Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Vývoj aplikace pro operační systém iOS

Tereza Fialová

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Tereza Fialová

Systémové Inženýrství a Informatika

Název práce

Vývoj aplikace pro operační systém iOS

Název anglicky

Application development for iOS

Cíle práce

Hlavním cílem bakalářské práce je praktická ukázka vývoje aplikace pro operační systém iOS. Dílčí cíle jsou:
- charakterizovat operační systém iOS
- porovnat programovací jazyky Swit a Objective-C
- analyzovat možnost psaní aplikace v Xcode a v jiných nativních programech
- v praktické části vysvětlit problematiku při nesplácení úvěrů a vytvořit aplikaci ”Kalkulačka úroků”
- formulovat obecné a specifické závěry

Metodika

Metodika řešené problematiky je založena na studiu a analýze odborných informačních zdrojů.

V praktické části bude vytvořena aplikace ”Kalkulačka úroků” v prostředí Xcode pomocí programovacího jazyka SWIFT.

Na základě syntézy teoretických poznatků a výsledků praktické části budou vytvořeny závěry bakalářské práce.

Doporučený rozsah práce

35

Klíčová slova

Swift, Objective-C, Apple, iOS, iPhone, Insolvence, Odstoupení od smlouvy

Doporučené zdroje informací

ALESSI, P. Vývoj her pro iPhone a iPad: Programování pro iOS = Beginning iOS game development. : Brno: Zoner soffware, a.s. 2012. ISBN 978-80-7413-199-8

Apple, Inc**.** *The Swift Programming Language (Swift 2.1).* Cupertino, CA : Apple Inc., 2015.

KOCHAN, S G. *Objective-C 2.0 : výukový kurz programování pro Mac OS X a iPhone.* Brno: Computer Press, 2010. ISBN 978-80-251-2654-7.

VÁVRŮ, J. iPhone vývoj aplikací: Praha: Grada Publishing, a.s. 2012. ISBN 978-80-247-4457-5

Předběžný termín obhajoby

2015/16 LS – PEF

Vedoucí práce

doc. Ing. Zdeněk Havlíček, CSc.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 13. 3. 2016

Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 13. 3. 2016

Ing. Martin Pelikán, Ph.D.

Děkan

V Praze dne 18. 03. 2016

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Vývoj aplikace pro operační systém iOS" jsem vypracovala samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autorka uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 14. března 2016 _____

Poděkování

Ráda bych touto cestou poděkovala doc. Ing. Zdeňku Havlíčkovi CSc. za vstřícné jednání, rady a ochotu, kterou mi projevil. Dále velké díky patří Josefu Hlaváčovi za pomoc s kódovou částí.

Vývoj aplikace pro operační systém iOS

Application development for iOS

Souhrn

Hlavním cílem této bakalářské práce je vývoj aplikace pro mobilní operační systéme iOS od společnosti Apple. Teoretická část se zabývá studiem podkladů a vymezením syntaxí programovacích jazyků a jejich následné zhodnocení. Následují informace o jednotlivých verzích operačního systému iOS.

V praktické části jsou definovány pojmy insolvence a zesplatnění / odstoupení od smlouvy. Dále vývoj samotné aplikace od designové stránky po samotný kód aplikace. Závěrem shrnutí celé bakalářské práce a doporučení.

Klíčová slova: Swift, Objective-C, Apple, iOS, iPhone, Insolvence, Odstoupení od smlouvy

Summary

The main aim of this bachelor thesis is development of applications for mobile iOS operating system from Apple. The theoretical part deals with the study of documents and defining the syntax of programming languages and their subsequent evaluation. The following information are about the different versions of the operating system iOS.

The practical part defines the terms of insolvency and matures / withdrawal. Next part is development of the application from website design after the application code. Finally is a summary of the whole thesis and recommendations.

Keywords: Swift, Objective-C, Apple, iOS, iPhone, insolvency, matures / withdrawal

Obsah

Úvod

Cílem této bakalářské práce je proces vývoje aplikace pro platformu iOS. Tato bakalářská práce ukazuje ucelený vývoj aplikace od výběru programovacího jazyka po zdůvodnění zvoleného tématu.

Řešení aplikace je navrženo tak, aby uživatelé měli snadnější a rychlejší možnost posoudit finanční náročnost nabízeného úvěru a jeho dopadu na vlastní finanční situaci. Záměr je prostý - předejít neschopnosti plnit své závazky plynoucí z finanční půjčky a následně možného upadnutí do „dluhové pasti“, která může končit insolvencí nebo „zesplatněním“ smlouvy.

Dále je třeba zhodnotit možnost vývoje aplikace přes nativní programovací prostředí a oficiální cestou doporučovanou společností Apple, následně zhodnotit funkcionalitu a použití pro tuto aplikaci a zvolit vhodné programovací prostředí.

Součástí této práce je popis základních charakteristik programovacích jazyků pro mobilní prostředí iOS, jejich porovnání a následný výběr nejvhodnější možnosti pro využití k vývoji této aplikace.

V neposlední řadě jsou zde zmíněny důvody výběru tématu aplikace, upřesnění možnosti jejího uplatnění v běžném životě, tedy aplikace by měla usnadňovat rozhodování o případném zvolení finančních produktů a zmapovat jejich dopady na finanční situaci jednotlivce.

Následně je zde uvedeno, jakým způsobem je možno vytvořenou aplikaci distribuovat k uživateli.

Cíl práce a metodika

Hlavním cílem této bakalářské práce je ukázat vývoje aplikace „Kalkulačka úroků“ pro operační systéme iOS od společnosti Apple.
Další cíle jsou:

- Charakteristika operačního systému iOS a informace o jednotlivých verzích.
- Porovnání programovacích jazyků Swift a Objective-C.
- Analýza zvoleného prostředí Xcode a v dalších nativních programech.
- V praktické části vysvětlit problematiky insolvence a odstoupení od smlouvy a zároveň samotná ukázka procesu vývoje aplikace „Kalkulačka úroků“.
- Popis publikace aplikace.
- Závěrem zhodnotit obecné poznatky, vytvořit specifické závěry a vložit doporučení pro konzumenty úvěrů.

Metodika řešené problematiky bakalářské práce je založena na studiu a analýze odborného textu. Praktická část je zaměřena na vytvoření aplikace „Kalkulačka úroků“ v prostředí Xcode pomocí programovacího jazyka Swift. Na základě syntézy teoretických poznatků a výsledků praktické části budou vytvořeny závěry a doporučení bakalářské práce.

1. Programovací jazyk Swift

V červnu roku 2014 představila společnost Apple programovací jazyk Swift, jež má plně nahradit Objective–C. Cíl společnosti Apple je však větší, snaží se totiž nahradit i jazyky typu C a C++, aby vzbudil zájem u více vývojářů. Od konce roku 2015 je již vyhotovena verze Swift 2.0. Tato verze přinesla velkou výhodu do světa Linuxu. Je totiž možné vyvíjet v ní aplikace i na platformě Linux. Je tedy volně ke stažení přes obchod pro Linuxové zařízení.

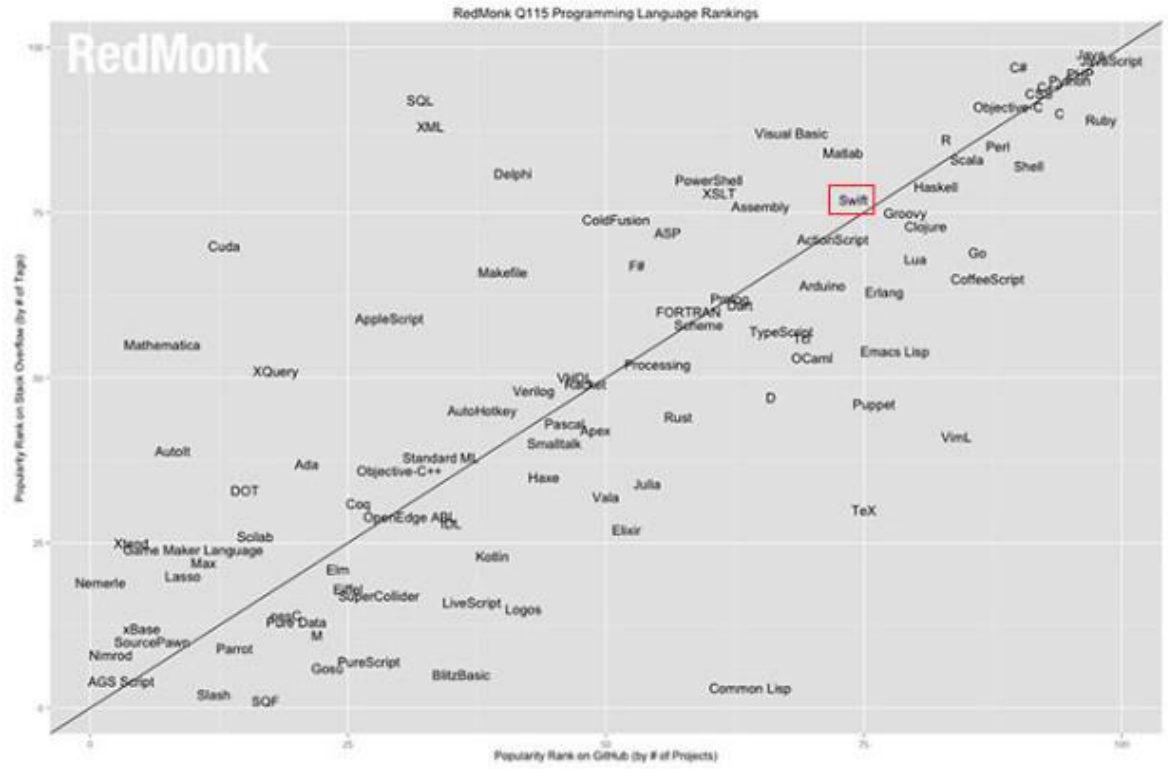
Swift is a new programming language for iOS, OS X, watchOS and tvOS apps that builds on the best of C and Objective-C, without the constraints of C compatibility. Swift adopts safe programming patterns and adds modern features to make programming easier, more flexible, and more fun. Swift´sclean slate, backed by the mature and much-loved Cocoa and Cocoa Touch frameworks, is an opportunity to reimagine how software development works. (1)

Swift je tedy nový programovací jazyk společnosti Apple pro tvorbu aplikací na platformu iOS, OS X. Jazyk se lze naučit buď přímo z publikací vydaných firmou Apple, které jsou volně ke stažení, nebo je možné dohledávat různé části kódu přímo na jejich webových stránkách. Dalším zdrojem informací může být i Youtube, kde je možnost najít spoustu tutoriálů, z nichž bylo pár přeloženo i do češtiny.

Ačkoliv je Swift poměrně nový programovací jazyk, tak na žebříček nejpopulárnějších programů RedMonk se již po třech měsících dokázal umístit na 68. místě. Půl roku po vydání jazyka Swift už mluvíme o rekordním 22. místě. Na obrázku 1.1 je vidět, mezi jakými jazyky se Apple nachází. Pro porovnání je dobré použít konkurenční jazyk Go od Googlu, který sice drží 20. příčku, ale šplhá na ní postupně už roku 2009.

Je tedy více než pravděpodobné, že stále více vývojářů začne tento jazyk využívat. K této teorii se staví i RedMonk, který se k výsledkům vyjádřil následovně – „Rapidní zájem o programovací jazyk Swift je bezprecedentní. Doposud bylo za výrazný skok považován posun o pět až deset míst.“ Apple tak drží další prvenství, kdy se během několika měsíců posunul o míst šestačtyřicet.

V první desítce žebříčku jsou například JavaScript, Java, PHP, Python, C#, C+ +, Ruby, CSS a C. Vysoko před Swiftem je také Objective-C, jehož je jazyk od Applu potenciálním nástupcem. (2)



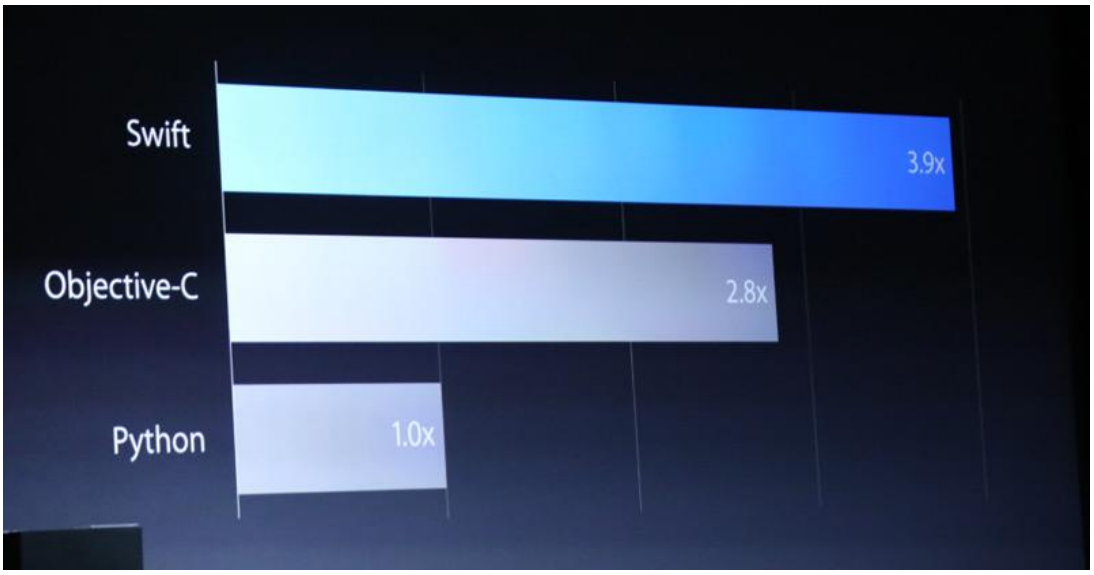
Obrázek 1.1- Umístění programovacího jazyku Swift

Zdroj: (3)

1.1. Syntaxe jazyka

Swift je velmi podobný svou syntaxí jiným programovacím jazyků, jako jsou Python, Ruby, C++ či C. Již není nutné, jako tomu bylo při zápise v Objective-C, rozdělovat zápis tříd do dvou různých souborů nebo importovat soubory s deklaracemi. Využívá bloky uzavřené složenými závorkami {}, dvojité lomeno jako komentář, na začátku dokumentu by měla být dokumentace, importování knihoven atd. Nevyžaduje středníky za příkazy a závorky u podmínek řídících konstrukcí (if, for, while). (4)

Pokud se podíváme na rozdíly mezi jazyky Objective-C viz obrázek 1.2 a 1.3 tak pro každého programátora je zcela zásadní. Neporovnáváme zde totiž jen délku kódu, je nutné si uvědomit celkový dopad na funkčnost. Následuje praktický příklad porovnání kódu mezi Objective-C a Swift.



Obrázek 1.2 – Rychlost programovacího jazyku Swift - srovnání

Zdroj: (5)

```
1if (myDelegate != nil) {
2
3    if ([myDelegate respondsToSelector: @selector(scrollViewDidScroll:)]) {
4
5        [myDelegate scrollViewDidScroll:myScrollView];
6
7    }
8
9}
```

Ten stejný zápis v Swift:

```
1myDelegate?.scrollViewDidScroll?(myScrollView)
```

Obrázek 1.3 – Rozdíl mezi zápisem v programovacím jazyce Swift a Objective-C

Zdroj: (5)

1.2. Objective-C

Jazyk Objective-C je doménou společnosti Apple. Celá jeho historie se s touto firmou úzce prolíná, ať už s přímo s Apple samotným, nebo s NexStep (společnost založená Stevem Jobsem po jeho nuceném odchodu z Apple)

1.2.1. Vznik Objective-C

Objective-C vytvořili Brad Cox a Tom Love jako objektově-orientované rozšíření populárního jazyka C. Základem návrhu jazyka byla úplná zpětná kompatibilita s „céčkem“. Objektovou syntax si vypůjčili ze Smalltalku (s přidanými hranatými závorkami kolem posílání zpráv). Původní verze byla jen jakýmsi preprocesorem převádějícím zdrojový kód

v ObjC do čistého C. Výsledný kód využíval malé běhové prostředí (runtime), jehož značně dokonalejší podobu máme k dispozici i v dnešních překladačích pro ObjC. (6)

V době vzniku Objective-C ještě neexistovalo C++ a jiný vhodný jazyk nebyl k dispozici. Zároveň funkce Smalltalku byly sice vhodné pro společnost Apple, ale protože Smalltalk byl velmi pomalý, došlo ke spojení dvou jazyků. – Smalltalk a C. Dalším důvodem bylo množství knihoven jazyka C, které by nebylo možné využít.

1.2.2. NeXT software

Společnost NeXT software si licencovala jazyk Objective-C a vyvinula jeho knihovny i vývojové prostředí označované NeXTSTEP. V roce 1992 byla podpora Objective-C doplněna do vývojového prostředí Free Software Founadtion´s GNU. Vlastnická práva ke všem produktům Free Software Foundation patří organizaci FSF. Produkt je k dispozici pod GNU General Public License. (Objective-C 2.0 Výukový kurz programovaní pro OS X a iPhone)

Roku 1996 přebrala společnost Apple software od společnosti NeXT software a vývojové prostředí pojala za své. To byl základ vývojového prostředí pro vznik operačního systému OS X a pak následně i iOS, což je jednodušším provedením pro mobilní platformy.

1.3. Výhody a nevýhody Swift vs. Objective-C

Výhody Swift oproti Objective–C

- Swift je mnohem jednodušší pro začínající vývojáře.
- Playground (prostředí pro první krůčky v novém jazyku) je velmi přívětivé a jednoduché.
- Swift již obsahuje Cocoa (sada objektově orientovaných framworků) a Cocoa Touch (framework pro uživatelské rozhraní).
- Je kompatibilní s Objective-C a zároveň v jedné aplikaci mohou být použity oba programovací jazyky.
- Je rychlejší, bezpečnější a moderní viz obrázeky 1.2 a 1.3.

Nevýhody Swift oproti Objective – C

- Swift je kompatibilní pouze s iOS 8 a výš dále s OS X 10.10 Yosemite a výš.
- Díky tomu, že Swift je poměrně mladý jazyk, nejsou zatím žádné odborné publikace v českém jazyce, k dispozici jsou pouze v jazyce anglickém.

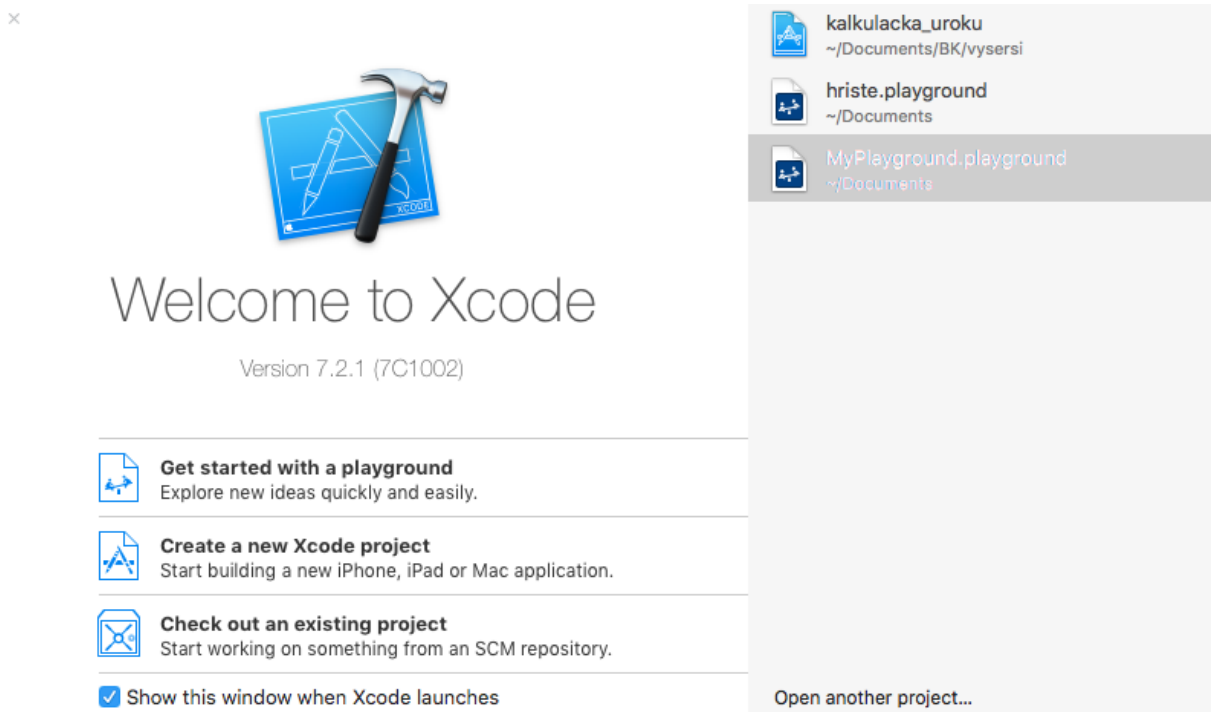
1.4. Playground

A playground is an interactive Swift coding environment that evaluates each statement and displays results as updates are made, without the need to create a project. (7)

Od verze Xcode 6 obsahuje Xcode nabídka viz obrázek 1.4 i možnost Playground viz obrázek 1.5. Jak již ze samotného názvu vyplývá (Hřiště) byl stvořen pro experimentování s kódem. Jeho nespornou výhodou je, že jsou vidět okamžitě změny - výpis se provádí do pravé části programu okamžitě. V konzoli v dolní části programu se vypisuje výsledek na základě příkazu, není tudíž již nutné spouštět stále simulátor při upravování kódu.

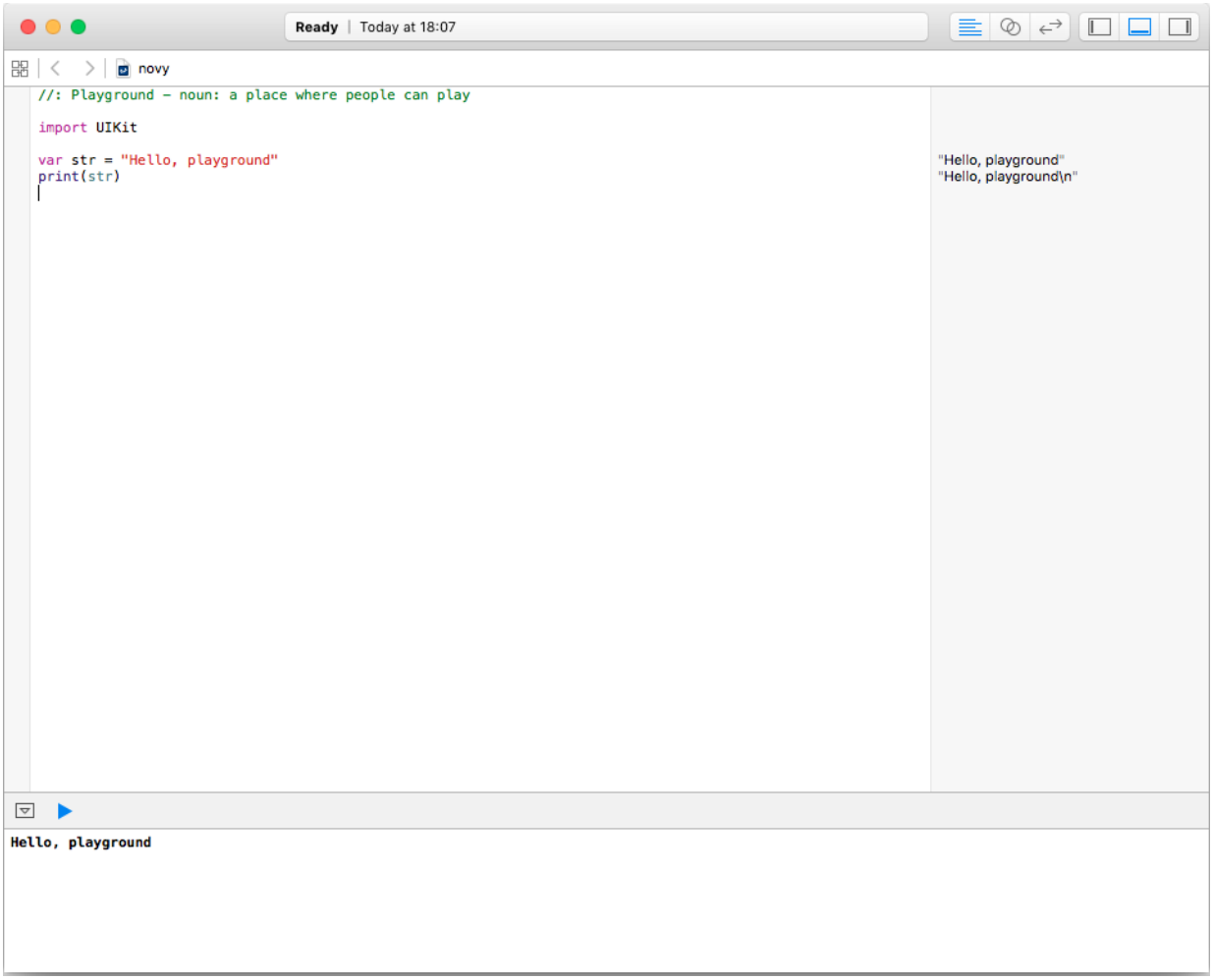
Playground je ideální možnost, jak se seznámit s jazykem Swift a naučit se ho podrobněji. Velmi příjemná je funkce nabízení upozornění na chyby, a to nejen kvůli kvalitě kódu, ale i kvůli jeho celkové funkčnosti.

Možnost grafického výstupu v Playground je další příjemnou funkcí viz obrázek 1.6. Tento obrázek představuje ukázku přímo z prezentace Keynote WWDC, více informací je níže v kapitole Keynote WWDC.



Obrázek 1.4 – Xcode

Zdroj: vlastní



Obrázek 1.5 – Playground – Vývojové prostředí součástí Xcode

Zdroj: vlastní

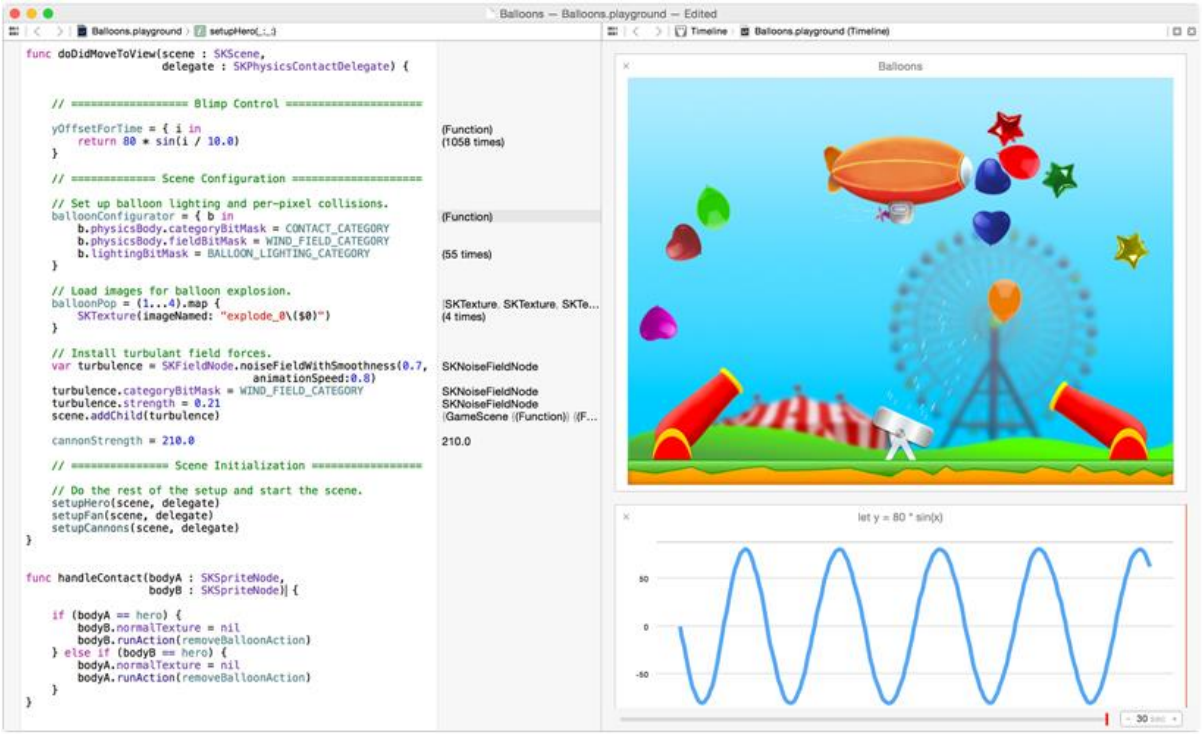
1.5. Keynote WWDC a Apple special event

Jedná se o prezentaci od společnosti Apple, na které tato společnost představuje své nové přístroje či software. V případě Keynote WWDC se jedná o nejočekávanější událost celého roku pro všechny nadšence Apple – napříč od uživatelů až po vývojáře. Každá prezentace je vytvořená takovým způsobem, který je jednoduchý a pochopitelný, aby i široká veřejnost dokázala bez problémů pochopit funkčnost nového softwaru či výrobku. Prezentace se koná pravidelně vždy v červnu každý rok. Nejedná se ale o jedinou prezentaci od společnosti Apple během roku.

Každý rok v září pořádá společnost Apple special event kde je opět možnost vidět jak nové přístroje společnosti Apple, tak software. Rozdíl mezi Keynote WWDC a Apple special event tkví v tom, že v červnu se společnost soustředí převážně na přestavení software a

hardware počítačů, případně poodhalí software pro mobilní přístroje. V září je pak důraz kladen hlavně na předvádění mobilních produktů a jejich funkčnost. (iPhone, iPad, iPod).

Obrázek 1.6 je proslavený především faktem, že programátor Apple dokázal hru zobrazenou na tomto obrázku naprogramovat během pouhých několika hodin. Hra nikdy nebyla vydaná na App Store jejím cílem je pouze demonstrovat funkčnost a obratnost jazyka Swift.



Obrázek 1.6 – Playground – jednoduchá hra naprogramovaná za pár hodin

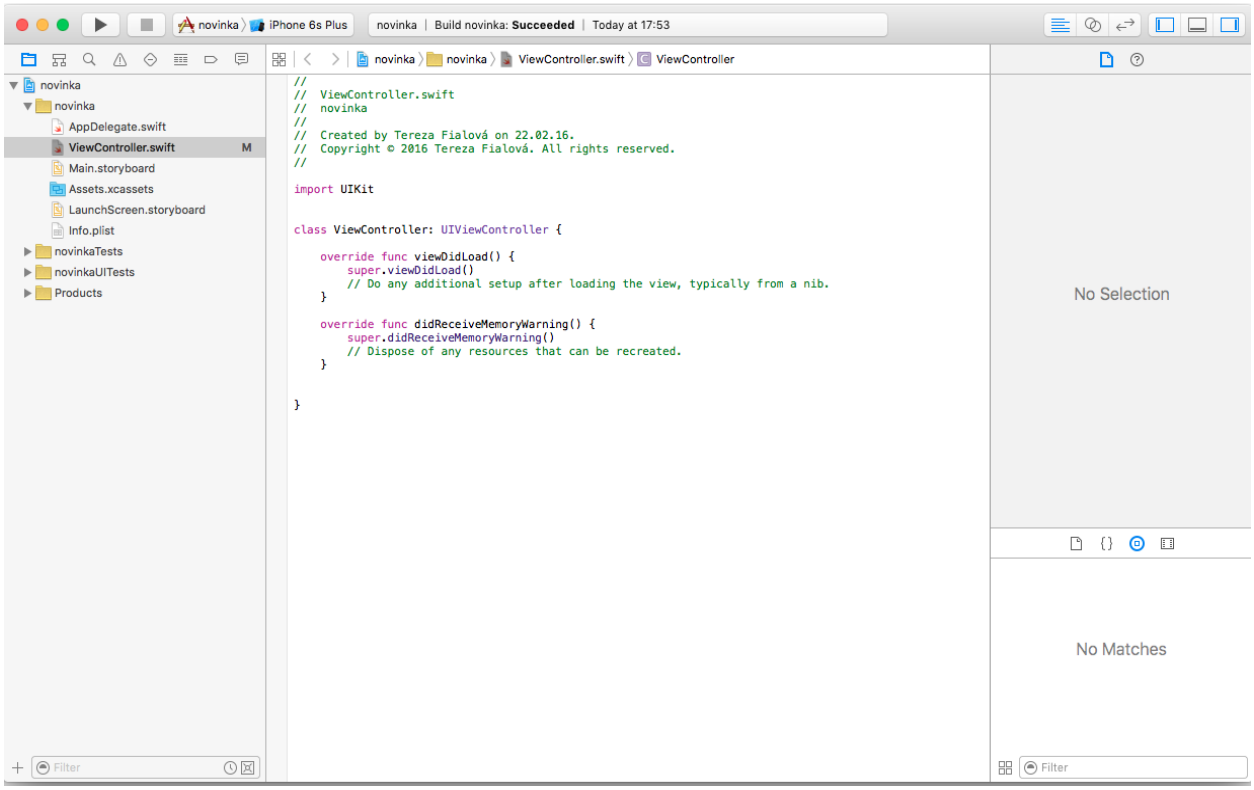
Zdroj: (8)

1.6. Xcode 7

V případě Xcode jako takového je integrované vývojové prostředí od společnosti Apple. Mezi typické nástroje patří správa souborů, integrace s kompilátorem kódu, funkcionalita pro navrhování uživatelského rozhraní a ladící schopnosti. (9)

Poté, co začne nový projekt, je pro celkové zjednodušení na výběr několik typů šablon, které mají za úkol zjednodušit případný návrh aplikace. Mezi nimi se nachází šablony pro hry, pro aplikaci s jedinou zobrazovací plochou, šablony složité s navigačními kontroléry, či šablonu pro aplikaci, která využívá záložky.

Jak je zřejmé z obrázku 1.7, jedná se v případě Xcode o složitější prostředí než Playground ale to na druhou stranu přináší mnohem více možností. Úplně vlevo se nachází navigační oblast. Ta slouží převážně pro přesun mezi kódem a zobrazení uživatelského rozhraní. Dále je zde oblast editoru, kde se upravuje kód, případně vzhled aplikace. Oblast ladění se nachází v části editoru dole, v tuto chvíli však není k náhledu. V pravé části je oblast náhledu rozdělena do dvou částí: oblast podokno inspektora, a pod ním lze nalézt protokol knihovny. V podokně inspektora se nastavují užitečné funkcionality, viz vlastní práce. Díky podoknu knihovny zase vložíme samotná tlačítka či okna pro psaní nebo posuvníky.



Obrázek 1.7 – Xcode – začátek vývoje aplikace

Zdroj: vlastní

1.7. Programování v Xcode vs. PhoneGap

1.7.1. Programování v Xcode 7

Samotné programování aplikace probíhá v Xcode 7, což je vývojové prostředí které jednak obsahuje jazyk Swift, ale i zároveň Objective-C. Xcode je mocné integrované vývojové prostředí, které má všechny funkce, jaké se od moderního IDE očekávají. Zahrnuje mnoho mocných funkcí, včetně editace kódu, ladění, řízení verzí a profilování softwaru. (9)

Přímo při zakládání nového projektu je potřeba vybrat, který jazyk bude zvolen. Mohou však být použity oba dva jazyky zároveň. Xcode je samozřejmě ke stažení zdarma na App Store. Toto je však podmíněno vlastnictvím počítače od Apple, protože Xcode je aplikace pro OS X zařízení, či nově i pro Linux platformu. Při vývoji aplikace pro telefon či tablet však již není potřeba daný přístroj vlastnit. Xcode totiž obsahuje Simulator všech přístrojů. V Xcode jsou obsaženy veškeré knihovny, které jsou zapotřebí pro vývoj aplikace. V Playground se dají chybějící knihovny vložit jednoduchými příkazy:
import UIKit či import Cocoa

1.7.2. Jednoduchá aplikace Hi Word v Xcode

Aplikace Hi Word je jedna z prvních, co se zobrazí v každém programovacím prostředí, ani v Xcode tomu není jinak. Ovšem důležitější je, o jak krátký kód se jedná, což je patrné na obrázku 1.8.

```
1 println("Hello, World!")
```

Obrázek 1.8 – Příklad výpisu pozdravu v Xcode

Zdroj: (10)

1.7.3. Programování v PhoneGap

PhoneGap je aplikační platforma založená na HTML5, která umožnuje autorům aplikací přistupovat k nativním funkcionalitám iPhone a tyto aplikace umístit do aplikačních obchodů, v našem případě Appstor. PhoneGap umožňuje vývoj aplikací i pro další platformy, jako Android, BlackBerry, Symbian, Windows Phone 7 a Bada, a to vše s minimálními změnami v kódu. (11) Za použití HTML a JavaScriptu lze velmi ušetřit čas při vývoji pro více

platforem. PhoneGap je stejně jako Xcode open source, lze tedy oběma způsoby naprogramovat aplikaci, která je zdarma či placená. Pokud tvořenou aplikaci chceme otestovat, je zapotřebí emulátor Ripple.

Dále je zapotřebí Framework jQuery Mobile - Unifikovaný Framework pro tvorbu uživatelského mobilního rozhraní využívajícího HTML5 a CSS3 vlastností, určený pro všechny současné populární mobilní platformy. Framework staví na jQuery a jQuery UI frameworku. (11)

Je potřeba použít editor vývojového prostředí, nejběžněji se používá Eclipse. Eclipse je určen primárně pro programování v jazyce Java, ale za pomocí pluginů ho lze rozšířit o UML, HTML či XML.

1.7.4. Jednoduchá aplikace Hi Word v PhoneGap

Na obrázku 1.9 je patrný nesrovnatelně delší a složitější kód. Kód je pěkně zarovnaný a dobře čitelný.

```
<!DOCTYPE HTML>
<HTML>
<head>
      <title>Titulek stránky</title>
      <meta name="viewport" content="width=device-width, initial-
scale=1">
</head>

<body>
<h1> Hi, World!</h1>
</body>
</HTML>
```

Obrázek 1.9 – Příklad výpisu pozdravu v PhoneGap

Zdroj: (11)

1.8. Zhodnocení

1.8.1. Zhodnocení Swift vs. Objective-C

Ačkoliv Swift vychází z Objective-C obsahuje totiž jeho strukturu. Společnost Apple ho zatím nepoužívá pro vývoj svých vlastních aplikací. Je použit pouze v jedné a tím je kalkulačka.

U vývojářů však došlo k velké oblibě jazyka Swift a je v něm programována většina aplikací pro zařízení Apple.

Velké pozitivum je v jeho jednoduchosti a to i pro absolutního začátečníka. Swift dokonce začaly používat pro své první aplikace i děti. Na kanále YouTube je k nalezení spousta tutoriálů, které vložily děti kolem věku 12ti let.

Swift i Objektiv-C mohou být v jedné aplikaci použity zároveň - pokud je v Xcode zvolena možnost vývoje aplikace v jazyce Swift – je tedy zpětně kompatibilní.

1.8.2. Zhodnocení Xcode vs PhoneGap

V případě programování pro více platforem je vždy lepší použít nativní vývojové prostředí typu PhoneGap, kdy je celý kód s minimálními úpravami převeden do jiného jazyka.

Pro vývoj aplikace pro zařízení od společnosti Apple však tato volba (PhoneGap) není optimální k náhledu viz obrázky 1.8 a 1.9.

Xcode oproti PhoneGap obsahuje všechny potřebné náležitosti a není potřeba nic dodávat či používat emulátory a jiné doplňky.

Xcode je velmi přehledný a uživatelsky přívětivější
(iPhone – vývoj aplikací)

2. iOS

OS (do června 2010 známý jako iPhone OS) je mobilní operační systém vytvořený společností Apple Inc. Původně byl určen pouze pro mobilní telefony iPhone, později se však začal používat i na dalších mobilních zařízeních této firmy, jako jsou iPod Touch, iPad a nejnověji Apple TV. (12)

iOS byl původně vytvořen z operačního systému Mac OS X. Obsahem iOS jsou 4 softwarové abstraktní vrstvy – Core OS layer, Core Service layer, Media layer a Cocoa Touch player. Všechny vrstvy vytváří jednotné rozhraní pro odlišný hardware.

2.1. Vrstvy iOS

2.1.1. Cocoa Touch player

V Cocoa Touch player jsou obsaženy frameworky sloužící jako podpora při vývoji aplikace, obsahuje ale i infrastrukturu pro grafické rozhraní. Nepostradatelnou součástí je multitasking nebo-li schopnost provádění více úkonů najednou. Frameworky této vrstvy jsou např. Address Book UI Framework (obsahující standardizované rozhraní pro zobrazování a úpravu kontaktních informací), iAd Framework (umožňuje zobrazení bannerové reklamy v aplikaci), Map Kit Framework (umožňuje používat mapové složky) a Message UI Framework (umožňuje vytvářet a odesílat e-maily a SMS zprávy). (12)

2.1.2. Media layer

Jak již z názvu vyplívá, jedná se o mediální vrstvu, nejčastěji je využívána grafickými a zvukovými aplikacemi. Grafickými technologiemi jsou tu třeba Core Graphics (Quartz) a Core Animation, zvukovými The Media Player framework, AV Foundation nebo OpenAL a audiovizuálními Media Player framework či Core Media. (12)

2.1.3. Core Service layer

Díky této vrstvě je možné provádět platby přímo v aplikacích nebo sledovat polohu uživatele. Z poskytovaných frameworků nabízí tato vrstva například Store Kit, poskytující přístup k iTunes Store a možnost nákupů, Core Location hledající aktuální geografickou polohu uživatele nebo Core Media, umožňující nízkoúrovňový přístup k audio a video souborům. (12)

2.1.4. Vrstva Core OS

Hlavní funkcí poslední vrstvy je podpora ostatním technologiím za pomocí Accelerate Framework nebo Security Framework. Poslední vrstva poskytuje nízkoúrovňové funkce ostatním technologiím - mezi jinými třeba Accelerate Framework, která nabízí rozhraní pro práci s matematickými funkcemi, nebo Security Framework, zaručující bezpečnost citlivých dat. (12)

2.2. Specifikace jednotlivých verzí iOS

2.2.1. iOS 6

Verze systému iOS 6 vyšla s prodejem telefonu iPhone 5 roce 2012. Mezi hlavní novinky byly mapy od TomTom, sdílení fotek pomocí iCloud či integrace Facebook. Právě kvůli rozepřím se společností Google byla odebrána aplikace YouTube..

Ze začátku byly s mapami od TomTomu drobnější problémy Pražský hrad se nacházel na Strahově a Zlín byl Gottwaldow, tento problém byl však velmi rychle vyřešen. Mapy však přinesly i novinku 3D zobrazení, jež je dostupná pro pěší i automobilové uživatele. Některá světová hlavní města budou moci uživatelé v USA zobrazit v takzvaném flyover módu, tedy ve 3D satelitním zobrazení. (13)

Díky integraci Facebooku je od verze iOS 6 možné synchronizovat kontakty z Facebooku přímo do zařízení. Integrací bylo taktéž docíleno, že v notifikačním centru se zobrazují zprávy.

Jednou z nejlepších funkcí, kterou nový software přinesl je sdílení fotek přes iCloud. Uživatel může nahlížet ze všech zařízení na své fotografie pomocí iCloud, nebo může své přátelé pozvat k nahlédnutí na jeho konkrétní fotografii či album.

2.2.2. iOS 7

iOS 7 sebou v roce 2013 přinesl velké změny převážně po designové stránce. Design je zcela nový avšak logika a vstřícnost uživatelského prostředí byla zachována. Mezi největší změnu se řadí dvě lišty (notifikační a centrum akcí), FaceTime Audio či přehledný multitasking

Horní vysouvací lišta byla upravena pro lepší přehlednost, důležitější je však lišta spodní. Zde bylo umístěno velmi praktické centrum akcí s nejpotřebnějšími funkcemi, jako jsou baterka, zákaz převracení obrazovky či rychlý přístup k Wi-Fi a Bluetooth.

Díky službě FaceTime Audio si uživatelé značky Apple již i volají zadarmo přes

Wi-Fi či mobilní síť.

Až do verze iOS 7 bylo nutné aplikace zavírat trefováním se do miniaturních křížků na spuštěných aplikací, proto velkým zpříjemněním je doslova „odhazování“ aplikací od sebe – tím se aplikace ukončí. Multitaskingové menu se opět aktivuje dvojitým stiskem ústředního tlačítka. (14)

2.2.3. iOS 8

Příchod operačního systému iOS 8 v roce 2014 nebyl takovou změnou po designové stránce jako po softwarové, jako je komunikace mezi platformami iOS a Mac OS X Yosemite, zpřístupnění Touch ID vývojářům či aplikace Health.

Komunikace mezi zařízeními s platformou iOS byla o poznání vylepšena, nyní však tato komunikace je i mezi zařízeními iOS a Mac OS X Yosemite a výš. Jedná se nejen o přesun mezi zařízeními dokumentů či fotografií ale i o možnosti plynule přecházet mezi zařízeními a stále pracovat na jedné věci – dopsat e-mail, zjišťování informací na internetu (stránky co jsou otevřené na iOS se dají okamžitě otevřít na Mac OS a podobně).

Nedílnou součástí nových zařízení s iOS 8 je Touch ID nebo-li identifikace za pomocí otisku prstu. Vývojářům byly zpřístupněné vývojářské nástroje pro Touch ID, které se tak nyní nemusí využívat pouze pro odemykání telefonu, k dispozici budou mít uživatelé několik alternativních klávesnic pro pohodlnější psaní, a zásadní novinka pro využití aplikací je možnost tzv. rozšíření, díky kterým půjde propojit aplikace mezi sebou mnohem jednodušeji než kdy dřív. (15)

Aplikace Health byla vytvořena pro připojení chytrých hodinek od stejnojmenné značky. Slouží jako zdravotní karta, obsahuje tedy veškeré informace, které uživatel zadá. V každém případě měří denní aktivitu uživatele a uživatel má tak možnost náhledu na svoje výsledky. Helath je propojen i s jinými aplikacemi třeba Sleep Cycle a zaznamená údaje o spánku nebo na kalorické diáře a promítá tak i údaje o přijatých kaloriích.

2.2.4. iOS 9

Nejaktuálnější celočíselná verze je prozatím iOS 9, která přinesla v roce 2015 opět významné změny v softwaru ale i mnoho vylepšení stávajících aplikací. Naprostou novinkou je funkce šetření baterie, Siri jako již 100% asistentka či iCloud Drive.

Nejen díky optimalizaci celého systému došlo ke zlepšení výdrže baterie, další možností je zaktivnit funkci šetření baterie, kdy po aktivace dojde k omezení výkonu a tím i vypnutí některých funkcí.

Velmi příjemnou změnou jsou poznámky, které se staly spíš textovým editorem. Jsou tu možnosti profesionálních TO-DO listů jako je checklist, záznam rozpracovaných úloh ale lze jednoduše vkládat fotky, odkazy na webové stránky nebo dokonce svoji polohu.

Nedílnou součástí nového iOS je iCloud Drive fungující jako aplikace zpřístupňující osobní úložiště iCloud. Jedna úplne nová aplikácia, ktorú si môžete dať na domovskú obrazovku (je totiž skrytá v nastaveniach iCloudu a tu si vyberiete, či ju chcete na ploche), je iCloud Drive. Podobne ako Dropbox, je to vlastne prístup k vášmu cloudovému úložisku s tým, že si ho môžete organizovať. (16)

Co se Siri(hlasem ovládaný asistent) týče, naučila se mnoho užitečných věcí. Doposud měla schopnosti převážně vyhledávací, s novou verzí však již zakládá schůzky, vytváří úkoly, odesílá zprávy či e-maily. Tento asistent se pozorováním denní rutiny uživatele naučí sledovat pro uživatele důležité informace a sama mu je předkládá. Siri však stále ještě neumí česky, Apple na jazykových podporách pracuje.

2.3. Android

2.3.1. Struktura os Android

Operační systém Android byl vytvořen společností Google již v roce 2007. Následně byla tato platforma předána sdružení firem Open Handset Alliance (výrobci hardwaru a softwaru telekomunikačních zařízení).

Programovací jazyk operačního systému Android je Java, která je zdarma dostupná pro několik operačních systémů. Zdrojové kódy programu nejsou překládány do strojového kódu procesoru, ale pouze předzpracovávány do tzv. byte-kódu. Ten ještě není závislý na konkrétním procesoru, ale časově náročné fáze kompilace jsou již provedeny. Takto předzpracovaný kód je pro člověka nečitelný. Při spuštění Java programu je byte-kód velmi rychle převeden na strojový kód daného procesoru (s ohledem na použitý operační systém) – to provádí tzv. Java Virtual Machine *(JVM)* (17)

2.3.2. Syntaxe jazyka Java

Jednotlivé proměnné mají přesně určené hodnoty, kterých mohou nabýt. Celkově je Java velmi podobná jazyku C++ až na to že Java je objektová. V Jave není možné mít globální proměnné, vše je vždy součástí objektu nebo třídy.

2.3.3. Porovnání operačních systémů iOS a Android

Ačkoliv má operační systém Android oproti ostatním mnoho výhod je pomalejší než iOS či Windows Phone. Důvodem je vykreslování aplikací v hlavním aplikačním vlákně s normální prioritou. Na rozdíl od operačního systému iOS, kde vykreslování probíhá ve vedlejším vlákně, jež má vyšší prioritu a je tudíž zpracováváno okamžitě. Pokud tedy na os Android dojde k náročnější práci na pozadí, pocítí uživatel tuto akci zpomalením celého systémů.

Toto nastavení vykreslovacího frameworku je velmi obtížně změnitelné. Pokud by došlo ke změně samotného systému většina aplikací by nebyla v tu chvíli kompatibilní.

Vlastní práce

3. Insolvence

Insolvence / Osobní bankrot, nebo-li oddlužení je soudem poskytnutý status pro osobu, která není schopna plnit své závazky vůči věřitelům. Může se jednat o osobu právnickou či fyzickou a především to může být i občan nepodnikající.

O insolvenci může zažádat každý, nebo může být na subjekt vyhlášena. V případě že fyzická osoba či občan neplní své závazky, mají její věřitelé možnost podat návrh na insolvenci. Pokud tak učiní a insolvenční soud návrh přijme, mají věřitelé jistotu, že během 5 let se jim dostane alespoň 30% z pohledávek. Insolvenční návrh musí podat alespoň dva věřitelé. Pokud subjekt na sebe insolvenci vyhlásí (činí tak přes právního poradce zabývajícím se insolvenčním řízením), platí stejné pravidlo - musí dokázat, že je schopen splatit během 5 let alespoň 30% svých závazků.

3.1. Výhody insolvence

Celkové závazky vůči věřitelům klesnou, a to až o 70%. To znamená, že zbytek musí být věřitelem odpuštěn na základně právních předpisů České republiky.
Jakmile je stanovená částka insolvenčním soudem uhrazena, minimálně se musí jednat o 30% závazků, je subjekt zproštěn veškerého placení, jako kdyby uhradil své pohledávky standartním způsobem.

Pokud se věřitel nepřihlásí k pohledávkám před ukončením insolvenčního řízení, jeho pohledávky zanikají.

Díky vyhlášené insolvenci není možné dražit nemovitost, zastaví se exekuční řízení a věřitelé nesmí hlavního dlužníka urgovat. Mohlo by dojít k upřednostnění věřitele, což je protiprávní jednání a na jeho základě může být insolvence zrušena. Pokud tato situace nastane je vyhlášen konkurz, při kterém je prodán veškerý majetek osoby, jež je v insolvenci. Veškeré pohledávky se „stopnou“, již se dále nemohou zvyšovat kvůli pokutám či poplatkům.

3.2. Nevýhody insolvence

Osoba v insolvenci nemá možnost spravovat svoje finance, dělá to za ni insolvenční správce. Ten z platu nechá pouze nezabavitelnou částku vypočtenou z životního minima,

veškeré ostatní finance jsou použity pro věřitele. S tímto životním minimem si dlužník musí vystačit po pět let nebo až do doby, než doplatí své závazky v určité procentuální výši stanovené insolvenčním soudem.

Dle nařízení vlády č. 595/2006 Sb. ve spojení s dalšími příslušnými právními předpisy (zákon č. 110/2006 Sb., nařízení vlády č. 409/2011 Sb., zákon č. 117/1995 Sb., nařízení vlády č. 440/2013 Sb.) se kalkuluje základní částka takto: na osobu dlužníka ve výši dvou třetin součtu částky životního minima a částky normativních nákladů na bydlení pro jednu osobu dle zvláštního právního předpisu („nezabavitelná částka“) - nezabavitelná částka = 2/3 * 9 283 Kč = 6 188,67 Kč. (18)

V případě, že si dlužník vzal hypotéku pro financování nemovitosti a zažádá o insolvenci, může bankovní ústav díky zástavě zažádat insolvenčního správce o prodej nemovitosti za účelem získat zpět co nejvíce ze své pohledávky.

Pokud je insolvence schválena, schvaluje se na osobu nikoliv na smlouvu. Tudíž pokud je hlavnímu dlužníkovi (vlastník smlouvy) schválena insolvence, ostatní účastníci smlouvy mají povinnost hradit zbylé závazky za něj.

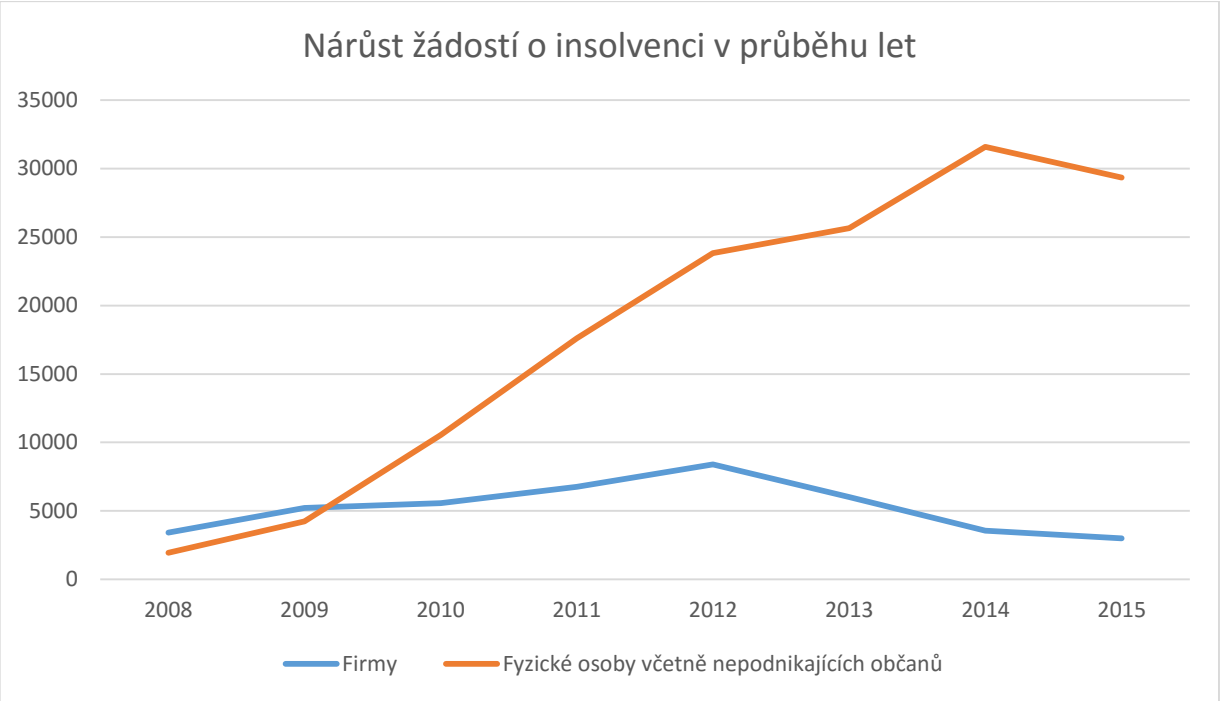
Veškeré finance, které osoba v insolvenci nabyde (výhra v loterii, dědictví….), jsou rozděleny mezi věřitele.

Po celou dobu trvání insolvence (a to i v mimořádných situacích) není možné dané osobě zapůjčovat finance.

3.3. Vývoj insolvence v ČR

Z následujícího grafu 1. je patrný rozsah problému. Od roku 2008 se trend žádostí u firem zvyšoval, a to až do roku 2012, kdy se ekonomická situace uklidnila, pak začal klesat. To bohužel neplatí pro fyzické osoby a občany, kde naopak insolvencí přibývá až do roku 2015. Lidé mají tendence půjčovat si stále více financí, aniž by měli jistotu, že budou schopni v pořádku hradit.

Jakmile se stanou dlužnými, ať už vlivem ztráty zaměstnání či vlivem mimořádné finanční situace, dostávají se díky dluhové pasti hlouběji do svých problémů a jako jedinou možnost vidí vyhlášení insolvence. Tito lidé se informují u insolvenčních správců, jakým způsobem postupovat. Bohužel je častým jevem, že insolvenční správce nesdělí klientovi všechny úskalí insolvence. Čím více má totiž insolvenční správce úkonů, tím více financí obdrží.



Graf 1 – Vývoj insolvencí v ČR

Zdroj dat: (19) a (20)

Vysvětlivky: Oranžová čára zobrazuje žádosti o insolvenci u fyzických osob (živnostníky) a osoby nepodnikající – občany české republiky.
Modrá čára ukazuje žádosti o insolvenci firem.

Pro ilustraci v roce 2014, kdy počet žádostí o insolvenci u fyzických osob a občanů byl nejvyšší (přes 30 tisíc), bylo oddlužení povoleno 24 900 případů. Což je skoro 80%!

4. Odstoupení od smlouvy věřitelem / zesplatnění smlouvy

Jak odstoupení od smlouvy věřitelem, tak její zesplatnění poukazují na tentýž fakt – na základě porušení smlouvy (ze strany klienta) věřitel smlouvu zruší a nechává si navrátit zbylou pohledávku i s úroky zpět (celkovou nesplacenou částku včetně úroků do konce smlouvy).

Každá bankovní i nebankovní společnost má jasně definováno ve smluvní části podmínky, kdy odstoupení od smlouvy učiní. U nebankovních společností většinou stačí dva měsíce nehradit. Oproti tomu u bankovních společností je situace mnohem mírnější.

4.1. Kroky při odstoupení

V případě, že klientovi byla odstoupena smlouva, obdrží korespondenci s vyčíslením ke konkrétnímu datu. Pokud neuhradí celou pohledávku v termínu, věřitel smlouvu předá svým právním zástupcům, kteří mají za úkol smlouvu připravit pro soudní jednání. Samozřejmě zde vznikají první poplatky (za právní zástupce).

Následuje soud, jež rozhodne podle podmínek smlouvy, které jasně definují porušení smlouvy ze strany klienta. Klient soud prohraje čili je povinen hradit soudní výlohy.

Jakmile soud vynese rozsudek ve prospěch věřitele, vykonavatelem je exekutor, který má za úkol obstavit účty, strhnout finance ze mzdy či rozprodat majetek za účelem navrácení financí věřiteli. Samozřejmě i exekutor si za své služby vezme odměnu a to opět od klienta.

4.2. Následky odstoupení / zesplatnění smlouvy

Mezi nejčastější následky při odstoupení od smlouvy (v případě že byla zajištěna nemovitostí) patří prodej nemovitosti. Prodej se uskutečňuje přes dražbu. Bankovní společnost nemá za cíl nemovitost prodat za nejvyšší cenu. Pro ni je prioritou prodat nemovitost co nejrychleji. To v praxi znamená, že se do dražby dává nemovitost hodně pod cenou. I přesto, že se nemovitost draží, nemusí dosáhnout prodaná nemovitost ceny pohledávky. Což není neobvyklé a právě díky tomu je na většině smluv více účastníků. Ti mají zajistit případné dorovnání financí (společně s hlavním dlužníkem) do výše nesplacené pohledávky. Zda a do jaké výše bude hradit jen hlavní dlužník či ostatní účastníci při zajištění nemovitostí i bez zajištění, je pouze na uvážení soudu.

5. Vývoj aplikace Kalkulačka úroků

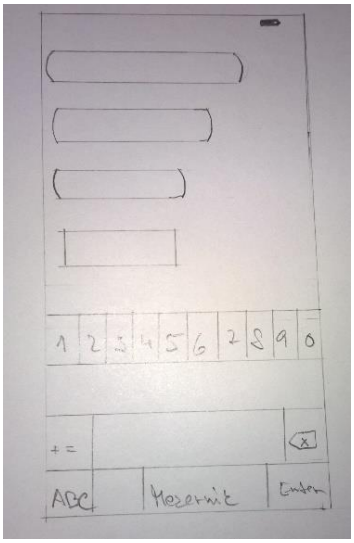
5.1. Důvody pro vznik aplikace

Kalkulačka úroků má sloužit jako aplikace pro mobilní platformu iOS. Jejím účelem je mít dostupnou možnost výpočtu konkrétní nabídky přímo při ruce i bez znalosti finanční matematiky.

Mezi nejčastější formy nabídky od nebankovních společností patří totiž telemarketing. Je velmi jednoduché takovouto půjčku obdržet, stačí si pro ni zavolat či napsat přímo sms zprávu. Většina lidí totiž již nezkoumá nabídky ostatních společností, či nehledá na internetu kalkulačku pro výpočet úroků. Těchto kalkulaček je opravdu nepřeberné množství, a to jak od dané bankovní či nebankovní společnosti, tak volně k dispozici. S touto aplikací bude mít možnost uživatel dále se rozhodovat pouze na svém místě a nemusí cokoliv vyhledávat na internetu. Kalkulačka úroků mu totiž spočítá, kolik přeplatí.

5.2. Paper Prototyping

Na obrázku 5.1 je vidět první návrh aplikace pomocí paper prototyping (návrh za pomocí papíru). První buňka slouží pro vložení celkové částky k půjčení, druhé pole je potřeba k vypsání řádné platby, která bude hrazena. Do posledního pole se bude vkládat úroková míra. Je jedno, jestli bude vkládána hodnota RPSN nebo p.a. obě tyto úrokové míry reprezentují roční úrokovou míru. V hodnotě RPSN jsou však obsaženy všechny poplatky za rok, typu vedení účtu. Proto je vždy hodnota p.a. o něco málo nižší. Na posledním místě se nachází tlačítko pro výpočet.

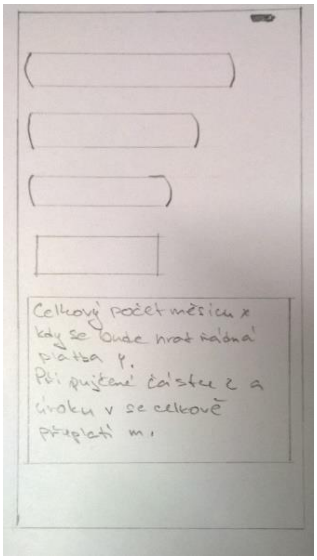


Obrázek 5.1 – Paper Prototyping – Náhled na aplikaci

Zdroj: vlastní

Klávesnice se zobrazí při poklepu na vkládací textové pole. Primárně se zobrazuje klávesnice čísel a znaků. Klient může přepnout na písmena, ale díky tomu, že se do těchto polí nebudou vkládat slova či písmena, je zbytečné, aby musel „překlikávat“ již při spuštění. Pokud zrovna není klávesnice používána (uživatel se nenachází v okně pro úpravu hodnot), klávesnice není aktivní, aby popřípadě nebránila zobrazení výsledku.

Při stisknutí tlačítka systém zpracuje zadaná data a zobrazí výsledek na místě, kde je klávesnice, jak je zřejmé z obrázku 5.2. Výsledek je reprezentován celkovou interpretací hodnot, aby byly všechny informace zřetelné a dobře pochopitelné.



Obrázek 5.2 – Paper prototyping - Zobrazení výsledků v aplikaci

Zdroj: vlastní

5.3. Výpočet a příprava pro programovací část.

Celý výpočet je charakterizován pevným splácením, které bankovní i nebankovní společnosti používají pro definování úroků. Čili z celkové částky se každý měsíc odečte řádná platba, dokud není celkový úvěr splacen. Z řádné platby se vypočítává úrok pomocí úrokové míry násobené aktuálním zůstatkem. Jednotlivé úroky se sečtou po celou dobu splácení a vznikne celkový úrok, který se splatí během celé doby splácení.

Pro přehlednější zobrazení poslouží příklady 1 a 2 s názornou ukázkou splácení.

Půjčka na 100 000 Kč při řádné platbě ve výši 2 860 Kč měsíčně a úrokové míře 3.7 % p.a.

Vzorec pro výpočet pevného splácení = $\sqrt[m]{(1 + i_{roční})}$ -1

Pokud dosadíme do vzorce hodnoty = $\sqrt[12]{(1 + 3{,}7\%)}$ - 1

Dostaneme úrokovou míru 0,003032249. Tuto úrokovou míru násobíme vždy aktuálním zůstatkem. Výsledné číslo je úrok, který instituce obdrží za své služby. Na následujícím příkladu 2 je vidět podrobně rozepsáno, jak výpočet vypadá pro prvních několik měsíců.

m = počet úrokovacích období

$i_{roční}$ = úroková míra roční

Příklad 1 – Postup výpočtu úroků

Zdroj: vlastní

Měsíc	Zůstatek	Řádná platba v Kč	Úrok z řádné platby v Kč vypočten z aktuálního zůstatku * úroková míra
1.	100000	2860 Kč	294,55 Kč
2.	97140	2860 Kč	285,88 Kč
3.	94280	2860 Kč	270,67 Kč
4.	91420	2860 Kč	262,46 Kč
5.	88560	2860 Kč	254,25 Kč
6.	85700	2860 Kč	237,83 Kč
7.	82840	2860 Kč	229,61 Kč

Příklad 2 – Ukázka úroků

Zdroj: vlastní

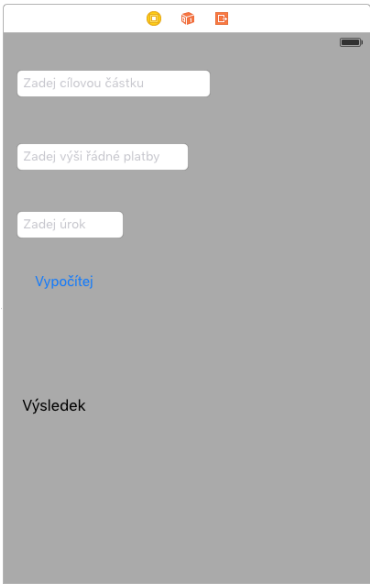
Příklad 2 vysvětluje, jak úrokování probíhá:

1) každý měsíc se od zůstatku odečte řádná platba
2) úrokovou míru je potřeba násobit aktuálním zůstatkem tím se zjistí úrok

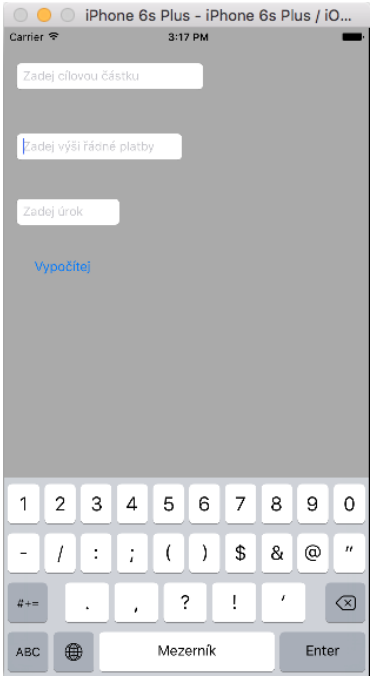
5.4. Vytvoření aplikace

5.4.1. Grafická struktura

Tvorbu aplikace započne vkládáním textových editorů (text), do kterých bude klient zadávat hodnoty. Dále je potřeba přidat tlačítko (button), které při zmáčknutí začne ze zadaných hodnot počítat výsledek. Jako poslední se vloží štítek (label), do kterého se zobrazují hodnoty. K náhledu v obrázku 5.3 a 5.4.



Obrázek 5.3 – Grafický návrh aplikace v Xcode



Obrázek 5.4 – Grafické znázornění aplikace v simulátoru

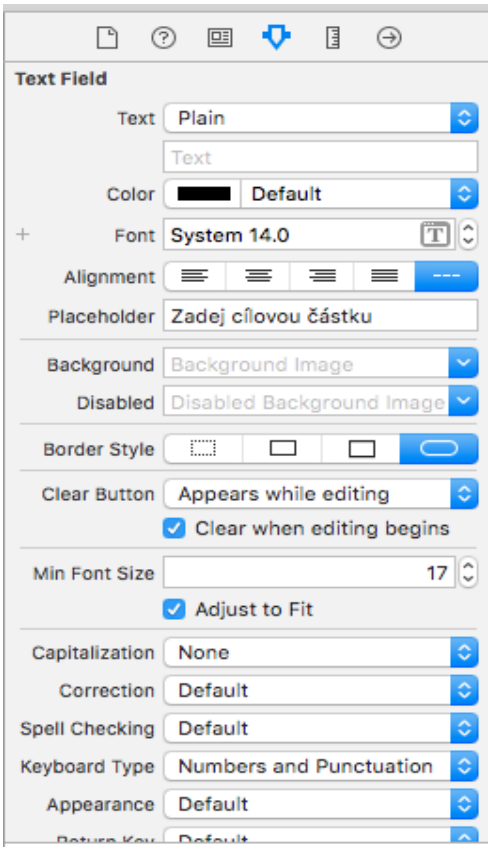
Zdroj: vlastní

5.4.2. Grafická funkcionalita

Pro úpravu funkcionality jednotlivých položek vložených do aplikace je nutné dostat v návrhu aplikace (grafické části) do sekce „Zobraz inspektora prvků“ (Show the atrributes inspector).

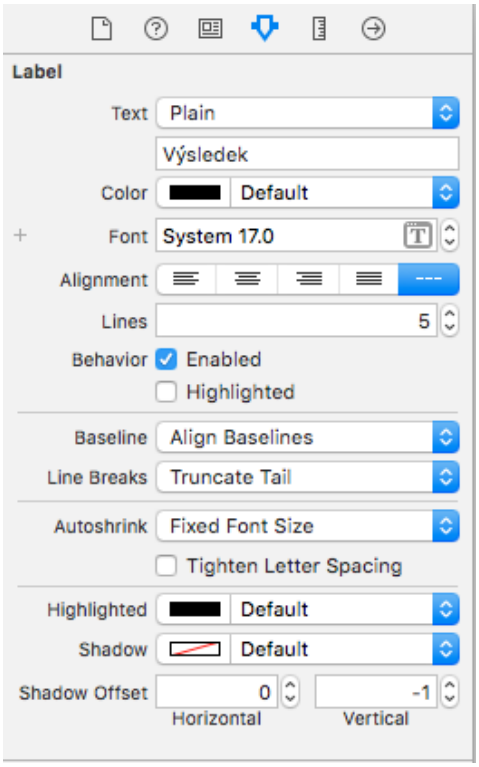
V části textové pole (text field) obrázek 5.5 je dobré zadat do pole Placeholder jakýkoliv text. Tento text se zobrazí v poli aplikace před zadáním hodnoty uživatelem. Není tudíž nutné přidávat popisky nad textová pole a tím si případně ubírat místo v aplikaci jako takové. Dále je praktické vybrat možnost „Smaž, jakmile se začne psát“ (Clear when editing begins), aby uživatel nemusel odmazávat text vložený do pole aplikace. Pole typ klávesnice (Keybord Type) je upraven pro tyto účely, aby zobrazovaly klávesnici s čísly a interpunkcí (Number and Punctuation).

V části štítek (label) je potřeba upravit počet řádků v případě, že aplikace bude vypisovat více textu. Je potřeba změnit opět přes sekci „Inspektora prvků“ pouze přidáním řádků (lines), jak vidíme na obrázku 5.6.



Obrázek 5.5 – Inspektor prvků – vkládání textu do Placeholder

Zdroj: vlastní

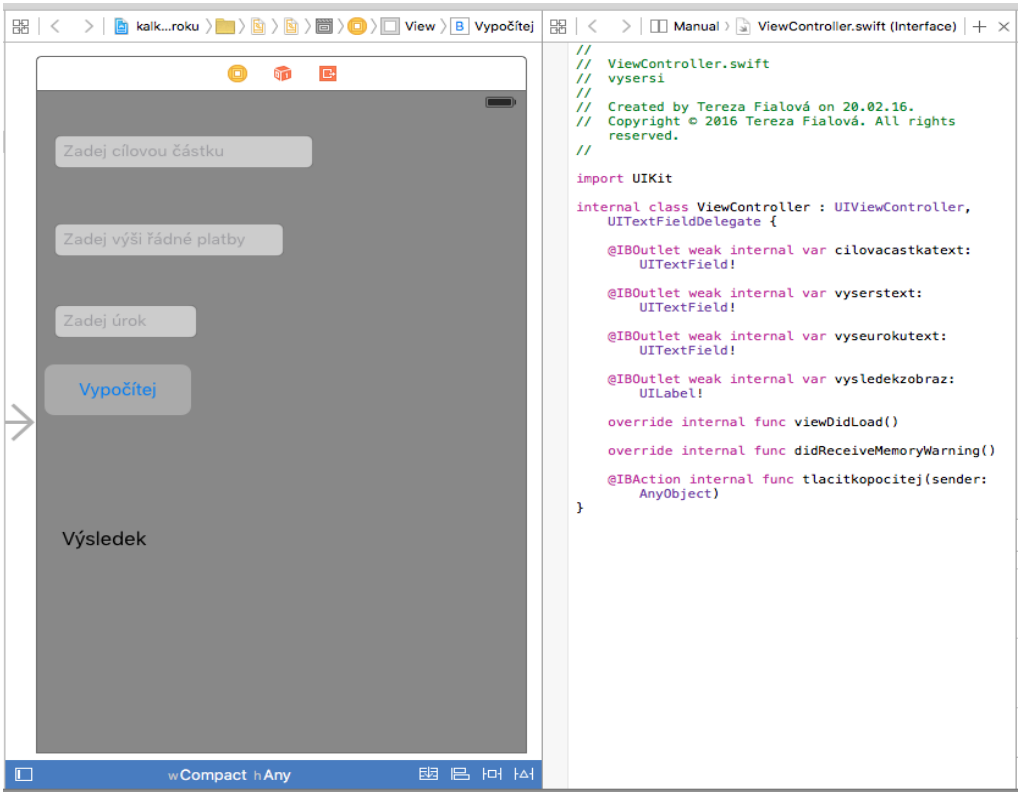


Obrázek 5.6 – Inspektor prvků – řádky

Zdroj: vlastní

5.4.3. Napojení funkcí do kódu

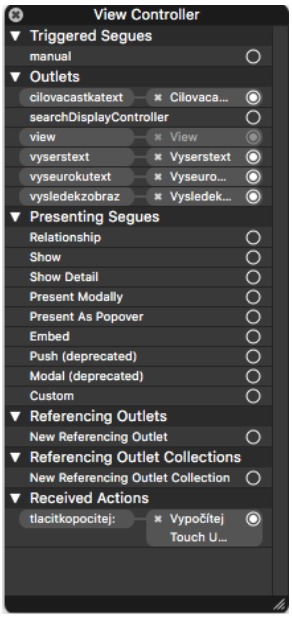
Po vytvoření grafické stránky projektu je potřeba jednotlivé části uživatelského rozhraní napojit do programovací části, aby bylo možné s nimi pracovat. To se provádí velmi jednoduše. Je potřeba si kromě grafické části zobrazit zároveň část programovací. Napojení se provede označením dané části uživatelského rozhraní a následným přidržením obou tlačítek myši a přetažením do programovací části se vytvoří zdrojový kód pro příslušné uživatelské rozhraní. Na obrázku 5.7 je k vidění názorná ukázka, jak vypadá, když jsou části uživatelského rozhraní napojeny na zdrojový kód.



Obrázek 5.7 – Napojení UI do programovací části

Zdroj: vlastní

Ověření, že napojení proběhlo zcela v pořádku, se dá zjistit jednoduchým stiskem obou tlačítek myši na plochu aplikace mimo uživatelské rozhraní. Zobrazí se následující „pohled dispečera“ (View Controller) obrázek 5.8 kde je jasně patrné, kam je která část uživatelského rozhraní kam napojena.



Obrázek 5.8 – Kontrola zda jsou všechny části napojeny

Zdroj: vlastní

5.4.4. Převedení textového pole na číslo s desetinou částí

V případě této kalkulačky uživatel zadává jako hodnotu číslo, avšak do textového pole. Swift tedy automaticky typ proměnné bere jako řetězec (string), což je problém pokud aplikace má pracovat s čísli a jejich desetinnými místy. Každé z textových polí se musí převést na typ proměnné dvojnásobek (double), jak je vidět z následující části kódu.

```
let cilova_castka = Double(cilovacastkatext.text!)
let vyse_rs = Double(vyserstext.text!)
let vyse_uroku = Double(vyseurokutext.text!)
```

Tímto postupem se zároveň vytvoří proměnné cilova_castka, vyse_rs a vyse_uroku již v typu double, které jsou zapotřebí v následující části programování aplikace.

5.4.5. Definice proměnných

Všechny hodnoty zadané ve zdrojovém kódu, které slouží k výpočtu, musí být zadané hodnotou double. V opačném případě nás Swift upozorní, že s proměnnými typu double a celé číslo (integer) nelze pracovat zároveň.

```
var pomocna = Double (1)
var pocet_mesicu = 0
var mesic = 0
var zustatek = cilova_castka
var urok = Double (0)
var vypocet = Double (0)
var celkovy_urok = Double(0)
```

Hodnota pomocna je připravena k počítání jednak pod odmocninou, kde je připočtena k úrokové míře a jednak za odmocninou. Jakmile se totiž spočte odmocnina je potřeba hodnotu pomocna odečíst jak bylo ve vzorečku v dodatku 4.3.I.

Díky tomu, že uživatel zadává hodnotu úroku jako 3.7, nikoliv jako 0,037 je zapotřebí upravit úrok aby byl použitelný pro kalkulačku úroků.

```
var urok_vyse_upravena = urok_vyse / 100
```

5.4.6. Pomocné funkce

Před vytvořením cyklu je dobré vytvořit pomocné výpočty pro celkové zjednodušení celého cyklu. Není nutné, aby se při každém cyklu dopočítávaly znovu všechny hodnoty, jež

budou potřeba být použity v samotném cyklu. V zaklad je deklarována funkce pod odmocninou tedy (1+0,037).

```
let zaklad = pomocna + urok_vyse_upravena
```

Pro samotnou odmocninu je potřeba definovat vlastní funkci. Již z prvního pohledu je zřejmé, že funkce není přímo pro odmocninu ale pro mocninu. Avšak z matematiky je jasné, že pokud je zapotřebí odmocnit 5 je to to samé jako když se 5 umocní na jednu polovinu.

```
infix operator ** { associativity left precedence 160}
func ** (left: Double, right: Double) -> Double {
   return pow(left, right)
}
```

Nyní je možné použít upravenou odmocninu pro výpočet skoro celé úrokové míry. Stačí pouze definovat hodnotu odmocnina, jež bude nadále využita a vložit proměnnou zaklad, která bude umocněna na jednu dvanáctinu.

```
let odmocnina = pow(zaklad, 1/12)
```

Po umocnění se mezikrokem už pouze přes proměnou pomocna odečte podle vzorce číslo 1, výsledkem je pak již zobrazení celé definované úrokové míry.

```
let urokova_mira = odmocnina - pomocna
```

Tímto byly definovány všechny proměnné pro vytvoření cyklu.

5.4.7. Cyklus

Princip této kalkulačky je poměrně jednoduchý, jak již bylo zobrazeno v dodatcích 1 a 2 Struktura cyklu je tedy následující.

Dokud zůstatek je vyšší než řádná platba, do proměnné měsíc přičítej +1.

```
while zustatek > vyse_rs {
      mesic = mesic + 1
```

Úrok vytvoř vynásobením zůstatku a úrokové míry, pak vždy odečti výši řádné platby. Jakmile se cyklus bude opakovat, aby bylo zajištěno, že úroková míra se bude snižovat v návaznosti na snižující se zůstatek.

```
      urok = zustatek * urokova_mira
```

```
zustatek = zustatek - vyse_rs
```

Celkový úrok čili částku, kterou klient zaplatí celkem za celé období, se bude přičítat do proměnné celkový úrok.

```
celkovy_urok = celkovy_urok + urok
```

Celkový počet měsíců se sečte v proměnné počet měsíců, aby klient věděl, kolik měsíců bude trvat případný závazek.

```
pocet_mesicu = mesic + 1
```

5.4.8. Návratové body

Jestliže jsou zadané hodnoty korektní, provede se výpočet. Klient se dozví, kolik bude celkově hradit na úrocích, dále kolik měsíců bude splácet. Zopakují se pro kontrolu i zadané hodnoty, jako je úroková výše, výše řádné platby a celková částka.

```
override func viewDidLoad() {
    super.viewDidLoad()

    vysledekzobraz.hidden = true

    vysledekzobraz.hidden = false
if cilova_castka != nil && vyse_rs != nil && vyse_uroku != nil {
    let vysledek = Double(celkovy_urok!)

    vysledekzobraz.text = "Celkový počet měsíců je \(pocet_mesicu), kdy se bude hradit řádná platba \(vyse_rs) Kč. Při půjčené částce \(cilova_castka)Kč a úroku \(urok_vyse) % se celkově přeplatí \(celkovy_urok) Kč"
```

Pokud uživatel zadá neplatnou hodnotu (třeba písmeno nebo úrok s čárkou mezi čísly), je důležité definovat návratové body, aby byla zajištěna funkčnost kalkulačky. Tedy pokud je hodnota zadaná uživatelem chybná (není číslo nebo obsahuje čárku), zobrazí se požadavek na zadání čísla či použití čárky při zadávání úroku.

```
}else{
      vysledekzobraz.text = "Musíte zadat čísla, při zadávání úroku použijte tečku."
    }

}
```

6. Publikace Aplikace

6.1. Podmínky pro publikování na App Store

Jakmile je aplikace hotová a funkční, tedy i s vytvořenou ikonou, protože právě tu vidí uživatel jako první a právě ikona aplikaci prodává, je čas vložení samotné aplikace do obchodu. Než však tato situace nastane, je potřeba splnit podmínky dané společností Apple.

V první řadě je pro vývojáře důležité, aby aplikace nebyla zamítnuta kvůli některému z následujících důvodů:

- *padá při provozu;*
- *vykazuje chyby;*
- *není v souladu s tím, co uvedl vývojář;*
- *zahrnuje nedokumentované nebo skryté funkce v rozporu s popisem aplikace;*
- *používá neveřejné API;*
- *čte nebo zapisuje data mimo svůj určený kontejner;*
- *stahuje kód jakýmkoliv způsobem nebo formou;*
- *instaluje nebo spouští jiný spustitelný kód;*
- *jde o beta, demo, nebo zkušební verzi;*
- *iPhone aplikace musí běžet na iPadu bez úprav, na iPhone rozlišení a na 2X iPhone 3GS rozlišení;*
- *je kopií aplikací v App Store již umístěných, zejména pokud takových aplikací existuje mnoho;*
- *není velmi užitečná, například jde o prosté webové stránky přibalené v rámci aplikace;*
- *je primárně zaměřena na marketing nebo reklamu;*
- *jde o trik nebo obsahuje falešné funkce, které nejsou jasně označeny;*
- *zobrazuje webové stránky, kde je nutné použít v rámci iOS WebKitu a JavaScript WebKitu;*
- *podporuje nadměrnou konzumaci alkoholu, nelegálních látek, nebo nabádá děti nebo mladistvé konzumovat alkohol či kouřit cigarety;*
- *poskytuje nesprávné diagnostické nebo nepřesné údaje;*
- *vývojáři „spameři“ s mnoha verzemi podobných aplikací budou odstranění z programu Developer iOS;*
- *obsahuje například jen jednu skladbu nebo film – takové by měly být nahrány do iTunes Store;*

- *obsahuje knihu – také by měly být nahrané do iBookstore;*
- *svévolně omezuje, kteří uživatelé mohou používat aplikaci, například podle msta nebo dopravce;*
- *neřídí se pokyny pro iOS pro ukládání dat;* (11)

Pokud není aplikace zamítnuta, kvůli některému z uvedených důvodů, je potřeba k publikování certifikát.

6.2. Vývojářský certifikát

Nejprve je potřeba u společnosti Apple založit developerský účet. To se provádí na těchto stránkách (developer.apple.com/programs/ios) poměrně snadno. Následuje platba 99 dolarů, která se opakuje každý rok. Zato obdrží žadatel certifikát od Apple a může začít vkládat své aplikace na App store.

Certifikát je potřeba stáhnout a elektronicky podepsat. Následně je třeba podepsaný nahrát zpět na svůj profil developerského účtu.

6.3. Vložení aplikace na App store

Vkládání probíhá pomocí App IDs sekce. V této sekci se musí vyplnit údaje o dané aplikaci. Tyto informace budou sloužit jednak pro firmu Apple tak i pro zákazníky.

Dále je potřeba další certifikát, tentokrát pro samotnou distribuci aplikace. Ten se vytvoří v sekci Provisioning.

Následně se vloží samotný zdrojový kód aplikace. Aplikace není schválena a vložena okamžitě. Firma Apple nejprve provede kontrolu dané aplikace, zda splňuje všechny podmínky a podle toho ji buď přijme, nebo zamítne.

7. Závěr

Cílem práce bylo vytvoření aplikace „Kalkulačka úroků“ pro snadný a rychlý přístup k porozumění nabídky finančních institucí a vysvětlení problematiky insolvence a odstoupení od smlouvy / zesplatnění.

Při zkoumání dílčích cílů byly získány tyto výsledky:

- *Porovnání programovacích jazyků Swift a Objective-C.*

Aplikace byla vytvořena pomocí programovacího jazyka Swift pro operační systém iOS, který byl vybrán na základě provedené analýzy jako nejvhodnější z důvodu jednoduchosti psaní kódu, bezpečnosti a jeho zpětné kompatibilitě s Objective-C, tudíž mohou být použity oba jazyky současně.

- *Charakteristika operačního systému iOS a informace o jednotlivých verzích.*

Mobilní operační systém iOS se skládá ze 4 vrstev, každá má na starosti jinou důležitou funkcionalitu. Jedná se o jeden operační systém pro více hardwarových zařízení od této společnosti (iPhone, iPod, iPad, Watch a Apple TV). S každou novou verzí operačního systému iOS jsou velké změny. Programátoři v Apple se snaží neustále zlepšovat jak funkčnost, intuitivnost tak design. V každé verzi je mnoho změn, při přechodu z iOS 6 na iOS 7 byla velká změna v designu. Siri jako 100% asistentka schopna reagovat a vykonávat povely přišla až s verzí iOS9.

- *Analýza zvoleného prostředí Xcode a v dalších nativních programech.*

Díky charakteristice jazyka Swift a Objetive-C je zřejmé, že jazyk Swift obsahuje spoustu výhod pro vývoj aplikace, i přes složitější možnost výuky pomocí knižní publikace pouze v angličtině. Pro potřeby této bakalářské práce byla zvolena možnost vývojového prostředí Xcode, která je ze všech posuzovaných vývojových prostředí nejvíce přívětivá z pohledu vybavenosti, do nějž se nemusí importovat žádné frameworky, pluginy či stahovat přímo emulátor, aby byla zajištěna funkčnost samotného programování.

- *Hlavním cílem této bakalářské práce je ukázat vývoje aplikace „Kalkulačka úroků“ pro operační systéme iOS od společnosti Apple.*

Vlastní vývoj aplikace probíhal dvěma směry. První byla práce na funkčnosti aplikace, jejíž vývoj probíhal v Playground, zatímco uživatelské rozhraní bylo vytvářeno a definováno přímo v Xcode.

Tato aplikace, jejíž pracovní název je „Kalkulačka úroků“, je navržena tak, že po zadání výše úvěru, zvolené částky měsíčního splácení a úrokové sazby (p.a. či RPSN) vypočte částku, kterou příjemce půjčky uhradí nad rámec jistiny. Dále zobrazí počet měsíců, po které bude půjčka splácena. S touto aplikací může uživatel mít přehled o tom, jak výhodný či nevýhodný je finanční produkt a zda je tento produkt pro něj dostupný.

- *V praktické části vysvětlit problematiky insolvence a odstoupení od smlouvy.*

Primárním důvodem proč byla realizována tato aplikace, je velké množství osob v České republice, jež se rozhoduje při pořízení jakékoliv formy úvěru velmi intuitivně a není tato záležitost řešena exaktně. Dochází pak k situacím jako je neschopnost plnit své závazky z těchto úvěrů plynoucích, což nejednou vede k opětovným finančním zápůjčkám a tímto způsobem se lze dostat do dluhové pasti. Toto jednání vede pro konzumenta úvěrů k odstoupení od smlouvy ze strany věřitele. Pokud dojde k nesplácení pohledávek, může věřitel podle smluvních podmínek smlouvy žádat o navrácení celé zbývající částky zpět včetně nesplacených úroků. Pokud se osoba dostane do situace, kdy nemůže dostát svým závazkům, hrozí pro něj reálná možnost insolvence, což znamená, že po dobu až 5-ti let nebude moci konzumovat své vlastní příjmy na základě svého rozhodnutí, ale tyto příjmy za něj bude spravovat insolvenční správce.

- *Popis publikace aplikace.*

Publikace vlastní aplikace je poměrně snadná avšak časově náročná. Postup je zřetelně popsán na stránkách společnosti Apple v sekci pro vývojáře aplikací. Pokud však vývojář nechce být prodělečný, musí vložená aplikace být populární, aby se mu vrátily roční náklady za vývojářský certifikát.

- *Závěrem zhodnotit obecné poznatky, vytvořit specifické závěry a vložit doporučení pro konzumenty úvěrů.*

Tato aplikace může pomoci při úmyslu realizace úvěru, avšak záleží na zkušenosti a obezřetnosti jednotlivých osob.

Obecná pravidla při zvažování úvěru jsou:

1) Mít finanční rezervu pro veškeré výdaje alespoň na půl roku při ztrátě zaměstnání či dlouhodobé pracovní neschopnosti.
2) Být schopen při výpadku platby uhradit dvě měsíční platby v jednom měsíci.

8. Seznam literatury

1. **Apple, Inc.** *The Swift Programming Language (Swift 2.1).* Cupertino, CA : Apple Inc., 2015.

2. **HOLZMAN, Ondřej.** jablickar.cz. [Online] 16. leden 2015. http://jablickar.cz/stale-vice-vyvojaru-pouziva-programovaci-jazyk-swift-od-applu/.

3. **O´GRADY, STEPHEN.** redmonk.com. [Online] 14. leden 2015. http://redmonk.com/sogrady/2015/01/14/language-rankings-1-15/.

4. **RŮŽIČKA, Jan.** itnetwork.cz. [Online] 2015. http://www.itnetwork.cz/programovani/swift/uvod-do-jazyka-swift/.

5. **ROBIME.** robime.it. [Online] 15. září 2014. http://robime.it/strucny-uvod-noveho-programovacieho-jazyka-swift/.

6. **ROOT.** root.cz. [Online] 20. září 2011. http://blog.root.cz/babel/strucna-historie-objective-c/.

7. **APPLE, Inc.** developer.apple.com. [Online] 21. říjen 2015. https://developer.apple.com/library/ios/recipes/Playground_Help/Chapters/AboutPlaygrounds.html.

8. **MIKO, Jaromír.** letemsvetemapplem.eu. [Online] 4. červen 2014. https://www.letemsvetemapplem.eu/2014/06/04/podivejte-se-na-prvni-hru-ktera-vznikla-v-programovacim-jazyku-apple-swift/ .

9. **ALESSI, Patrick.** *Vývoj her pro iPhone a iPad: Programování pro iOS = Beginning iOS game development.* Brno : Zoner soffware, a.s., 2012. ISBN 978-80-7413-199-8.

10. **CODINGEXPLORER.** codingexplorer.com. [Online] 6. leden 2015. http://www.codingexplorer.com/hello-world-first-ios-app-swift/.

11. **VÁVRŮ, Jiří.** *iPhone - Vývoj aplikací.* Praha : Granada Publishig, a.s., 2012. ISBN 978 - 80- 247 - 4457-5.

12. **AKTUALNE.cz.** aktualne.cz. [Online] 19. července 2011. http://www.aktualne.cz/wiki/veda-a-technika/ios-apple/r~i:wiki:1558/.

13. **NOVÁK, Adam.** idnes.cz. [Online] 20. září 2012. http://mobil.idnes.cz/ios-6-pro-iphone-je-venku-0wq-/iphone.aspx?c=A120919_210614_iphone_ada.

14. —. idnes.cz. [Online] 19. září 2013. mobil.idnes.cz/ios-7-pro-iphone-0gy-/iphone.aspx?c=A130918_031931_iphone_ada.

15. **HOLZMAN, Ondřej.** jablickar.cz. [Online] 17. září 2014. http://jablickar.cz/apple-vydal-ios-8-pro-iphone-ipad-a-ipod-touch/.

16. **MIKO, Jaromír.** letemsvetemapplem.eu. [Online] 16. září 2015. https://www.letemsvetemapplem.eu/2015/09/16/kompletni-prehled-novinek-ktere-prinasi-ios-9/.
17. **SEMECKÝ, Jiří.** interval.cz. [Online] 26. dubna 2002. https://www.interval.cz/clanky/naucte-se-javu-uvod/.
18. **Ministerstvo spravedlnosti, ČR.** insolvencni-zakon.cz. *Insolvenční zákon.* [Online] 14. březen 2006. http://insolvencni-zakon.justice.cz/kalkukator-splatek/kalkulacka-vypocet.html.
19. **OSVALDOVÁ, Miroslava, MORES, Cyril a MENŠÍKOVÁ, Stanislava.** Creditforem. [Online] 2016. http://www.creditreform.cz/fileadmin/user_upload/CR-International/local_documents/cz/Presseartikel/Vyvoj_insolvenci_v_Ceske_republice_v_roce_2015.pdf.
20. **SAMKOVÁ, Dana.** cfoworld.cz. [Online] 6. leden 2014. http://cfoworld.cz/financni-sluzby/vyvoj-insolvenci-v-cr-v-roce-2013-2803.

9. Seznam příloh