УДК 004.415

ФРЕЙМВОРК ДЛЯ РАЗРАБОТКИ НАТИВНЫХ МОБИЛЬНЫХ ПРИЛОЖЕНИЙ С ДОПОЛНЕННОЙ РЕАЛЬНОСТЬЮ

Дмитрий Евдокименко[1], Ринат Ханов[2]

Высшая школа информационных технологий и интеллектуальных систем Казанского (Приволжского) федерального университета

[1] evdodima@gmail.com, [2]rinat@khanov.com

Аннотация

Представлен фреймворк для разработки нативных мобильных приложений с дополненной реальностью. В частности, проведен анализ фреймворков из различных сфер разработки – игр, мобильных приложений. Предложена концепция нового фреймворка и набора инструментов для построения мобильных приложений с дополненной реальностью. Предложены способы внедрения данного фреймворка.

Ключевые слова*: дополненная реальность, разработка, фреймворк, augmented reality, AR, ARKit, iOS, development, framework.*

ВВЕДЕНИЕ

Несмотря на большой интерес со стороны крупных компаний, таких, как Apple и Facebook [1], а также быстрый рост числа мобильных приложений, использующих дополненную реальность в последние годы [2], технология дополненной реальности на мобильных устройствах в большинстве случаев остается недостаточно востребованной и распространенной среди обычных пользователей [3, 4]. Одной из причин этого является трудозатратность развития и поддержки нативных мобильных приложений с дополненной реальностью ввиду отсутствия паттернов проектирования, учитывающих особенности этой предметной области.

На данный момент времени приложения с дополненной реальностью для iOS реализуются на основе фреймворка MVC [5], который предназначен для работы с пользовательским интерфейсом приложения и его данными, а не 3D-объ-

ектами и окружением пользователя. Использование фреймворка MVC для разработки приложений с дополненной реальностью вызывает множество неудобств, а также трудности с дальнейшей поддержкой приложения и добавлением нового функционала.

В данной статье предложена концепция фреймворка, который бы позволял максимально эффективно решать задачи разработки мобильных приложений с дополненной реальностью для операционной системы iOS, а также обеспечивал простоту поддержки таких приложений и внесения в них изменений.

НЕДОСТАТКИ ТЕКУЩЕЙ АРХИТЕКТУРЫ

Разбор архитектур существующих AR-приложений

Для более подробного рассмотрения недостатков текущей архитектуры возьмем несколько существующих мобильных приложений с дополненной реальностью с открытым кодом и примеры демо приложений от компании Apple:

- MeasureKit – приложение для проведения измерений с помощью AR;
- SwiftStrike – демо приложение от Apple; игра в дополненной реальности [6];
- CoreML-in-ARKit – приложение с открытым кодом для детекции объектов реального мира [7].

Все перечисленные приложения используют архитектуру MVC. Данная архитектура не учитывает специфику AR: наличие объектов реального мира, разделение объектов на реальные и виртуальные, наличие только одного View. Как следствие, ненадобность сущности ViewController.

Далее на примере указанных приложений будут рассмотрены нарушения принципов объектно-ориентированного подхода в программировании (SOLID), а также возможные нарушения паттернов проектирования и наличие антипаттернов в архитектурах этих приложений.

Нарушения SOLID и Антипаттерны

- Принцип единственной ответственности

По этому принципу каждый класс приложения должен отвечать за решение только одной конкретной задачи.

Во всех рассматриваемых приложениях большинство функционалов, связанного с дополненной реальностью, реализовано внутри класса ViewController, из-за чего ViewController содержит большое число методов, не относящихся к его сфере ответственности. Вследствие этого нарушается принцип единственной ответственности, а ViewController становится очень громоздким, что в свою очередь затрудняет дальнейшую поддержку и внесение новых изменений.

- Принцип подстановки Барбары Лисков

По данному принципу при замене объектов класса на объекты его подкласса поведение программы не должно изменяться. В рассмотренных приложениях данный принцип нарушается, например, из-за того, что при наследовании от класса, описывающего виртуальные объекты, находящиеся на сцене (SCNNode), также переопределяются методы и параметры, связанные с размещением этого объекта на сцене (например, изменяется положение начала системы координат внутри этого объекта). Таким образом, при замене объекта класса SCNNode на такого наследника изменяется способ размещения этого объекта на сцене.

- Принцип разделения интерфейса

Этот принцип говорит о том, что интерфейсы необходимо разделять на более мелкие и специфические.

В текущей архитектуре мобильных приложений с дополненной реальностью, использующих фреймворк ARKit, все методы, связанные с распознаванием окружающего мира, находятся внутри интерфейса ARSessionDelegate. Данный интерфейс содержит методы для получения уведомлений и информации о распознанных плоскостях, распознанных маркерах, а также об обновлениях местоположения устройства. Таким образом, если в приложении используется только распознавание маркеров, то реализации остальных методов приходится оставлять пустыми, что противоречит принципу разделения интерфейсов. Если следовать данному принципу, то интерфейс ARSessionDelegate должен быть разделен на более мелкие интерфейсы, каждый из которых отвечает за распознавание отдельных объектов реального мира. Таким образом, при разработке приложения можно будет использовать один или несколько подходящих интерфейсов.

Также в рассмотренных приложениях присутствуют антипаттерны, например, Божественный объект (God object) – большая концентрация функций в одном классе. Таким классом во всех рассмотренных приложениях является ViewController.

Сложности при дальнейшей поддержке

В связи с тем, что область дополненной реальности очень стремительно развивается, приложения должны быть адаптированы для расширения, добавления новых возможностей и поддержки новых устройств. В связи с тем, что текущие фреймворки разработки приложений не учитывают специфику дополненной реальности, а также из-за нарушения вышеперечисленных принципов, внесение изменений и поддержка мобильных приложений с дополненной реальностью являются дорогостоящим и трудоемким процессом.

В частности, в рассматриваемых нами приложениях наибольшие сложности вызывают поддержка и изменение класса ViewController. В связи с большим количеством различных функций, которые выполняются этим классом в приложениях с дополненной реальностью, внедрение нового функционала в такие приложения требует все больших трудозатрат. Также усложняется процесс поиска возможных ошибок в приложении и тестирования.

Таким образом, текущая архитектура мобильных приложений с дополненной реальностью не учитывает специфику предметной области и не позволяет обеспечивать должное качество разрабатываемых приложений при сопоставимых трудозатратах.

Помимо всех перечисленных проблем, отсутствие специальной архитектуры для разработки приложений с дополненной реальностью приводит к тому, что каждой команде разработки приходится самостоятельно продумывать архитектуру создаваемого приложения, что приводит к дополнительным трудозатратам, а также порождает множество различных подходов и методов для разработки одного и того же функционала. Такое многообразие реализаций, в свою очередь, ведет к повышению трудозатрат на внедрение в команду нового разработчика. Даже если у него уже имеется большой опыт в разработке похожих приложений, разработчику приходится каждый раз заново разбираться в новой архитектуре и использованных методах при реализации стандартного функционала.

ОПИСАНИЕ ПРЕДЛАГАЕМОГО ФРЕЙМВОРКА

Особенности предметной области

Для лучшего понимания специфики разработки мобильных приложений с дополненной реальностью, необходимых функций и возможных ответственностей фреймворка, была составлена карта предметной области дополненной реальности (рис. 1).

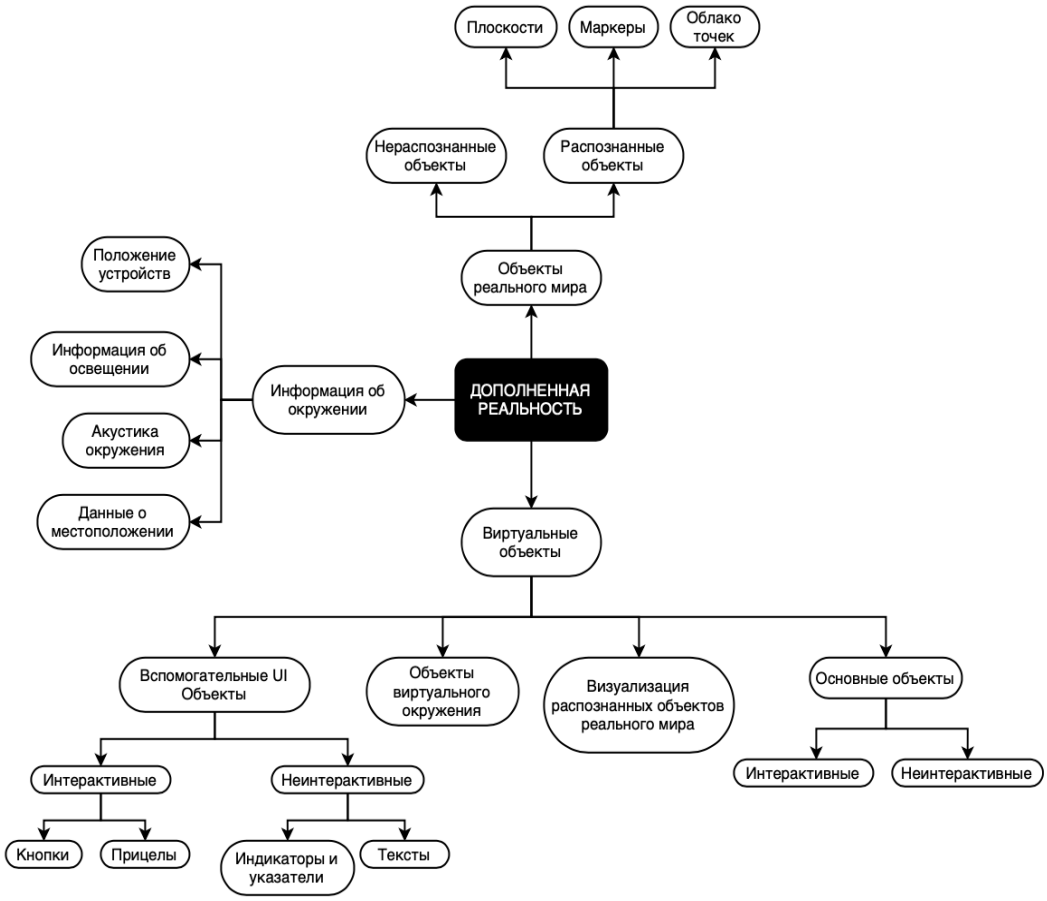

Рисунок 1. Карта предметной области дополненной реальности

Для составления концепции фреймворка, который бы позволил упростить разработку мобильных приложений с дополненной реальностью, необходимо знать, какие именно функции требуют реализации при создании таких приложений. На основе карты предметной области был составлен список функций (ответственностей), требующих реализации при разработке мобильных приложений с дополненной реальностью.

Ответственности:

- Обработка информации от ARKit и ее представление
  - Плоскости
  - Прицел
  - Распознавание 3D-маркеров
  - Распознавание 2D-маркеров
  - Информация об освещении
  - Информация об акустике окружения
  - Распознавание лиц
  - Распознавание людей
  - Информация о положении устройств
  - Информация об облаке точек
- Обработка 3D- и 2D-жестов
  - Выбор объекта для манипуляции
  - Предоставление информации о 2D- и 3D-смещении
  - Предоставление информации о количестве касаний
  - Выполнение манипуляции над заданным объектом
  - Настройка жестов для каждого объекта
- Рендеринг сцены и ее частей
  - Настройки геометрии объектов
  - Настройки материалов
  - Эффекты сцены
  - Анимации
- Синхронизация сессий дополненной реальности

Ключевые элементы и правила предлагаемого фреймворка

Основываясь на карте, представленной в предыдущем параграфе, а также сформулированном наборе ответственностей, можно построить набор ключевых принципов и основных модулей фреймворка для разработки мобильных приложений с дополненной реальностью.

Основные принципы предлагаемого фреймворка:

- Выделение функционала, связанного с AR, в отдельный класс, отличный от ViewController (ARSessionController);
- Разделение сущностей на объекты реального мира (Anchors) и объекты виртуального мира (Entities);
- Использование отдельного класса для работы с жестами (ARGesture);
- Создание специального модуля, отвечающего за визуальное отображение объектов, текстуры, геометрии, эффекты и анимации (ARAppearance);
- Использование ARKit и SceneKit в качестве низкоуровневых технологий, на которых базируется фреймворк.

Основные структурные модули фреймворка:

- ARSessionController – основной класс, инкапсулирующий всю логику AR-сессии. Координирует работу всех остальных компонентов, что позволяет реализовывать логику AR-сессии, не нарушая принципа единственной ответственности, как это было в случае использования класса ViewController;
- ARAnchorController – реализует логику, связанную с распознаванием объектов реального мира: облака точек, плоскостей, 2D-маркеров, 3D-маркеров. Содержит информацию об освещении, акустике окружения, положении устройств. Предоставляет функционал распознавания лиц, людей;
- Anchor – распознанный объект реального мира;
- Entity – виртуальный объект, ьожет быть составным (содержать внутри себя другие Entity);
- ARGesture – класс, отвечающий за обработку жестов над виртуальным объектом (Entity);
- ARAppearance – модуль, содержащий классы для работы с геометриями, текстурами и анимациями.

Описание фреймворка

ARSessionController

Отвечает за работу сессии дополненной реальности ARKit. Является основным классом, через который происходит реализация бизнес логики приложения. Схема работы данного класса представлена на рис. 2.

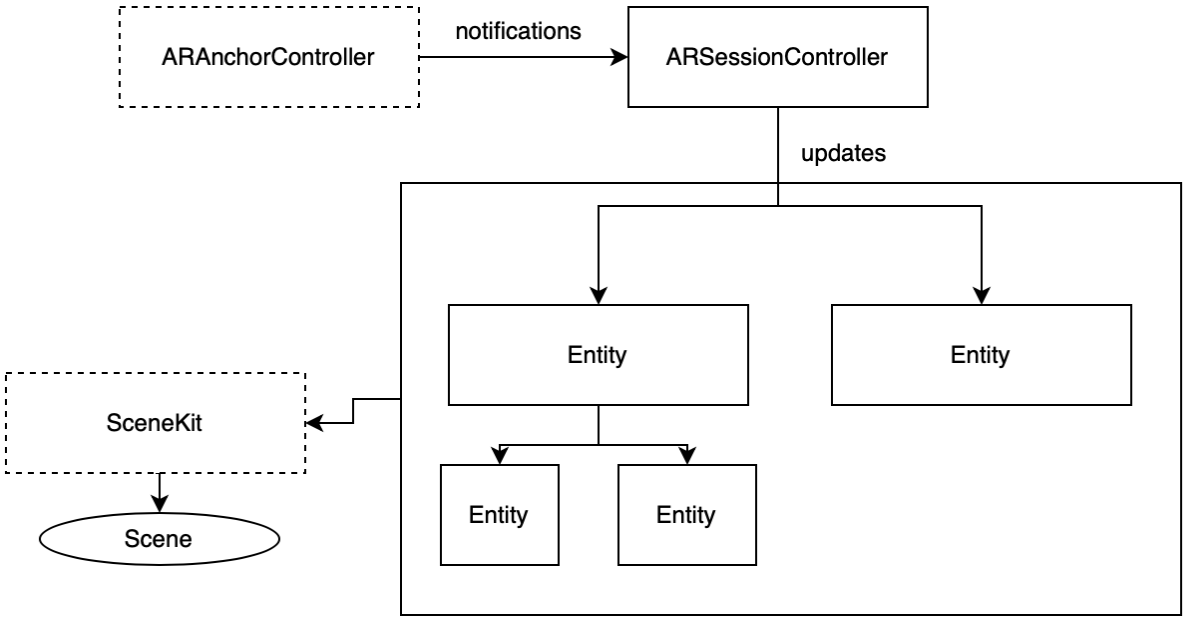

Рисунок 2. Схема работы класса ARSessionController

Содержит следующие поля и методы:

- *scene: SCNScene Объект SceneKit сцены. Содержит информацию о всех виртуальных объектах сцены;*
- *anchorController:* ARAnchorController; контроллер распознанных объектов;
- *didFoundAnchor(:Anchor, :ARAnchorController).* Метод, оповещающий о распознавании нового объекта реального мира;
- *didUpdateAnchor(:Anchor, ARAnchorController).* Метод, оповещающий об обновлении объекта реального мира.

ARAnchorController

Реализует следующие ARKit и SceneKit протоколы: ARSessionDelegate, ARSessionObserver, ARSCNViewDelegate. Отвечает за обработку событий, получаемых от ARKit и SceneKit. Перенаправляет эти события в ARSessionController для

дальнейшей реализации бизнес-логики. Схема работы данного класса представлена на рис. 3.

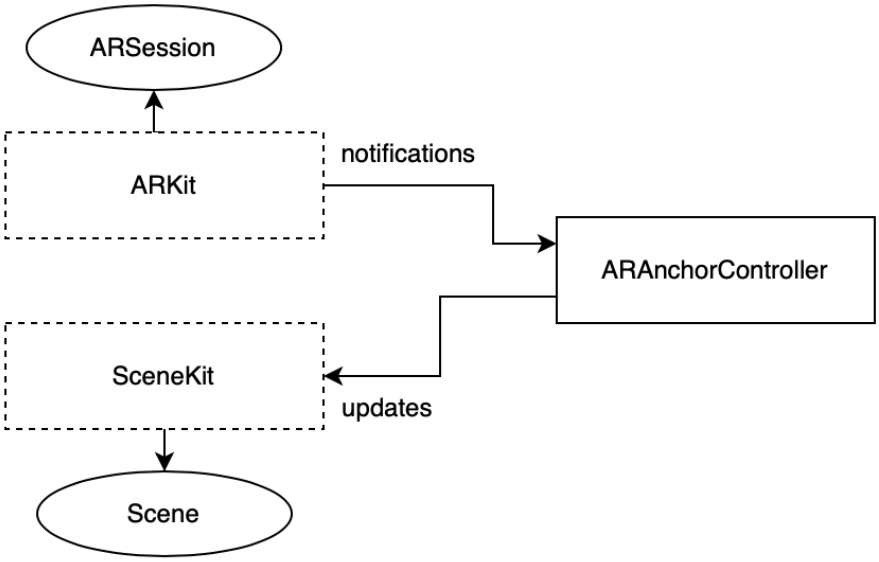

Рисунок 3. Схема работы класса ARAnchorController

Anchor

Представляет распознанный объект реального мира.

- *type: AnchorType.* Тип распознанного объекта;
- *transform: SCNMatrix4x4.* Матрица расположения объекта. Содержит положение, поворот, масштаб объекта.

Entity

Представляет виртуальный объект.

ARGesture

Отвечает за обработку жестов. Автоматически перемещает, поворачивает и масштабирует объекты.

- *gestures:[GestureType].* Массив доступных жестов для заданного объекта.

ARApperance

Модуль, содержащий классы, отвечающие за визуальное отображение объектов. Подробная схема данного модуля представлена на рис. 4.

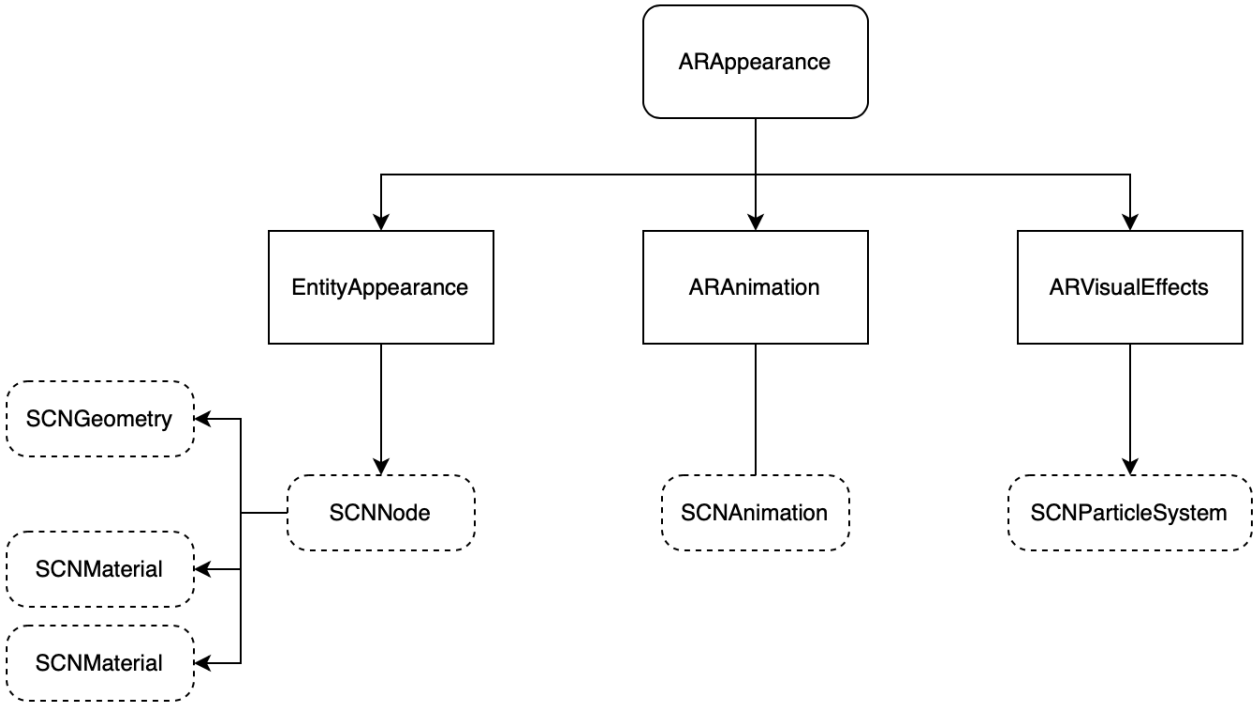

Рисунок 4. Составные части модуля ARAppearance

EntityApperance

Отвечает за геометрию и материалы объектов сцены.

ARAnimation

Отвечает за анимации объектов сцены.

ARVisualEffects

Позволяет создавать визуальные эффекты, используя объекты SCNParticleSystem.

Способ внедрения фреймворка

Фреймворк будет доступен для установки через систему управления зависимостями CocoaPods [8]. Благодаря тому, что работа данного фреймворка базируется на нативных библиотеках ARKit и SceneKit, он может быть встроен в стандартный Xcode-проект приложения с дополненной реальностью.

После установки необходимых модулей в проект нужно назначить объект ARAnchorController в качестве делегата сессии дополненной реальности, а затем создать новый класс – наследник ARSessionController. Внутри него следует реализовывать функционал дополненной реальности создаваемого мобильного приложения.

ЗАКЛЮЧЕНИЕ

Представленный выше фреймворк для разработки мобильных приложений с дополненной реальностью позволяет упростить процесс разработки за счет более высокоуровневого подхода и готовой реализации некоторых повторяющихся функций. Использование данного фреймворка при разработке мобильного приложения позволяет сфокусироваться на реализации уникальных функций приложения благодаря упрощению взаимодействия с ARKit и SceneKit, а также готовой реализации основных повторяющихся функций таких приложений.

Также при использовании представленной архитектуры значительно упрощается процесс поддержки приложения и внедрения нового функционала, так как данная архитектура учитывает специфику предметной области дополненной реальности и не содержит некоторых структурных ошибок, которые затрудняют разработку при использовании архитектуры MVC.

Возможность использования высокоуровнего фреймворка для разработки мобильных приложений с дополненной реальностью снижает трудозатраты на разработку и порог вхождения для разработчиков, а также повышает качество разрабатываемых приложений за счет предоставления готового набора функций и автоматической обработки жестов.

СПИСОК ЛИТЕРАТУРЫ

1. *Facebook Makes Another Billion-Dollar Bet on AR/VR.* URL: https://www.fool.com/investing/2019/09/24/facebook-makes-another-billion-dollar-bet-on-arvr.aspx.

2. *Mobile AR is evolving faster than you think* URL: https://venture-beat.com/2018/05/19/mobile-ar-is-evolving-faster-than-you-think/.

3. *Virtual and Augmented Reality Users 2019.* URL: https://www.emar-keter.com/ content/virtual-and-augmented-reality-users-2019.

4. *5 challenges to mainstream AR adoption.* URL: https://www.al-lerin.com/blog/5-challenges-to-mainstream-ar-adoption.

5. *Building an AR app with ARKit and SceneKit.* URL: https://blog.pusher.com/building-an-ar-app-with-arkit-and-scenekit/.

6. *What Apple's new AR bowling game taught us about the future.* URL: https://www.cnet.com/news/what-apples-new-ar-bowling-game-taught-us-about-the-future/.

7. *CoreML-in-ARKit* // Github. URL: https://github.com/hanleyweng/CoreML-in-ARKit.

8. *CocoaPods* // Github. URL: https://github.com/CocoaPods/CocoaPods.

NATIVE MOBILE APPLICATIONS WITH AUGMENTED REALITY DEVELOPMENT FRAMEWORK

Dmitriy Evdokimenko [1], Rinat Khanov [2]

The Higher Institute of Information Technology and Intelligent Systems, Kazan Federal University

[1]evdodima@gmail.com, [2]rinat@khanov.com,

Abstract

Framework for development of native mobile applications with augmented reality. Analysis of various development spheres: game development, mobile applications development. The concept of a new framework and a set of tools for developing mobile applications with augmented reality is presented.

Keywords*: augmented reality, AR, ARKit, iOS, development, framework.*

REFERENCES

1. Facebook Makes Another Billion-Dollar Bet on AR/VR. URL: https://www.fool.com/investing/2019/09/24/facebook-makes-another-billion-dollar-bet-on-arvr.aspx.

2. *Mobile AR is evolving faster than you think* URL: https://venturebeat.com/2018/05/19/mobile-ar-is-evolving-faster-than-you-think/.

3. *Virtual and Augmented Reality Users 2019.* URL: https://www.emarketer.com/ content/virtual-and-augmented-reality-users-2019.

4. *5 challenges to mainstream AR adoption.* URL: https://www.allerin.com/blog/5-challenges-to-mainstream-ar-adoption.

5. *Building an AR app with ARKit and SceneKit.* URL: https://blog.pusher.com/building-an-ar-app-with-arkit-and-scenekit/.

6. *What Apple's new AR bowling game taught us about the future.* URL: https://www.cnet.com/news/what-apples-new-ar-bowling-game-taught-us-about-the-future/.

7. *CoreML-in-ARKit* // Github. URL: https://github.com/hanleyweng/CoreML-in-ARKit.

8. *CocoaPods* // Github. URL: https://github.com/CocoaPods/CocoaPods.

СВЕДЕНИЯ ОБ АВТОРАХ

ЕВДОКИМЕНКО Дмитрий Андреевич – студент 2 курса магистратуры Высшей школы информационных технологий и интеллектуальных систем Казанского федерального университета, направление подготовки – «Программная инженерия».

Dmitriy Andreevich EVDOKIMENKO, student of 2 grade of master program of Higher Institute of Information Technologies and Intelligent Systems of Kazan Federal University.

email: evdodima@gmail.com

ХАНОВ Ринат Гафурович – студент 2 курса магистратуры Высшей школы информационных технологий и интеллектуальных систем КФУ, направление «Программная инженерия».

Rinat Gafurovich KHANOV, student of 2 grade of master program of Higher Institute of Information Technologies and Intelligent Systems of Kazan Federal University.

email: rinat@khanov.com

Материал поступил в редакцию 7 апреля 2020 года