Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Logická hra pro iOS

2019

Ivo Michalec

Vedoucí práce:
Mgr. Tomáš Kühr, Ph.D.

Studijní obor: Aplikovaná informatika,
prezenční forma

Bibliografické údaje

Autor:	Ivo Michalec
Název práce:	Logická hra pro iOS
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2019
Studijní obor:	Aplikovaná informatika, prezenční forma
Vedoucí práce:	Mgr. Tomáš Kühr, Ph.D.
Počet stran:	33
Přílohy:	1 CD
Jazyk práce:	český

Bibliograhic info

Author:	Ivo Michalec
Title:	Logic game for iOS
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2019
Study field:	Applied Computer Science, full-time form
Supervisor:	Mgr. Tomáš Kühr, Ph.D.
Page count:	33
Supplements:	1 CD
Thesis language:	Czech

Anotace

Cílem práce je vytvořit jednoduše ovladatelnou logickou hru Ubongo pro iOS. V úvodu jsou popsána pravidla originální stolní hry Ubongo. Dále práce obsahuje informace o všech technologiích použitých při návrhu a implementaci aplikace. Součástí práce je i uživatelská příručka a přehled možností dalšího rozšíření hry.

Synopsis

The aim of the thesis is to create an iOS game which has easy controls. The game is based on a logic game called Ubongo. The introduction describres the rules of the original board game. Furthermore, the thesis contains information about all technologies used in design and implementation of the application. The work also includes a user manual and an overview of the possible further extention of the game.

Klíčová slova: iOS; Swift; Xcode; logická hra; Ubongo

Keywords: iOS; Swift; Xcode; logic game; Ubongo

Rád bych tímto poděkoval vedoucímu bakalářské práce Mgr. Tomášovi Kührovi, Ph.D. za ochotu a cenné rady při vytváření této práce. Chtěl bych také poděkovat všem učitelům a vědeckým pracovníkům za předané vědomosti, zkušenosti a přátelský přístup.

Místopřísežně prohlašuji, že jsem celou práci včetně příloh vypracoval samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

datum odevzdání práce podpis autora

Obsah

Seznam obrázků

Seznam tabulek

Seznam zdrojových kódů

1 Úvod

Práce se zabývá návrhem a implementací logické hry pro operační systém iOS na platformě iPhone od společnosti Apple. V úvodní kapitole je popsána původní desková hra, kterou jsem se nechal inspirovat při vytváření vlastní verze. Kapitola také popisuje všechny použité technologie i vývojové prostředí pro programování a návrh hry včetně jejího grafického uživatelského prostředí. Pro implementaci hry byl použit programovací jazyk Swift. Grafické prostředí bylo vytvořeno pomocí online MetaPost previewer, jehož tvůrcem je Troy Henderson.[10] Dále práce obsahuje programátorskou část, kde je popsán výběr herní technologie a struktura celého projektu. Nacházejí se tam ukázky kódů ve Swiftu, v MetaPostu a podrobný popis všech tříd. Součástí práce je i uživatelská příručka, která uživatele seznámí s prostředím a ovládáním hry. Celý popis ovládání je doplněn obrázky ze hry. V poslední kapitole jsou možnosti dalšího rozšíření hry.

1.1 Desková hra Ubongo

Ubongo je vlastně spíše hlavolam než-li hra. Nejlépe ji vystihuje označení logická hra. Je to herní žánr, u kterého je potřeba využití logického myšlení hráče. Autorem této hry je Grzegorz Rejchtman.[1]

Obrázek 1: Obsah balení deskové hry Ubongo. Převzato z [1]

1.1.1 Pravidla a principy hry

Hra je určena pro 2 až 4 hráče starší osmi let. Každý hráč má k dispozici 20 kostek (dílků skládačky). Kostky jsou tvořeny jednotlivými čtverečky, které jsou poskládané do různých tvarů. Tvary kostek připomínají kostky z celosvětově známé ruské hry Tetris [3]. Každý z hráčů má před sebou hrací kartičku, na které je prázdná deska a 6 sad kostek (obrázek 2). Hodem klasické hrací kostky se na začátku kola zvolí sada kostek, která se při skládání použije. Každý z hráčů poté skládá desku pouze pomocí kostek z určené sady. Kostky se mohou libovolně otáčet i převracet. Na složení desky mají hráči časový limit. Hráči, kteří

stihnou desku složit v limitu, jsou odměněni možností vylosovat si drahokam. Hráč, který nestihl složit desku v časovém limitu, nemá možnost losovat o drahokamy. O vítězi pak rozhodne hodnota samotných drahokamů, nikoliv jejich počet.

Obrázek 2: Hrací kartička. Převzato z [2]

1.2 Vlastní pojetí hry

Vytvořená hra, která je v této práci popsána, je pouze inspirovaná originální hrou Ubongo. Některé herní principy byly úplně odstraněny. Například zde není žádné házení kostkou, sbírání drahokamů nebo měření času. Hra má také pozměněná pravidla a je pouze pro jednoho hráče. Už se tedy nejedná o společenskou zábavu, ale o oddychovou skládací logickou hru. Dokonce je hra o trochu jednodušší, protože zde není možné kostky převracet, ale pouze otáčet. Důvodem pro zrušení převracení kostek bylo nejen jednodušší ovládání, ale i snížení obtížnosti při skládání desky s více kostkami.

1.3 Vývojové prostředí

Společnost Apple má vlastní vývojové prostředí zvané Xcode [4]. Lze v něm vytvářet aplikace pro iPhone, iPad, Mac, Apple Watch nebo Apple TV. Xcode podporuje programovací jazyky Swift, C, Objective-C, C++, Objective-C++, Java, AppleScript, Python, Ruby a Rez.

1.3.1 Xcode

Xcode [4] je integrované vývojové prostředí (IDE) vytvořené společností Apple. Obsahuje sadu nástrojů pro vývoj softwaru operačních systémů Mac OS X [5] a iOS [6]. Byl poprvé představen v roce 2003. Aktuální verze Xcode je zdarma k dispozici na Mac App Store. Registrovaní vývojáři si mohou stáhnout předchozí verze přes vývojářské webové stránky https://developer.apple.com. Xcode zahrnuje editory, kompilátory a další nezbytné nástroje potřebné pro usnadnění vývoje. Obsahuje také simulátory všech Apple zařízení pro testování vyvíjeného softwaru. Prostředí Xcode je navrženo tak, aby fungovalo nejlépe v jednom okně. Xcode umí sestavit univerzální binární soubory, které umožňují běh softwaru na PowerPC a na platformách s procesory Intel zahrnující 32-bit a 64-bit verze. Vývojové prostředí Xcode je velmi přívětivé a je možné si ho přizpůsobit.

Obrázek 3: Vývojové prostředí Xcode 10.2.1 a simulátor iPhonu 6 Plus

1.3.2 iOS simulátor

Simulátor operačního systému iPhonu umožňuje použít všechny funkce iPhonu, včetně otáčení zařízení na šířku a simulaci dotyku pomocí kurzoru. Xcode je vybaven simulátory nejen pro všechny modely zařízeni iPhone, ale i pro iPad, iPad Air, iPad Pro, Apple TV a Apple Watch. Registrovaní vývojáři mají možnost bezplatně nainstalovat vyvíjený software (aplikaci) na 3 zařízení. Po instalaci je možné aplikaci testovat 48 hodin. Poté je potřeba aplikaci instalovat znovu. Tohle omezení je možné obejít zaplacením vývojářské licence na rok za 99 USD.

1.3.3 Swift

Swift [7] je programovací jazyk od společnosti Apple. Poprvé byl uveden 2. června 2014 na WWDC konferenci. Autorem jazyka Swift je vývojář Apple Inc. Chris Lattner. Jazyk je kompilovaný pomocí LLVM [8] a je staticky typovaný. Umožňuje hned několik programovacích paradigmat: objektově orientované, multiparadigmatické, strukturované, imperativní (procedurální). Veškerá dokumentace k jazyku Swift je dostupná na vývojářských stránkách Applu https://developer.apple.com/swift.

Tabulka 1: Historie verzí jazyka Swift

Datum vydání	Verze
9.9.2014	Swift 1.0
22.10.2014	Swift 1.1
8.4.2015	Swift 1.2
21.9.2015	Swift 2.0
13.9.2016	Swift 3.0
19.9.2017	Swift 4.0
29.3.2018	Swift 4.1
17.9.2018	Swift 4.2
25.3.2019	Swift 5.0

1.4 Návrh uživatelského rozhraní

Uživatelské rozhraní aplikace Ubongo je navrženo tak, aby bylo pro uživatele co nejvíce přehledné a jednoduše pochopitelné. Tlačítka na úvodní obrazovce jsou velká, aby bylo jednoduché na ně kliknout. Díky tomu, že jsou tlačítka jednotlivých úrovní velká, tak bylo možné do nich umístit i nějaké informace o dané úrovni.

Při návrhu uživatelského rozhraní jsem se držel těchto pravidel:

- minimalizace zátěže paměti uživatele;
- maximální konzistence ovládacích prvků;
- umístění uživatele do centra ovládání;
- maximální poznání uživatelů – uživatelská testování.

Právě uživatelská testování mne stále nutila vylepšovat vzhled aplikace. Nechal jsem aplikaci testovat mezi přáteli a zaznamenával jsem si jejich připomínky k ovladatelnosti a orientaci v aplikaci. Rozumné připomínky jsem vzal v potaz a následně jsem uživatelské prostředí podle nich upravil. Jedna z těch větších změn byla právě zvuková odezva po kliknutí na tlačítko. Zvuk kliknutí jsem vytvořil a vygeneroval pomocí programu GarageBand [9]. Další připomínka se týkala chybějícího stavového řádku, který slouží například pro sledování aktuálního času. Jak zvuky klikání, tak i možnost zobrazení stavového řádku si může uživatel v možnostech aplikace zapnout, respektive vypnout podle vlastní potřeby. Více informací o možnostech aplikace je popsáno níže v uživatelské příručce 3.5.

1.4.1 MetaPost

Veškerá grafika ve hře je vytvořena pomocí MetaPostu [10]. MetaPost je programovací jazyk a zároveň interpret tohoto jazyka. Jazyk vznikl v roce 1994. Jeho autorem je John D. Hobby a jazyk byl vyvinut pomocí programovacího jazyka C vývojářem Taco Hoekwater. Po zkompilování kódu v MetaPostu je na výstupu vygenerován soubor s vektorovou grafikou. Ukázka kódu v MetaPostu a výsledného grafického souboru je v programátorské části 2.2 této práce.

2 Programátorská část

Tato sekce stručně popisuje strukturu a technologii, kterou je sestavena aplikace Ubongo. Jsou zde ukázky kódů, popis jednotlivých tříd a jejich instancí.

2.1 Výběr herní technologie

Při zakládání nového projektu v Xcode je možné si vybrat mezi herními technologiemi SpriteKit [12] nebo SceneKit [13]. Obě technologie se používají k přidávání animací do aplikací a her pro iOS.

Obrázek 4: SpriteKit & SceneKit. Převzato z [11]

2.1.1 SpriteKit

SpriteKit je framework[1] navržený pro 2D animace v aplikacích, které se vyznačují nízkou spotřebou baterie. Poskytuje základní funkce, mezi které patří podpora přehrávání zvuku, simulace fyziky, síly, kolize a další speciální efekty. SpriteKit slouží nejen pro herní aplikace, ale je vhodný i pro přidání poutavého pohybu a efektů do jakéhokoli obsahu. Xcode poskytuje vestavěnou podporu pro SpriteKit. Oba tyto nástroje společně vytvářejí výkonnou platformu pro vytváření složitých speciálních efektů pro hry nebo aplikace.

2.1.2 SceneKit

SceneKit je výkonnější varianta z obou frameworků. Vyžaduje dobrou znalost matematiky a geometrie. Používá se pro 3D aplikace a hry, které mají více úhlů pohledu do scény. Byl vydán společně s operačním systémem OS X Lion a je navržen tak, aby vývojářům usnadnil vytváření složitých 3D scén. SceneKit pracuje společně s technologiemi, jako je Quartz [14], Core Animation [15] a GLKit [16], a integruje se s dalšími grafickými technologiemi.

[1] Knihovna ulehčující práci při programování aplikace.

2.1.3 SpriteKit & SceneKit

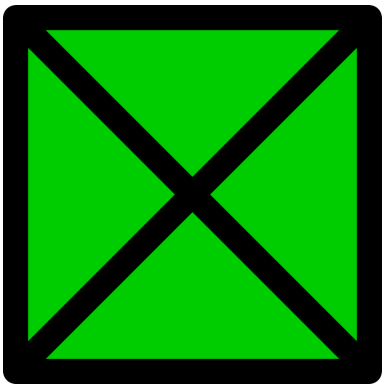
Je důležité poznamenat, že oba frameworky lze použít také společně. 2D SpriteKit scéna může být vrstvena na 3D SceneKit scénu nebo může být přidána jako pozadí, překrytí popředí nebo textura objektu.

2.2 Ukázka kódu v MetaPostu

Následující kód je ukázkou, jakým stylem jsou vytvořeny všechny textury ve hře. Přímo tento kód vygeneruje část zelené kostky, která překrývá jinou kostku. Přeškrtnutá část kostky má uživatele upozornit na to, že kostku je potřeba umístit jinak.

```
1  beginfig(0);
2    pair A[];
3    A[0]:=( 0cm, 0cm);
4    A[1]:=( 2cm, 0cm);
5    A[2]:=( 2cm, 2cm);
6    A[3]:=( 0cm, 2cm);
7    fill A[0]--A[1]--A[2]--A[3]--A[0]--cycle withcolor .8 green;
8    draw A[0]--A[1]--A[2]--A[3]--A[0]--cycle withpen pencircle scaled
         4bp;
9    draw A[0]--A[2] withpen pencircle scaled 4bp;
10   draw A[1]--A[3] withpen pencircle scaled 4bp;
11 endfig;
```

Zdrojový kód 1: Ukázka kódu v jazyce MetaPost generující texturu části kostky



Obrázek 5: Překrytá část kostky

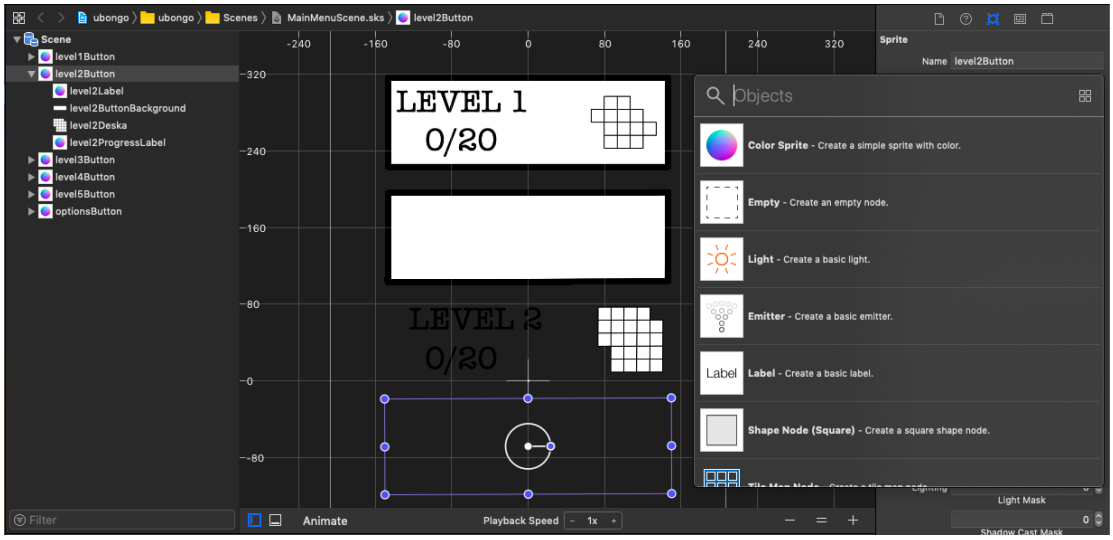
2.3 Popis struktury projektu

Aplikace Ubongo se skládá z několika scén. Scéna je instancí třídy `SKScene`, která definuje její vzhled. Každá taková scéna je propojena s třídou, která definuje její funkcionalitu a propojení s ostatními scénami.

Například `MainMenuScene.sks` je scéna úvodní obrazovky ve hře Ubongo a její funkcionalita je definována třídou `MainMenuClass.swift`. Třída definuje co se má stát, případně jaká jiná scéna se má načíst, po kliknutí na některé z tlačítek úvodní obrazovky. Také se stará o čítače už složených kol.

2.4 Ukázka práce s editorem scény

Tlačítka na úvodní obrazovce se skládají z pěti komponentů a jsou naskládány ve třech vrstvách. Na obrázku 6 je vidět, že spodní vrstva slouží pouze jako pozadí celého tlačítka. Je to objekt typu `Color Sprite`, kterému je nastavena textura vytvořená v MetaPostu. U tohoto objektu je pozice ve směru osy `Z` nastavena na hodnotu 0.



Obrázek 6: Editor scény v Xcode

V prostřední vrstvě se na levé straně nachází dva objekty typu `Label`. Jeden s názvem úrovně, druhý s počtem složených kol. Na pravé straně je objekt `Color Sprite`, u kterého je nastavena textura náhledu desky dané úrovně, která je také vygenerována pomocí MetaPostu. Všechny tyto tři objekty mají nastavenou pozici `Z` na hodnotu 1.

Poslední vrstvou je znovu objekt `Color Sprite`, který je zprůhledněný a pozici `Z` má nastavenou na hodnotu 3. Objekt zakrývá celé tlačítko a slouží pro identifikaci kliknutí do oblasti tlačítka. V třídě `MainMenuClass.swift`, která se stará o funkcionalitu úvodní obrazovky, se pracuje pouze s průhlednou vrstvou a čítačem kol.

2.5 Přidání objektu do scény

V prostředí Xcode má programátor na výběr jak uvést objekt do scény. Jeden ze způsobu je pomocí editoru, kde stačí vybrat objekt z knihovny a umístit ho na pracovní plochu editoru. Poté objektu může, přímo v editoru a jen za pomocí klikání, nastavit požadované vlastnosti, jako například pozici, průhlednost, barvu nebo texturu. Ve třídě, která scénu ovládá potom stačí jen nastavit objekt jako potomka. To je ukázáno v následujícím zdrojovém kódu.

```
1  class MainMenuClass: SKScene {
2
3   private var progressLabel: SKLabelNode!
4
5   override func didMove(to view: SKView) {
6     self.progressLabel = self.childNode(withName: "level1
          ProgressLabel") as? SKLabelNode
7   }
8  }
```

Zdrojový kód 2: Přidání objektu z editoru do scény

Druhý způsob je vynechání editoru a vytvoření celého objektu čistě za pomocí kódování. Vytvářet objekty pomocí editoru je pohodlné a programátorovi to šetří čas. V aplikaci Ubongo jsou všechny objekty, kromě samotných kostek, vytvořeny editorem. Následující kód je ukázkou toho, jak lze vytvořit objekt a nastavit mu nějaké vlastnosti bez použití editoru.

```
1  class MainMenuClass: SKScene {
2
3   private var progressLabel: SKLabelNode!
4
5   override func didMove(to view: SKView) {
6     progressLabel = SKLabelNode(fontNamed: "American Typewriter")
7     progressLabel.name = "level1ProgressLabel"
8     progressLabel.fontColor = UIColor.black
9     progressLabel.fontSize = 33
10    progressLabel.text = "0/20"
11    progressLabel.position.x = -70
12    progressLabel.position.y = 240
13    progressLabel.zPosition = 1
14    addChild(progressLabel)
15  }
16 }
```

Zdrojový kód 3: Vytvoření objektu a přidání do scény

2.6 Popis jednotlivých tříd

Tato sekce se věnuje třídám aplikace Ubongo. Jsou zde podrobně popsány jednotlivé třídy, jejich atributy a funkce.

2.6.1 Třída Cube

Instancí této třídy je kostka. Třída se stará o vytvoření objektů `SKSpriteNode` z nichž je kostka složena. Těmto objektům je nastavena textura, která udává vzhled a barvu kostky. Následně je jednotlivým dílkům nastavena pozice, aby kostka dostala požadovaný tvar. Třída také přepočítává pozice dílků kostky při posunech, nebo rotacích. Kostka má následující atributy:

`texture: String` – název určující texturu kostky
`countOfPieces: Int` – počet z kolika dílků se kostka skládá
`arrayOfNodes: [SKSpriteNode]` – pole objektů reprezentující dílky kostky
`arrayOfPoints: [CGPoint]` – pole všech bodů určující pozice dílků kostky
`data: [[Int]]` – dvourozměrné pole (matice) určující tvar kostky
`rotationNumber: Int` – číslo rotace kostky (hodnoty 1 až 4)
`inMove: Bool` – `true`, pokud je kostka v pohybu, jinak `false`
`isPlaced: Bool` – `true`, pokud je kostka umístěna na desce, jinak `false`
`isOverlapped: Bool` – `true`, pokud je kostka překrytá, jinak `false`
`distances: [[Int]]` – pole hodnot rozestupů mezi dílky kostky

Metody:

`setZPosition(n: CGFloat)`
– nastavuje `Z` pozici v závislosti na překrývání kostek

`rotate()`
– stará se o rotaci všech dílků kostky tak, že přepočítá jejich pozice, aby kostka měla požadovaný tvar a byla otočená o 90 stupňů

`dataToPoints()`
– z matice určující tvar kostky (`data`) vypočítá pomocí `distances` poziční body (`arrayOfPoints`)

`turnOver(data: [[Int]], n: Int)`
– převrátí matici (`data`) určující tvar kostky, aby mohla být vytvořena zrcadlově obrácená kostka, kterou některá kola vyžadují

`getCountOfPiecesFromTexture(texture: String) -> Int`
– metoda, která vrací hodnotu `countOfPieces` určující z kolika dílků se má kostka skládat. Pomocí názvu textury vyhledá v databázi kostek matici `data`, která určuje tvar kostky, a spočítá kolik je v matici nenulových hodnot

2.6.2 Třída Board

Instancí této třídy je deska. Třída se stará o vytvoření objektů `SKSpriteNode` z nichž je deska složena. Těmto objektům je nastavena textura desky. Následně je jednotlivým dílkům nastavena pozice, aby deska dostala tvar požadované úrovně. Deska má následující atributy:

`arrayOfNodes: [SKSpriteNode]` – pole objektů reprezentující dílky desky
`texture: String` – název textury desky
`countOfPieces: Int` – počet z kolika dílků se deska skládá
`arrayOfCubes: [Cube]` – pole určující sadu kostek
`arrayOfPoints: [CGPoint]` – pole všech bodů určující pozice dílků desky
`centerPoint: CGPoint` – středový bod desky
`xPositions: [Int]` – x-ové souřadnice poloh kostek
`yPositions: [Int]` – y-ové souřadnice poloh kostek
`distances: [[Int]]` – pole hodnot rozestupů mezi dílky desky

Metody:

`setShape(type: String)`
– určí pozice dílků desky (`distances`) podle typu desky a nastaví středový bod (`centerPoint`)

`setPositions()`
– nastaví pozici dílků desky (`arrayOfNodes`)

`setDistances()`
– k aktuální pozici jednotlivých dílků (`arrayOfNodes`) přičte odstup od středu desky, aby deska získala požadovaný tvar

`createArrayOfCGPoints()`
– metoda, která vytvoří pole bodů desky (`arrayOfPoints`) pro kontrolu překrývání kostek

2.6.3 Třída Game

Game je třída reprezentující herní scénu. Její hlavní funkcí je kontrola pozic dílků desky a kostek, aby bylo možné zjistit, jestli je kostka správně umístěna na desce, nebo jestli se kostky navzájem nepřekrývají. Také se stará o zarovnání kostek na desku, o veškerý pohyb kostek a o to, jestli už je deska složená nebo ne.

Atributy:

`completeLabel: [SKLabelNode]` – štítek s nápisem "Game Complete"
`levelName: String` – název kola
`boardType: String` – název tvaru desky
`cubeTextures: [String]` – pole názvů textur kostek (sada kostek)
`turnOverArr: [Int]` – pole určující, které kostky je nutné předem převrátit
`rotateArr: [Int]` – pole určující, které kostky je potřeba předem otočit
`arrayOfPlacedCubes: [Cube]` – pole kostek, které jsou na desce
`actualCube: Cube` – kostka, která je aktuálně aktivní
`actualBoard: Board` – deska aktuálně spuštěného kola
`doubleTap: Bool` – `true`, pokud je k dispozici dvojí kliknutí, jinak `false`
`isOverlapped: Bool` – `true`, pokud je překrytá kostka, jinak `false`
`movingCube: Bool` – `true`, pokud je kostka v pohybu, jinak `false`

Metody:

`createBoard()`
– volání metod pro vytvoření desky, zavedení objektů desky do scény a nastavení středu desky

`createCube(cube: Cube, x: Int, y: Int, cPoints: [CGPoint])`
– volání metod pro vytvoření kostek, zavedení kostek do scény a nastavení bodů zobrazení ve hře

`distanceOfPoints(pt1: CGPoint, pt2: CGPoint) -> Double`
– výpočet vzdálenosti dvou bodů

`alignmentCheck(cube: Cube) -> CGPoint`
– zarovnání kostky na desku. Metoda se zavolá v momentě, kdy uživatel dokončí tažení kostky. Vypočítá se vzdálenost mezi nejbližším bodem desky a středem kostky. Pokud je vzdálenost menší než 40, střed kostky se nastaví na zmiňovaný nejbližší bod desky

`completenessCheck()`
– metoda se zavolá v okamžiku, kdy jsou na desce umístěny všechny kostky a kontroluje se, jestli jsou všechna políčka desky zakrytá. Pokud ano, zviditelní se nápis "Game Complete". Nakonec se perzistentně uloží postup ve hře

`overlappingCheck()`
– metoda, která identifikuje překrývající se části kostek a nastaví jim texturu s křížkem

`textureReset()`
– uvedení textur kostky do původní podoby

`standOutCheck(cube: Cube)`
– metoda zjistí, jestli kostka nepřesahuje z desky. Pokud ano, tak u částí kostky, které přesahují z desky je textura zprůhledněna

`setAlpha1(cube: Cube)`
– nastavení neprůhlednosti textur u kostky

`saveGame()`
– metoda, která uloží rozehranou hru

`loadGame()`
– metoda, která načte rozehranou hru

2.7 Perzistentní uložení uživatelských dat

Aplikace pro iOS mají vestavěný slovník (`dictionary`[2]), do kterého je možné ukládat uživatelská data.[7] Data zůstanou perzistentně uložena tak dlouho, dokud je aplikace nainstalována. Tento slovník se nazývá `UserDefaults` a může ukládat hodnoty typu `bool`, `integer`, `double`, `string`, `array` a jiné.

V aplikaci Ubongo jsem použil `UserDefaults` pro uložení rozehrané hry a celého postupu ve hře. Pro uložení rozehrané hry je potřeba u všech kostek na desce uložit jejich název, číslo rotace, x-ové a y-ové souřadnice středu kostky a nakonec název rozehraného kola. V následujícím kódu je ukázáno ukládaní dat do `UserDefaults` a způsob, jak se zpětně čtou.

```
1  let defaults = UserDefaults.standard
2
3  // uložení dat
4  defaults.set(true, forKey: levelName)
5  defaults.set(cube.position.x, forKey: cube.name + "x")
6  defaults.set(cube.position.y, forKey: cube.name + "y")
7  defaults.set(cube.rotation, forKey: cube.name + "rotation")
8
9  // čtení dat
10 if(defaults.bool(forkey: levelName)) {
11   cube.position.x = defaults.float(forkey: cube.name + "x")
12   cube.position.y = defaults.float(forkey: cube.name + "Y")
13   cube.rotationNumber = defaults.integer(forkey: cube.name + "
         rotation")
14 }
```

Zdrojový kód 4: Ukázka práce s UserDefaults

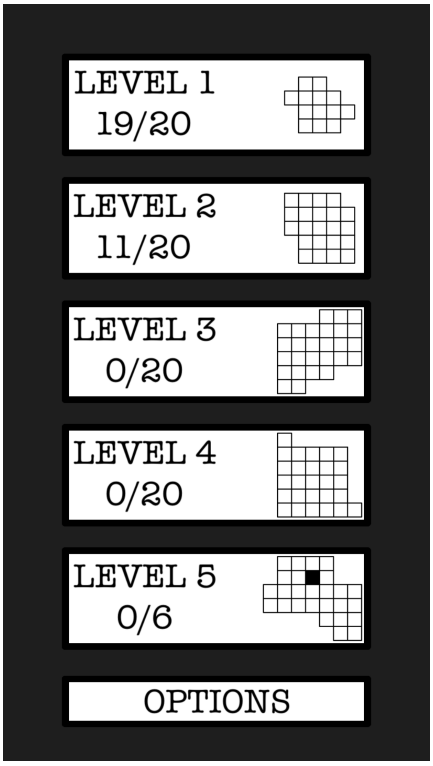
[2]kolekce dvojic klíč – hodnota

3 Uživatelská příručka

Uživatelská příručka poskytuje uživateli základní informace k používání a ovládání aplikace Ubongo. Pro snadnější popis funkcionality jsou zde k dispozici také náhledy z aplikace. Příručka slouží k seznámení uživatele se všemi možnostmi aplikace a s uživatelským prostředím.

3.1 Úvodní obrazovka

Na úvodní obrazovce (obrázek 7) je vidět přehled všech úrovní hry. U každé úrovně je napravo vyobrazen vzhled desky dané úrovně a nalevo je uveden počet složených kol z celkového počtu kol dané úrovně. Ve spodní části obrazovky jsou pak možnosti aplikace.



Obrázek 7: Úvodní obrazovka

3.2 Popis jednotlivých úrovní

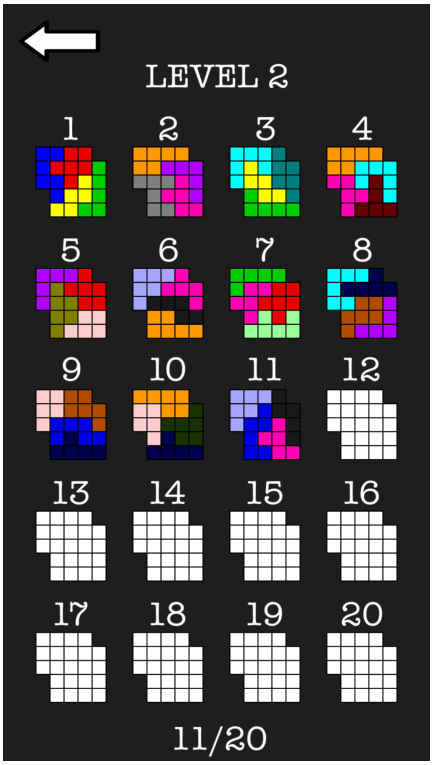
Každá úroveň má odlišnou obtížnost. Úrovně jsou seřazeny od nejsnadnější až po nejvíce náročnou. Rozdíly v obtížnosti jsou dány jak velikostí hrací desky, tak i počtem kostek pro její složení, a také tvarem samotné desky. Údaje o rozměrech desek a počtů kostek jsou uvedeny v tabulce 2.

Tabulka 2: Údaje o všech úrovních hry

úroveň	rozměr desky	počet kostek	počet kol
1	4×5	3	20
2	5×5	4	20
3	6×6	5	20
4	6×6	5	20
5	6×7	6	6

3.3 Náhled úrovně

Každá úroveň, kromě poslední, se skládá z dvaceti kol. Jednotlivá kola jsou očíslována. Na začátku jsou všechny desky bílé. Po složení některé z desek se deska vybarví. Nesložené desky zůstávají bílé (obrázek 8).

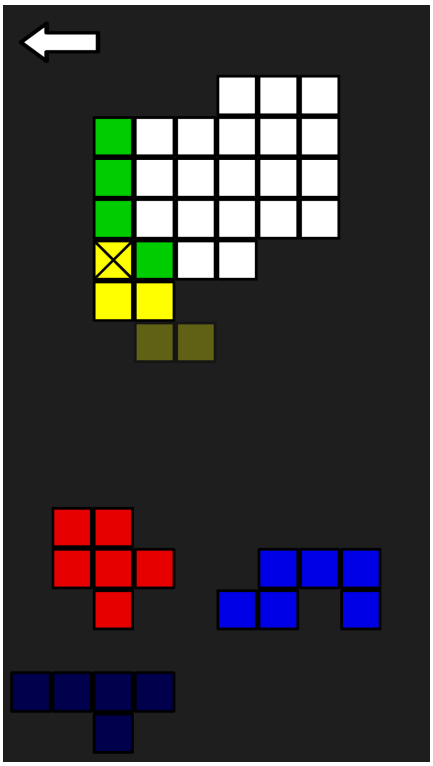


Obrázek 8: Náhled úrovně

3.4 Ovládání hry

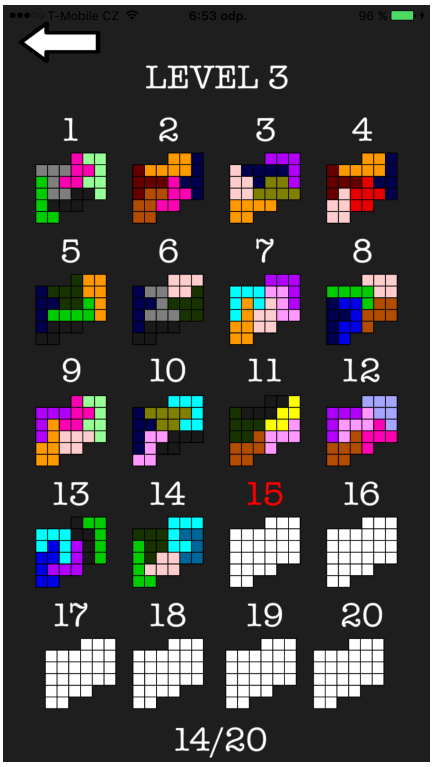
Po kliknutí na jednu z desek se uživatel dostane do hry. V horní části obrazovky je herní deska, a v dolní části obrazovky jsou seřazené kostky. Kostku na desku dostaneme tak, že na ni položíme prst a tažením ji posuneme na desku. Pro pootočení kostky o 90 stupňů je potřeba na ni dvakrát kliknout.

Pokud uživatel položí kostku na desku tak, že část přesahuje z desky ven, potom ta část kostky, která není na desce, je zprůhledněna. V tomto případě je potřeba umístit kostku jinak. Také se může stát, že je jedna kostka překryta druhou. V místě překrytí kostek se objeví křížek. To znamená že jednu z nich je potřeba umístit jinak. Obě situace jsou vyobrazeny na obrázku 9.



Obrázek 9: Pohled ze hry

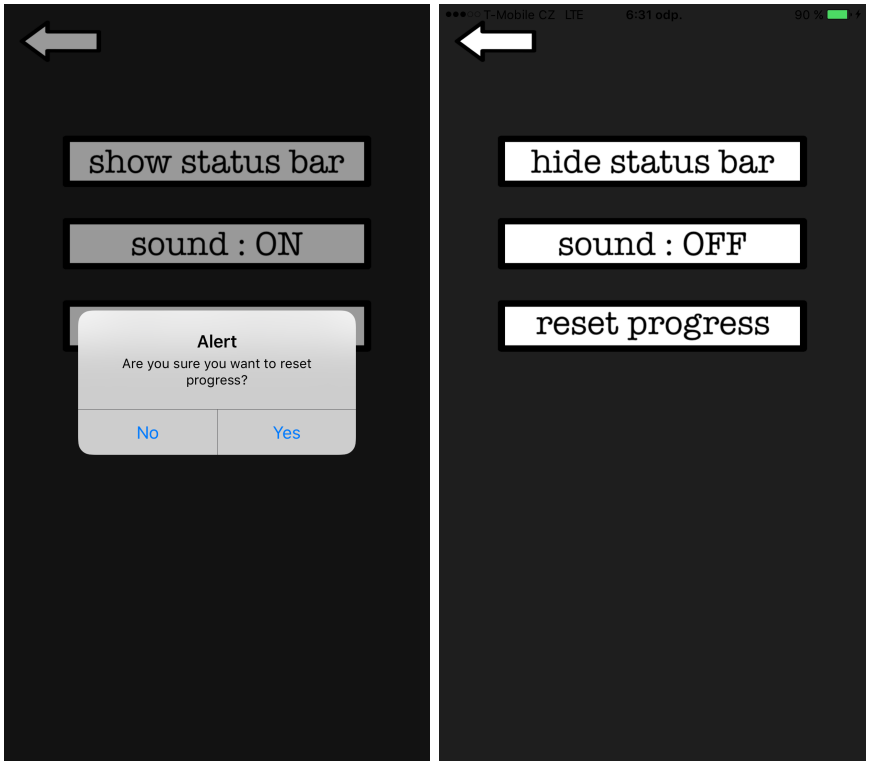
Pokud si uživatel neví rady jak desku složit, může hru přerušit a začít skládat jinou desku. Přerušená hra zachová pozice všech kostek na desce a v přehledu kol dané úrovně je číslo kola vyobrazeno červeně (obrázek 10).



Obrázek 10: Ukázka rozehrané hry

3.5 Možnosti hry

V možnostech hry Ubongo si uživatel může zobrazit, nebo případně skrýt stavový řádek zobrazující čas, stav baterie a sílu signálu. Také může vypnout/zapnout zvukový efekt kliknutí, a nebo smazat veškerý pokrok ve hře (obrázek 11). Smazání pokroku ve hře je nevratná akce, na kterou je uživatel upozorněn a musí ji potvrdit.



Obrázek 11: Možnosti

4 Možnosti rozšíření

Hru Ubongo jsem se snažil naprogramovat tak, aby případné rozšíření aplikace neznamenalo přepracování celé struktury implementace.

4.1 Rozšíření na větší zařízení

Velmi vhodné rozšíření by bylo hru přizpůsobit pro zařízení iPad, nebo iPad mini. Hra by se stala přehlednější a daleko lépe ovladatelnou. Na větší obrazovku se vejde více kostek, a díky tomu je možné skládat ještě větší a náročnější desky. Samotné kostky mají větší rozměr, a tak se výrazně zlepší ovladatelnost a nebudou se ztrácet pod prsty uživatele.

4.2 Rozšíření o nové úrovně a funkce

Rozšířit hru o nové úrovně také není problém. Pokud by bylo úrovní tolik, že by nebylo možné je umístit na úvodní obrazovku, řešením by bylo dát úrovně do vertikálně posuvného seznamu.

Dále by bylo možné rozšířit hru o funkci měření času skládání desky a propojení se sociálními sítěmi. Díky propojení hry se sociálními sítěmi by bylo možné s přáteli sdílet nejen pokrok ve hře, ale i žebříčky nejlepších časů.

Závěr

Cílem této práce bylo navrhnout a implementovat logickou hru pro iOS, inspirovanou sérií stolních her Ubongo. Hru jsem programoval pomocí jazyku Swift ve vývojovém prostředí Xcode. Prostředí Xcode už jsem dobře znal díky tomu, že jsem v něm už dříve programoval úkoly v jazyce C.

Grafické uživatelské rozhraní se od prvotního návrhu moc nezměnilo. Všechnu grafiku jsem vytvořil pomocí online MetaPostu. Čím více je poskládaných kol, tím více je prostředí hry barevnější. Díky tomu, že má hra 96 kol, dokáže zabavit i na několik hodin.

Pro testování jsem využil možnost nainstalovat hru na 3 zařízení s iOS. Pro distribuci hry na neomezený počet zařízení je potřeba zakoupení licence developerského účtu a umístit hru na Apple Store.

Conclusions

The aim of the thesis was to design and implement a logic game for iOS, inspired by series of board game called Ubongo. The game was programmed using the Swift language in the Xcode development programme. I already knew Xcode because I had previously completed homework using C language.

The graphical user interface has not changed much since the initial design. All graphics were created using online MetaPost. The more levels are completed, the more colourful the game environment is. The game can entertain for hours thanks to the fact that the game has 96 levels.

For testing I used the option to install the game on 3 different iOS devices. To distribute the game to an unlimited number of devices, you need to purchase a developer account license and upload the game to the Apple Store.

A Obsah přiloženého CD

`doc/`
: Složka obsahuje samotný text práce ve formátu PDF a veškeré zdrojové soubory potřebné pro vygenerování tohoto PDF dokumentu.

`src/`
: V této složce se nachází kompletní Xcode projekt aplikace Ubongo, a také veškerá grafika i s kódy pro vygenerování v MetaPostu.

`readme.txt`
: Instrukce pro instalaci a spuštění aplikace Ubongo, včetně všech požadavků pro její bezproblémový provoz.

B Instrukce pro instalaci a spuštění aplikace

Pro spuštění aplikace Ubongo je zapotřebí:

- počítač s operačním systémem Mac OS X 10.13.6 a novější;
- vývojové prostředí Xcode 10.1 a novější.

Kroky pro spuštění aplikace:

1. spustit Xcode;
2. na úvodní obrazovce zvolit možnost "Open another project...";
3. v průzkumníku souborů vyhledat složku s projektem aplikace Ubongo;
4. v levé horní části okna Xcode zvolit simulátor iPhonu 6/7/8 Plus, aby byla obrazovka simulátoru velká (aplikace je primárně určena pro tato zařízení s označením "Plus", která mají větší obrazovkou);
5. v levé horní části okna Xcode klikněte na tlačítko spustit s popiskem "Build and then run the current scheme".

Upozornění: Při používání aplikace v simulátoru, může docházet k zadrhávání a opožděným reakcím. Používáte-li aplikaci na fyzickém zařízení, aplikace běží stabilně na 60 FPS.

Pro instalaci aplikace na fyzické zařízení je zapotřebí:

- zařízení iPhone s iOS 10.3 a novější;
- vývojářský účet pro iOS aplikace.

Nejdříve je potřeba se přihlásit/registrovat jako vývojář v nastavení aplikace. Toto nastavení najdete v levé nabídce. Klikněte na tlačítko složky s popiskem "Show the Project navigator", a potom klikněte na první modrou ikonu s názvem "ubongo". Následně v horní části obrazovky klikněte na "General"a níže najděte sekci "Signing". Pokud vývojářský účet nemáte, můžete se registrovat pomocí vašeho Apple ID.

Po úspěšném přihlášení pak místo simulátoru zvolte Vaše zařízení. Nejdříve zapojte zařízení pomocí originálního kabelu do počítače. Následně je potřeba kliknout na rozbalovací menu se simulátory, které se nachází v levé horní části obrazovky, a úplně nahoře zvolit Vaše zařízení. Poté stačí spustit kompilaci, jak je uvedeno v kroku 5.

Seznam zkratek

IDE	Integrated Development Environment
OS	Operating System
WWDC	Worldwide Developers Conference
LLVM	Low Level Virtual Machine
USD	United States Dollar
FPS	Frames Per Second
ID	Identity Document

Bibliografie

[1] REJCHTMAN, Grzegorz. Ubongo. Albi 2005. Svět deskových her 2006 - 2019. [cit. 2019-07-24].
Dostupné z: https://www.svet-deskovych-her.cz/produkty/79/ubongo

[2] KURIMSKY, Jakub. Ubongo. Listopad 2009. [cit. 2019-07-24].
Dostupné z: http://www.hrej.cz/clanky/ubongo-2914

[3] ACKERMAN, Dan. The Tetris Effect: The Game that Hypnotized the World. Září 2016. ISBN: 978-1610396110. [cit. 2019-07-24].
Dostupné z: https://www.goodreads.com/book/show/29101491-the-tetris-effect

[4] XCODE. Apple Inc. 2019. [cit. 2019-07-24].
Dostupné z https://developer.apple.com/xcode

[5] DEVELOPER DOCUMENTATION ARCHIVE. Mac OS X. Apple Inc. 2016. [cit. 2019-07-24]. Dostupné z https://1url.cz/SMgwx

[6] DEVELOPER DOCUMENTATION ARCHIVE. iOS. Apple Inc. 2016. [cit. 2019-07-24]. Dostupné z https://1url.cz/iMgwD

[7] THE SWIFT PROGRAMMING LANGUAGE. Apple Inc. 2019. [cit. 2019-07-24]. Dostupné z https://docs.swift.org/swift-book

[8] LATTNER, Chris; ADVE, Vikram. The LLVM Compiler Framework and Infrastructure Tutorial. West Lafayette, Indiana, Září 2004. [cit. 2019-07-24].
Dostupné z http://llvm.org/pubs/2004-09-22-LCPCLLVMTutorial.html

[9] GARAGEBAND. Apple Inc. 2019. [cit. 2019-07-24].
Dostupné z https://www.apple.com/cz/mac/garageband

[10] HENDERSON, Troy. MetaPost online previewer. Tex Live 2017. [cit. 2019-07-24]. Dostupné z: http://www.tlhiv.org/mppreview/

[11] OGAWA, Hideko. SpriteKit & SceneKit. 2015. [cit. 2019-07-24].
Dostupné z: https://blog.sorausagi.org/2015/07/spritekitscenekit.html

[12] SPRITEKIT. Apple Inc. 2019. [cit. 2019-07-24].
Dostupné z: https://developer.apple.com/spritekit/

[13] SCENEKIT. Apple Inc. 2019. [cit. 2019-07-24].
Dostupné z: https://developer.apple.com/scenekit/

[14] GELPHMAN, David; LADEN, Bunny. Programming with Quartz. San Francisco: Morgan Kaufmann, 2006. ISBN: 978-0-12-369473-7. [cit. 2019-07-24].
Dostupné z: https://www.sciencedirect.com/book/9780123694737/programming-with-quartz

[15] CORE ANIMATION. Apple Inc. 2019. [cit. 2019-07-24].
Dostupné z: https://developer.apple.com/documentation/quartzcore

[16] GLKIT. Apple Inc. 2019. [cit. 2019-07-24].
Dostupné z: https://developer.apple.com/documentation/glkit