Katedra informatiky
Přírodovědecká fakulta
Univerzita Palackého v Olomouci

BAKALÁŘSKÁ PRÁCE

Využití rozšířené reality při prezentaci katedry

2023

Dalibor Janeček

Vedoucí práce:
Mgr. Roman Vyjídáček

Studijní program: Aplikovaná informatika,
kombinovaná forma

Bibliografické údaje

Autor:	Dalibor Janeček
Název práce:	Využití rozšířené reality při prezentaci katedry
Typ práce:	bakalářská práce
Pracoviště:	Katedra informatiky, Přírodovědecká fakulta, Univerzita Palackého v Olomouci
Rok obhajoby:	2023
Studijní program:	Aplikovaná informatika, kombinovaná forma
Vedoucí práce:	Mgr. Roman Vyjídáček
Počet stran:	34
Přílohy:	elektronická data v úložišti katedry informatiky
Jazyk práce:	český

Bibliographic info

Author:	Dalibor Janeček
Title:	Use of augmented reality in the presentation of the department
Thesis type:	bachelor thesis
Department:	Department of Computer Science, Faculty of Science, Palacký University Olomouc
Year of defense:	2023
Study program:	Applied Computer Science, combined form
Supervisor:	Mgr. Roman Vyjídáček
Page count:	34
Supplements:	electronic data in the storage of department of computer science
Thesis language:	Czech

Anotace

Tato bakalářská práce se zaměřuje na implementaci aplikace UPOL AR pro zařízení s operačním systémem iOS. Cílem této aplikace je představit Katedru informatiky Univerzity Palackého v Olomouci pomocí rozšířené reality. Uživatelé mají možnost využívat aplikaci jak v budově Přírodovědecké fakulty, tak i mimo ni. V budově aplikace rozšiřuje a oživuje prostor, zatímco mimo ni umožňuje vzdálenou prohlídku prostor katedry. Aplikace nabízí různé režimy rozšířené reality, z nichž každý poskytuje unikátní přístup k prozkoumání obsahu. Tyto režimy jsou podrobně popsány v této práci.

Synopsis

This bachelor thesis focuses on the implementation of the UPOL AR application for iOS devices. The aim of this application is to present the Department of Computer Science of Palacký University in Olomouc using augmented reality. Users have the possibility to use the application both inside and outside the Faculty of Science building. Inside the building, the application extends and animates the space, while outside it allows remote viewing of the department's premises. The app offers different augmented reality modes, each providing a unique approach to explore the content. These modes are described in detail in this thesis.

Klíčová slova: ARKit; SwiftUI; rozšířená realita; iOS; 3D

Keywords: ARKit; SwiftUI; augmented reality; iOS; 3D

Děkuji svému vedoucímu práce Mgr. Romanovi Vyjídáčkovi za jeho cennou podporu, vedení, rady a za trpělivost a motivaci, kterou mi projevoval při řešení výzev, které se v průběhu této bakalářské práce objevily. Dále děkuji všem svým blízkým a kolegům za jejich trpělivost a podporu během mého studia.

Odevzdáním tohoto textu jeho autor/ka místopřísežně prohlašuje, že celou práci včetně příloh vypracoval/a samostatně a za použití pouze zdrojů citovaných v textu práce a uvedených v seznamu literatury.

Obsah

1 Úvod

Tato práce se věnuje využití technologie rozšířené reality (AR, z anglického augmented reality) a jejímu využití k prezentaci Katedry informatiky Univerzity Palackého v Olomouci.

První část práce se věnuje popisu AR včetně příkladů jejího využití v různých oblastech. Představuje také potenciál budoucího rozvoje AR a ukazuje možnosti, které přináší pro další zlepšení interakce mezi člověkem a digitálním světem.

Druhá část se zaměřuje na konkrétní platformu vybranou pro implementaci prezentace Katedry informatiky, jíž je iOS. Obsahuje popis platformy a jejich vlastností, které ji činí vhodnou pro využití AR technologie. Dále se zaměřuje na programovací jazyky, které jsou používány pro tuto platformu, a představí také frameworky, které slouží k návrhu uživatelského rozhraní pro iOS aplikace.

Samostatná část této práce se zabývá nativním frameworkem pro AR na platformě iOS, který umožňuje efektivní vývoj AR aplikací.

Poslední část této práce popisuje samotnou aplikaci, která využívá AR technologii k představení oboru informatiky a Katedry informatiky. V rámci aplikace jsou implementovány různé přístupy k AR v závislosti na poloze uživatele. Pokud se uživatel nachází v blízkosti budovy Přírodovědecké fakulty, kde se Katedra informatiky nachází, aplikace nabídne možnosti k oživení prostoru a navigaci po budově fakulty. Pokud je však uživatel vzdálen od budovy, aplikace poskytne uživateli vzdálenou prohlídku budovy a interaktivní seznámení s Katedrou informatiky.

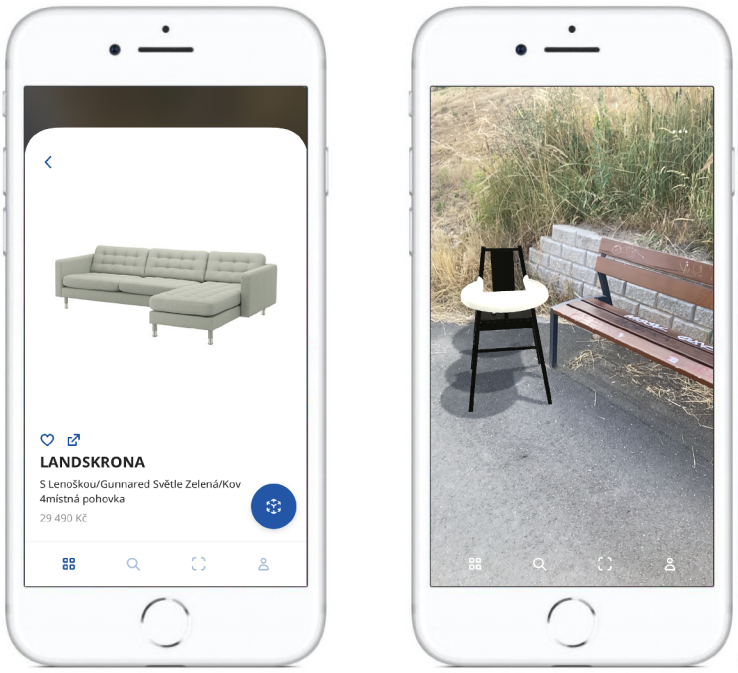
2 Rozšířená realita

Rozšířená realita je technologie, která kombinuje digitální 3D grafiku a realitu na displeji v reálném čase. [1] Dostupné mobilní telefony se stále výkonnějšími čipy, displejem, kamerou, gyroskopem a dalšími senzory přinášejí AR většímu množství uživatelů. Díky těmto technologiím jsou schopny na displeji vykreslit 3D objekty na pozadí výstupu z kamery a ukotvit je v prostředí takovým způsobem, že se zdají reálné. Takto je možné rozšířit realitu. [2]

Obrázek 1: Hra Pokémon GO na displeji Apple iPhone 8

2.1 Příklady aplikací využívajících AR

Prvním příkladem aplikace využívající AR je hra *Pokémon Go* od společnosti Niantic (viz obrázek 1). Hra umisťuje fiktivní tvory nazvané Pokémoni po celém světě. Hráč tyto tvory může vyhledávat za použití kamery a displeje mobilního telefonu. Po několik týdnů od vydání se veřejná média zabývala hlavně dvěma tématy kolem Pokémon Go. První téma se věnovalo tomu, jak je tento typ hry prospěšný pro člověka tím, že i fyzicky neaktivní hráče motivoval k pohybu, aby našel nové Pokémony a získal ve hře body. Druhé téma upozorňovalo na nutnost neustálého sledování displeje mobilního telefonu místo toho, aby se uživatel věnoval bezpečnosti pohybu po okolí. Vývojáři na tuto výtku reagovali a implementovali do hry sledování pohybu na pozadí a notifikace na nové Pokémony.[3] V roce 2016 se hra dostala na první místo v žebříčku měsíčních aktivních uživatelů po celém světě. [4]



Obrázek 2: Aplikace IKEA Place na displeji Apple iPhone 8

Druhým příkladem je aplikace společnosti IKEA, *IKEA Place* (viz obrázek 2). Ta umožňuje svým uživatelům virtuálně umisťovat nábytek z obchodního domu do svého pokoje a vidět ho díky AR ještě před nákupem.

2.2 Budoucnost využití AR

Očekává se, že technologie AR způsobí revoluci ve způsobu interakce mezi digitálním světem a člověkem. Může být použita lékaři k provádění operací na dálku nebo k výuce. Využít ji mohou architekti k předvedení svého díla na místě, kde se teprve bude stavět. Další využití je možné v kanceláři, a to při rozšíření pracovní plochy, která se teď omezuje pouze na monitor. Díky AR ji lze rozšířit na celou pracovní plochu stolu a okolí. Může pomoci také lidem se zhoršeným zrakem. Zařízení jim přečte text a promítne ho zvětšený, přeložený nebo jakkoliv upravený, aby to uživateli pomohlo. Ve světě, kde má každý uživatel celý den nasazené brýle pro AR, se mohou reklamní plochy přizpůsobovat na míru každému uživateli nebo může existovat muzeum, ve kterém bude každý den díky digitální vrstvě jiná expozice. [5]

3 Apple iPhone

Jedním ze zařízení, které v současné době dokáže zobrazit AR, je telefon Apple iPhone. V roce svého představení (2007) ho nazval časopis TIME vynálezem roku.

Telefon byl podle redaktora to nejlepší, co bylo vynalezeno v roce 2007 z pěti důvodů: byl pěkný, příjemný na dotek, díky němu jsou ostatní telefony lepší, byla to nová platforma a byl to začátek dalších generací iPhonů. Do té doby byly smartphony přizpůsobeny pro ovládání stylusem a ekosystém byl podřízený podnikové sféře. [6]

Telefon v roce 2008 rozšířil své funkce o 3G konektivitu, možnost instalovat aplikace třetích stran a přejmenoval se na iPhone 3G. V roce 2010 byl v nové generaci iPhone 4 představen retina displej s vysokým rozlišením. Šestá generace iPhone přišla v roce 2014 s možností placení pomocí bezkontaktních tokenizovaných platebních karet Apple Pay a větším displejem. Prvním iPhonem s displejem přes celou přední stranu zařízení byl iPhone X v roce 2017. Přišel tak o tlačítko Home a odemykání otiskem prstu bylo nahrazeno skenováním obličeje Face ID.

Apple iPhone dokáže zobrazit AR v zařízeních s čipem Apple A9 a vyšším. Phil Schiller, senior viceprezident společnosti Apple pro celosvětový marketing, uvedl při představení iPhone 8 v září 2017, že je to první chytrý telefon navržený pro rozšířenou realitu.

3.1 Operační systém iOS

Uvnitř iPhone běží operační systém iOS, který se během vývoje přejmenoval. Do verze 4 se jmenoval iPhone OS a v první verzi nabízel jen multitouch gesta, základní aplikace a uživatelské rozhraní. Možnost nahrávat aplikace třetích stran díky SDK a digitálnímu obchodu s aplikacemi AppStore se objevila v iPhone OS 2 na telefonu iPhone 3G.

Další verze operačního systému přinášely do telefonu nové funkce. Například multitasking, videohovory, notifikace, možnosti pro hlídání zdraví a soustředění.

iOS se během svého vývoje rozšířil i na zařízení s větším displejem. Apple ho až do roku 2019 používal v zařízeních iPad. Po tomto roce jej Apple oddělil a operační systém pro iPad nazval iPadOS.

4 Programovací jazyky

4.1 Objective-C

Objective-C je vysokoúrovňový univerzální objektově orientovaný programovací jazyk, který doplňuje programovací jazyk C o přenos zpráv ve stylu Smalltalku. Má syntaxi pro definování tříd a metod a podporuje dynamické běhové prostředí. Původně jej vyvinuli Brad Cox a Tom Love na počátku 80. let 20. století a společnost NeXT si jej vybrala pro svůj operační systém NeXTSTEP. Společnost Apple používá jazyk Objective-C od doby krátce po založení firmy NeXT. K vývoji aplikací pro iPhone se do nedávna používal výhradně Objective-C. Postupem času se ale moderní jazyky naučily nové vlastnosti, které bylo obtížné přidat do jazyka Objective-C. Jako příklad lze zmínit generické funkce společné

pro celá čísla a hodnoty s plovoucí desetinnou čárkou. Jazyk sledoval objekty pomocí ukazatelů stejně jako jazyk C a to mohlo způsobovat problémy při vývoji složitých aplikací. [7] [8]

4.2 Swift

V roce 2014 uvedl Apple nový programovací jazyk Swift, který nabídl alternativu vývoje pro iOS a macOS. Swift je objektově orientovaný, open source, multi-paradigmatický, kompilovaný a imperativní programovací jazyk. Oproti Objective-C zahrnuje funkční programování. Swift nepoužívá ukazatele a má jednodušší syntaxi, takže pomáhá vývojářům k menšímu počtu chyb. Inspiruje se v nových programovacích jazycích jako například C#, Java, JavaScript, PHP, Python, Ruby a Scala. Swift lze použít společně s jazyky C, Objective-C a Objective-C++. [9] Řetězce ve Swiftu nejsou pouhá pole znaků, ale základní součást jazyka. Více se podobají objektům se svými vlastnostmi a metodami. Spousta věcí ve Swiftu je volitelná, například není potřeba psát středníky na konci řádků, i když je to možné, proměnné jsou deklarovány s typem, nebo bez něj s přiřazenou hodnotou, podle které si Swift sám zjistí o jaký typ proměnné jde (viz zdrojový kód 1).

```
1 // proměnná s~deklarovaným typem
2 let mojePromenna: String = 1
3
4 // proměnná bez deklarovaného typu
5 let mojePromenna = 1
```

Zdrojový kód 1: Proměnná v jazyce Swift

Je možné odstranit závorky kolem logických vyhodnocení, jako je příkaz `if` (viz zdrojový kód 2).

```
1 if tvrzeni {
2    vysledek = 1
3 } else {
4    vysledek = 0
5 }
```

Zdrojový kód 2: Příkaz IF bez závorek kolem logických vyhodnocení.

Swift má dva nové operátory pro porovnávání objektů. Logický operátor, který zjistí, jestli jsou objekty ekvivalentní, například jestli pole obsahuje stejné znaky a logický operátor, který zjistí, jestli se jedná o stejný objekt na stejném místě v paměti (viz zdrojový kód 3).

```
1 // logický operátor zjistí, zda jsou dva objekty ekvivalentní
2 objekt1 == objekt2
3
4 // logický operátor zjistí, zda jsou dva objekty na stejném místě v~
     paměti
5 objekt1 === objekt2
```

Zdrojový kód 3: Porovnání objektů

K dispozici jsou také operátory rozsahu (viz zdrojový kód 4). Příkaz `0...50` poskytne programu každé celé číslo v intervalu <0, 50>, což je užitečné při psaní jednoduché smyčky.

```
1 for cislo in 0...5 {
2   print(cislo)
3 }
```

Zdrojový kód 4: Operátor rozsahu

Při definování výčtových typů (viz zdrojový kód 5) stačí přidat index k první položce a následující indexy přidá Swift automaticky.

```
1 enum cisla {
2   case jedna = 1, dva, tri, ctyri, pet
3 }
```

Zdrojový kód 5: Výčtové typy

V definici funkce je možné nastavit výchozí hodnoty a dávat argumentům různé interní a externí názvy (viz zdrojový kód 6). Je také možné externí názvy zcela odstranit.

Při použití příkazů `switch/case` se nemusí jednotlivé bloky přerušovat, aby se neprovedlo více příkazů `case`. V jazyce Swift je potřeba explicitně přikázat provedení dalších případů `case` pomocí klíčového slova `fallthrough`. Tento mechanismus umožňuje vytvářet komplexní programy bez nutnosti implementace zbytečné smyčky a podmínky.

Struktury v jazyce Swift jsou podobné třídám, ale nemají dědičnost, deinicializátory a počítání referencí jako třídy. Třídy ve Swiftu se chovají podobně jako třídy v Objective-C a předávají se jako reference.

Soubory Objective-C a Swift je možné používat společně v jednom projektu bez ohledu na to, v jakém jazyce byl projekt původně vytvořen. [10]

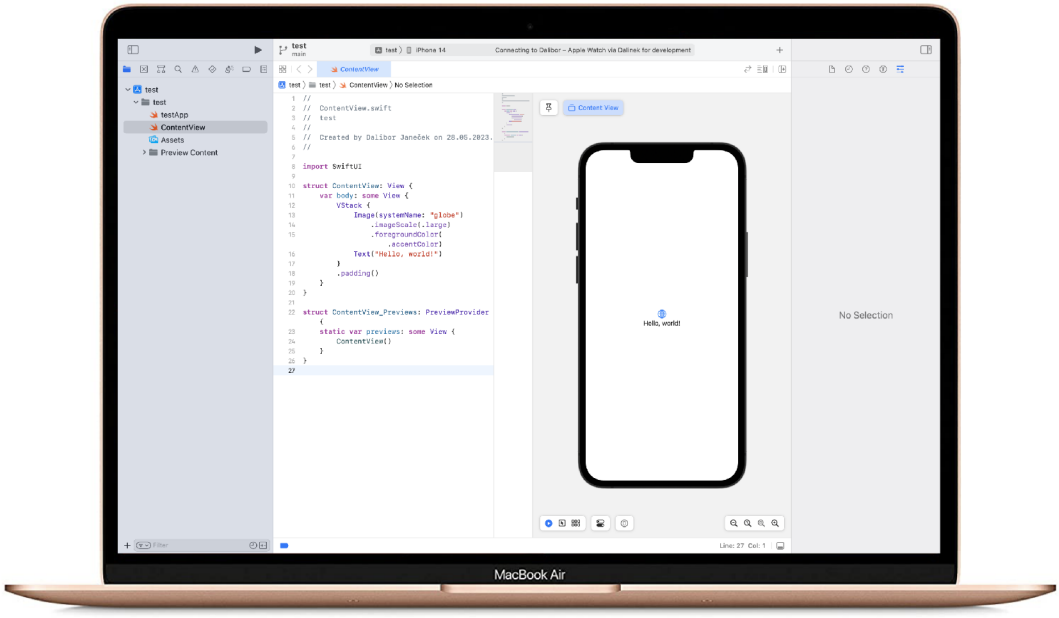
```swift
// v těle funkce se pracuje s interními názvy
func mojeFunkceA(externiNazev interniNazev: Int = 1) {
    let a = interniNazev
}
mojeFunkceA(externiNazev: 1)

// v těle funkce se pracuje s interními názvy, externí název je odstraněný
func mojeFunkceB(_ interniNazev: Int) {
    let b = interniNazev
}
mojeFunkceB(1)
```

Zdrojový kód 6: Funkce

4.3 Xcode

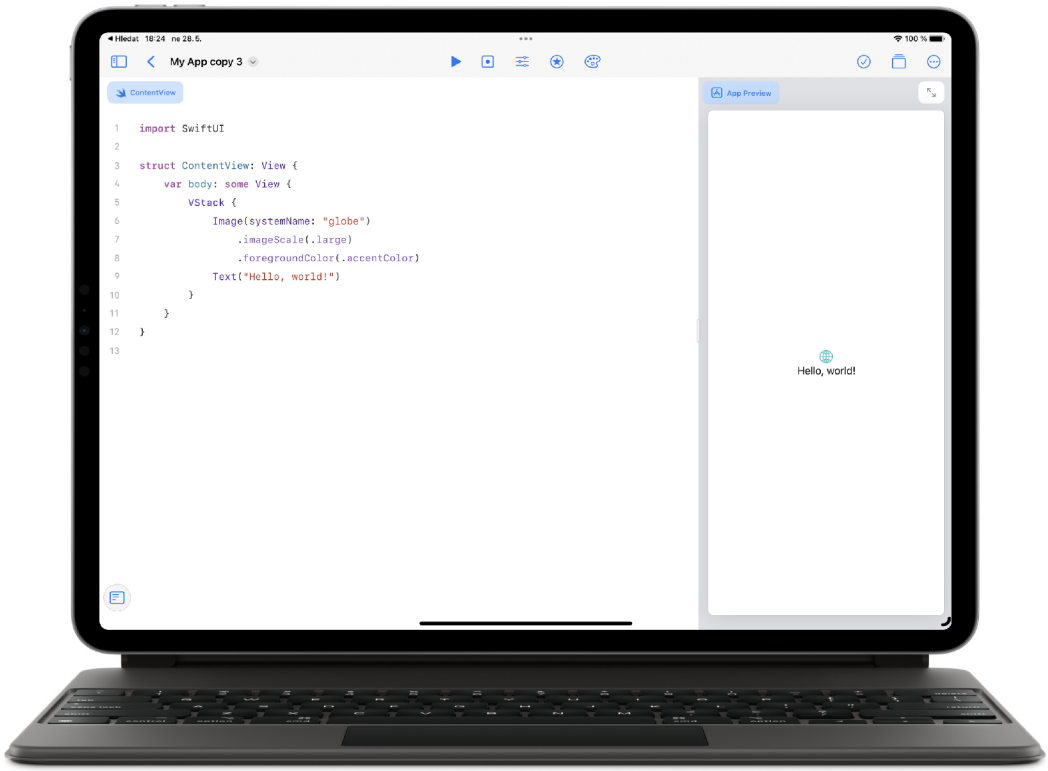
Pro vývoj aplikací pro iOS, iPadOS, macOS, watchOS i tvOS se používá vývojové prostředí Xcode. Xcode je dostupný pouze pro macOS a podporuje programovací jazyky C, C++, Objective-C, Objective-C++, Java, AppleScript, Python, Ruby, Rez a Swift. V Xcode lze aplikace spouštět v simulátorech všech aktuálně podporovaných zařízení a při psaní kódu ve SwiftUI si nechat zobrazovat náhled aplikace ve vedlejším okně (viz obrázek 3).[10]



Obrázek 3: Vývojové prostředí Xcode 14 s náhledem aplikace Hello World ve SwiftUI

4.4 iOS Swift Playgrounds

V roce 2014 přenesl Apple programování iOS aplikací i na Apple iPad. Vytvořil aplikaci Swift Playgrounds (viz obrázek 4), která poskytuje vývojářům prostředí, ve kterém mohou zkoumat možnosti jazyka Swift, učit se kódovat a vytvářet jednoduché aplikace a odesílat je do AppStore. Swift Playgrounds nabízí kurzy pro začátečníky, v nichž uživatel řeší zadané úkoly pomocí kódu.[10]



Obrázek 4: Aplikace Swift Playgrounds na iPad Pro

5 Uživatelské rozhraní

5.1 UIKit

S vývojem mobilních aplikací se neustále vyvíjí i kvalita nástrojů nabízených vývojářům aplikací, kteří mají na výběr z různých integrovaných vývojových prostředí. Také prostředí Xcode vyvinuté společností Apple je obohacováno o nové doplňky. Xcode se neustále vyvíjí a pracuje se na odstraňování nedostatků. Původní jazyk pro psaní nativních aplikací Objective-C byl z velké části nahrazen novějším jazykem Swift.

Při vývoji aplikací pro iOS byl dříve standardem vývojový framework UIKit. Tento framework nabízí mnoho předdefinovaných uživatelských prvků, jako jsou

tlačítka, seznamy, pole pro vstup dat, tabulky a další, které usnadňují vývoj aplikací. Je také poměrně snadný k použití a lze se jej rychle naučit. V porovnání se SwiftUI je ale pomalejší, hlavně pokud jde o generování zobrazení pro základní aplikace.[11]

5.2 SwiftUI

Nedávnou významnou událostí v souvislosti s programováním pro iOS je představení nového frameworku pro návrh uživatelského rozhraní s názvem SwiftUI. Společnost Apple ho propaguje jako moderní nástroj pro vývoj aplikací. Je nástupcem dříve používaného frameworku UIKit. SwiftUI používá deklarativní syntaxi, takže je možné jednoduše pomocí kódu uvést, co má uživatelské rozhraní dělat. V programu Xcode lze také vybírat jednotlivé prvky uživatelského rozhraní myší a pomocí nabídek je měnit. Zároveň se při změnách v uživatelském rozhraní synchronizuje kód ve vedlejším okně. [11]

Kompilátor a běhové prostředí Swiftu jsou integrovány do programu Xcode. Při každé změně kódu je Aplikace napsaná ve SwiftUI na pozadí kompilována a zobrazována v náhledu. Díky tomu je návrhové plátno, které je vedle editoru kódu, přímo živá aplikace. [12] Pro potřeby ladění aplikace ve tmavém režimu, v dalších lokalizacích, s jinou velikostí písma nebo na různých zařízeních je možné si vytvořit jeden nebo více náhledů v návrhovém plátně současně.

Ve SwiftUI je hlavní prvek označovaný jako `View` (viz zdrojový kód 7). Je to typ, který představuje jakoukoliv část uživatelského rozhraní aplikace. Vlastní část lze vytvořit novou strukturou, která je typu `View`. Obsahuje vlastnost `body`, která poskytuje obsah zobrazení. Vlastní část uživatelského rozhraní se sestaví s použitím vestavěných prvků, jako jsou `VStack`, `Hstack` a `ZStack` pro uspořádání prvků, `Text`, `Image`, `Color` pro vlastní obsah a další. `View` poskytuje také modifikátory, které lze použít ke konfiguraci zobrazení.

```
1 struct MyView: View {
2    var body: some View {
3       Text("Hello, World!")
4    }
5 }
```

Zdrojový kód 7: SwiftUI View

Pro ukládání stavů částí uživatelského rozhraní a jejich změnu se ve SwiftUI používají obaly vlastností `@State`. Každý `View` má svoje vlastnosti a při jejich změně se překreslí (viz zdrojový kód 8).

Tyto vlastnosti lze použít ve vnořených prvcích `View` pomocí obalu vlastnosti `Binding` ve vnořeném prvku.

Jestliže je potřeba uchovávat stav ve složitějších typech dat, jako jsou třídy, lze použít obal vlastnosti `@StateObject`. Objektem této třídy musí být potomek třídy `ObservableObject`. Vlastnosti objektu, se kterými může prvek

View pracovat, musí být v obalu @Published (viz zdrojový kód 9). Pomocí modifikátoru environmentObject lze potom tento objekt sdílet s celou hierarchií prvků View. [12] [13]

```
struct PlayButton: View {
    @State private var isPlaying: Bool = false // Vytvoření
        vlastnosti

    var body: some View {
        Button(isPlaying ? "Pause" : "Play") { // Čtení vlastnosti
            isPlaying.toggle() // Změna vlastnosti
        }
    }
}
```

Zdrojový kód 8: SwiftUI View s vlastností obalenou @State

```
class DataModel: ObservableObject {
    @Published var name = "Some Name"
    @Published var isEnabled = false
}

struct MyView: View {
    @StateObject private var model = DataModel() // Vytvoření objektu

    var body: some View {
        Text(model.name) // Čtení vlastnosti objektu
        MySubView()
            .environmentObject(model)
    }
}
```

Zdrojový kód 9: SwiftUI View se sdílenou instancí třídy ObservableObject

5.2.1 UIViewRepresentable

UIViewRepresentable je protokol frameworku SwiftUI, který umožňuje propojit prvky uživatelského rozhraní z UIKit (např. UIView) s prvky ve SwiftUI. To umožňuje využívat v aplikaci jak prvky z UIKit, tak prvky ze SwiftUI. Tento protokol je užitečný pro vývojáře, kteří chtějí postupně přepsat své aplikace z UIKit na SwiftUI, nebo pro ty, kteří chtějí využívat některé funkce z UIKit, které nejsou v SwiftUI zatím k dispozici. Struktury podřízené protokolu UIViewRepresentable obsahují dvě metody: makeUIView, která vytváří pohled v UIKit, a updateUIView, která aktualizuje pohled v UIKit na základě dat ze SwiftUI. Pokud je potřeba přenést data opačným směrem z UIKit do SwiftUI, je k tomu použitá metoda makeCoordinator. [13]

5.2.2 MVVM - Návrhový vzor Model-View-ViewModel

Návrhové vzory jsou oblíbené, protože poskytují strategie pro řešení problémů. V 80. letech byl ve Smalltalku zaveden návrhový vzor MVVM, který má překonat omezení MVC a využít jeho silné stránky. MVVM odděluje zájmy více než MVC, což snižuje složitost a maximalizuje testovatelnost a znovupoužitelnost. Studie ukazuje, že dělá kód kompaktnější a činí jej flexibilním.

Existují různé verze návrhových vzorů MVVM, které se používají v různých programovacích komunitách. V případě komunity iOS je běžné, že vývojáři využívají implementaci MVVM od společnosti Microsoft.

Záměrem MVVM je oddělit logické zájmy aplikace od zájmů uživatelského rozhraní rozdělením odlišných odpovědností mezi tři objekty: objekty View pro vykreslování uživatelského rozhraní, objekty ViewModel přebírající odpovědnost za interakce s uživatelem a stav zobrazení a objekty Model pro zpracování dat aplikace.

Objekty ViewModel vlastní objekty Model a mají k nim přímý přístup. Data binding zlepšuje logické oddělení mezi objekty View a objekty ViewModel. Objekty ViewModel shromažďují data z objektů Model, připravují data pro prezentaci, ověřují vstupy a drží síťovou logiku. [14]

6 ARKit

Existuje mnoho knihoven pro vytváření AR aplikací. Mají různé funkcionality, používají různé technologie detekce a sledování objektů, bodů nebo prvků ve scéně. Jejich cíl je však společný, a to propojit digitální obsah s reálným světem. Některé z nich jsou multiplatformní a je tedy možné napsat jednu aplikaci pro iOS, Android a další zařízení. Tato vývojová prostředí mají své výhody i nevýhody, ale všechna vyžadují určitou vrstvu abstrakce nad nativním řešením dané platformy.

Vrstva abstrakce může způsobit to, že aplikace nebude působit nativně a bude mást uživatele zvyklé na svoji vybranou platformu. Je potřeba také hlídat, které z nově uvedených funkcionalit nativních řešení je, nebo není implementované do vrstvy abstrakce. V některých případech může tato vrstva způsobit pád aplikace po vydání nové verze nativního řešení. Výhodou multiplatformního řešení je, že aplikaci je možné použít na všech platformách. [15]

Nativní řešení od společnosti Apple, poprvé oznámené na vývojářské konferenci WWDC17, se jmenuje ARKit. Od té doby technologie ARKit pokračuje v pokroku, je stále přesnější ve sledování světa a přibývají do ní nové funkce. ARKit umožňuje sledování vertikální a horizontální roviny, rozpoznávání obrazu, detekce objektů a obličejů, podporuje formát souborů USDZ, sledování pohybu, zeměpisné polohy a další.

Pro nativní řešení je výhodou také to, že technologie společnosti Apple jsou dobře zdokumentovány. Společnost Apple poskytuje archiv dokumentace pro vývojáře, který obsahuje popis rozhraní API, článků a ukázkových kódů. [5]

ARKit spolupracuje s těmito nativními grafickými frameworky, které je možné použít k vytvoření 3D zážitku v AR:

- SpriteKit
- SceneKit
- Metal
- RealityKit

6.1 SpriteKit

SpriteKit je grafický framework pro vytváření 2D her. Díky podpoře vlastních shaderů a osvětlení OpenGL ES, integraci se SceneKitem a novým pokročilým fyzikálním efektům a animacím je možné do her přidávat silová pole, detekovat kolize a generovat nové světelné efekty. Pokud jde o AR, SpriteKit je možné promítnout na plochu ve 3D prostoru. [10]

6.2 SceneKit

SceneKit je 3D grafický framework vyšší úrovně. Zvládne animace, fyzikální simulace, částicové efekty, realistické vykreslování na základě fyzikálních zákonů a další. SceneKit je cestou pro aplikace, které nevyžadují graficky náročný výkon. Pro začínající vývojáře je vhodné nejdříve pracovat se SceneKitem a až později se věnovat optimalizaci výkonu a frameworku nižší úrovně (Metal). [10]

6.3 Metal

Framework Metal dává aplikaci přímý přístup ke grafickému procesoru (GPU) zařízení. Díky nástroji Metal mohou aplikace využívat GPU k rychlému vykreslování složitých scén a paralelnímu spouštění výpočetních úloh. [10]

Metal je složitější k naučení. Programátor se musí starat o podrobnější konfiguraci, i když chce vytvořit něco jednoduchého. Výkon Metalu využívá mnoho vysokoúrovňových frameworků společnosti Apple, například RealityKit, SceneKit a SpriteKit. Tyto vysokoúrovňové frameworky implementují programovací detaily pro GPU za programátora. Se zavedením frameworku RealityKit a Reality Composer se drasticky snížila náročnost vytváření zážitků AR pomocí Metalu. Je to proto, že RealityKit je vytvořen pro AR s ohledem na snadné použití. [5]

6.4 RealityKit

Framework RealityKit byl vytvořen speciálně pro AR. Poskytuje fotorealistické vykreslování, kamerové efekty, animace, fyziku a další funkce. RealityKit má nativní rozhraní API Swiftu a integraci ARKitu. [10] [5] Framework nabízí spoustu funkcí pro zjednodušení vývoje aplikací s AR.

6.4.1 Zachycení objektu

Pomocí Object Capture API na macOS Monterey a Photogrammetry API je možné ze série snímků pořízených na iPhonu nebo iPadu vytvořit 3D modely, které jsou vhodné k použití v projektu s AR nebo zobrazení pomocí QuickView.

6.4.2 Shadery

Aby byl virtuální obsah téměř k nerozeznání od reality propojuje RealityKit virtuální obsah se skutečným světem pomocí realistických, fyzicky založených materiálů, odrazů prostředí, stínů, šumu kamery a rozmazání pohybu.

6.4.3 Object occlusion

Kombinací informací ze skeneru LiDAR na zařízeních iPad Pro a iPhone Pro a detekce hran v RealityKitu jsou virtuální objekty schopny interagovat s fyzickým okolím přesně tak, jak uživatel očekává. Virtuální objekty lze umístit pod stoly, za stěny nebo kolem rohů.

6.4.4 Video texture

Do jakékoliv části scény RealityKitu lze přidat textury zobrazující video.

6.4.5 Škálovatelný výkon

RealityKit využívá funkce Metalu pro maximální využití GPU a plně využívá mezipaměti CPU a více jader. RealityKit automaticky přizpůsobuje výkon AR pro každý iPhone nebo iPad.

6.4.6 Sdílené zážitky

RealityKit přebírá práci se sítí, udržuje konzistentní stav a optimalizuje síťový provoz a tím zjednodušuje vytvářet sdílené scény mezi více uživateli a jejich zařízeními.

6.5 USDZ

Universal Scene Description (USD) je softwarová platforma, která řeší potřebu robustní a škálovatelné výměny a rozšiřování libovolných 3D scén. Tyto scény mohou být složeny z mnoha elementárních prvků nazvaných „Prims“. USD obsahují sadu schémat, která pokrývají oblasti jako geometrie, stínování, osvětlení a fyzika. [16]

USD vyvinula společnost Pixar a poprvé byla zveřejněna jako open source software v roce 2016. Apple USD používá v balíčcích USDZ pro svůj framework ARKit a RealityKit.

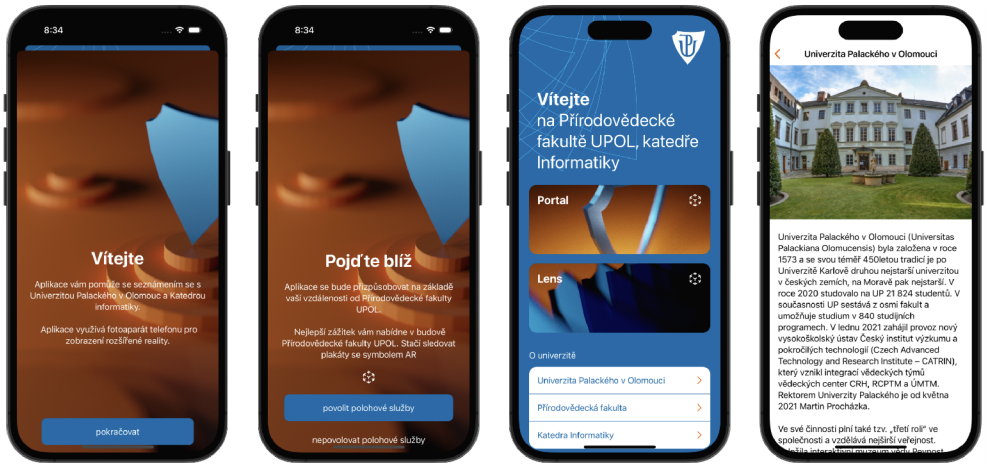
Obrázek 5: Ukázka prostředí aplikace Blender

Převod 3D objektů do formátu USDZ lze provést pomocí nástroje příkazového řádku, který volně poskytuje společnost Apple pod názvem USDZ Tools nebo grafickým nástrojem Reality Converter. Ke tvorbě originálních 3D objektů lze použít open source program Blender[1] (viz obrázek číslo 5). [17]

7 Aplikace UPOL AR

Výsledkem této práce je aplikace, která slouží k prezentaci Katedry informatiky pomocí AR, a to jak přímo v budově Přírodovědecké fakulty, tak i mimo ni. Aplikace nese název UPOL AR, který ji předurčuje k rozšíření na celou Univerzitu Palackého v Olomouci. Aplikace je naprogramovaná v jazyce Swift a uživatelské rozhraní je vytvořené pomocí frameworku SwiftUI. V aplikaci je pro zobrazení AR použitý framework RealityKit a složitější 3D modely jsou vytvořené v programu Blender s exportem do USDZ.

[1]Blender je open-source 3D grafický software, který poskytuje funkce pro modelování, animaci, tvorbu vizuálních efektů a renderování. Byl vyvinut jako komplexní nástroj pro tvorbu 3D obsahu a je dostupný zdarma.



Obrázek 6: První spuštění aplikace UPOL AR v simulátoru iOS.

7.1 Návrh uživatelského rozhraní

První návrh aplikace byl vytvořený ručními náčrty (viz obrázek 7).

Pro tvorbu přesnějších návrhů uživatelského rozhraní byla využita aplikace Figma[2] (viz obrázek 8). Dále byl v aplikaci Figma vytvořen jednoduchý prototyp pro iPhone SE. V této fázi byly rovněž vytvářeny grafické bannery v aplikaci Blender s využitím prvků z manuálu jednotného vizuálního stylu Univerzity Palackého v Olomouci. Tyto grafické bannery slouží jako základ pro tvorbu plakátů, které budou v aplikaci použity jako kotvy[3] pro AR.

7.2 Struktura aplikace

Aplikace pracuje ve dvou režimech podle vzdálenosti od budovy Přírodovědecké fakulty (viz příloha A):

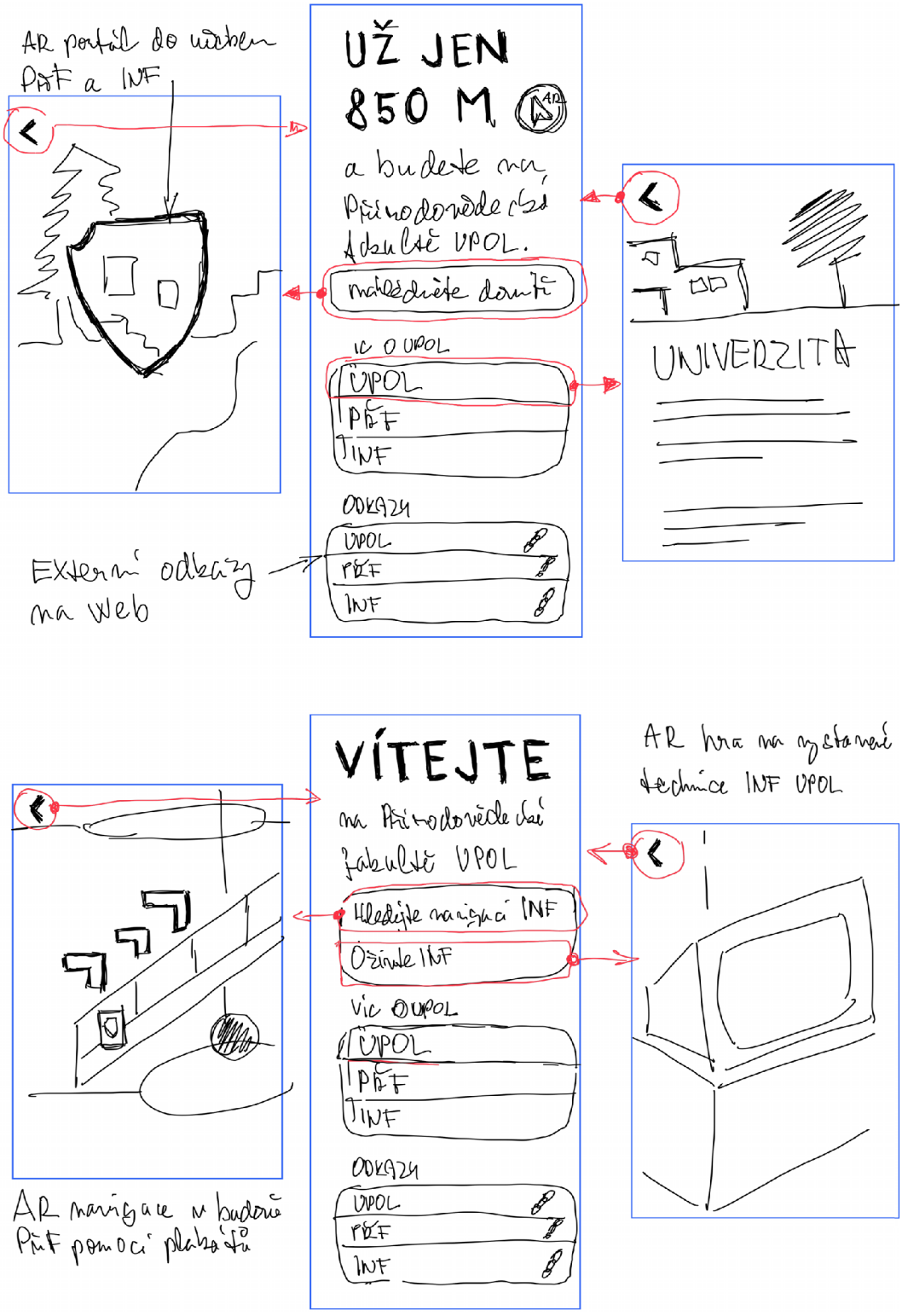
- Režim v budově Přírodovědecké fakulty, kde rozšiřuje a oživuje prostor.
- Režim mimo budovu představuje prostor Přírodovědecké fakulty a Katedru informatiky na dálku.

Při prvním spuštění je uživatel přivítán a požádán o přístup aplikace k polohovým službám telefonu (viz obrázek 6).

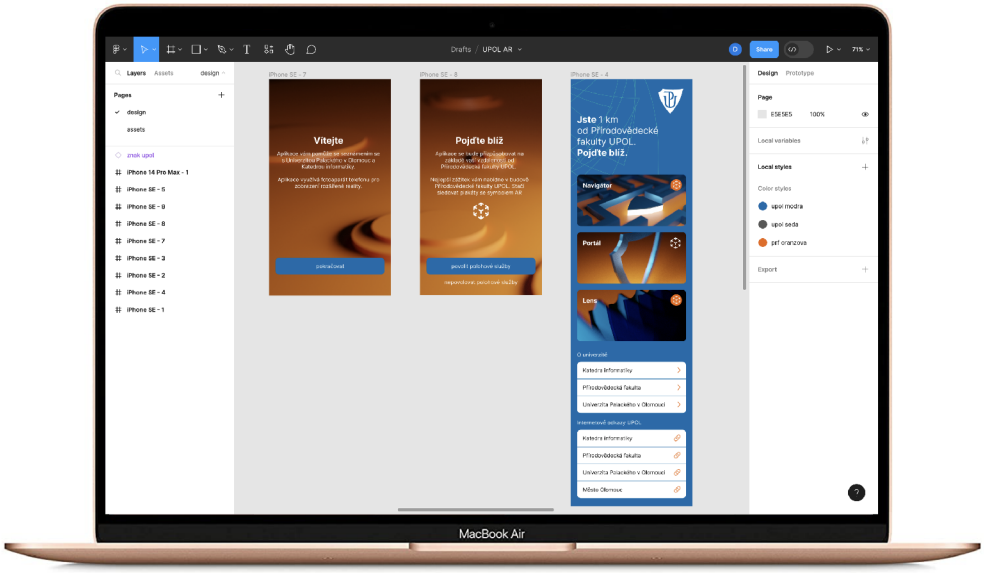
Aplikace určuje vzdálenost od budovy pomocí frameworku CoreLocation. Rozhodující vzdálenost od fakulty je nastavená tak, aby byla aplikace v režimu blízko budovy, i když se uživatel dostane s aplikací do krajních částí budovy, nebo do částí, kde není přesný GPS signál.

[2]Aplikace Figma poskytuje vývojářům prostředí, ve kterém mohou vytvářet rozhraní, návrhy a wireframy aplikací. Nabízí tvorbu a úpravu grafiky, přidávání interakcí a vytváření prototypů.

[3]Bodové reference, které umožňují přesné umístění a uchycení digitálních objektů v reálném prostoru.



Obrázek 7: Ruční návrh aplikace UPOL AR. Dva režimy aplikace.



Obrázek 8: Návrh uživatelského rozhraní v aplikaci Figma.

V režimu blízko budovy jsou dostupné tyto AR části aplikace:

- Navigator
- Computer
- Tetris
- Portal
- Lens

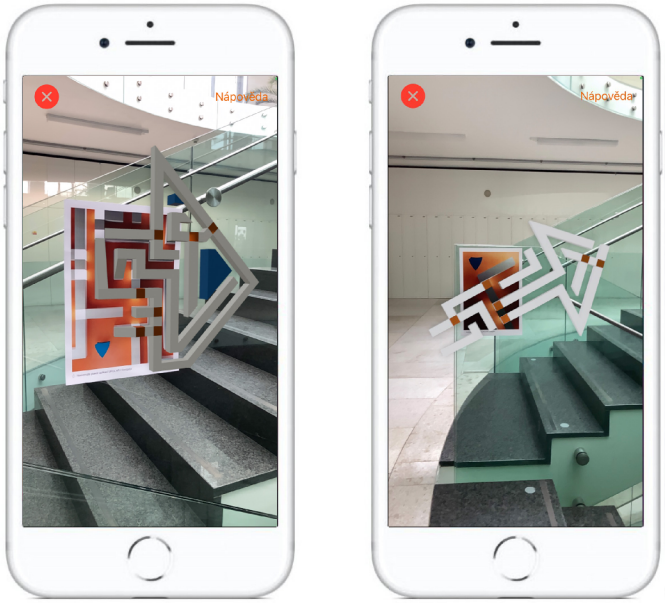
V režimu vzdáleném od budovy jsou dostupné tyto AR části aplikace:

- Portal
- Lens

V obou režimech jsou dostupné části popisující Univerzitu Palackého v Olomouci, Přírodovědeckou fakultu i Katedru informatiky. Uživatel má také možnost se v obou režimech jednoduše dostat na webové stránky univerzity i města Olomouc.

7.2.1 Navigator

Část nazvaná „Navigator“ se věnuje navigaci v budově Přírodovědecké fakulty. Pro ukotvení obsahu AR jsou zde použity plakáty s motivy bludiště. Po namíření fotoaparátu telefonu na plakát se rozšíří o 3D model, který ukazuje pomocí šipky směr ke Katedře informatiky. Díky vhodnému umístění těchto plakátů lze uživatele navigovat správným směrem (viz obrázek 9).



Obrázek 9: Zobrazení AR scény Navigátor v budově.

Scéna pro každý plakát byla vytvářena v grafickém programu Reality Composer. V této aplikaci lze pro každou scénu vytvořit obrázkovou kotvu, kterou aplikace potom vyhledává a umístí na ni 3D model. 3D modely jsou vytvořené pomocí aplikace Blender a exportem do USDZ přenesené do Reality Composeru. Scény z Reality Composeru je možné jednoduše přidávat do scény pomocí dvou řádků kódu (viz zdrojový kód 10).

```
1 let arrowAnchor = try! Navigator.loadArrowStairs()
2 arView.scene.addAnchor(arrowAnchor)
```

Zdrojový kód 10: SwiftUI View se sdílenou instancí třídy ObservableObject.

Část „Navigator“ měla být původně zpřístupněna i uživatelům, kteří se pohybují mimo budovu. V plánu bylo umisťovat v AR šipku, která by ukazovala směr, kterým se nachází budova Přírodovědecké fakulty. Při implementaci ale nastal problém aktualizace dat z kompasu telefonu. Bylo potřeba znát přesné směrování uživatele ve chvíli, kdy ARKit najde vhodnou plochu k umístění 3D modelu. Když se uživatel s telefonem nepohyboval, tak byl výsledek otočení v pořádku, ale při rychlejších pohybech nebo otáčení přístroje byla data z kompasu opožděna a neodpovídala nasměrování plochy, kterou našel ARKit. Proto se stávalo, že se model otočil špatně. Tato funkce byla z finální aplikace odstraněna. Po uživateli by totiž aplikace v jednu chvíli vyžadovala, aby pohyboval telefonem pro nalezení vhodné plochy a zároveň, aby byl telefon v klidu pro získání co nejpřesnějších dat z kompasu.

7.2.2 Computer

Část „Computer“ oživí historické počítače ve vitrínách v 5. patře před vstupem na Katedru informatiky.

V tomto případě je použita jako kotva grafika open-source filmu Big Buck Bunny [18]. Po zachycení kotvy fotoaparátem mobilního telefonu se před obrazovkou promítne 3D plocha o velikosti podkladové obrázku a na ní video textura s ukázkou z filmu. Je možné rozšířit tento model na ostatní obrazovky a demonstrovat na nich pomocí videí původní operační systémy. (viz obrázek 10).

Obrázek 10: Zobrazení AR scény Computer.

7.2.3 Tetris

Na jednom monitoru z předchozí části je místo grafiky z filmu motiv ze hry Tetris. Po namíření fotoaparátu na tento obrázek může uživatel hrát na monitoru hru na motivy hry Tetris (viz obrázek 11).

V této části jsou modely tvořené pomocí `MeshResource.generateBox()` a ty se skládají do běžných tvarů hry tetris na 2D poli. Hru uživatel spustí poklepáním na virtuální tlačítko start zobrazené na monitoru. Této interakce s virtuálním objektem je umožněno vysláním virtuálního paprsku z fotoaparátu mobilního telefonu na místo souřadnic, kterých se uživatel dotkl na displeji telefonu. Pokud se tento paprsek při své cestě střetne s 3D modelem, informuje o tom aplikaci. Ta pomocí třídy `Coordinator` vyhodnotí, jestli se jedná o model s názvem "startButton". Pokud ano, provede start hry tetris.

Diagram tříd, který modeluje strukturu objektů v části Tetris je v příloze B.

Obrázek 11: Zobrazení AR scény Tetris.

Na začátku celé této práce bylo v plánu využít fyziku a materiály v RealityKit a s jejich pomocí vytvořit hru podobnou hře Breakout z roku 1972. V této hře bylo úkolem pomocí odrážecí plošiny na jedné straně hrací plochy a míčku sestřelit všechny cihličky, které byly na druhé straně hrací plochy. Během implementační fáze se objevilo několik problémů, které bránily správnému fungování hry. Byl zaznamenán nedostatečný odraz kuličky z materiálu s názvem "guma". Ačkoli byly tření a gravitace nastaveny na hodnotu 0, kulička se nepohybovala konstantní rychlostí. Při každé kolizi se stěnami hrací plochy byla nastavena nová síla pohybu kuličky, avšak ani tato úprava nepřinesla očekávanou hratelnost. Kromě těchto problémů se při prudkých pohybech telefonem míček často dostával mimo hrací plochu a bez zaznamenání kolize procházel přes stěny. Z těchto důvodů nebyla tato hra začleněna do aplikace.

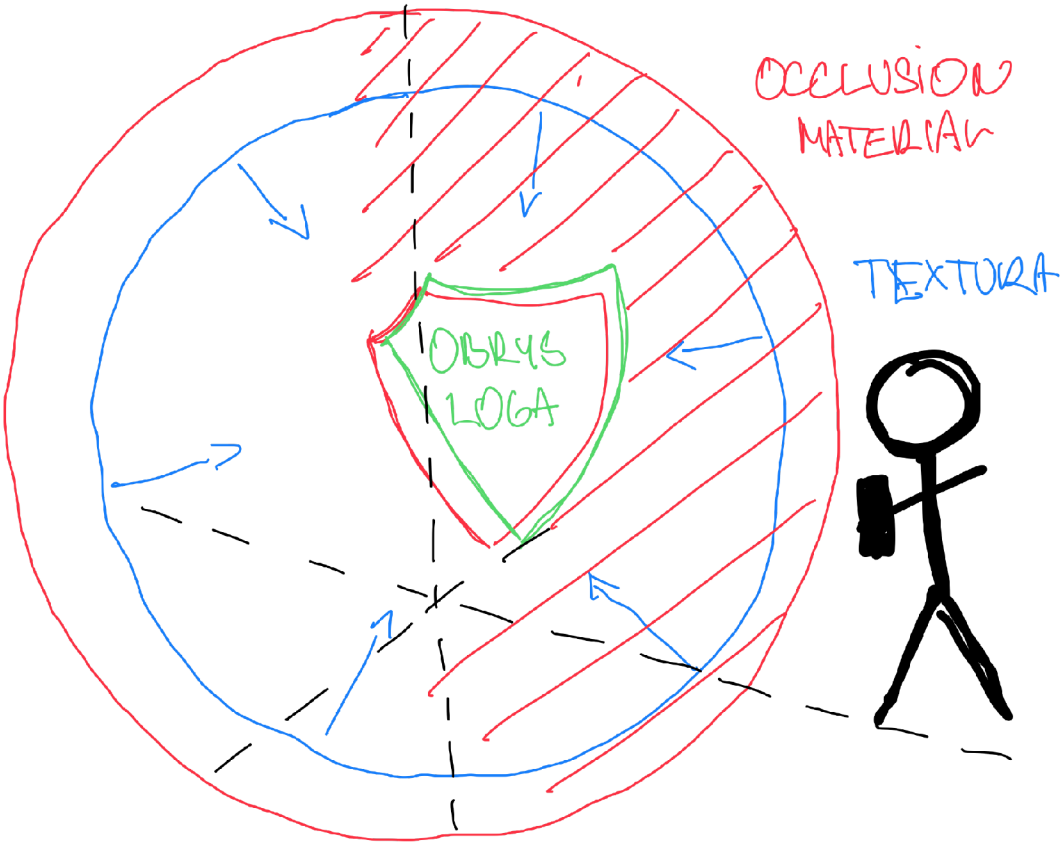
7.2.4 Portal

Část „Portal“ nevyžaduje ke svému fungování přítomnost v budově fakulty. K ukotvení AR stačí jakýkoliv prostor. V tomto prostoru se vykreslí obrys loga univerzity a v něm k nahlédnutí interiér nebo exteriér fakulty. Pomocí pohybu telefonu je možné si tento prostor prohlížet nebo do něj vejít a rozhlížet se v něm.

Tato scéna se skládá z několika USDZ modelů (viz obrázek 5 a 12). První viditelný model je obrys loga univerzity, použitý jako vstup do portálu. Dalším modelem je 3D koule s průchodem, která obsahuje materiál `OcclusionMaterial()`, tímto materiálem je model a vše co je skryté za ním neviditelné. Uvnitř koule je

další koule, která promítá svoji texturu směrem ke středu a je na ní zobrazená sférická fotografie prostoru fakulty.

Jednotlivé prostory se přepínají gestem `DragGesture` ve SwiftUI a do AR je tato hodnota předaná díky obalu vlastnosti `@Binding`.



Obrázek 12: Návrh scény Portal

7.2.5 Lens

Stejně jako předchozí „Portal“ ani „Lens“ nevyžaduje přítomnost uživatele na fakultě. Tato část jako jediná používá přední fotoaparát mobilního telefonu a promítání AR na obličej uživatele.

K ukotvení 3D modelů na obličej je použitá kotva Face v programu Reality Composer. 3D modely jsou vytvořené v aplikaci Blender a importované do Reality Composer. Načtení celé scény probíhá stejně jako v případě části „Navigator“.

Závěr

V této bakalářské práci byla vytvořena aplikace využívající technologii (AR) k prezentaci Katedry informatiky Univerzity Palackého v Olomouci. Hlavním cílem této práce bylo představit Katedru informatiky prostřednictvím interaktivních a zábavných AR prvků a umožnit uživatelům prozkoumávat prostor budovy Přírodovědecké fakulty.

Během vývoje aplikace byly popsány základní principy technologie AR a představeny možnosti jejího využití. Byla provedena analýza vhodné platformy a prostředí pro implementaci aplikace, a to včetně výběru programovacích jazyků a frameworků. Implementace aplikace zahrnovala vytvoření interaktivního uživatelského rozhraní.

V rámci aplikace bylo využito různých přístupů k AR v závislosti na poloze uživatele. Při přítomnosti uživatele v budově fakulty byla aplikace zaměřena na oživení prostoru a navigaci po budově. Pokud byl uživatel vzdálen od budovy, byla nabídnuta možnost vzdálené prohlídky budovy a interaktivního seznámení s Katedrou informatiky.

Výsledkem této práce je funkční aplikace, která umožňuje uživatelům prozkoumat prostor Katedry informatiky prostřednictvím AR technologie. Tato aplikace přináší nový způsob interakce a prezentace, který je zábavný, inovativní a přispívá k lepšímu seznámení s Katedrou informatiky.

V dalších krocích je možné dále rozšiřovat funkcionalitu aplikace na ostatní katedry, provádět uživatelské testování a evaluaci, a využít získané poznatky pro další vývoj a zdokonalování aplikace.

Věřím, že představená aplikace přinese uživatelům nový zážitek a zlepší prezentaci a seznámení s Katedrou informatiky Univerzity Palackého v Olomouci prostřednictvím technologie rozšířené reality.

Conclusions

In this bachelor thesis, an application using augmented reality (AR) technology was created to present the Department of Informatics at Palacký University in Olomouc. The main goal of this work was to present the Department of Computer Science through interactive and entertaining AR elements and to allow users to explore the space of the Faculty of Science building.

During the development of the application, the basic principles of AR technology were described and the possibilities of its use were presented. An analysis of the appropriate platform and environment for implementing the application was conducted, including the selection of programming languages and frameworks. The implementation of the application included the creation of an interactive user interface.

Different approaches to AR were used within the application depending on the location of the user. When the user was in the faculty building, the application focused on animating the space and navigating the building. When the user was away from the building, the user was offered the opportunity to remotely tour the building and interactively learn about the Department of Computer Science.

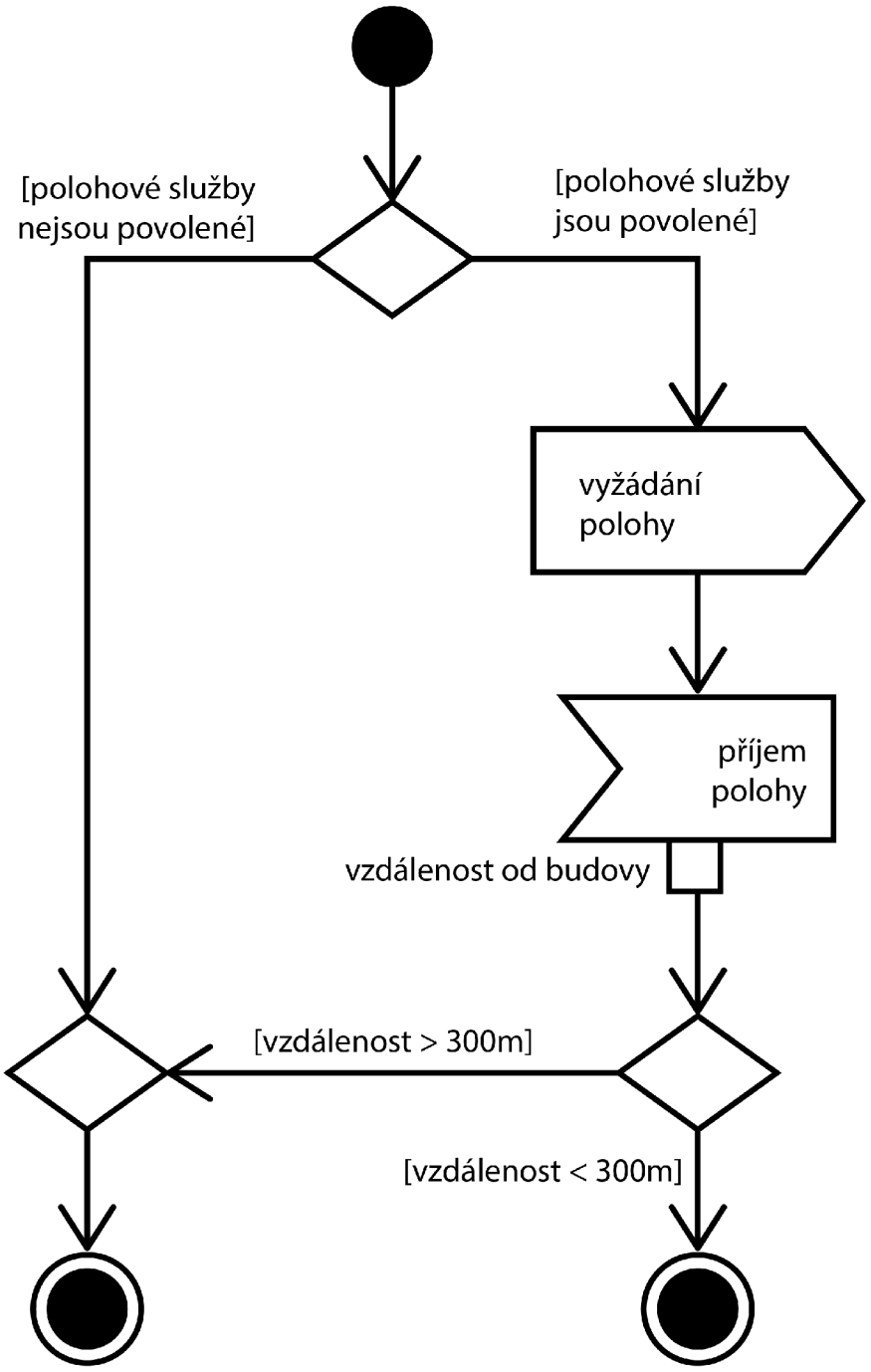
The result of this work is a functional application that allows users to explore the space of the Department of Computer Science through AR technology. This application brings a new way of interaction and presentation that is fun, innovative and contributes to a better familiarity with the Department of Computer Science.

The next steps are to further extend the functionality of the app to other departments, conduct user testing and evaluation, and use the findings to further develop and improve the app.

I believe that the presented application will bring a new experience to the users and improve the presentation and familiarization with the Department of Informatics of Palacký University in Olomouc through augmented reality technology.

A Diagram aktivit

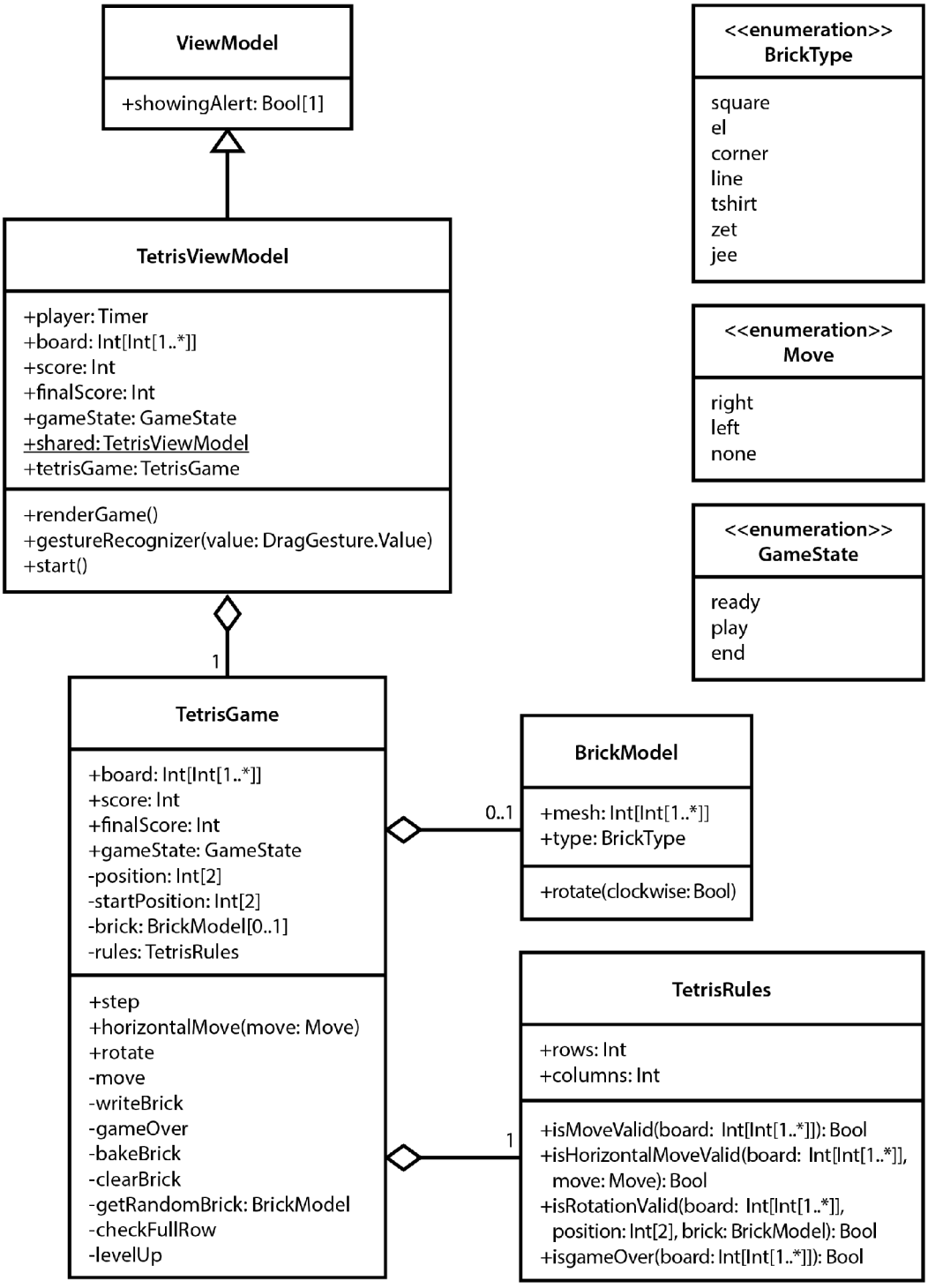
Diagram aktivit znázorňuje průběh získání a zpracovaní vzdálenosti od budovy Přírodovědecké fakulty Univerzity Palackého v Olomouci.



Obrázek 13: Diagram aktivit

B Diagram tříd části Tetris

Diagram tříd modeluje strukturu objektů v části Tetris.



Obrázek 14: Diagram tříd

C Obsah elektronických dat

`text/`
: Adresář s textem práce ve formátu PDF, vytvořený s použitím závazného stylu KI PřF UP v Olomouci pro závěrečné práce, včetně všech (textových) příloh, a všechny soubory potřebné pro bezproblémové vytvoření PDF dokumentu textu (případně v ZIP archivu), tj. zdrojový text textu a příloh, vložené obrázky, apod.

`README.txt`
: Textový soubor s informacemi o postupu zprovoznění software vytvořeného v rámci práce, tzn. jeho instalace a spuštění, včetně uvedení všech požadavků pro bezproblémový provoz.

`*`
: Adresáře a soubory s veškerými ostatními autorskými daty práce (případně v ZIP archivu) – typicky spustitelné a další soubory software vytvořeného v rámci práce potřebné pro bezproblémový provoz software, případně jeho instalační program, a kompletní zdrojové texty software a další data nutná pro plně reprodukovatelné korektní vytvoření spustitelných souborů.

`anchors/`
: Adresář s grafickými kotvami potřebnými pro zprovoznění AR.

`UpolAR/`
: Zdrojový kód aplikace pro program Xcode.

U veškerých cizích obsažených materiálů jejich zahrnutí dovolují podmínky pro jejich veřejné šíření nebo přiložený souhlas držitele práv k užití. Pro všechny použité (a citované) materiály, u kterých toto není splněno a nejsou tak obsaženy, je uveden jejich zdroj, např. webová adresa, v bibliografii nebo textu práce nebo souboru `README.txt`.

Literatura

[1] Chen, Yunqiang; Wang, Qing; Chen, Hong aj. An overview of augmented reality technology. *Journal of Physics: Conference Series*. 2019, roč. 1237, č. 2, s. 022082. Dostupný také z: ⟨https://dx.doi.org/10.1088/1742-6596/1237/2/022082⟩.

[2] Xiong, Jianghao; Hsiang, En-Lin; He, Ziqian; Zhan, Tao; Wu, Shin-Tson. Augmented reality and virtual reality displays: emerging technologies and future perspectives. *Light: Science & Applications*. 2021, roč. 10, č. 1, s. 216.

[3] Baranowski, Tom. Pokémon Go, go, go, gone? *Games for health journal*. 2016, roč. ePub, č. ePub, s. ePub–ePub. Dostupný také z: ⟨http://dx.doi.org/10.1089/g4h.2016.01055.tbp⟩. ISSN 2161-783X.

[4] Hindy, Joe. *2016 recap: 90% of Google Play's revenue came from games (and more fun stats!)* [online]. 2017 [cit. 2023-5-10]. Dostupný z: ⟨https://www.androidauthority.com/2016-recap-90-percent-google-play-revenue-gaming-fun-stats-743626/⟩.

[5] Nhan, Jayven. Why Augmented Reality? In. *Mastering ARKit: Apple's Augmented Reality App Development Platform*. 2022, s. 1–15.

[6] Grossman, Lev. *Invention Of the Year: The iPhone* [online]. 2007 [cit. 2023-5-6]. Dostupný z: ⟨https://content.time.com/time/specials/2007/article/0,28804,1677329_1678542_1677891,00.html⟩.

[7] Garcıa, Cristian González; Espada, Jordán Pascual; Bustelo, Begoña Cristina Pelayo Garcıa; Lovelle, Juan Manuel Cueva. Swift vs. objective-c: A new programming language. *IJIMAI*. 2015, roč. 3, č. 3, s. 74–81.

[8] Timmer, John. A fast look at Swift, Apple's new programming language. *Ars Technica*. 2014, roč. 5, č. 6.

[9] Lattner, Chris. Swift (programovacı jazyk). Dostupný také z: ⟨https://czwiki.cz/Lexikon/Swift_(programovac%C3%AD_jazyk)⟩.

[10] *Apple Developer*. [online]. [cit. 2023-5-14]. Dostupný z: ⟨https://developer.apple.com/⟩.

[11] Wiertel, Piotr; Skublewska-Paszkowska, Maria. Comparative analysis of UIKit and SwiftUI frameworks in iOS system. *Journal of Computer Sciences Institute*. 2021, roč. 20, s. 170–174. Dostupný také z: ⟨https://ph.pollub.pl/index.php/jcsi/article/view/2662⟩.

[12] Varma, Jayant. *SwiftUI for Absolute Beginners*. 2019.

[13] Bello, A.; Laszkowicz, P.; Morefield, B. *SwiftUI by Tutorials (First Edition): Declarative App Development on the Apple Ecosystem*. 2019. Dostupný také z: ⟨https://books.google.cz/books?id=wxFHzAEACAAJ⟩. ISBN 9781942878834.

[14] Aljamea, Mariam; Alkandari, Mohammad. MMVMi: A validation model for MVC and MVVM design patterns in iOS applications. *IAENG Int. J. Comput. Sci.* 2018, roč. 45, č. 3, s. 377–389.

[15] Oufqir, Zainab; El Abderrahmani, Abdellatif; Satori, Khalid. ARKit and ARCore in serve to augmented reality. In. *2020 International Conference on Intelligent Systems and Computer Vision (ISCV)*. 2020, s. 1–7. Dostupný také z: ⟨http://dx.doi.org/10.1109/ISCV49265.2020.9204243⟩.

[16] *USD Home - Universal Scene Description 23.05 documentation*. Dostupný z: ⟨https://openusd.org/release/index.html⟩.

[17] Sanii, Babak. Creating Augmented Reality USDZ Files to Visualize 3D Objects on Student Phones in the Classroom. *Journal of Chemical Education*. 2020, roč. 97, č. 1, s. 253–257. Dostupný také z: ⟨https://doi.org/10.1021/acs.jchemed.9b00577⟩. ISSN 0021-9584.

[18] *Big Buck Bunny*. Dostupný z: ⟨https://peach.blender.org⟩.