možné zabalit do výsledné aplikace připravené pro distribuci na konkrétní platformu. Uživatelské prostředí není nativní, nicméně framework může mít předpřipravenou sadu ovládacích prvků nastylovaných v CSS, které ty systémové věrně napodobují. Pro vývoj her lze použít některý z široké nabídky herních a fyzikálních enginů. I pro čistě webové HTML5 aplikace existují nástroje usnadňující jejích vývoj na mobilní zařízení, příkladem je **Sensa Touch**.

5.6 Výběr frameworku

Jak jsem již uvedl v kapitole 4.4, největší potenciál vidím v nativním multiplatformním vývoji. Pominu-li profesionální herní enginy, vybíral jsem pouze mezi třemi frameworky Marmelade, Unity a Xamarin. První dva z nich jsou určené primárně pro vývoj her a pokud bych měl mezi nimi rozhodnout, zvolil bych rozhodně Unity, především díky jeho perfektnímu studiu, kde designování a testování probíhá v reálném čase. Mě ovšem nejvíce oslovil Xamarin. V kombinaci s Cocos2D-XNA (viz níže) tvoří velmi solidní základ pro kvalitní 2D hry, zároveň však jako jediný nástroj ze všech zkoumaných poskytuje stoprocentní podporu systémového API a nativního UI. Z mého pohledu se proto jedná o nejperspektivnější variantu a vybral jsem Xamarin pro implementaci praktické části práce.

Podmínkou výběru je samozřejmě také podpora grafického, herního a fyzikálního enginu, bez kterých by hra byla velmi obtížně realizovatelná. V případě Xamarinu je možnost v podstatě jen jedna, ale po všech stránkách vyhovující. K dispozici je volně dostupný herní engine Cocos2D-XNA, portace oblíbeného Cocos2D do jazyka C# vytvořená speciálně pro spolupráci s XNA, což je herní framework od Microsoftu postavený na .Net Frameworku. Podporu iOS a Androidu zajišťuje MonoGame, což je implementace XNA. O fyziku se stará engine Box2D.

					Technologie	**Vhodné využití**
Xamarin	✓	✓	✓		Nativní	Aplikace a hry
Unity	✓	✓	✓	✓	Nativní	Hry
Marmelade	✓	✓	✓	✓	Nativní	Hry
Unreal Engine	✓	✓			Nativní	Hry
UniGine	✓	✓			Nativní	Hry
ShiVa3D	✓	✓		✓	Nativní	Hry
Appcelerator	✓	✓	✓	✓	Nativní	Aplikace a hry
PhoneGap	✓	✓	✓	✓	Hybridní	Aplikace a hry
Apache Cordova	✓	✓	✓	✓	Hybridní	Aplikace a hry
Telerik AppBuilder	✓	✓	✓	✓	Hybridní	Aplikace a hry
Sencha Touch	✓	✓	✓	✓	HTML5	Aplikace a hry

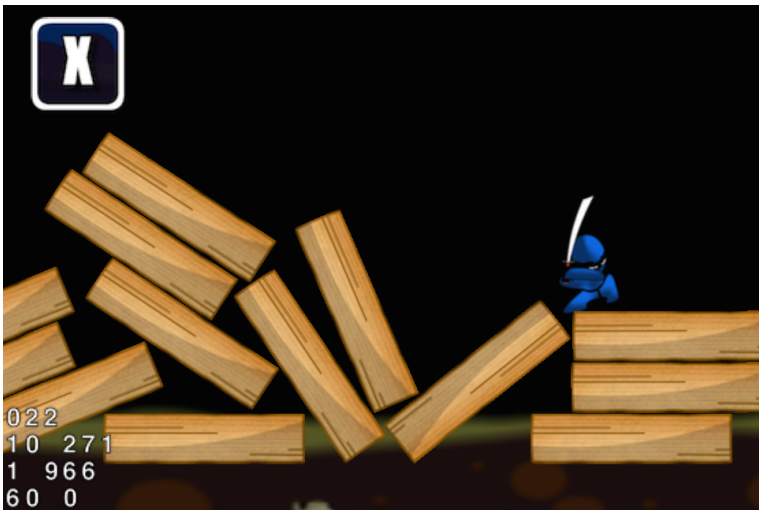
Tabulka 5.1: Srovnání vybraných multiplatformních frameworků

5.7 Herní prototyp

Před implementací samotné hry jsem chtěl ověřit, jak bude Xamarin použitelný v praxi. Vytvořil jsem pro tento účel jednoduchý herní prototyp. Vychází z projektu AngryNinjas, což je oficiální vzorový příklad hry pro Cocos2d-XNA. Zdrojové kódy jsou veřejně dostupné na `<http://github.com/xamarin/AngryNinjas>`.

Na obrázku 5.3 je ukázka tohoto prototypu. Je zde vykreslena malá postava, která při stisknutí displeje v levé či pravé části provádí skok daným směrem. U horního okraje obrazovky se v pravidelném intervalu vytváří objekty a silou gravitace padají k zemi. Postava s objekty interaguje a při skoku je může svou setrvačností posouvat. Při podržení tlačítka v levém horním rohu jsou předměty při kolizi s postavou zničeny. V levé spodní části se vypisují některé významné hodnoty. Důležitý je první řádek, který značí aktuální počet vykreslovaných předmětů a poslední, kde je zobrazen framerate.

V prototypu jsem ověřil vykreslování grafiky a animací, použití fyzikálního modelu, rozpoznávání doteků na obrazovce a především výkonové možnosti frameworku. Hra při testování na telefonu iPhone 5 plynule dokázala vykreslovat více než 60 předmětů, což je dostačující hodnota. Tento prototyp je přiložen jako bonus na CD (viz příloha A.1).



Obrázek 5.3: Ukázka herního prototypu

6 Vývoj hry

V této kapitole popisuji postup při vývoji hry pomocí vybraných frameworků Xamarin a Cocos2d-XNA. Zmiňuji veškeré nástroje, které jsem při implementaci praktické části použil a věnuji se také nastavení celého projektu tak, aby bylo možné hru úspěšně spustit na všech požadovaných platformách.

6.1 Vývojové prostředí

Vývoj hry jsem cílil na všechny platformy podporované Xamarinem, tedy iOS, Android a Windows Phone. Xamarin zajišťuje multiplatformní vývoj až na úroveň vytvoření spustitelné aplikace, ale pro jejich deploy do zařízení a testování používá nativní nástroje jednotlivých systémů. Neobsahuje proto ani vlastní simulátory či emulátory, což je logické jednak z důvodu licenčních podmínek, ale také z pohledu aktuálnosti řešení. Plyne z toho ale jedna zásadní nevýhoda. Jednotlivé platformy mají specifické požadavky na operační systém, na kterém pro ně lze vyvíjet (viz tab. 4.1, str. 15). Android je sice z tohoto pohledu bezproblémový, ale pro iOS lze použít pouze Mac OSX a pro Windows Phone počítač s operačním systémem na Windows.

Dle podmínek společnosti Apple jejich operační systém smí být používán pouze na jejich hardwaru a je tím vyloučeno použití PC s virtualizovaným Macem [37]. Naštěstí to jde ale obráceně a na počítači Mac lze legálně používat Windows buď pomocí některého z virtualizačních nástrojů a nebo použitím programu Boot Camp, který nainstaluje Windows na vlastní diskový oddíl a lze ho poté spustit přímo. Chce-li člověk vyvíjet na jediném stroji pro všechny tři zmíněné mobilní platformy, je to v podstatě jediná možnost. I já jsem se vydal touto cestou a přiklonil jsem se k možnosti virtualizace pomocí nástroje Parallels Desktop. Ten funguje s dostatečným množstvím RAM naprosto spolehlivě a mezi systémy lze jednoduše a okamžitě přepínat. Programy je tak možné současně testovat na všech platformách.

6.2 Použité nástroje

V tabulce 6.1 jsou uvedeny všechny použité nástroje a jejich verze aktuální při dokončení hry. Jak je z ní patrné, pro Android, u kterého jsem měl na výběr, jsem pro vývoj zvolil Mac OSX. Spíše než vývoj je korektnější výraz testování, protože vývoj jako takový lze díky konceptu Xamarinu a tomu, že se jedná o grafickou hru, provádět pouze na jedné platformě, protože velká většina kódu je společná (viz kap. 6.4). Já jsem primárně vyvíjel pro iOS a pro testování jsem kromě simulátorů a emulátorů jednotlivých platforem používal i skutečná zařízení.

Protože se jedná hru, bylo velmi důležité použití nástrojů pro návrh designu a tvorbu grafiky. Pro úpravu obrázků mi velmi dobře posloužil program Pixelmator nabízející práci ve vrstvách a všechny obvyklé grafické nástroje. Hlavním programem pro návrh celkového vzhledu se pro mě stala aplikace Sketch. Ta primárně slouží pro vektorové kreslení, ale ve spolupráci se zařízením iOS na něm dokáže zobrazovat živé náhledy aktuálně upravovaných scén a člověk má tak okamžitou představu o finální podobě hry. Velmi dobře se zde designují i tlačítka a další herní prvky a výsledné elementy se jednoduše exportují do požadovaného formátu.

Počítač	Macbook Pro, Intel Core i5 2,4GHz, 16GB RAM
Primární operační systém	Mac OS X 10.9.2
Vývoj iOS	Xamarin Studio 4.2.4 + Xamarin.iOS 7.2.1.42
Podpůrné nástroje iOS	Xcode 5.1.1, iOS SDK 7.1
Vývoj Android	Xamarin Studio 4.2.4 + Xamarin.Android 4.12.3
Podpůrné nástroje Android	Java 1.7.0_45, Android SDK 4.0.3
Virtualizační nástroj	Parallels Desktop 9.0.24229
Virtualizovaný operační systém	Windows 8.1 64bit
Vývoj Windows Phone	Visual Studio Ultimate 2013 12.0.30110.00
Podpůrné nástroje WP	Windows Phone SDK 8.0
Herní framework	Cocos2d-XNA verze stable z 31.12.2013
Součásti frameworku	Cocos2d, Box2d, MonoGame
Grafické nástroje	Sketch 3.0.1, Pixelmator 3.1, Gimp 2.8.10
Testovací zařízení	viz tab. 9.1, str. 55

Tabulka 6.1: Specifikace vývojového prostředí

6.3 Xamarin Studio

Xamarin pro vývoj nabízí vlastní kompletní řešení Xamarin Studio. Tento nástroj v sobě integruje podporu C#, Mono (viz kap. 5.1, str. 22) a knihoven pro konkrétní platformy Xamarin.iOS a Xamarin.Android, případně Xamarin.Mac, což ale pro tuto práci není podstatné. Podporován je na systémech Mac OSX a Windows. Při instalaci jsou staženy všechny potřebné chybějící závislosti, tedy např. iOS a Android SDK. Pro otestování aplikace se studio dokáže připojit k nativnímu iOS simulátoru, Android emulátoru i k fyzickým zařízením těchto dvou platforem. Zobrazuje konzoli, kam lze pro ladění provádět výpisy aplikace a umožňuje i pozastavení, krokování programu a zobrazování stavu proměnných.

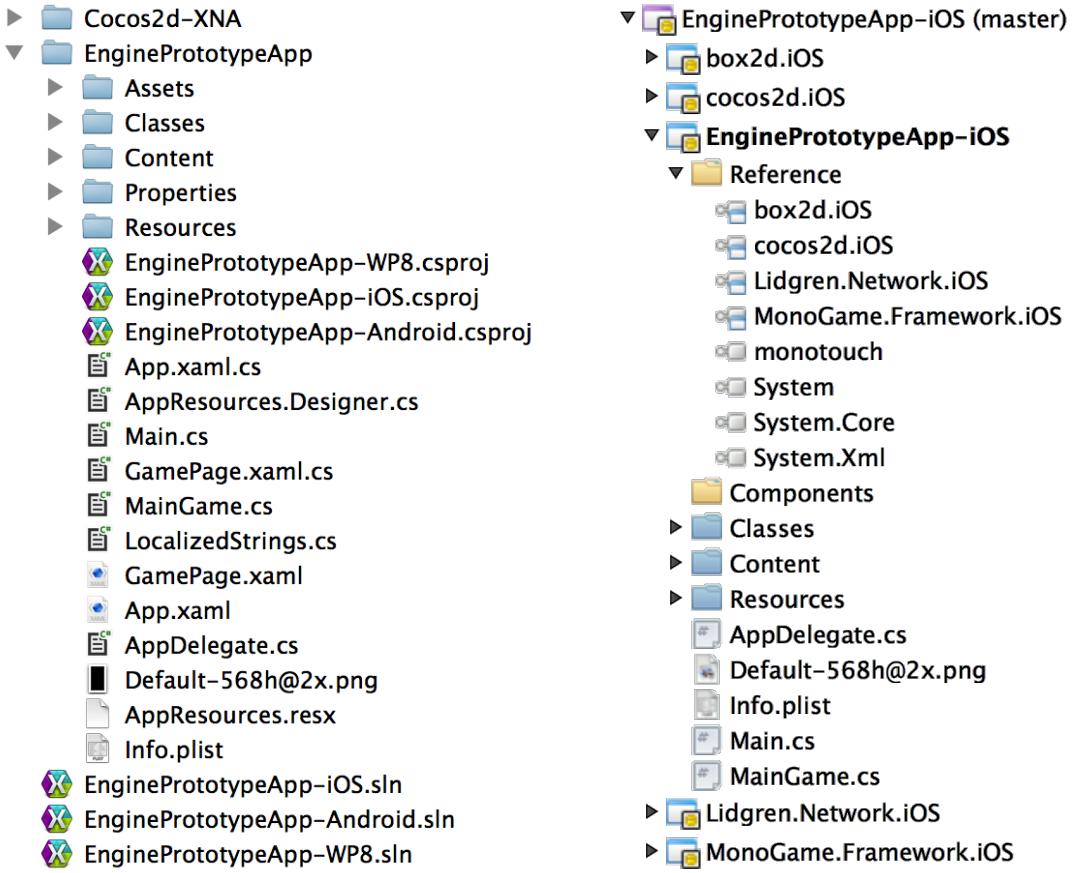
Nástroj jako takový je zdarma, ovšem pro jeho používání je nutné být přihlášen pod Xamarin účtem, ke kterému je vázána licence. Nabízeno je několik variant lišící se cenou a funkcemi. Nejzákladnější z nich je zdarma, dovoluje ovšem vyvinout aplikace pouze s omezenou výslednou velikostí a neumožňuje použít žádné externí knihovny. Já jsem použil verzi Business se studentskou slevou za cenu 99 $ na rok pro každou platformu iOS a Android. Pro Windows Phone není potřeba licence žádná, vývoj probíhá standardním nativním způsobem ve Visual Studiu výhradně na systému Windows. Použil jsem verzi Ultimate 2013. Xamarin pro Visual studio nabízí i iOS a Android plugin, ovšem deploy na iOS zařízení i tak probíhá pomocí vzdáleného počítače Mac, takže jeho vlastnictví se ani v tomto případě nevyhneme.

6.4 Struktura projektu

Celý projekt hry pro všechny tři platformy se nachází na jednom místě, na obrázku 6.1a je zobrazena jeho adresářová struktura. Snažil jsem se vše zorganizovat co nejpřehledněji a tak, aby bylo možné sdílet mezi platformami co nejvíce zdrojů. Následuje seznam nejvýznamnějších složek a souborů:

- `*.sln` - tzv. solution soubory, které tvoří sestavu pro Xamarin Studio v případě Androidu a iOS a pro Visual Studio u Windows Phone. Obsahují informace o použitých projektech a jejich vzájemných referencích. Např. tedy u iOS soubor `EnginePrototype-iOS.sln` v sobě obsahuje referenci na projekt hry, tedy `EnginePrototypeApp-iOS.csproj` a dále pak na projekty všech potřebných frameworků a knihoven Cocos2d.iOS, Box2d.iOS, Lidgren.Network.iOS a MonoGame.iOS (viz obr. 6.1b).

- `*.csproj` - soubory jsou projekty samotné hry. Nemusí v nich být nutně zahrnuty reference na všechny dostupné soubory, čímž se jednotlivé platformy odlišují. Jak je vidět na výše zmíněných obrázcích, u iOS jsou např. úplně ignorovány složky Assets, Properties nebo soubory `*.xaml` a `*.resx`, které jsou specifické pro Android a Windows Phone.
- `Cocos2d-XNA/` - složka obsahující kompletní sadu použitých frameworků včetně Cocos2d, Box2d, MonoGame, Lingren.Network nebo SharpDX. Podobně jako u mého řešení struktury obsahuje každý z nich projekty pro konkrétní platformu, na které je následně ze hry odkazováno.
- `EnginePrototypeApp/` - složka s vlastní hrou.
- `Assets/` - obsahuje grafiku, hudbu a zvuky ve formátu `xnb` (viz kapitola 6.5.2, str. 34) pro použití na Androidu a Windows Phone. Jsou zde např. umístěny i aplikační ikony pro Windows Phone.
- `Classes/` - zde jsou uloženy všechny zdrojové kódy hry a pro všechny platformy jsou zcela společné.
- `Content/` - obsahuje grafiku, hudbu a zvuky v originálních formátech `png` a `mp3` pro použití na iOS.
- `Properties/` - obsahuje soubory specifické pro Android a Windows Phone.
- `Resources/` - tuto složku využívají všechny platformy, iOS zde má např. uloženy aplikační ikony, Android a Windows Phone některé konfigurační soubory.
- `Main.cs` - vstupní bod programu pro iOS a Android.
- `App.xaml` - centrální třída Windows Phone aplikace.
- `Info.plist` - konfigurační soubor iOS aplikace.



(a) Adresářová struktura (b) Struktura projektu iOS

Obrázek 6.1: Struktura projektu

6.5 Postup při vývoji

6.5.1 Nastavení projektů

Prvním krokem je vytvoření prázdných herních projektů pro každou platformu. Díky šablonám, které jsou součástí Cocos2d-XNA je to poměrně jednoduché, k dispozici jsou ve složce:

```
Cocos2d-XNA/ProjectTemplates/Templates/StarterTemplates
```

Tyto projekty je potřeba zkompletovat do podoby uvedené na obr. 6.1a. K jediné kolizi dochází u souboru `Program.cs`, v mém projektu pojmenovaném jako `Main.cs`, který slouží jako vstupní bod aplikace u iOS a Androidu a svým obsahem se odlišují. Jejich sjednocení lze dosáhnout pomocí direktivy překladače (viz kód 1).

Obdobným způsobem je možné rozdělovat kód kdekoliv v aplikaci a na několika málo místech tak skutečně činím. Projekt pro Windows Phone tento soubor neobsahuje. Dále je nezbytné u všech projektů nastavit správné reference na všechny vyžadované frameworky. Podrobnosti jsou uvedeny v tabulce 6.2.

```
...
#if IPHONE || IOS
using MonoTouch.Foundation;
...
#endif
#if ANDROID
using Android.Content.PM;
...
#endif

namespace EnginePrototypeApp
{
#if IPHONE || IOS   // Tato část je určena pro iOS
    [Register ("AppDelegate")]
    public class Application : UIApplicationDelegate {
        private MainGame game;
        public override void FinishedLaunching(UIApplication app){
            game = new MainGame();
            game.Run();
        }
        static void Main (string[] args) {
            UIApplication.Main(args, null, "AppDelegate");
        }
    }
#endif
#if ANDROID          // A tato část pro Android
    [Activity( MainLauncher = true, ... )]
    public class Activity1 : AndroidGameActivity {
        protected override void OnCreate(Bundle bundle) {
            ...
            game.Run(GameRunBehavior.Asynchronous);
        }
    }
#endif
}
```

Kód 1: Oddělení kódu pro různé platformy - soubor Main.cs

Framework	**Reference**
iOS - Box2d	`Cocos2d-XNA/box2d/box2d.iOS.csproj`
iOS - Cocos2d	`Cocos2d-XNA/cocos2d/cocos2d.iOS.csproj`
iOS - MonoGame	`Cocos2d-XNA/MonoGame/MonoGame.Framework/MonoGame.Framework.iOS.csproj`
iOS - Lidgren.Network	`Cocos2d-XNA/MonoGame/ThirdParty/Lidgren.Network/Lidgren.Network.iOS.csproj`
Android - Box2d	`Cocos2d-XNA/box2d/box2d.Android.csproj`
Android - Cocos2d	`Cocos2d-XNA/cocos2d/cocos2d.Android.csproj`
Android - MonoGame	`Cocos2d-XNA/MonoGame/MonoGame.Framework/MonoGame.Framework.Android.csproj`
Android - Lidgren.Network	`Cocos2d-XNA/MonoGame/ThirdParty/Lidgren.Network/Lidgren.Network.iOS.csproj`
WP - Box2d	`Cocos2d-XNA/box2d/box2d.WP8.csproj`
WP - Cocos2d	`Cocos2d-XNA/cocos2d/cocos2d.WP8.csproj`
WP - MonoGame	`Cocos2d-XNA/MonoGame/MonoGame.Framework/MonoGame.Framework.WindowsPhone.csproj`
WP - SharpZipLib	`Cocos2d-XNA/cocos2d/external lib/ICSharpCodeSource/src/ICSharpCode.SharpZLib.WP8.csproj`

Tabulka 6.2: Projektové reference na frameworky

6.5.2 Multimediální soubory

Poslední věcí, ve které se projekty zásadně liší, jsou grafické a hudební soubory. U iOS lze použít klasický formát obrázků png a zvuky v mp3, Android a Windows Phone ovšem v MonoGame frameworku vyžadují `xnb`, což je speciální formát souborů používaný v XNA. Převod do tohoto formátu je komplikovaný z toho pohledu, že XNA jako takové už není Microsoftem podporováno a neexistuje žádný specializovaný nástroj např. ve formě pluginu do Xamarin Studia. Jeden ze způsobů, který jsem dohledal a ukázal se jako funkční, je použití Visual Studia a provizorního Windows Game projektu. Do něj lze veškeré multimediální soubory přidat a po sestavení aplikace se automaticky převedou do `xnb` formátu. Poté je stačí překopírovat do skutečné složky s projektem. Instalaci XNA jsem provedl pomocí neoficiálního balíku ze stránky `<https://msxna.codeplex.com>`, jelikož poslední oficiální balík je dostupný pouze pro Visual Studio 2010.

Aby zůstal projekt přehledný, jsou originální multimediální soubory umístěny ve složce `Content` a xnb pak v `Assets/Content` (viz kap. 6.4). V programu jsou k určení této složky opět použity direktivy překladače (viz kód 2), konkrétně v souboru `MainGame.cs`. Díky tomu se nemusí formát souborů rozlišovat na žádném jiném místě v kódu.

```
#if WINDOWS_PHONE
    Content.RootDirectory = "Assets/Content";
#else
    Content.RootDirectory = "Content";
#endif
```

Kód 2: Určení složky s multimediálním obsahem - soubor MainGame.cs

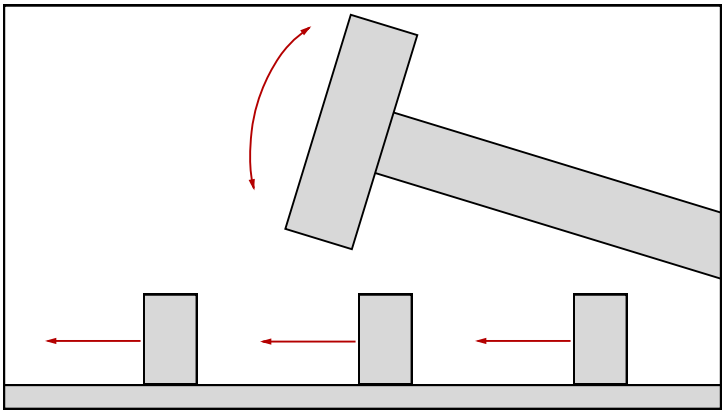
7 Realizace hry

V této části práce se věnuji návrhu a realizaci vlastní hry. Popisuji zde herní objekty, významné třídy, princip výsledné hry a její nejdůležitější prvky.

7.1 Návrh hry

Při návrhu hry jsem se snažil jít konceptuálně co nejjednodušší cestou, kdy z pohledu hráče k ovládání stačí jen několik málo intuitivních gest bez použití tlačítek, na které se na malých displejích mobilních telefonů obtížně trefuje. Tématicky se má jednat o arkádu, u které jsou spíše kratší herní cykly. Že takovéto hry patří na mobilních telefonech mezi ty nejúspěšnější dokazují tituly jako např. Tiny Wings nebo Flappy Bird. Rozsáhlejší, časově náročné hry s dlouhým příběhem, nebo složitým ovládáním dle mého názoru stále patří spíše na velké obrazovky herních konzolí a stolních počítačů.

Přišel jsem tedy s jednoduchým konceptem, kdy se po obrazovce pohybují předměty a hráč má k dispozici kladivo, kterým je může rozbíjet. Cílem je být při úderech co nejpřesnější a vyvinout také co největší rychlost. Kladivo je na svém konci v jednom bodě upevněno a otáčí se tak kolem své osy (viz obr. 7.1). Při dopadu na předmět se odrazí do vzduchu a po chvíli se silou gravitace začne vracet zpátky k zemi. Hráč může kdykoliv přidržením prstu na displeji stlačit kladivo a urychlit tak jeho pád, čímž ovlivní čas a sílu dopadu. Je tedy potřeba postřeh a správné načasování. Za dobře cílené údery do předmětů hráč získává body a naopak v případě minutí přichází o životy, kterých je omezené množství.



Obrázek 7.1: Konceptuální návrh hry

7.2 Herní objekty a významné třídy

Cocos2d je herní engine vhodný přesně pro hry tohoto typu. Tvoří dvourozměrný svět obsahující objekty, které mají své specifické vlastnosti a mohou mezi sebou interagovat. Engine řeší jejich kolize, umožňuje jejich pohyb či různé transformace a s použitím fyzikálního enginu Box2d lze dosáhnout i chování podobající se reálnému světu. Základní objektem celého frameworku je `CCNode`. Sám o sobě není ve hře nikde využitý, ale prakticky všechny herní prvky jsou z něj odděděny, např. herní scény, obrazovky, tlačítka menu nebo obrázky (viz obr. 7.5, str. 41). Tvoří základ i pro konkrétní herní objekty.

7.2.1 BodyNode

Jako základ herních objektů slouží abstraktní třída `BodyNode`. Ta rozšiřuje základní `CCNode` o fyzikální model a texturu. Dědí z ní dva předměty, `DestroyableObject` a `BaseHammer` popsané níže. Následuje přehled jejích nejdůležitějších vlastností a metod:

- `b2Body body` - fyzikální model objektu.
- `CCSprite sprite` - textura objektu.
- `CreateBodyWithSpriteAndFixture()` - používá se pro inicializaci v herním světě. Určuje fyzikální vlastnosti objektu, jeho texturu a pozici, kde má být vytvořen.
- `RemoveCompletely()` - odstraní objekt z herního světa včetně těla a textury.

7.2.2 DestroyableObject

`DestroyableObject` oddděděný od `BodyNode` reprezentuje předmět pohybující se po obrazovce, který lze zničit kladivem. Má fyzické vlastnosti, odolnost a skládá se z určitého množství snímků (třída `Frame`). Jedná se o abstraktní třídu, z níž jsou odděděny konkrétní předměty použité ve hře, které už jen přepisují atributy uvedené v tabulce 7.1. Snímky představují jednotlivé fáze předmětu při destrukci a každý z nich má své rozměry a texturu. Jejich počet není striktně daný, minimálně však musí být dva, jeden pro nový předmět a druhý pro úplně zničený. Ve chvíli nárazu předmět podle aktuální síly kladiva změní svoji fázi a provede se animace destrukce. Na obrázku 7.2 jsou vidět snímky jednoho předmětu použitého

ve hře, kterých je celkem 8. Červený rám vyznačuje rozměry snímku. Pokud má např. tento předmět nastavený atribut `staminaMax` na hodnotu 100, potom úder kladiva se silou 70 způsobí přepnutí na šestý snímek. Předmět se tak fyzicky zmenší a zároveň se spustí rychlá animace od první do šesté textury, čímž vznikne dojem destrukce. Vše je připraveno genericky pro jednoduchou tvorbu nových předmětů (viz kap. 8.1).

Atribut	**Popis**
`staminaMax`	Odolnost předmětu
`objectFrames`	Snímky předmětu
`density`	Hustota
`friction`	Tření
`restitution`	Pružnost

Tabulka 7.1: Atributy definující předměty ke zničení

Obrázek 7.2: Ukázka snímků objektu

7.2.3 BaseHammer

Druhou třídou, která dědí z `BodyNode` je `BaseHammer` představující kladivo. Podobně jako `DestroyableObject` se jedná o abstraktní třídu, kterou pouze pomocí přepsání parametrů lze použít k vytvoření specifického kladiva (viz kap. 8.1). Kladivo použité ve hře je znázorněno na obrázku 7.3. Skládá se ze dvou hlavních objektů, červeně je vyznačeno hlavní tělo kladiva (převzato z `BodyNode`), zeleně potom statický bod třídy `StaticPoint`, kolem kterého se otáčí. Navzájem jsou spojeny pomocí `b2RevoluteJointDef`, což je třída sloužící přesně k tomuto účelu. Je vidět, že textura je oproti tělu výrazně posunuta, prakticky o polovinu své šířky, tak aby se kladivo chovalo přirozeně. V tabulce 7.2 jsou vypsány všechny atributy, které je nutné definovat ve třídě odvozené z `BaseHammer`. K dispozici je několik metod, akcí, kterými kladivo reaguje na události ve hře:

- `ActionPutOnGround()` - kladivo minulo předmět a zůstane ležet na zemi.
- `ActionLiftUp()` - hráč inicioval zdvihnutí kladiva. Pokud leží na zemi, dostane silový impuls a tím se vznese se do vzduchu.
- `ActionStartPushingDown()`- hráč tlačí kladivo k zemi. Kromě gravitace tak na něj působí i další síla urychlující pád.
- `ActionStopPushingDown()` - hráč přestal tlačit kladivo k zemi. To bude dále klesat pouze působením gravitace.
- `ActionStopRising()` - k této události dojde ve chvíli, kdy je dosažena maximální výška. Kladivo se zastaví a silou gravitace následně začne padat k zemi.
- `ActionMakeHit()` - událost nárazu do předmětu, při které dojde k odražení směrem vzhůru úměrně podle aktuální rychlosti. V případě přesného úderu se zvýší aktuální síla a spustí se efekty exploze.

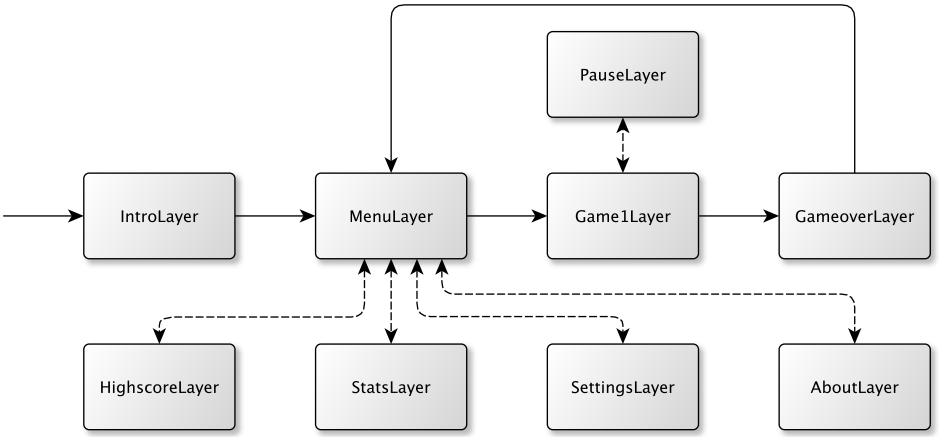
Obrázek 7.3: Ukázka objektu kladiva

Atribut	**Popis**
`liftUpVector`	Vektor určující počáteční zdvih kladiva
`pushDownAccelerate`	Zrychlení při stlačování kladiva
`pushDownSpeed`	Maximální rychlost, kterou kladivo může padat
`maxHeight`	Maximální výška, které kladivo může dosáhnout
`powerIncreasePerfectHit`	Kolik síly kladivo získá při přesném úderu
`powerDecreaseHit`	Snížení síly kladiva při nepřesném úderu
`powerDecreaseMiss`	Snížení síly kladiva při minutí
`powerMin`	Minimální síla kladiva
`powerMax`	Maximální síla kladiva
`hammerLength`	Vzdálenost od těla ke statickému bodu
`hammerEndHeight`	Výška, ve které se nachází statický bod
`hammerHeadHeight`	Výška těla
`hammerHeadWidth`	Šířka těla
`hammerCentering`	Vzdálenost těla od středu obrazovky
`hammerSpritePos`	Pozice textury vůči tělu
`density`	Hustota těla
`friction`	Tření těla
`restitution`	Pružnost těla
`spriteImageName`	Název obrázku pro texturu

Tabulka 7.2: Atributy definující objekt kladiva

7.2.4 Scény a vrstvy

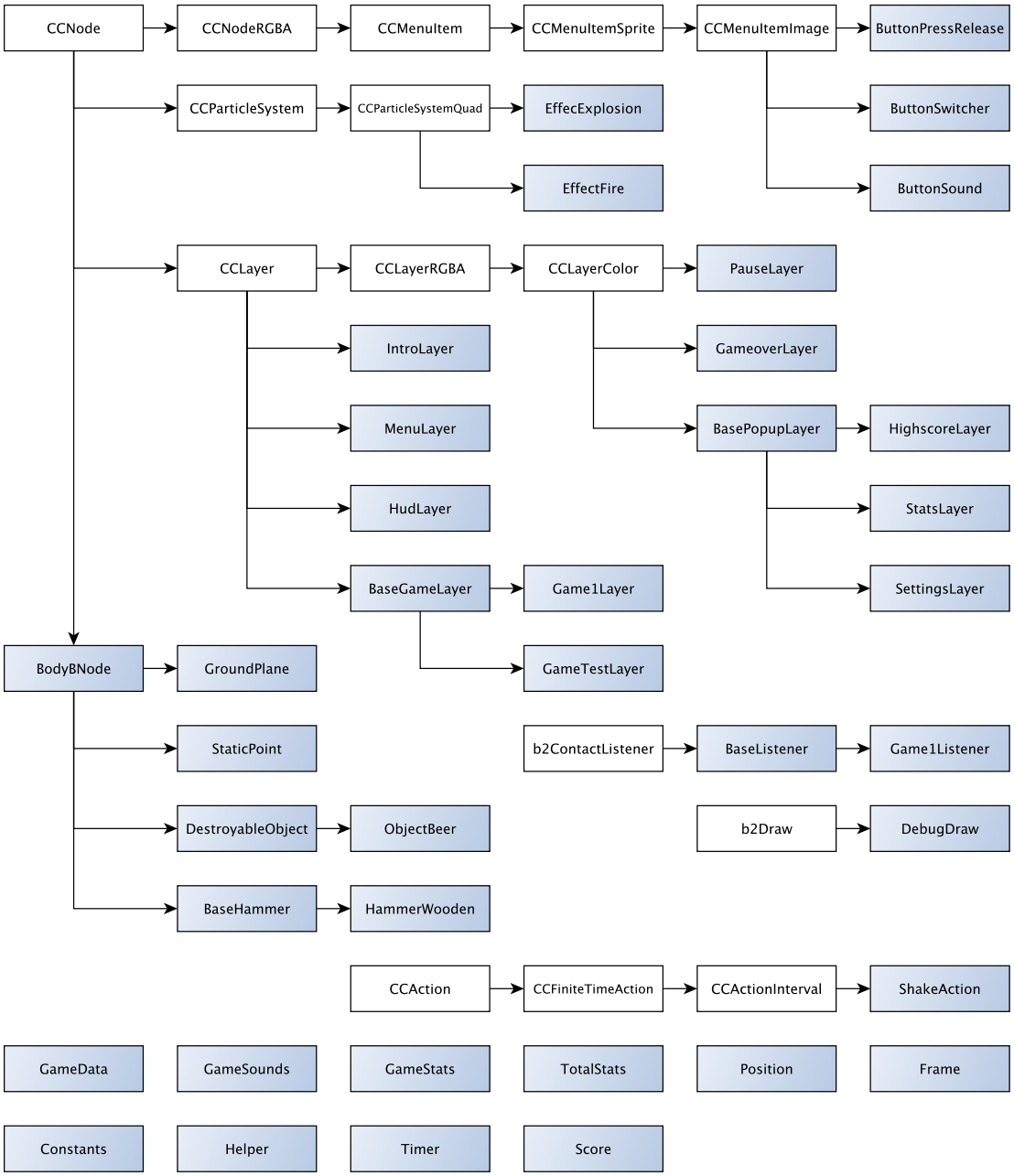
Pro správu herních obrazovek je použita třída `CCDirector`, která vytváří a obsluhuje hlavní okno aplikace a stará se o přechody mezi jednotlivými scénami. Na obr. 7.4 je přehled všech scén použitých ve hře a přechodů mezi nimi.



Obrázek 7.4: Herní obrazovky a přechody mezi nimi

7.2.5 Struktura tříd

Většina použitých tříd ve hře je oděděna z objektů Cocos2d (předpona `CC`) a Box2d (předpona `b2`). Na obr. 7.5 je zobrazena struktura celého projektu, přičemž bíle jsou značeny třídy frameworků a modře třídy vlastní. Šipka vždy míří směrem k potomkovi.



Obrázek 7.5: Struktura použitých tříd

7.3 Popis výsledné hry

Hra se skládá z celkem devíti obrazovek, jejichž souslednost již byla znázorněna výše na obr. 7.4. Dále jsou popsány její nejvýznamnější části.

7.3.1 Design

Věnoval jsem velkou pozornost grafickému návrhu a zpracování tak, aby vše tvořilo kompaktní celek. Naprosto nepostradatelným pomocníkem mi byl přitom vektorový nástroj Sketch, který je určen mimo jiné právě k návrhu mobilních aplikací. První věcí, ke které jsem ho použil, byla výroba tlačítek. Pro každé z nich bylo potřeba vytvořit dva obrázky, pro normální stav a stisknutí. V případě přepínačů v nastavení dokonce čtyři. Jedná se o poloprůhledné png, díky čemuž nepůsobí rušivým dojmem (viz obr. 7.6). Při návrhu designu Sketch umožňuje přes místní síť připojení iOS zařízení, na kterém lze následně sledovat živé náhledy aktuálně upravovaných obrazovek, díky čemuž má člověk jasnou představu o výsledku.

Obrázek 7.6: Ukázka použitých tlačítek

Pro tvorbu předmětu určeného ke zničení jsem vymyslel vlastní techniku. Kamerou jsem na kontrastním pozadí natočil rozbíjení skutečné plechovky a z videozáznamu jsem následně vybral vhodné snímky, které jsem převedl do formátu png. U nich jsem pomocí nástroje Gimp vytvořil průhledné pozadí a dále použil program Pixelmator pro aplikování několika grafických efektů tak, aby předmět působil komiksovým dojmem (viz obr. 7.2, str. 38). V poslední řadě bylo nutné ručně zvýraznit jeho hrany. Celkově včetně všech tlačítek, nápisů, pozadí a herních objektů je ve hře použito 80 různých obrázků.

7.3.2 Herní prvky

Výsledná hra se velmi podobá původnímu návrhu, úkolem hráče je zničit pomocí kladiva co nejvíce předmětů pohybujících se po obrazovce (viz obr. 7.7). Následuje popis všech herních prvků vyznačených na zmíněném obrázku:

a) Kladivo, hlavní objekt hry, na obrázku je zobrazeno v základní poloze. Pokud hráč švihne prste směrem vzhůru, získá impuls a vznese se do vzduchu. Následně ho může přidržením prstu stlačit rychleji k zemi a ovlivnit tak úder do předmětu.

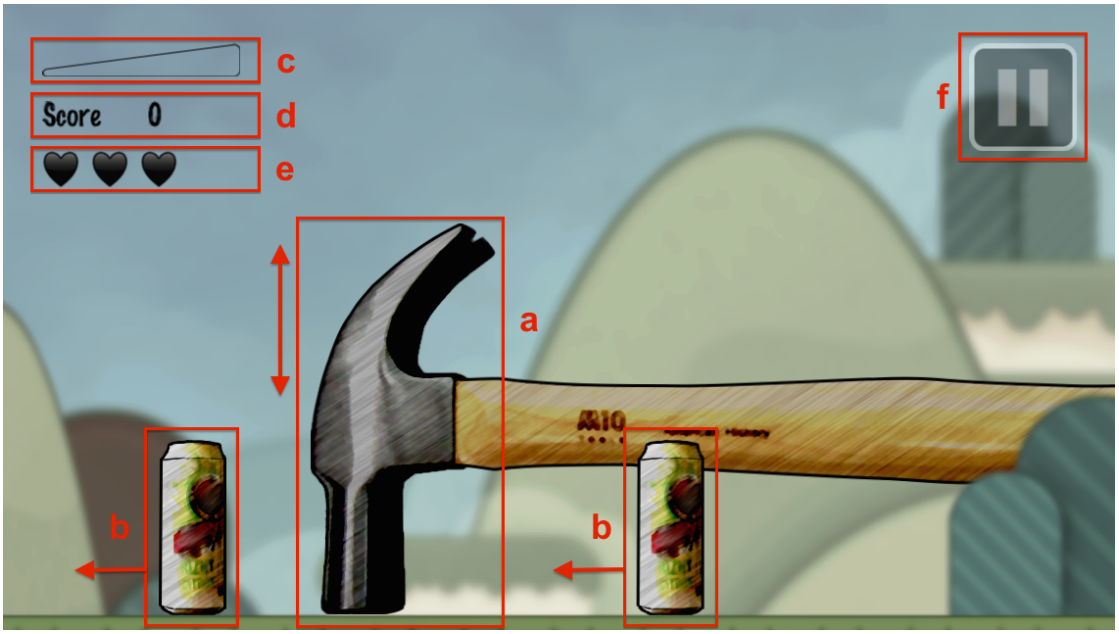
b) Předměty, které lze zničit. V pravidelném intervalu se objevují v pravé části obrazovky a přesouvají se k levému okraji.

c) Síla kladiva znázorněná progress barem. Za každý přesný úder vzroste o 10 jednotek, za nepřesný naopak o 5 klesne. Pokud hráč předmět mine úplně, klesne o 20. Maximální síla může být u každého kladiva odlišná.

d) Skóre, připočítává se při každém úderu do předmětu. Velikost jeho navýšení určují tři faktory: rychlost kladiva, aktuální síla a přesnost úderu. Pokud se některý předmět posune k levému okraji aniž by byl zasažen, odečítá se 100 bodů.

e) Počet životů, odečítají se při každém minutí předmětu a po ztrátě posledního hra končí.

f) Tlačítko pause pro pozastavení hry.



Obrázek 7.7: Herní prvky

7.3.3 Vizuální efekty

Ve hře je pro lepší dojem použito několik vizuálních efektů:

- **Efekt výbuchu** - nastane při přesném úderu do středu předmětu, zachycen je na obrázku 7.12.
- **Rozžhavení kladiva** - úměrně se zvyšuje se silou kladiva. Příklad kladiva s minimální silou je na obrázku 7.7, maximální možné rozžhavení lze vidět stejně jako efekt výbuchu na obrázku 7.12.
- **Chvění obrazovky** - nastane při jakémkoliv úderu do předmětu a umocňuje pocit z nárazu. Jeho intenzita je přímo úměrná síle kladiva. Na obrázku tento efekt bohužel nelze ilustrovat.

7.3.4 Testovací mód

V nastavení hry lze zapnout testovací mód (viz příloha A.3.4). Ve hře jsou poté barevně vykreslovány použité fyzické objekty. Na obrázku 7.8 je znázorněna ukázka v momentě po úderu. Je vidět, že velikost těla zničeného objektu odpovídá velikosti obrázku. Zajímavá je i pravá část obrazovky, kde je již vytvořený nový objekt, ale je zatím skrytý za obrázkem na popředí (vidět je jen fyzický objekt bez textury). Tím je dosaženo stejné hratelnosti na displejích s různými poměry stran (viz kap. 7.4). Testovací mód je ve hře ponechaný především pro demonstraci fyzikálních modelů, zároveň je to ale nezbytná pomůcka při vytváření nových předmětů, kdy je potřeba správně pozicovat obrázky snímků předmětu (viz kap. 8.1.1).

Obrázek 7.8: Ukázka herní obrazovky se zapnutým testovacím módem

7.3.5 Žebříček a statistiky

Skóre je klíčovým údajem ve hře, určuje se podle něj se pořadí hráčů. K dispozici je proto žebříček deseti nejlepších, kde je kromě skóre uveden i herní čas. Kromě toho se po každé dokončené hře zobrazují aktuální statistiky (viz tab. 7.3).

Údaj	**Popis**
Score	Výše skóre
Time	Odehraný čas
Objects	Počet vytvořených předmětů
Hits	Počet zásahů
TOP speed	Nejvyšší rychlost při úderu
TOP accuracy	Nejvyšší přesnost úderu
AVG accuracy	Průměrná přesnost úderu
Destroyed	Počet úplně zničených předmětů

Tabulka 7.3: Statistiky aktuální hry

Dále se také shromažďují statistiky vztahující se ke všem doposud odehraným hrám (viz tab. 7.4). Celkové statistiky a žebříček jsou trvale uložené v zařízení a jsou tak k dispozici i po opětovném spuštění hry.

Údaj	**Popis**
Total games	Celkový počet odehraných her
Total time	Celkový čas strávený hraním
Longest game	Nejdelší čas hry
AVG game	Průměrný čas hry
Total objects	Celkový počet předmětů
Most objects	Nejvyšší počet předmětů ve hře
AVG objects	Průměrný počet předmětů ve hře
TOP speed	Nejvyšší dosažená rychlost při úderu
AVG score	Průměrné skóre ve hře
AVG precision	Průměrná přesnost zásahů
Total hits	Celkový počet zásahů
Most hits	Nevíce zásahů ve hře
AVG hits	Průměrný počet zásahů ve hře
Total destroy	Celkový počet zničených předmětů
Most destroy	Nejvíce zničených předmětů ve hře
AVG destroy	Průměrný počet zničených předmětů

Tabulka 7.4: Celkové herní statistiky

7.4 Kompatibilita

To, že je hra multiplatformní, znamená mimo jiné, že musí být připravena na různé rozlišení displejů a jejich poměry stran. Problém s rozlišením usnadňuje Cocos2d tím, že se v něm nepracuje s fyzickým rozlišením displeje zařízení, nýbrž s relativními jednotkami. Pomocí metody `SetDesignResolutionSize()` se při inicializaci hry nastaví velikost herní obrazovky v těchto jednotkách a framework se automaticky stará o škálování pro konkrétní zařízení (viz kód 3). Co se týká poměru stran, je k dispozici několik možností nastavení:

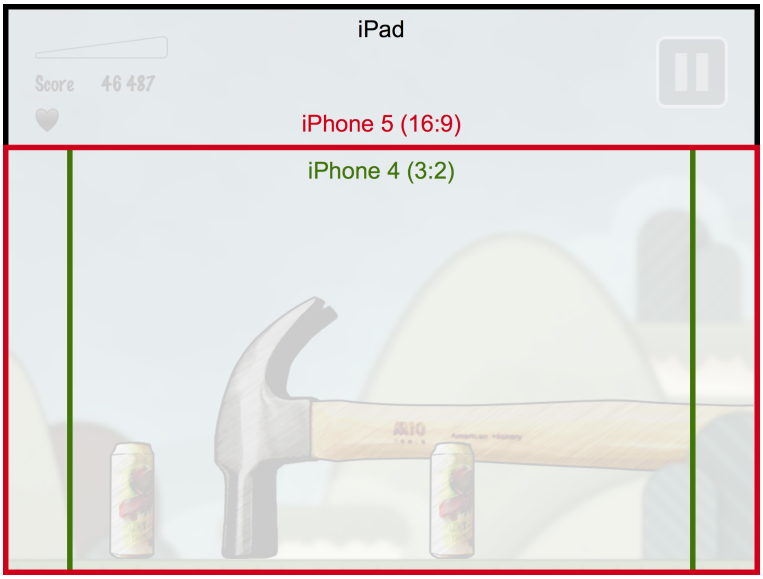
- `ExactFit` - je vyplněna celá plocha displeje a zobrazena je celá herní obrazovka. V případě, že nesouhlasí poměry stran, může u ní dojít k deformaci.
- `NoBorder` - zobrazení 1:1, kdy každý herní bod odpovídá jednomu pixelu. Nedochází proto k deformaci, ale liší-li se rozměry, může být zobrazena jen část herní obrazovky.
- `ShowAll` - vždy je zobrazena celá herní obrazovka bez deformace. V případě rozdílných poměrů stran vzniknou po stranách černé pruhy.
- `FixedHeigh` - vždy je zobrazena celá výška herní obrazovky a šířka je upravena podle displeje tak, aby nedošlo k deformaci. U širokoúhlých displejů je tak zobrazeno více plochy.
- `FixedWidth` - obdoba FixedHeigh, ovšem zachována je v tomto případě šířka. Více plochy je proto k dispozici u méně širokoúhlých displejů.

Na obr. 7.9 je k dispozici schéma ukazující pokrytí herní scény různými zařízeními. Nechtěl jsem žádným způsobem deformovat obraz a zároveň jsem považoval za vhodné co nejvíce využít plochu displeje, aby však zůstala zachována na každém zařízení stejná hratelnost. Jako základní velikost herní obrazovky jsem zvolil 1136 x 640, což je nativní rozlišení iPhonu 5 (značeno červeně). Dále se rozhoduji, zda se jedná o mobilní telefon nebo tablet a podle toho nastavuji výše zmíněné hodnoty:

- **Telefon** - použito nastavení `FixedHeight`. Výška je tedy vždy 640 bodů a šířka závisí na konkrétním zařízení. Na zmíněném obrázku je zeleně vyznačeno zúžení obrazu v případě iPhonu 4 nebo jiného zařízení s podobným poměrem stran displeje. Aby měli hráči na všech zařízeních stejné podmínky, je v pravé spodní části obrazovky umístěn objekt v popředí, který překrývá

vyjíždějící předměty ke zničení. Vzdálenost mezí místem objevení předmětu a potenciálního úderu je tedy vždy konstantní.

- **Tablet** - k dispozici je větší zobrazovací plocha a je tak logické použití nastavení `FixedWidth`. Šířka zůstane plných 1136 bodů a zvětší se prostor nad původní scénou, čímž herní prostor zůstává nedotčen.



Obrázek 7.9: Herní obrazovka na různých zařízeních

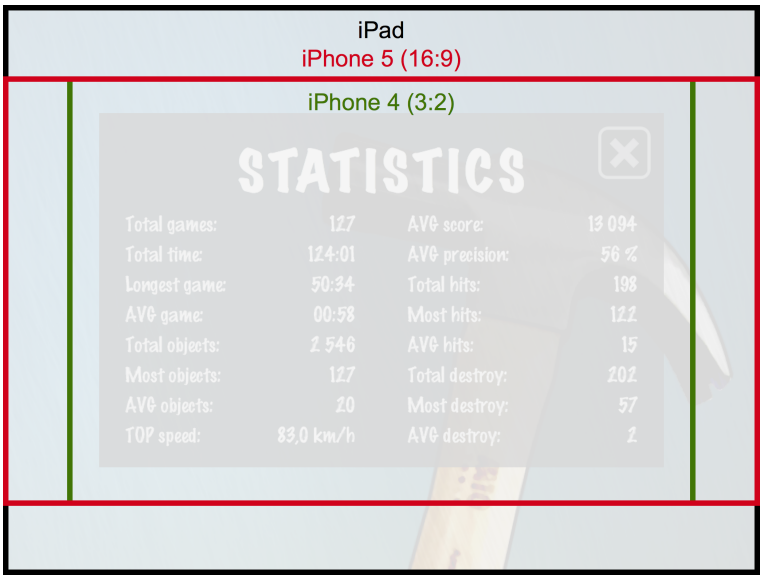
```
if (Helper.IsTablet()) { // Tablet
    CCDrawManager.SetDesignResolutionSize(1136, 640,
                                  CCResolutionPolicy.FixedWidth);
}
else {                      // Telefon
    CCDrawManager.SetDesignResolutionSize(1136, 640,
                                  CCResolutionPolicy.FixedHeight);
}
```

Kód 3: Nastavení velikosti herní obrazovky - soubor AppDelegate.cs

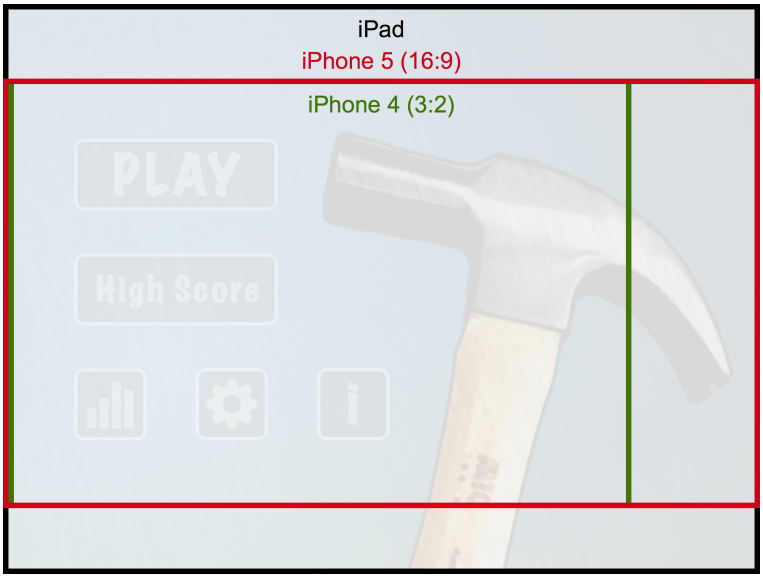
Pozicování objektů

Každý prvek ve hře má svou pozici, přičemž střed souřadnic je v levém spodním rohu obrazovky. Kvůli rozdílnému zobrazování scény ale není možné pozicování provádět absolutně. Např. ve hře je umístění informací o síle, skóre a životech zobrazeno v relativní pozici k levému hornímu rohu obrazovky, tlačítko pro pozastavení

hry naopak k pravému hornímu rohu a předměty se pohybují vždy v konstantní vzdálenosti od spodního okraje. Ve statistikách, nastavení a ostatních obrazovkách s popup oknem jsou veškerá tlačítka a texty pozicovány od středu, tablet v tomto případě přidává prostor do všech stran (viz obr. 7.10). V hlavním menu je použit způsob znázorněný na obrázku 7.11.



Obrázek 7.10: Obrazovka statistik na různých zařízeních



Obrázek 7.11: Obrazovka hlavního menu na různých zařízeních

7.5 Ukázky hry

Následují ukázky hry v následujících fázích:

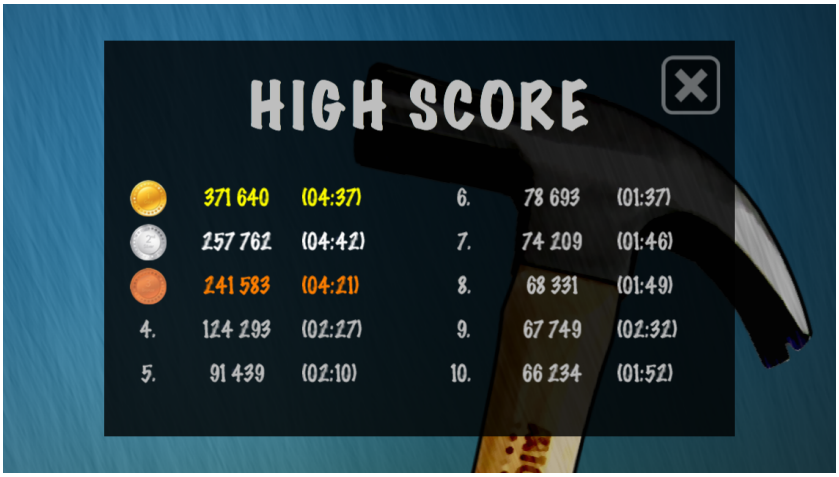
- Herní scéna (viz obr. 7.12)
- Výsledky hry (viz obr. 7.13)
- Žebříček nejvyššího skóre (viz obr. 7.14)
- Celkové statistiky (viz obr. 7.15)
- Hlavní menu (viz obr. 7.16)
- Nastavení (viz obr. 7.17)
- Informace o hře (viz obr. 7.18)

Obrázek s herní scénou zachycuje moment, kdy kladivo udeřilo do předmětu a odrazilo se od něj. Jsou zde vidět částicové efekty výbuchu, jelikož se jednalo o přesný úder. Síla kladiva je téměř na maximu, což je znázorněno téměř plným indikátorem v levé horní části obrazovky a také efektem rozžhavení kladiva. Předmět je po úderu zdeformovaný.

U obrazovky s výsledkem hry se jedná o okamžik, kdy hráč dosáhl nejvyššího skóre a získal tím zlatou medaili. Alternativně se zobrazuje i stříbrná a bronzová, od čtvrtého do desátého místa se pak umístění vypisuje pouze číselnou formou.

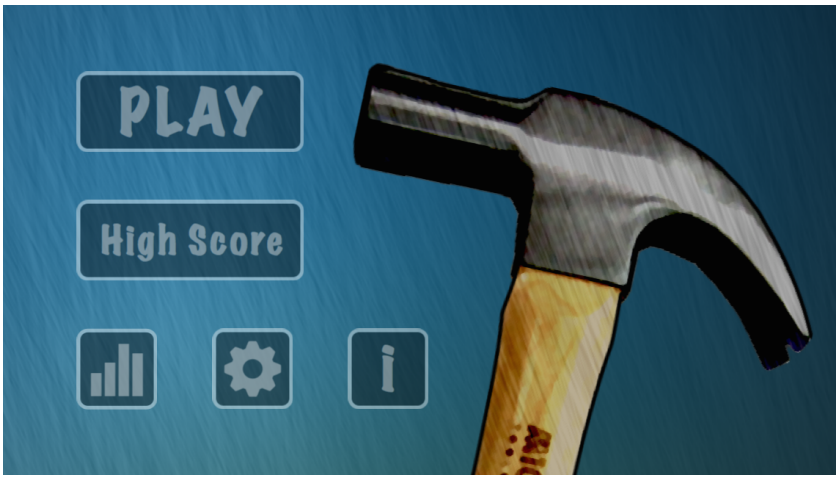
Obrázek 7.12: Ukázka herní scény

Obrázek 7.13: Ukázka výsledků hry

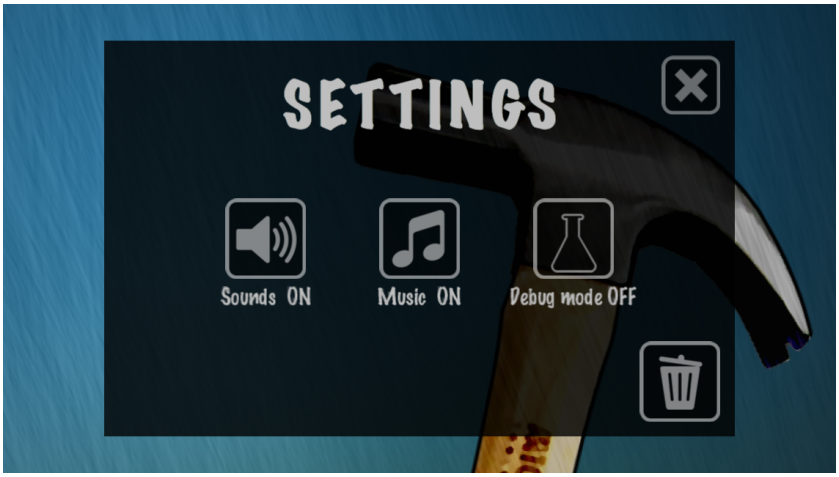


Obrázek 7.14: Ukázka žebříčku nejvyššího skóre

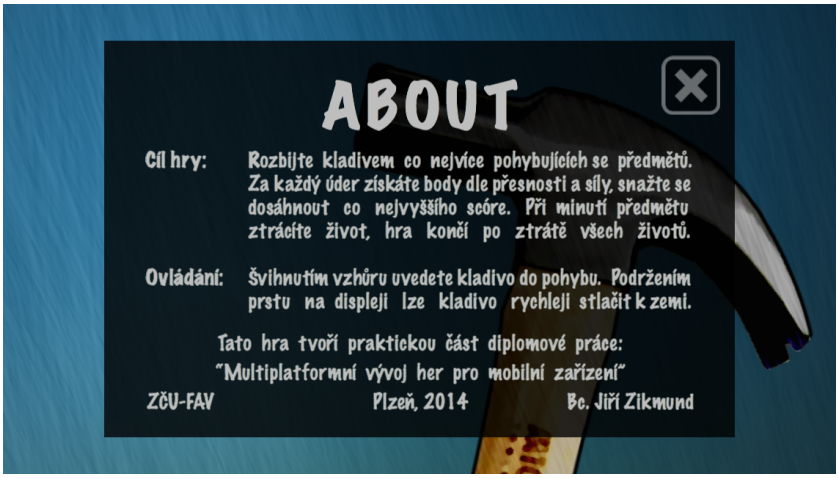
Obrázek 7.15: Ukázka celkových statistik



Obrázek 7.16: Ukázka hlavního menu



Obrázek 7.17: Ukázka nastavení



Obrázek 7.18: Ukázka informací o hře

8 Rozšíření hry

Přestože je hra ve své aktuální podobě funkční a dá se považovat za finální produkt, stále se pro ní dají navrhovat další rozšíření a vylepšení. V této části práce některé z nich popisuji.

8.1 Nové herní objekty

Jednoznačně se pro rozšíření nabízí přidání nových herních objektů. Hra byla od začátku koncipována tak, aby to bylo možné co nejjednodušším způsobem. Následuje návod, jak konkrétně toho docílit.

8.1.1 Předměty pro zničení

V kapitole 7.2.2 popisuji princip fungování předmětů, které lze zničit, jejich vlastnosti a význam tzv. snímků. Základem pro vytvoření nového předmětu jsou obrázky reprezentující vzhled předmětu v jednotlivých fázích destrukce. Jejich počet není striktně daný, ale pro kvalitní vizuální dojem se jich dá doporučit 5 až 10. Příklad postupu při jejich vytváření je popsán ve zmíněné kapitole. Výsledkem musí být série obrázků ve formátu `png` s průhledným pozadím Výška nejvyššího obrázku (typicky prvního snímku) je doporučena kolem 200 pixelů, pro přehlednost je vhodné zvolit názvy jako `obj_predmet_xx.png`, kde `xx` značí číslo snímku.

Další postup se liší v závislosti na platformě. U iOS se musí hotové obrázky zkopírovat do složky `Content/objects/`. V případě Androidu a Windows Phone je nutné provést nejdříve jejich převod do formátu xnb (viz kap. 6.5.2) a zkopírovat do `Assets/Content/objects/`. Ve všech případech se musí v Xamarin Studiu resp. Visual Studiu zahrnout do projektu, aby se staly součástí výsledné aplikace.

V tuto chvíli je již možné vytvořit konkrétní třídu. Její umístění by mělo být ve složce `Classes/Bodies`, dědit musí z `DestroyableObject`. Nejjednodušší je zkopírovat třídu již hotového předmětu `ObjectBeer` (viz kód 4), který je použitý ve hře. Tu pak stačí vhodným způsobem přejmenovat a v metodě `SetProperties()` nastavit požadované vlastnosti. Snímek předmětu reprezentuje objekt Frame, v jehož konstruktoru se nastavují název obrázku (bez přípony png nebo xnb), velikost těla a souřadnice posunu obrázku, pokud jeho střed nesouhlasí se středem těla.

Předmět je také nutné začlenit do hry. Ve třídě Game1Layer, což je hlavní herní obrazovka, stačí upravit metodu generující náhodné předměty (viz kód 5).

```
class ObjectBeer : DestroyableObject
{
    // Kontruktor by měl zůstat nezměněný
    public ObjectBeer(
      b2World world,
      CCPoint position,
      BaseGameLayer gameLayer) : base(world, position, gameLayer)
    { }

    // Metoda definující vlastností předmětu
    protected override void SetProperties ()
    {
        this.staminaMax = 100;          // Odolnost
        this.density = 0.02f;           // Hustota
        this.friction = 0.5f;           // Tření
        this.restitution = 0.1f;        // Pružnost
        this.objectFrames = new Frame[] // Snímky předmětu
        {
            // (obrázek, šířka, výška, offset_x, offset_y)
            new Frame("obj_beer_01", 70,  180,  14, 0),
            new Frame("obj_beer_02", 70,  180,  14, 0),
            new Frame("obj_beer_03", 74,  179,  12, 0),
            new Frame("obj_beer_04", 88,  174,  12, 0),
            new Frame("obj_beer_05", 95,  170,  13, 0),
            new Frame("obj_beer_06", 98,  162,  10, 0),
            new Frame("obj_beer_07", 115, 120,   5, 0),
            new Frame("obj_beer_08", 120, 70,   -5, 0)
        };
    }
}
```

Kód 4: Třída konkrétního předmětu pro zničení - soubor ObjectBeer.cs

```
private DestroyableObject RandomObject {
    get {
        int objectCount = 1; // int objectCount = 2;
        int rand = CCRandom.GetRandomInt(1,objectCount);
        if (rand == 1)
            return new ObjectBeer(world,createPosition,this);
//      else if (rand == 2)
//          return new NovyObjekt(world,createPosition,this);
        else
            return null;
    }
}
```

Kód 5: Metoda pro vytvoření náhodného předmětu - soubor Game1Layer.cs

8.1.2 Objekt kladiva

Přidání nového kladiva je obdobné jako u výše popsaných předmětů ke zničení. Obrázek je v tomto případě potřeba jen jeden, jeho formát na konkrétních platformách, umístění i nutnost přidání do všech projektů jsou ale totožné. Vytvořená třída musí dědit z `BaseHammer` a nastavují se v ní všechny atributy uvedené v tabulce 7.2. Opět je nejsnadnější cestou zkopírování a přejmenování již existující třídy `HammerWooden`, což je kladivo použité ve hře. Pro použití stačí jednoduše upravit metodu `CreateHammer()` ve třídě `Game1Layer` (viz kód 6).

```
private void CreateHammer()
{
    hammer = new HammerWooden(world, this);
//  hammer = new NoveKladivo(world, this);
    AddChild(hammer, Constants.GameLayer_Hammer);
}
```

Kód 6: Metoda pro vytvoření kladiva - soubor Game1Layer.cs

8.2 Žebříček nejvyššího skóre

Ve hře jsou lokálně ukládány herní statistiky a žebříček deseti nejlepších hráčů. V dnešní době je ale oblíbené sdílet a porovnávat výsledky her se svými přáteli. Nabízí se proto možnost hru rozšířit o on-line službu, která by to umožňovala. Např. iOS má k dispozici Game Center, který by byl pro tento účel vhodný. Ukazuje žebříček výsledků mezi přáteli i globálně, umožňuje posílat výzvy pro překonání skóre a hráč může také sledovat dosažené úspěchy přátel.

Protože je ale hra multiplatformní, bylo by zapotřebí mít řešení fungující na všech platformách. V úvahu by tak připadlo provozování vlastního serveru, kam by se výsledky ukládaly a vhodná by byla integrace s některou ze sociálních sítí. Hráči by mohli například porovnávat své skóre s přáteli z Facebooku nebo by mohli sdílet své dosažené výsledky na Twitteru.

9 Ověření funkcionality

Pro ověření správné funkcionality jsem hru otestoval na všech třech vyvíjených platformách, v tabulkách 9.1 a 9.2 jsou uvedena použitá zařízení, simulátory a emulátory[1]. Jako minimální verzi operačního systému jsem v případě iOS stanovil 7.0. Ta je podporována i iPhonem 4 z roku 2010 a ke dni 4.5.2014 tento systém funguje v 88% všech přístrojů [38]. U Androidu jsem jako nejnižší verzi systému určil 2.2, tedy API level 8. Ke dni 1.5.2014 to znamenalo 99% všech používaných zařízení [39]. Windows Phone je podporovaný od verze 8.0.

Zařízení	**Systém**
iPhone 5S	iOS 7.1.1
iPhone 5	iOS 7.1
iPhone 4	iOS 7.1.1
iPad 3	iOS 7.1
LG Optimus L9 II	Android 4.1.2
HTC One V	Android 4.0.3
Samsung GT-S5690	Android 2.3.6
Nokia Lumia 520	WP 8.0

Tabulka 9.1: Zařízení použité při testování

Simulátor / Emulátor	**Systém**
iPhone Retina (3.5-inch)	iOS 7.1
iPhone Retina (4-inch)	iOS 7.1
iPhone Retina (4-inch 64-bit)	iOS 7.1
iPad	iOS 7.1
iPad Retina	iOS 7.1
iPad Retina (64-bit)	iOS 7.1
Windows Phone WVGA 512MB	WP 8.0

Tabulka 9.2: Simulátory použité při testování

Na všech zařízeních fungovala hra dle očekávání, pouze u Windows Phone nebylo možné přehrávat hudbu na pozadí hry, což přisuzuji chybě ve frameworku MonoGame. Ukázky hry na reálných zařízeních jsou k dispozici v příloze A.5.

[1]Emulátor napodobuje softwarové i hardwarové prostředí skutečného zařízení. Simulátor napodobuje pouze softwarové prostředí a má přístup k hardwarovým prostředků hostujícího systému. Pro iOS je k dispozici simulátor, pro Android a Windows Phone emulátory.

10 Závěr

V diplomové práci jsem se věnoval multiplatformnímu vývoji mobilních her. V první části jsem popsal historii mobilních platforem s důrazem kladeným na jejich herní možnosti a přes ně jsem se dostal až k dnešním systémům. Zmapoval jsem jejich současné pozice na trhu z pohledu aktuálního rozšíření, dlouhodobého vývoje a potenciální výnosnosti a určil jako primární platformy pro vývoj iOS, Android a Windows Phone. V další fázi jsem porovnával nativní vývoj pro jednotlivé systémy s multiplatformním způsobem a hledal jsem vhodné nástroje pro vývoj hry. Zvolil jsem framework Xamarin společně s herním enginem Cocos2d-XNA.

Výsledkem práce je mobilní hra fungující na všech třech vybraných platformách, otestovaná na reálných zařízeních. Konceptem se jedná o poměrně jednoduchou záležitost, kdy hráč pomocí kladiva rozbíjí pohybující se předměty. To ovšem nijak nesnižuje samotnou hratelnost, ve hře je využíván fyzikální model, který přináší do jisté míry realistický dojem a vizuální zpracování je také na vysoké úrovni.

Hra je koncipována tak, aby se dala snadno rozšiřovat o nové předměty. Součástí práce je návod, jak toho docílit. Mohla by být také pro snadnou distribuci publikována v obchodech s aplikacemi jednotlivých platforem.

Vývoj mobilních her je dnes velkým byznysem a vývojáři se proto snaží pokrýt co největší část trhu. Nástroje, které jsem v diplomové práci vybral a následně použil při implementaci hry ukazují, že nativní multiplatformní vývoj je dobrou cestou, jak toho dosáhnout. Vývoj se díky nim značně urychluje a vyšší počáteční finanční náklady se v případě úspěšného produktu snadno vrátí. Vzhledem ke svým možnostem je dnes použití multiplatformních nástrojů v podstatě rovnocenné nativnímu vývoji.

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

Seznam použitých pojmů a zkratek

- **Akcelerometr** - součástka nebo přístroj měřící akceleraci, používá se např. pro detekci orientace.
- **AOT,** *Ahead Of Time* - převod do kódu pro cílovou platformu se provádí v době kompilace.
- **API**, *Application Programming Interface* - rozhraní pro interakci aplikací.
- **Arkáda** - žánr počítačové hry upřednostňující hratelnost před realističností.
- **CSS**, *Cascading Style Sheets* - jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML.
- **ECMA**, *European Computer Manufacturers Association* - mezinárodní standardizační organizace.
- **Engine** - softwarový framework tvořící jádro programu.
- **Framerate** - počet vykreslovaných snímků za sekundu.
- **Garbage collector** - metoda automatické správy paměti programu.
- **GPS**, *Global Positioning System* - družicový systém sloužící pro určení polohy kdekoliv na Zemi.
- **Gyroskop** - zařízení pro měření orientace v prostoru.
- **HTML**, *HyperText Markup Language* - značkovací jazyk využívaný zejména pro tvorbu webových stránek.
- **IDE**, *Integrated Development Environment* - vývojové prostředí pro tvorbu programů.
- **In-App nákup** - nákup provedený v rámci aplikace.

- **Infraport** - rozhraní pro přenos dat pomocí infračerveného světla.
- **JavaScript** - objektově orientovaný programovací jazyk.
- **JIT**, *Just In Time* - kompilace programu prováděná při jeho běhu.
- **Joint-venture** - forma spolupráce dvou či více podniků nebo organizací.
- **kB** - kilobyte.
- **Multiplayer** - hra pro více hráčů.
- **Multitouch** - dotykové ovládání založeno na schopnosti rozeznat více dotyků najednou.
- **PDA**, *Personal Digital Assistant* - malý kapesní počítač.
- **QWERTY** - rozložení kláves na klávesnici.
- **RAM**, *Random-Access Memory* - druh počítačové paměti.
- **SDK**, *Software Development Kit* - softwarový vývojový nástroj umožňující tvorbu aplikací pro určitou platformu.
- **Smartphone** - mobilní telefon využívající pokročilý operační systém umožňující instalaci aplikací třetích stran.
- **SMS**, *Short Message Service* - služba pro zasílání krátkých textových zpráv mezi mobilními telefony.
- **Stylus** - drobný nástroj pro ovládání rezistivních dotykových obrazovek.
- **UI**, *User Interface* - uživatelské rozhraní.
- **WAP**, *Wireless Application Protocol* - standard pro přístup k informacím přes mobilní síť.
- **WiFi** - technologie pro bezdrátovou komunikaci v počítačových sítích.
- **WP** - Windows Phone.
- **XNA** - herní framework společnosti Microsoft.

Literatura

[1] Fagerberg, J. *C# Programming: The ultimate way to learn the fundamentals of the C# language*. CreateSpace Independent Publishing Platform, 2013. ISBN 9781494208394.

[2] GSMHistory.com, Temple, S. *Vintage mobiles* [online], 2013 [cit. 10.3.2014]. Dostupné z: <http://www.gsmhistory.com/vintage-mobiles/#hagenuk_mt2000_1994>.

[3] Recombu, Schilling, C. *From Snake to Tegra: the evolution of mobile phone gaming* [online], 2011 [cit. 10.3.2014]. Dostupné z: <http://recombu.com/mobile/news/from-snake-to-tegra-the-evolution-of-mobile-phone-gaming_M14965.html>.

[4] Organic Chemistry. *Snake – The First Mobile Phone Game* [online], 2012 [cit. 10.3.2014]. Dostupné z: <http://ochemclothing.com/way-back-whensdays-snake-the-first-mobile-phone-game/>.

[5] Steel Media Ltd., Wright, C. *A Brief History of Mobile Games: In the beginning, there was Snake* [online], 2008 [cit. 11.3.2014]. Dostupné z: <http://www.pocketgamer.biz/feature/10619/a-brief-history-of-mobile-games-in-the-beginning-there-was-snake/>.

[6] Steel Media Ltd., Wright, C. *A Brief History of Mobile Games: 2000 - A brave new world* [online], 2008 [cit. 11.3.2014]. Dostupné z: <http://www.pocketgamer.biz/feature/10626/a-brief-history-of-mobile-games-2000-a-brave-new-world/>.

[7] Steel Media Ltd., Wright, C. *A Brief History of Mobile Games: 2001 - A Mobile Odyssey* [online], 2008 [cit. 11.3.2014]. Dostupné z: <http://www.pocketgamer.biz/feature/10691/a-brief-history-of-mobile-games-2001-a-mobile-odyssey/>.

[8] Steel Media Ltd., Wright, C. *A Brief History of Mobile Games: 2002 - Wake up and smell the coffee* [online], 2008 [cit. 11.3.2014]. Dostupné z: <http://www.pocketgamer.biz/feature/10705/a-brief-history-of-mobile-games-2002-wake-up-and-smell-the-coffee/>.

[9] SOFTONIC INTERNACIONAL S.A., Wrzosiński, P. *Rajd WRC w autobusie* [online] [cit. 12.3.2014]. Dostupné z: <http://rally-master-pro.softonic.pl/java>.

[10] AOL Inc., Fingas, J. *Nokia ships its last Symbian phones this summer* [online], 2013 [cit. 14.3.2014]. Dostupné z: <http://www.engadget.com/2013/06/12/nokia-ships-its-last-symbian-phones-this-summer/>.

[11] The New York Times, Stone, B. *Play It Again, Nokia. For the 3rd Time.* [online], 2007 [cit. 14.3.2014]. Dostupné z: <http://www.nytimes.com/2007/08/27/technology/27nokia.html>.

[12] Blomedia.pl Sp. z o.o., Żołyniak, M. *Nokia N-Gage – minęło już 10 lat, a my wciąż nie chcemy konsol-smartfonów* [online], 2013 [cit. 15.3.2014]. Dostupné z: <http://komorkomania.pl/2013/10/07/nokia-n-gage-minelo-juz-10-lat-a-my-wciaz-nie-chcemy-konsol-smartfonow>.

[13] Extra Publishing, s.r.o., Smrček, J. *OS Windows Mobile/Phone: strmá cesta historií* [online], 2011 [cit. 14.3.2014]. Dostupné z: <http://www.cnews.cz/os-windows-mobilephone-strma-cesta-historii>.

[14] Apple Inc. *iOS Developer Program* [online], 2014 [cit. 5.4.2014]. Dostupné z: <https://developer.apple.com/programs/ios/distribute.html>.

[15] Visually Inc., Trail, B. *Apple App Store Stats 2013 Infographic* [online], 2013 [cit. 3.4.2014]. Dostupné z: <http://visual.ly/apple-app-store-stats-2013-infographic>.

[16] Apple Inc. *Choosing an iOS Developer Program* [online], 2014 [cit. 10.5.2014]. Dostupné z: <https://developer.apple.com/programs/start/ios/>.

[17] OpenSignal, Inc. *Android Fragmentation Visualized* [online], 2013 [cit. 14.3.2014]. Dostupné z: <http://opensignal.com/reports/fragmentation-2013/>.

[18] Sophos Ltd. *When Malware Goes Mobile* [online], 2013 [cit. 2.4.2014]. Dostupné z: <http://www.sophos.com/en-us/security-news-trends/security-trends/malware-goes-mobile/why-ios-is-safer-than-android.aspx>.

[19] Google Inc. *Registrace vývojáře* [online], 2014 [cit. 10.5.2014]. Dostupné z: <https://support.google.com/googleplay/android-developer/answer/113468?hl=cs>.

[20] Google Inc. *Transakční poplatky* [online], 2014 [cit. 10.5.2014]. Dostupné z: <https://support.google.com/googleplay/android-developer/answer/112622?hl=cs>.

[21] Microsoft. *Microsoft to acquire Nokia's devices and services business* [online], 2013 [cit. 10.5.2014]. Dostupné z: <http://www.microsoft.com/en-us/news/press/2013/sep13/09-02announcementpr.aspx>.

[22] GeekWire LLC, Bishop, T. [online], 2014 [cit. 10.5.2014]. Dostupné z: <http://www.geekwire.com/2014/leak-shows-microsoft-unifying-windows-phone-windows-rt-app-development/>.

[23] CBS Interactive, Woods, B. *A history of BlackBerry in nine iconic handsets* [online], 2013 [cit. 1.4.2014]. Dostupné z: <http://www.zdnet.com/a-history-of-blackberry-in-nine-iconic-handsets-and-one-meh-tablet-photos-7000009777>.

[24] StatCounter. *StatCounter Global Stats* [online], 2014 [cit. 30.3.2014]. Dostupné z: <http://gs.statcounter.com/#mobile_os-ww-monthly-200812-201403>.

[25] Statistic Brain Research Institute. *Mobile Phone App Store Statistics* [online], Statistic Brain, 2.3.2014 [cit. 4.4.2014]. Dostupné z: <http://www.statisticbrain.com/mobile-phone-app-store-statistics/>.

[26] Firtman, M. *HTML5 compatibility on mobile and tablet browsers* [online], 2014 [cit. 9.4.2014]. Dostupné z: <http://mobilehtml5.org>.

[27] Imura, T. *HTML5 Mobile: Hardware Access and Device APIs* [online], 2013 [cit. 8.4.2014]. Dostupné z: <http://girliemac.com/presentation-slides/html5-mobile-approach/deviceAPIs.html>.

[28] Khronos Group. *WebGL Specification* [online], 2014 [cit. 8.4.2014]. Dostupné z: <http://www.khronos.org/registry/webgl/specs/latest/1.0/>.

[29] Webglstats.com. *WebGL Statistics* [online], 2014 [cit. 9.4.2014]. Dostupné z: <http://webglstats.com>.

[30] Business Insider, Inc., Ballve, M. *The Future Of Mobile Development: HTML5 Vs. Native Apps* [online], 2013 [cit. 9.4.2014]. Dostupné z: <http://www.businessinsider.com/html5-vs-native-apps-for-mobile-2013-6?op=1>.

[31] Mono. *What is Mono What is Mono What is Mono What is Mono* [online], 2014 [cit. 15.4.2014]. Dostupné z: <http://www.mono-project.com/What_is_Mono>.

[32] Gigaom, Inc, Taylor, C. *How Xamarin gave Mono a life after Novell* [online], 2011 [cit. 15.4.2014]. Dostupné z: <http://gigaom.com/2011/12/12/xamarin-mono/>.

[33] Xamarin Inc. *Building Cross Platform Applications* [online], 2013 [cit. 15.4.2014]. Dostupné z: <http://docs.xamarin.com/guides/cross-platform/application_fundamentals/building_cross_platform_applications/>.

[34] Xamarin Inc. *Xamarin pricing* [online], 2013 [cit. 16.4.2014]. Dostupné z: <https://store.xamarin.com>.

[35] Unity Technologies. *Unraveling the unending oink* [online], 2012 [cit. 16.4.2014]. Dostupné z: <http://unity3d.com/showcase/profiles/rovio-badpiggies>.

[36] Unity Technologies. *Unity Pricing* [online], 2014 [cit. 17.4.2014]. Dostupné z: <https://store.unity3d.com/products/pricing>.

[37] Apple Inc. *Software license agreement for OS X Mavericks* [online], 2013 [cit. 30.4.2014]. Dostupné z: <https://www.apple.com/legal/sla/docs/OSX109.pdf>.

[38] Apple Inc. *App Store Distribution* [online], 2014 [cit. 10.5.2014]. Dostupné z: <http://developer.apple.com/support/appstore/>.

[39] Google Inc. *Platform Versions* [online], 2014 [cit. 10.5.2014]. Dostupné z: <https://developer.android.com/about/dashboards/index.html>.

A Přílohy

A.1 Struktura CD

K práci je přiložené CD s následující strukturou:

- `/dokument/` - obsahuje tento dokument ve formátu pdf.
- `/dokument/source/` - zdrojové kódy tohoto dokumentu pro aplikaci Lyx.
- `/aplikace/bin/` - sestavená hra pro Android a Windows Phone.
- `/aplikace/source/` - projekty hry pro Xamarin Studio a Visual Studio.
- `/aplikace/source/Cocos2d-XNA/` - framework potřebný k přeložení hry.
- `/aplikace/source/EnginePrototypeApp/` - zdrojové kódy hry.
- `/bonus/` - projekt herního prototypu.
- `/bonus/source/` - zdrojové kódy herního prototypu.

A.2 Instalační příručka

Hra není publikována v žádném oficiálním obchodě s aplikacemi. Pro Android a Windows Phone jsou na CD připravené soubory `apk` resp. `xap` (viz příloha A.1), které lze nainstalovat do zařízení např. z paměťové karty. U Androidu je nutné mít povolenou instalaci aplikace z jiných zdrojů. Aplikace pro iOS musí být podepsaná pro konkrétní zařízení, instalaci proto lze provést pouze přes vývojové prostředí.

Instalace pomocí vývojového prostředí

Projekty s hrou jsou umístěny na přiloženém CD ve složce `/aplikace/source/` (viz příloha A.1):

- `EnginePrototypeApp-iOS.sln` - iOS projekt, určen pro Xamarin Studio ve verzi pro systém Mac OSX. Je nutné mít nainstalované iOS SDK a aktivovanou licenci Xamarin.iOS.
- `EnginePrototypeApp-Android.sln` - Android projekt, určen pro Xamarin Studio ve verzi pro systém Mac OSX nebo Windows. Je nutné mít nainstalované Android SDK a aktivovanou licenci Xamarin.Android.
- `EnginePrototypeApp-WP8.sln` - Windows Phone 8 projekt, určen pro nástroj Visual Studio 2012 nebo 2013. Je nutné mít nainstalované Windows Phone 8 SDK, žádná licence není potřebná.

Projekty mají v sobě uložena všechna potřebná nastavení a reference na frameworky. Po jejich otevření v příslušném studiu tak lze hru ihned jednoduše sestavit a nahrát do simulátoru nebo skutečného zařízení, které je k počítači připojeno. U iOS je situace složitější v tom, že pro spuštění na zařízení je potřeba mít aktivovaný vývojářský účet Apple a pro požadovaný přístroj mít vytvořený tzv. provisioning profil. Detailnější informace ohledně této problematiky popisuje Apple na `<https://developer.apple.com/programs/ios/gettingstarted>`.

Xamarin Studio

Instalační balík Xamarin Studia lze bezplatně stáhnout z `<http://xamarin.com/download>`. Je nutné mít vytvořený Xamarin účet a v rámci něj aktivovanou licenci Xamarin.iOS nebo Xamarin.Android, minimálně ve verzi Indie (viz příloha A.4). Pro úspěšné sestavení je potřeba být pod tímto účtem ve studiu přihlášen.

A.3 Uživatelská příručka

A.3.1 Spuštění

Hra není publikována na oficiálních obchodech s aplikacemi, pro instalaci je proto nutné použít instalační příručku v příloze A.2. Hra se poté objeví ve standardní nabídce mezi ostatními programy. Spuštění proběhne stisknutím příslušné ikony.

A.3.2 Hlavní menu

Po načtení hry hráče přivítá hlavní menu (viz obr. A.1), které nabízí následující možnosti:

a) PLAY - spustí hru.

b) High Score - zobrazí žebříček deseti nejlepších hráčů.

c) Statistics - zobrazí celkové herní statistiky.

d) Settings - zobrazí nabídku s nastavením.

e) About - zobrazí informace o hře.

Obrázek A.1: Návod k použití - hlavní menu

A.3.3 Cíl hry a ovládání

Úkolem je získat co nejvíce bodů za rozbíjení předmětů pomocí kladiva. Při spuštění hry se kladivo nachází v základní poloze, leží na zemi. Gestem švihnutí prstu vzhůru dostane impulz, vznese se do vzduchu a po chvíli začne silou gravitace opět klesat. Hráč může kdykoliv během letu přidržením prstu na obrazovce stlačit kladivo rychleji k zemi a tím ovlivnit moment nárazu. Předměty se objevují v pravé části obrazovky a posouvají se k levému okraji, při jejich zásahu se kladivo odrazí opět vzhůru. V případě tzv. perfektního úderů, kdy kladivo zasáhne střed předmětu, se zvýší síla kladiva. Za každý úder hráč získává body, čím větší je rychlost, přesnost a síla kladiva, tím více bodů se připočítá. v případě, že předmět nezasažen opustí obrazovku, odečítá se 100 bodů. Při minutí předmětu a dopadu na zem hráč přichází o jeden svůj život, při ztrátě všech životů hra končí.

A.3.4 Nastavení

Nastavení (viz obr. A.2) nabízí následující volby:

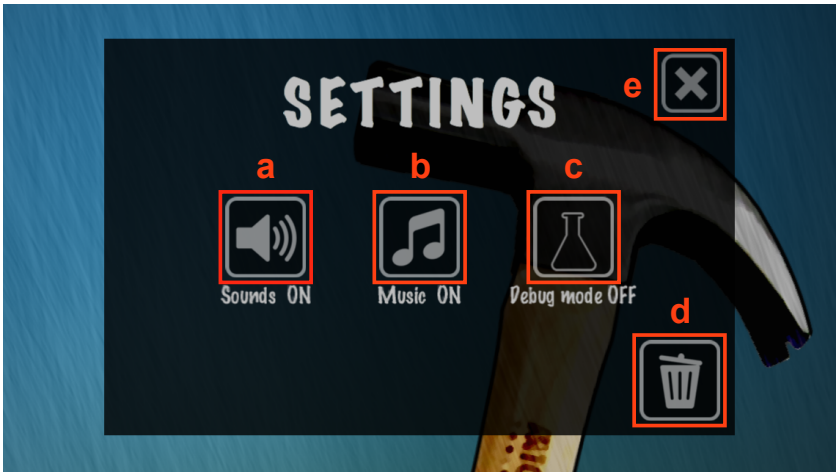
a) Zvuky - zapnutí/vypnutí zvuků.

b) Hudba - zapnutí/vypnutí hudby.

c) Testovací mód - zapnutí/vypnutí testovacího módu. Při hře jsou při aktivaci tohoto módu viditelné použité fyzikální objekty.

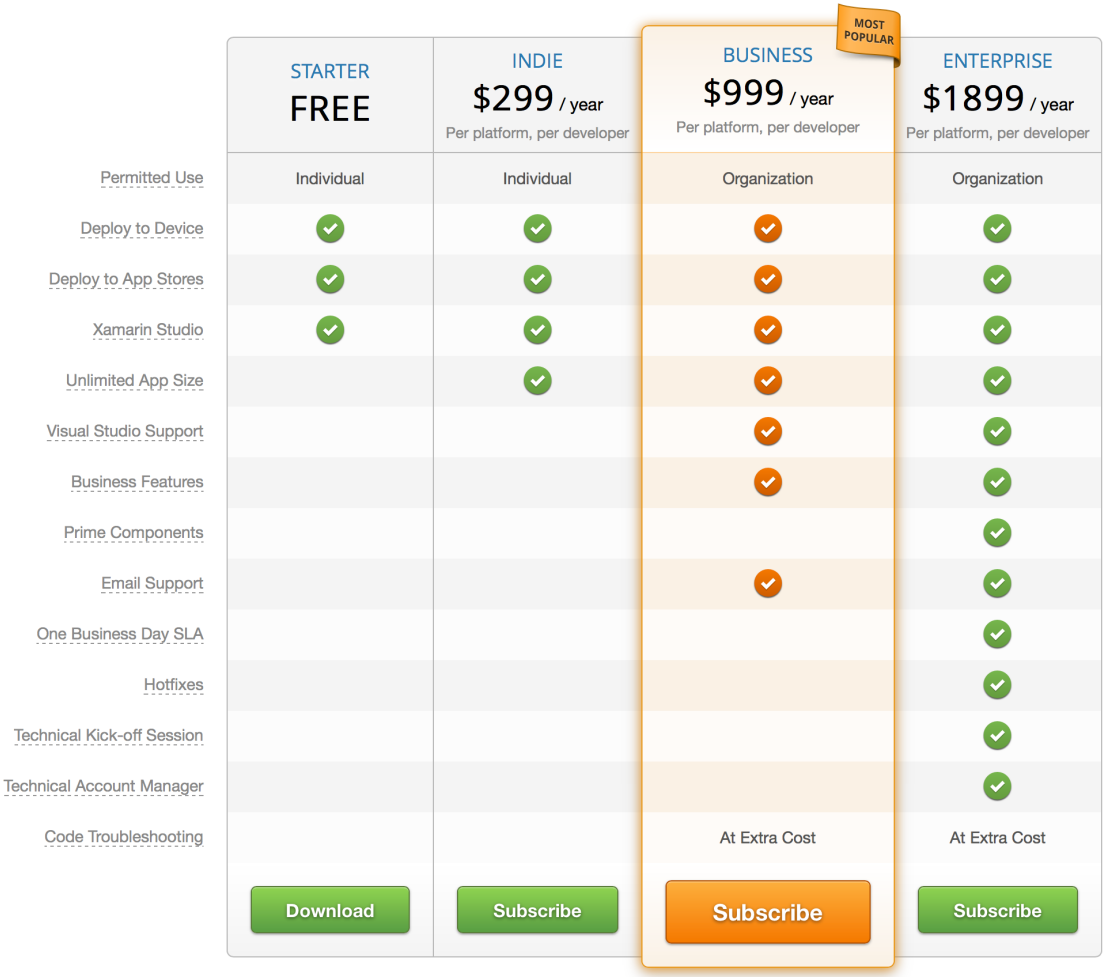
d) Reset hry - smazání všech herních statistik a žebříčku deseti nejlepších hráčů.

e) Návrat - návrat do hlavní nabídky.



Obrázek A.2: Návod k použití - nastavení

A.4 Kompletní ceník Xamarin



Obrázek A.3: Kompletní ceník nástroje Xamarin [34]

A.5 Ukázky hry na reálných zařízeních

Obrázek A.4: Ukázka hry na zařízení iPhone 5

Obrázek A.5: Ukázka hry na zařízení HTC One V

Obrázek A.6: Ukázka hry na zařízení Nokia Lumia 520

Obrázek A.7: Ukázka statistik na zařízení iPhone 4

Obrázek A.8: Ukázka hlavního menu na zařízení iPhone 5S

Obrázek A.9: Ukázka žebříčku nejvyššího skóre na zařízení LG Optimus L9 II

Obrázek A.10: Ukázka hry na zařízení iPad 3