Оглавление

Аннотация 3

1. Введение 3

1.1 Глоссарий 3

1.2 Рынок мобильных игр 4

1.3 Неформальная постановка задачи 6

1.4 Обзор существующих решений 7

1.5 План работ 12

2. Требования к окружению 12

2.1 Требования к аппаратному обеспечению 12

2.2 Требования к программному обеспечению 12

2.3 Требования к пользователям 12

3. Функциональные требования 12

4. Требования к интерфейсу 14

5. Проект 16

5.1 Средства реализации 16

5.2 Структура данных 17

5.3 Модули и алгоритмы 18

5.4 Проблемы и решения 23

5.5 Проект интерфейса 26

6. Реализация и тестирование 35

Заключение 38

Список литературы 39

# Аннотация

В данной работе представлены этапы создания мобильной игры iOS: исследование рынка мобильных игр в жанре «головоломка», проектирование и реализация игры, опубликование игры в магазине мобильных игр App Store, сбор статистики, выпуск обновлений на основе найденных недочётов и отзывов пользователей, а также на основе статистики. Также описаны средства реализации и используемые модули. Игра имеет уникальную механику, что делаем данный продукт перспективным.

# Введение

Разрабатываемая система — мобильная игра «Стратегист». Жанр игры — головоломка. Механика игра: необходимо провести персонажа из точки А (начальная позиция) в точку Б (финишная черта [алмаз]) на игровом поле, которое разделено на клетки одинакового размера (размер поля по умолчанию — 5 на 5). Проводить персонажа нужно с помощью tap и swipe, прокладывая путь. На игровом поле помимо главного персонажа, присутствуют другие объекты (враги), которые усложняют прокладывание пути из начальной позиции в конечную. После того, как путь проложен игрок должен нажать кнопку «Старт», после чего запускается основной цикл игры и все объекты (главный персонаж, враги) начинают пошагово перемещаться. Каждый вражеский объект имеет свою собственную механику поведения (например, пчела перемещается по игровому полю, если главный персонаж попадает на пчелу, то игрок проигрывает). Помимо перемещающихся объектов, на игровом поле могут присутствовать статичные объекты (например, бомба, которая взрывается через N ходов и, если главный персонаж попадает в радиус одной клетки вокруг бомбы во время взрыва, то игрок проигрывает). В каждом новом уровне увеличивается сложность и добавляются новые объекты.

## 1.1 Глоссарий

**AppStore** — интернет-магазин приложений для устройств под управлением iOS от компании Apple;

**Flurry Analytics** — библиотека для сбора и анализа статистических данных;

**Swift** — открытый компилируемый язык программирования общего назначения. Создан компанией Apple в первую очередь для разработчиков iOS и macOS;

**Xcode** —интегрированная среда разработки (IDE) программного обеспечения для платформ macOS, iOS, watchOS и tvOS, разработанная корпорацией Apple;

**Tap** — касание пальцем экрана смартфона;

**Swipe** — tap с дальнейшем проведением по экрану смартфона;

**Endless игра** — игра, которую нельзя выиграть. Бесконечная игра (прохождение на рекорд);

**Github** — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. Основан на системе контроля версий Git и разработан на Ruby on Rails и Erlang компанией GitHub, Inc (ранее Logical Awesome).

## 1.2 Рынок мобильных игр

Аналитики компании Newzoo утверждают, что рынок игр (компьютерных, мобильных и прочих) по итогам текущего года достигнет отметки в 137,8 млрд долларов, что на 13,3% превысит показатель 2017 года[14].

Самым крупным сегментом остаются мобильные игры. Если точнее, по итогам года ожидается, что этот рынок достигнет отметки в 70,3 млрд долларов, то есть на мобильные игры будет приходиться более половины всего рынка. В годовом выражении этот сегмент вырастет на 25,5%[2][3].

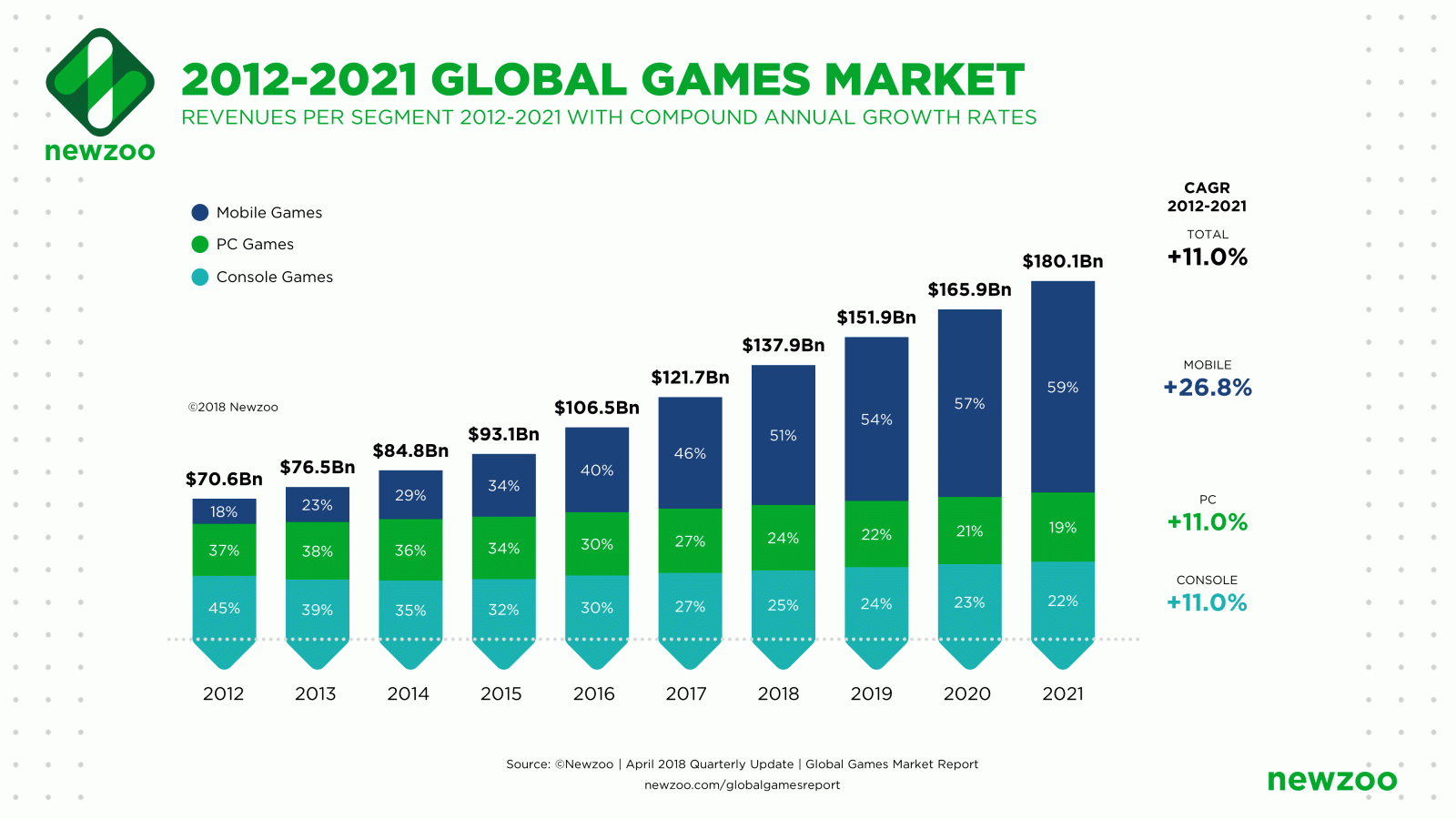


Рисунок 1 — Развитие рынка мобильных игр

Если рассматривать рынок мобильных игр по регионам, то больше всего на игры тратят в Китае. По итогам текущего года китайцы потратят 37,9 млрд долларов. Жители США потратят 30,4 млрд, а японцы — 19,2 млрд долларов[4][15].

В США, [67% пользователей](http://services.google.com/fh/files/misc/omp-2013-us-en.pdf) используют смартфон, чтобы выходить в интернет каждый день, и большинство никогда не выйдут из дома без своего телефона[9]. Аналогичный рост использования смартфонов наблюдается и в России – даже пользователи с доходами ниже среднего чаще выбирают смартфон вместо компьютера, чтобы всегда оставаться на связи с миром. [Исследования рынка](http://www.pewinternet.org/2010/09/14/the-rise-of-apps-culture/) показывают, что около половины всех пользователей мобильных телефонов загрузили приложения, а две трети из них регулярно их используют. Большинство пользователей мобильных приложений находятся в возрастном промежутке от 25 до 30 лет, женаты или замужем, живут в пригородных районах, и имеют высшее образование. Таким образом, пользователи мобильных приложений в целом моложе, более образованы и имеют более высокий доход, нежели пользователи, не использующие мобильные приложения[5].

Жанр головоломка был выбран, так как данный жанр способен развивать дедуктивные способности человека. Данный жанр, помимо развлекательной составляющей, также может содержать ещё и познавательный контент. В данной игре, чтобы построить маршрут, необходимо учесть все варианты перемещения вражеских объектов, что способствует развитию стратегических способностей, продумывание будущих действий.

### Жанр мобильных игр «головоломка»

Головоломка — название жанра компьютерных игр, целью которых является решение логических задач, требующих от игрока задействования логики, стратегии и интуиции.

Основные условия при разработке игр в жанре головоломка:

Понятность: в правилах и управлении можно легко разобраться.

Распознаваемость системы: системы внутри головоломки должны быть понятными, чтобы игроки могли распознать серию возможных действий, необходимых для решения головоломки. Обычно используются цвета и формы, а также другие сферы общих знаний (правописание или простые функции вещей);

Простота использования: для качественного взаимодействия, игроку необходим хороший интерфейс;

Поощрение навыка игрока: Если игрок встречал головоломку много раз, он должен иметь возможность совершенствоваться в своём умении[1];

### Цель

Цель данной работы: минимальное количество пользователей игры должно быть больше 100, а также набрать минимальный уровень показателя ретенции 7 дней — 5%.

## 1.3 Неформальная постановка задачи

Проанализировав аналогичные приложения в магазине мобильных игр AppStore, были выявлены следующие требования, которым должна отвечать разработанная мобильная игра:

1. Иметь понятный интерфейс;
2. Иметь бесплатную модель распространения со встроенными покупками;
3. Включать встроенную рекламу;
4. Иметь различные способы управления (tap, swipe);
5. Иметь высокую сложность прохождения;
6. Иметь различный игровой контент (объекты, с которыми пользователь взаимодействует);

## 1.4 Обзор существующих решений

Перед разработкой игры было проанализировано более 200 мобильных игр в категории «головоломка». Были найдены 5 схожих игр, которые максимально схожи с разработанной игрой:

### Two dots

Популярная головоломка от студии Playdots, которая находится в ТОП-100 в жанре настольные. Механика игры: необходимо собирать цепочки из шариков с помощью swipe. В игре приятный дизайн. Поддерживается русский язык. Игра достаточно часто обновляется (каждые две недели), что говорит о том, что данная игра ещё развивается[10].

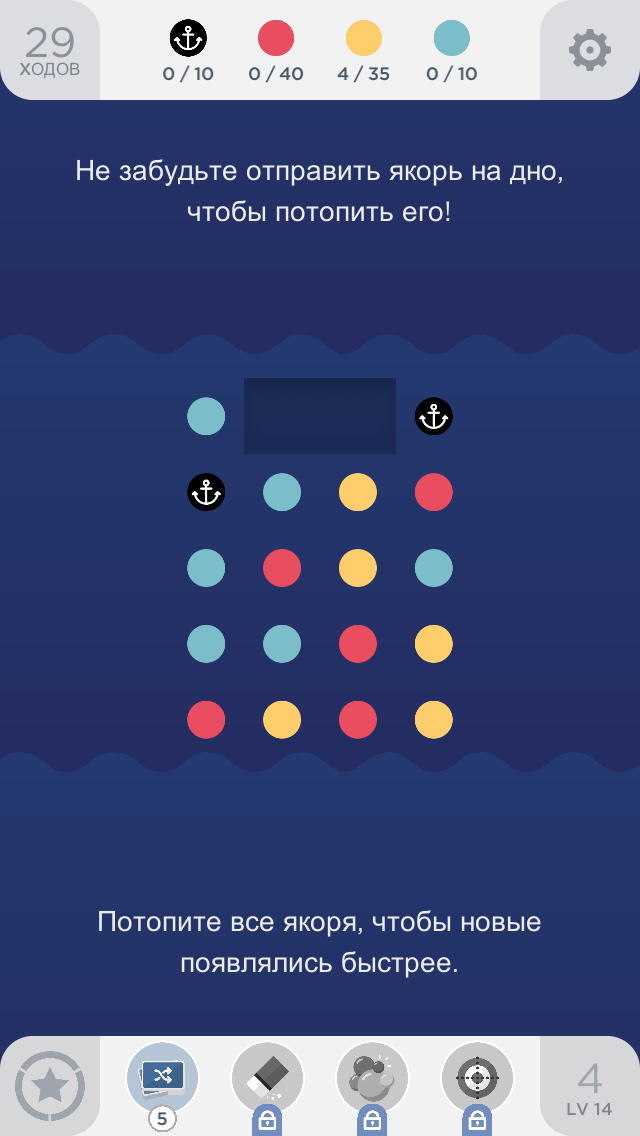


Рисунок 2 — Скриншот интерфейса мобильной игры "Two Dots"

### Flow Free

Классическая игра своего жанра с понятной механикой: необходимо соединить все точки одинакового цвета, чтобы не одна линия не пересекала другую. Простой дизайн игры, с достаточно непростым интерфейсом, который может сбить толку рядового пользователя. Игру русифицирована, хотя название игры не переведено. Достаточно много однотипных уровней. Кнопки интерфейса небольшие, что может вызывать некоторые проблемы при нажатии на них[8].

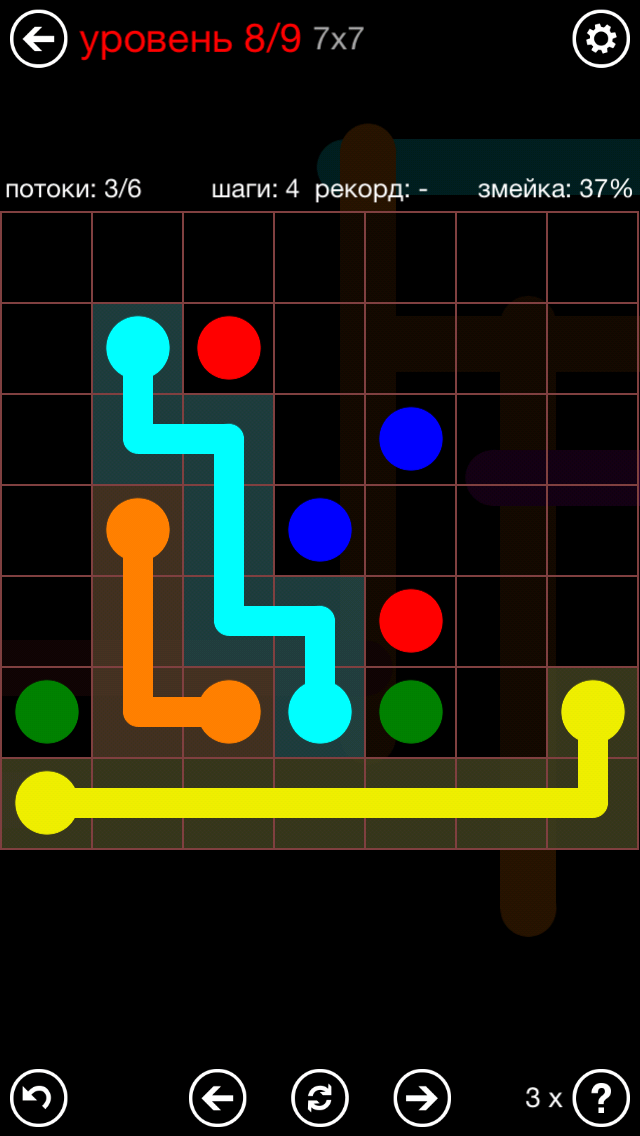


Рисунок 3 — Скриншот интерфейса мобильной игры "Flow Free"

### Water pipes / Водопровод

Головоломка с большим количеством однотипных уровней. Механика игры несложная: если нажать на трубу, то она развернётся на 90 градусов по часовой стрелке, необходимо собрать из небольших кусочков одну большую трубу. Интерфейс игры – несложный, присутствуют внутриигровые покупки, с помощью которых можно пройти сложные уровни. Игра имеет русскую версию[13].

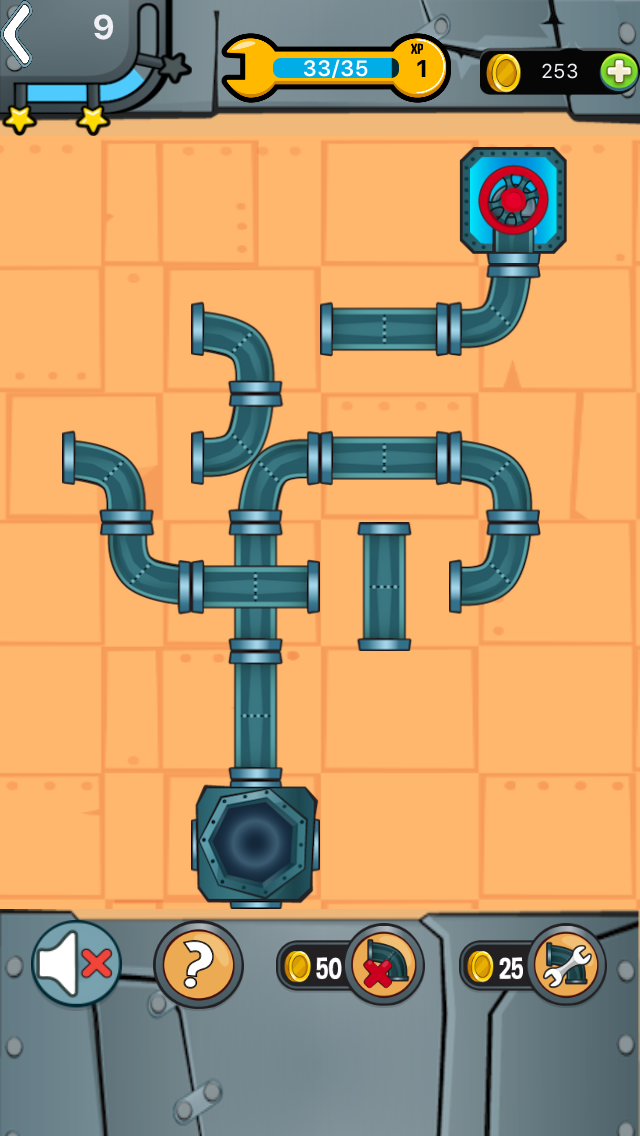


Рисунок 4 — Скриншот интерфейса мобильной игры "Water Pipes"

### Chanquan

Игра, которая не поддерживает русский язык, но разработчики сделали такой интерфейс, что все основные функции игры можно выполнить без знания английского языка (кнопка «Начало» отображена в виде треугольника, который указывает в правую сторону). Механика игры: необходимо с помощью swipe привести кулак к финишу, после кулака остаётся след (линия), которую нельзя пересекать. Очень небольшое разнообразие игровых механик и контента. Добавляется всего лишь один вспомогательный игровой блок – стена[7].



Рисунок 5 — Скриншот интерфейса мобильной игры "Chanquan"

### Smove

Endless игра, которая имеет приятный дизайн. Механика игры: с помощью swipe необходимо избегать движущихся шаров, а также собирать появляющиеся ромбы на игровом поле. Отсутствует локализация на русский язык, что не особо ухудшает игру, так как интерфейс игры очень простой и имеет всего несколько активных действий: начать игру, поставить на паузу[12].



Рисунок 6 — Скриншот интерфейса мобильной игры "Smove"

Ниже приведены критерии сравнения конкурентных игр:

Таблица 1 — Сравнительная таблица всех представленных игр



Как можно метить из таблицы выше, большинство игр в данном жанре не имеет метаигру.

## 1.5 План работ

* Представить концепт игры руководителю;
* Разработать базовый функционал;
* Разработать первую версию игры (альфа-версия);
* Разработать вторую версию игры (бета-версия);
* Локально протестировать игру, исправить ошибки;
* Выпустить игру в магазин мобильных приложений AppStore;
* Собрать статистику и выпустить обновления на её основе;
* Написать отчет для защиты ВКР;
* Переработать, исправить отчёт для ВКР;

# Требования к окружению

## 2.1 Требования к аппаратному обеспечению

Необходимое устройство: iPhone, iPad, iPod touch начиная с версии iOS 10.0 и выше; Для эффективной работы требуется iPhone 5 и выше;

## 2.2 Требования к программному обеспечению

Для нормального функционирования системы необходима операционная система iOS 10.0 и выше.

## Требования к пользователям

Для использования игры необходимы базовые навыки пользования мобильным устройством, а также знания русского или английского языков.

Ролевая система в данном мобильной игре отсутствует. Единственная доступная роль – игрок (пользователь игры).

# Функциональные требования

Система должна:

* Иметь прямоугольное игровое поле размером MxN (где, M > 2, N > 2);
* Иметь светлые оттенки цветов, преимущественно светло-зелёный;
* Иметь доску достижений;
* Иметь окно настроек;
* Позволять играть бонусный уровень на рекорд в окне настроек;

Система должна позволять пользователю:

* Входить на игровой уровень;
* Совершать игровые действия внутри уровня;
* Строить маршруты с помощью swipe, tap;
* Проходить процесс обучение;
* Переигрывать уровень;
* Активировать режим предпросмотра уровня;
* Просматривать доступное количество жизней на уровне;
* Получать бесплатный ежедневный алмаз;
* Искать главного персонажа в окне выбора уровня;
* Включать/выключать режим предпросмотра на игровом поле;
* Оставлять отзыв об игре в окне настроек;
* Узнавать дополнительную информацию об объектах, при клике на них;
* Делать долгий tap (зажатие), чтобы узнать информацию об объекте, если маршрут уже построен;
* Пропускать прохождение некоторых уровней;
* Покупать внутриигровую валюту за реальные деньги;
* Покупать дополнительные жизни за внутриигровую валюту;
* Покупать выигрышные стратегии на уровнях за внутриигровую валюту;
* Восстанавливать одну жизнь каждые 10 минут, на уровне, на котором осталось 0 жизней;
* Играть бонус-уровень для получения дополнительной внутриигровой валюты;
* Ставить бонус-уровень на паузу;
* Смотреть рекламу для получения дополнительной внутриигровой валюты;
* Стирать выбранный путь (стратегию);
* Смотреть информацию об объектах (врагах) на игровой сцене;
* Получать ежедневную, бесплатную внутриигровую валюту;
* Зарабатывать достижения;
* Получать вознаграждения за заработанные достижения;
* Выключать/включать фоновую музыку, звуки;
* Выключать/включать подсказки на уровнях;
* Оставлять оценку и отзыв о мобильной игре;
* Просматривать количество внутриигровой валюты;
* Просматривать доступное количество ходов на уровне;
* Записывать текущий прогресс игрока;
* Сохранять и загружать игровой процесс;



Рисунок 7 — Варианты переходов между экранами в игре

# Требования к интерфейсу

Основные требования к интерфейсу:

* взаимодействие пользователей с системой посредством визуального графического интерфейса;
* удобный доступ к основным функциям и операциям, выполняемым системой;
* интуитивно понятный интерфейс, реализованный с учетом привычных для пользователя задач;
* при возникновении ошибок или ошибочных действиях пользователей необходимо, чтобы система показывала вид ошибка на экране, а также показывала о необходимых дальнейших действиях;
* все экранные формы пользовательского интерфейса в рамках подсистем необходимо выполнить в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;
* для обозначения сходных операций должны использоваться сходные графические значки, кнопки и другие управляющие (навигационные) элементы;
* необходимы унифицированные термины, используемые для описания типовых понятий, операций и действий пользователя;
* адаптивность для различных разрешений экранов и различных мобильных устройств;
* при проигрыше игрока должно выводиться сообщение о причине проигрыша;
* основой шрифт интерфейса — Avenir Next, был выбран, так как присутствует на всех устройствах с операционной системой iOS, а также является читабельным для пользователей;
* название игры — «Стратегист». Название было выбрано, так как игроку необходимо прокладывать пути (выбирать стратегию);
* в случае отсутствия интернета, должно выводиться окно с сообщением, о том, что реклама не может быть загружена для показа;

# Проект

## 5.1 Средства реализации

### Swift

Помимо языка программирования Swift, рассматривались также языки Objective C и C#. Так как разработка мобильной игры проходила только на систему iOS, то изначально из выбора был исключён C#, так как C# используется Unity и Xamarin — интегрированными средами разработки для различных операционных систем[6].

В дальнейшем проходил выбор между Objective C и Swift. Objective C — более старый язык, чем Swift и документации по данному языку написано больше, но данный язык имеет очень непростой синтаксис, объём кода на Objective C значительно превышает тот же самый код на Swift. Также язык программирования Swift – значительно быстрее компилируется и выполняется, чем Objective C[11].

### Xcode

Помимо интегрированной среды разработки Xcode, рассматривались также Xamarin и Unity. Так как разработка велась только для операционной системы iOS, то было решено использовать Xcode, как наиболее подходящий. Xamarin и Unity — являются межплатформенными средами разработки приложений.

### CocoaPods

Менеджер зависимостей уровня приложений для Objective-C, Swift и любых других языков, работающих в среде выполнения Objective-C.

CocoaPods было решено использовать как самый распространенный и удобный менеджер зависимостей.

### Flurry analytics

Бесплатный сервис аналитики от компании Yahoo для измерения охвата аудитории, вовлечения, возвратов, конверсий, доходов и прочих игровых параметров. Данный сервис был выбран, так как распространяется на бесплатной основе.

### AdMob

Библиотека, разработанная компанией Google, которая позволяет показывать в приложениях объявления из различных источников, таких как сеть AdMob, внешние рекламные сети и кампании с собственными объявлениями. Этот инструмент отправляет запросы в разные сети и выбирает самые выгодные варианты, что позволяет повысить заполняемость и доход от показа рекламы.

## Структура данных

Ниже представлен JSON формат уровня:

{

"Финиш": [X, Y],

"ГлавныйПерсонаж": [X, Y],

"РазмерДоски": [M, N],

"КоличествоХодов": Число,

"КоличествоАлмазовВыигрыш": Число,

"ИспользоватьВсеХоды": Флаг,

"УровеньОбучение": Флаг,

"ВыигрышныйПуть": [Массив[X, Y]]

"ПеремещающиесяОбъекты": [

{

"Тип": "Перечисление.ТипОбъекта",

"Перемещение": [Массив[X, Y]]

}

],

"СтатичныеОбъекты": [

{

"Тип": "Перечисление.ТипОбъекта",

"Точка": [X, Y]

}

]

}

## Модули и алгоритмы

**class GameScene: SKScene**

Класс, который отвечает за работу с игровым полем. У класса более 5 расширений: обработчики tap/swipe, предпросмотр, обработка уровней, обучение и другие. Далее приведены основные свойства класса:

* public var move: Int

Текущий игровой ход. Увеличивается на 1 единицу каждый игровой ход.

* public var moves: Int

Количество доступных ходов, чтобы построить выигрышную стратегию. У игрока есть определённое количество ходов на уровень, за который его (уровень) можно пройти.

* public var movingObjects: Set<Object>

Сет всех перемещающихся объектов.

* public var losingReason: String?

Причина проигрыша. После каждого проигрыша данная строка принимает значение причины, по которой игрок не смог пройти уровень.

* public var boardSize: Point

Размер доски. Значение задаётся из JSON файла. Если значение не задано, то устанавливается значение по умолчанию – *Point(column: 5, row: 5)*

* public var character: Character

Главный персонаж. Переменная инициализируется в функции *createLevel()*

Основные методы:

* func createLevel()

Создаёт игровую сцену. Инициализируем все игровые объекты. Добавляет задний фон на игровую сцену.

* func startLevel()

Запускает уровень или режим предпросмотра. Блокирует/активирует элементы интерфейса.

* func loseLevel()

Обработчик проигрыша уровня. Завершает все игровые процессы/таймеры. Отнимает жизнь на уровне. Запускает рекламу каждый шестой проигрыш. Показывает причину проигрыша. Отображает модальное окно о проигрыше.

* func winLevel()

Обработчик выигрыша уровня. Добавляет выигранное количество алмазов игроку. Отображает модальное окно о выигрыше.

* func goToLevel(\_ level: Int)

Функция загрузки текущего уровня из JSON файла. Обрабатывает и задаёт начальные свойства всех игровых объектов (размер игровой доски, начальная позиция, финишная позиция).

* func modalWindowPresent(type: modalWindowType)

Функция отображает модальное окно поверх всех других окон.

* func checkMovingObjectPos(

object: Object,

characterMove: Int,

isLoseLevelByDefault: Bool = true) -> Bool

Функция проверяет столкновение главного персонажа и перемещающегося объекта.

* func checkStatisObjectPos(object: StaticObject) -> Bool

Функция проверяет столкновение главного персонажа и статичного объекта.

* func worldMove()

Функция выполняет перемещения игровых объектов. Также выполняются основные проверки столкновений главного персонажа с объектами (врагами).

* func worldPreview(move: Int)

Функция запускает предпросмотр сцены.

* func checkFinish()

Функция проверяет попадание главного персонажа на финишный блок.

**class Model**

Связующий класс, используется, в основном, через его свойство *shaderInstance* для быстрого доступа из любого другого класса. Содержит, в основном. все переменные, которые сохраняют прогресс пользователя/настройки, а также функции для работы с ними.

Основные свойства:

* public let countLevels: Int

Константа, которая содержит общее количество уровней.

* public var currentLevel: Int

Текущий уровень (если активно окно выбора уровня, то указывает на расположение главного персонажа, если активно окно игрового поля, то указывает на уровень, на котором находится игрок).

* private var countLives: [Int]!

Массив, который содержит количество жизней на каждом уровне.

* private var completedLevels: [Bool]!

Массив, который содержит значения завершённости уровней (true – уровень пройден, false – уровень не пройден), количество элементов равно константе *countLevels*.

Основные методы

* func getLevelLives(\_ level: Int) -> Int

Функция возвращает количество жизней на уровне.

* func setLevelLives(level: Int, newValue: Int = -1)

Функция устанавливает количество жизней на уровне.

* func isCompletedLevel(\_ level: Int) -> Bool

Функция возвращает true, если выбранный уровень пройден, в противном случае false.

* func setCompletedLevel(\_ level: Int, value: Bool = true)

Функция устанавливает завершённость уровня, true – уровень завершён, false – уровень не завершён.

* func sendNotification(id identifier: String,

time: Double,

title: String,

message: String)

Функция отправляет уведомление с название *title* и текстом *message*, через время в секундах *time*.

**class IAPHandler:**

**NSObject,**

**SKProductsRequestDelegate,**

**SKPaymentTransactionObserver**

Класс, который отвечает за обработку встроенных покупок.

Основные методы:

* func purchaseProduct(id: String)

Функция, которая подготавливает встроенную покупку.

* func paymentQueue(

\_ queue: SKPaymentQueue,

updatedTransactions transactions: [SKPaymentTransaction])

Функция, которая добавляет транзакцию в очередь встроенных покупок.

**class BossLevel: NSObject, SKPhysicsContactDelegate**

Класс, который отвечает за логику бонус-уровней.

Основные методы:

* func showCountdown(timeToStart: Int, labelTimeToStart: UILabel)

Функция вызывает таймер перед началом бонус-уровня или после снятия игры с паузы.

* func pickUpCoin(nodeToRemove: SKNode)

Функция срабатывает, если главный персонаж подбирает монетку.

* func cleanTimers()

Функция, которая очищает все запущенные таймеры.

**class AchieveViewContoller: UIViewController**

Класс, который отвечает за отображение и обработку достижений, их получение.

Основные методы:

* func countCompletedLevels(\_ maxLevel: Int) -> Int

Функция, которая считает количество пройденных уровней в интервале [0; maxLevel].

* func addGems(amount: Int) {

Функция увеличивает текущее количество алмазов игроку на *amount*. Также отображает новое значение в интерфейсе.

**class ChooseLevelViewController:**

**UIViewController,**

**GADRewardBasedVideoAdDelegate**

Класс, который отвечает за работу меню выбора уровня.

Основные свойства:

* var distanceBetweenLevels:Int

Расстояние в клетках между ближайшими уровнями.

* var character: UIImageView!

Переменная, которая сохраняет текстуру главного персонажа.

* var moveCharacterToNextLevel: Bool

Флаг, который указывает нужно ли главному персонажу перемещаться. Если значение *true*, то главный персонаж перемещается от текущего уровня к следующему. Данный флаг был заведён, так как после выигранного уровня персонаж автоматически перемещается на уровень выше, но если перейти в меню выбора уровней, то флаг установлен в значение *false* и главный персонаж остаётся на месте.

* var modalWindow: UIView!

Окно, на котором отображается модальное окно (переход к уровню)

Основные методы:

* @objc func getFreeEveryDayGem(\_ sender: UITapGestureRecognizer)

Функция, которая срабатывает при клике пользователя на бесплатный ежедневный алмаз.

* func moveToPoint(from: Point, to: Point, delay: CFTimeInterval = 0)

Функция, которая вызывает анимацию перемещения главного персонажа по кривой от одного уровня к другому.

* func goToLevel()

Функция, которая обеспечивает переход пользователя к выбранному уровню.

* func pathToPoint(from: Point, to: Point) -> (bezier: UIBezierPath, count: Int)

Функция, которая возвращает ближайший путь между двумя точками на меню выбора уровня.

* func convertPoint(point: CGPoint) -> (success: Bool, point: Point)

Функция конвертирует CGPoint в позицию на игровом поле, если клик был сделан по игровому полю.

**class MenuViewController:**

**UIViewController,**

**GADRewardBasedVideoAdDelegate**

Класс, который отвечает за работу игрового меню.

Основные методы:

* @IBAction func showTips(sender: UISwitch)

Включение/выключение подсказок на игровом поле при клике на объект.

* @IBAction func buyGems(sender: UIButton)

Покупка алмазов в меню.

* func addGems(amount: Int, animation: Bool = true)

Функция добавляет алмазов после покупки и отображает новое значение в интерфейсе.

* @IBAction func rateApp(sender: UIButton)

Функция вызывает окно оценки приложения и написания отзыва.

## Проблемы и решения

### Встроенные покупки

Самое сложное для реализации в данной работе – это реализовать обработчик встроенных покупок. Сложнее всего было отлаживать данную функцию приложения, так как реальные транзакции можно проверять только на реальных устройствах.

В первой версии приложения при низком качестве интернет соединения, при покупке внутриигровой валюты пользователь мог потратить деньги, но не получить внутриигровую валюту.

### Босс–уровни — неудачная попытка

В первых версиях приложения каждый 16 уровень являлся босс-уровнем. Механика босс-уровня: необходимо с помощью *swipe* перемещать главного персонажа, избегая движущихся объектов (вражеских). Если главный персонаж сталкивается с движущимся на него объектом, то игрок проигрывает. Необходимо собрать N (где N ≥ 10) движущихся навстречу звёзд, чтобы выиграть уровень.

Так как механика босс-уровня значительно отличается от механики самой игры (обычных уровней), то многие пользователи могли завершать игру на первом босс-уровне (первый босс-уровень встречался на 16 уровне). Так как босс-уровень был обязательны к прохождению, то некоторые пользователи, которые не могли пройти его просто удаляли игру с мобильного устройства.

В одном из первых обновлений было решено босс-уровни переименовать в бонус-уровни, за которые можно получить немного внутриигровой валюты. Также бонус-уровни необязательны к прохождению и их можно пропускать.

### Недопонимание пользователей причины проигрыша

Пользователи не всегда понимали причину проигрыша на уровне. Например, пользователь мог собрать не все звёзды, но попасть на финишный блок (алмаз) и не понять почему он проиграл уровень. Чтобы убрать недопонимание пользователей были добавлены подсказки при каждом проигрыше уровня, которые показывали причину проигрыша игрока (например, «Причина проигрыша: вы столкнулись с пчелой»).

### Отсутствие мотивации

В игре была слабая мотивирующая система. В первом обновлении была добавлена доска достижений. Некоторые достижения:

* Завершите N (где N = 5, 25, 50, 100) уровней;
* Завершите N (где N = 5, 10, 25, 50) уровней без использования выигрышного пути;
* Завершите N (где N = 5, 10, 25, 50) уровней с первой попытки;
* Соберите N (где N = 5, 25, 50, 100) ежедневных алмазов;
* Соберите N (где N = 15, 100, 250, 500) алмазов на бонусных уровнях;

Во втором обновлении были добавлены ежедневные алмазы. Алмаз можно получить каждые 24 часа. Данная функция была добавлена, чтобы мотивировать пользователей к повторному возвращению. Позже были добавлены уведомления, которые оповещали пользователей о доступном бесплатном алмазе.

### Сложная игра

Первоначальный вариант игры был достаточно сложным для пользователей: так как жизни на уровнях не восстанавливались, получить бесплатную внутриигровую валюту было невозможно. Поэтому были добавлены ежедневные бесплатные алмазы. Также были добавлены бесплатные жизни на уровнях, которые восстанавливаются каждые 10 минут, если пользователь проигрывает все жизни на уровне и цены на все внутриигровые покупки (покупка выигрышного пути, восстановление жизней на уровне) были уменьшены.

В четвёртом обновлении появилась новая функция: получение алмазов за рекламу. Зайдя в меню, можно посмотреть 30 секундный рекламный видеоролик, за который получить 1 алмаз.

В последующих обновлениях была добавлена ещё одна функция получения внутриигровой валюты – бонус-уровень. Бонус-уровень – это бесконечный уровень, в котором за каждые собранные 5 звёзд можно получить 1 алмаз, с каждым полученным алмазом скорость всех перемещающихся объектов увеличивается.

## 5.5 Проект интерфейса

### Окно выбора уровня

Каждый уровень обозначен соответствующим числом. Главный персонаж находится на уровне, который следует после последнего пройденного уровня. Из данного окна можно попасть в разделы: достижения, меню, игровое поле.

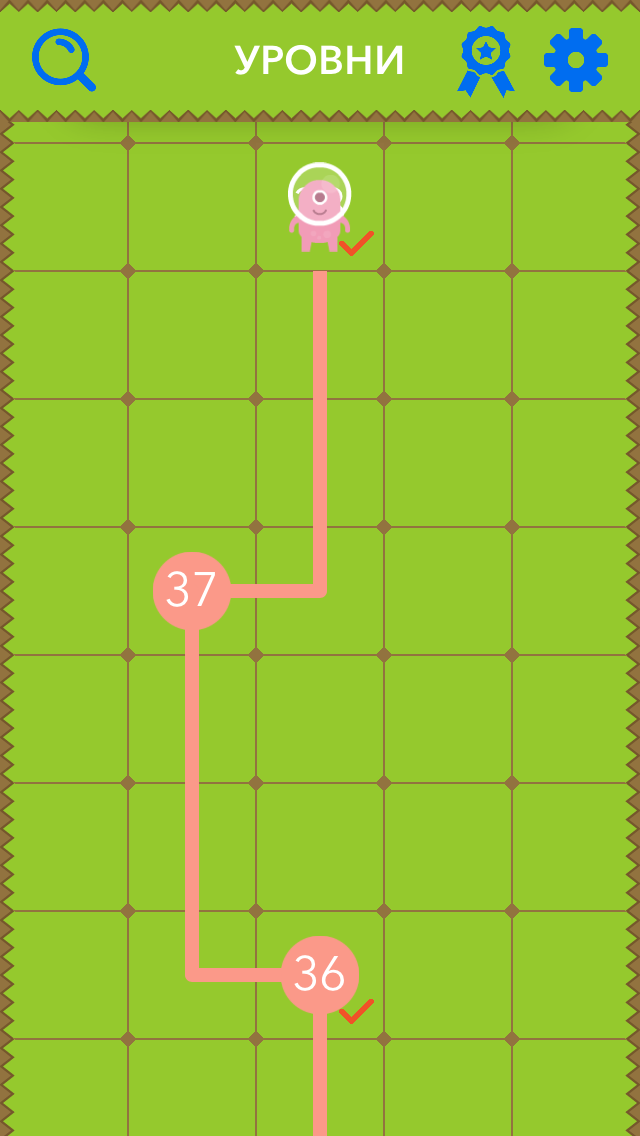


Рисунок 8 — Скриншот главного меню

### Окно «Достижения»

В правом верхнем углу отображается текущее количество алмазов (внутриигровая валюта). Окно можно пролистывать вниз. Если вознаграждение за достижение было получено, то рядом с достижением отображается красная галочка. Если вознаграждение ещё не получено, то рядом с достижением отображается текущее число, которое было получено в данном достижении, а также число, которое необходимо набрать, чтобы получить достижение. Из данного окна можно попасть в разделы: окно выбора уровня.



Рисунок 9 — Скриншот доски достижений

### Игровое меню

В данном окне находятся основные настройки: включение/выключение музыки/звуков, включение/отключение подсказок на игровом поле. Можно купить внутриигровую валюту, а также посмотреть рекламу за вознаграждение. В данном окне можно сыграть бонус-уровень на рекорд, в котором можно получить 1 алмаз за каждые 5 собранных звёзд.



Рисунок 10 — Скриншот меню

### Бонус–уровень

Данный уровень необязателен к прохождению. Существует два вида бонус–уровней: обычный бонус-уровень (переход из окна выбора уровня) и бонус–уровень на рекорд (переход из окна меню). В обычном бонус–уровне необходимо собрать определённое количество звёзд, чтобы получить небольшое вознаграждение, располагаются каждый 16 уровень в окне выбора уровня. Бонус–уровень на рекорд невозможно завершить, так как он бесконечен. За каждые 5 собранных звёзд игрок может получить 1 алмаз.



Рисунок 11 — Скриншот бонус-уровня

### Игровое поле

Основное поле, на котором происходит игровой процесс. Поле представляет квадрат размеров MxN. На игровом поле располагаются все объекты, основное взаимодействие пользователя происходит с игровым полем и объектами, которые находятся на нём. В верхней части находятся кнопки интерфейса: (слева-направо) меню, подсказка, доступное количество ходов, запустить уровень.



Рисунок 12 — Скриншот игрового поля

### Игровое поле с выбранной стратегией

Голубая линия — выбранный путь пользователем, по котором главный персонаж будет перемещаться после нажатия кнопки старт. При нажатии кнопки старт, синяя линия исчезает и главный персонаж начинает перемещаться, все остальные объекты начинают выполнять свои действия (пчела перемещается, бомба взрывается и т.п.).

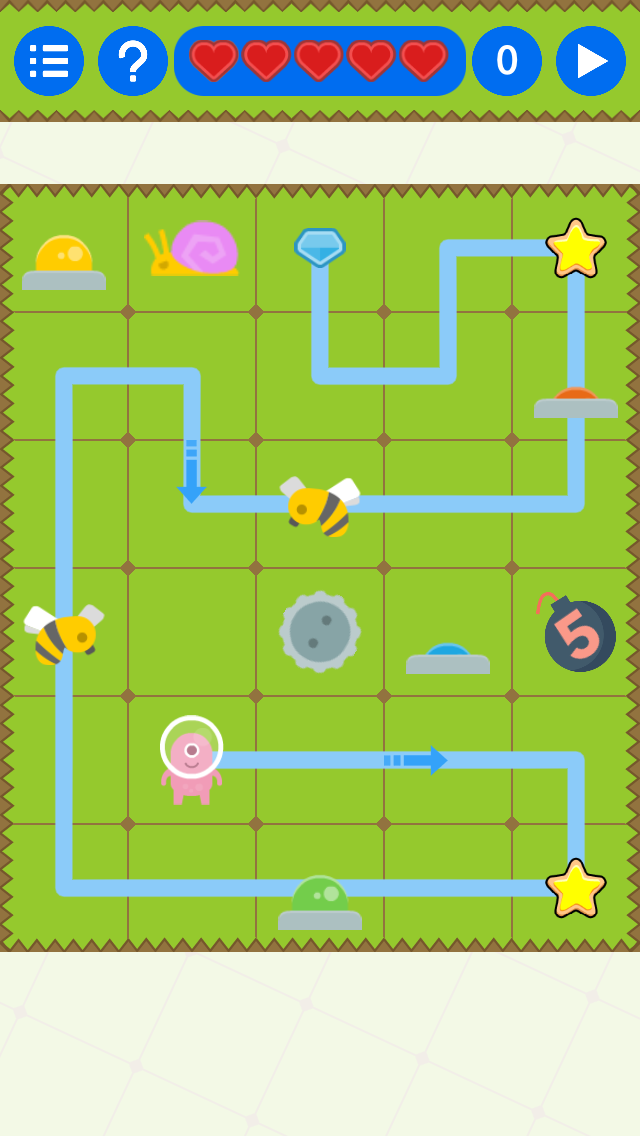


Рисунок 13 — Скриншот игрового поля с выбранной стратегией

### Модальное окно

Модальное окно появляется поверх всех других окон. Модальное окно появляется на некоторых экранах: игровое поле, меню выбора уровня. При клике на затемнённую область окно исчезает.

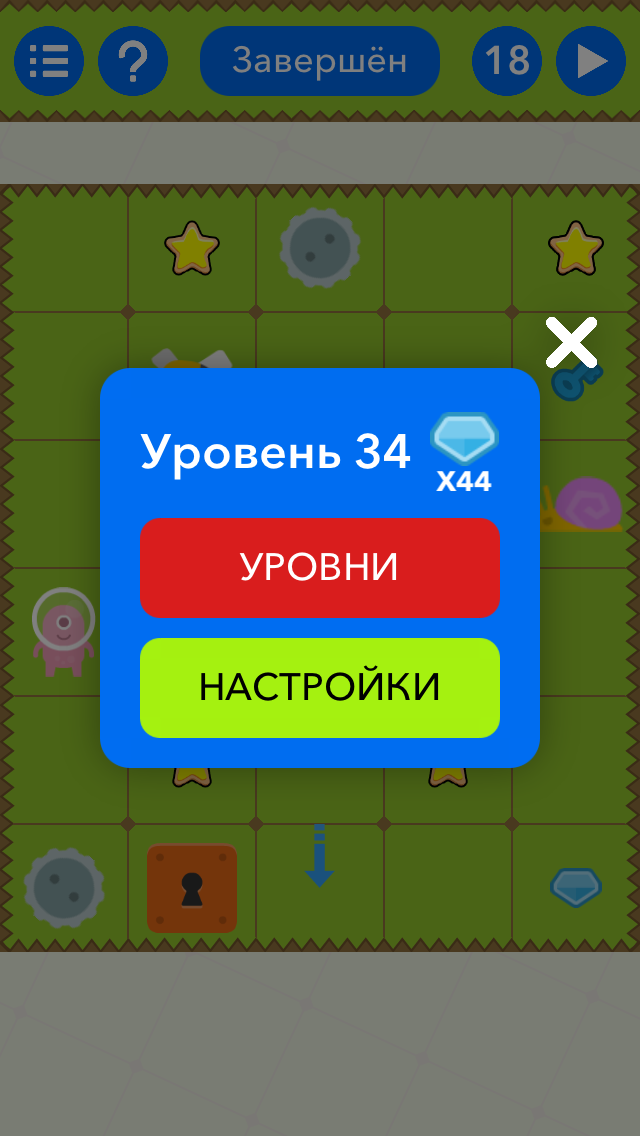


Рисунок 14 — Скриншот игрового поля с открытым модальным окном

### Окно покупки внутриигровой валюты

При переходе в окно «настройки», пользователь может за реальные деньги купить некоторое количество внутриигровой валюты. При нажатии на кнопку «купить за Nруб» у пользователя появляется всплывающее окно, которое показывает необходимую информацию о покупке. После нажатия кнопки «купить», происходит ожидание ответа от сервера, после чего покупка применяется.

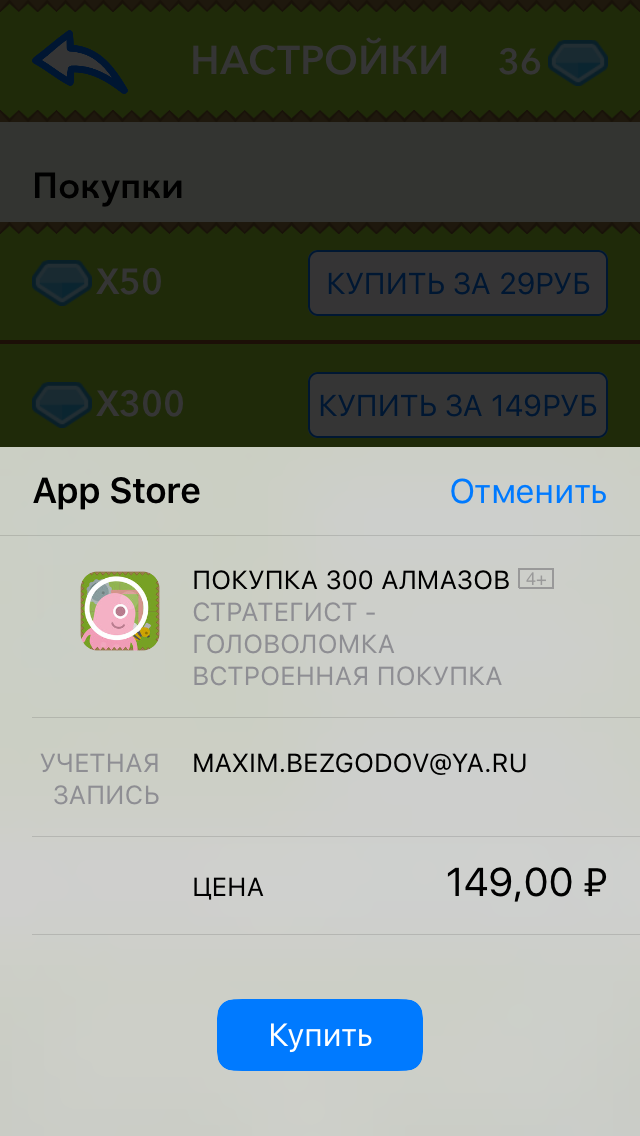


Рисунок 15 — Скриншот интерфейса покупки внутриигровой валюты

### Push-уведомления

В версии 1.07 были добавлены push-уведомления. Уведомления напоминают пользователям о том, что бесплатный алмаз уже доступен для получения. Уведомления были добавлены для повышения возврата пользователей в игру через определённые промежутки времени.

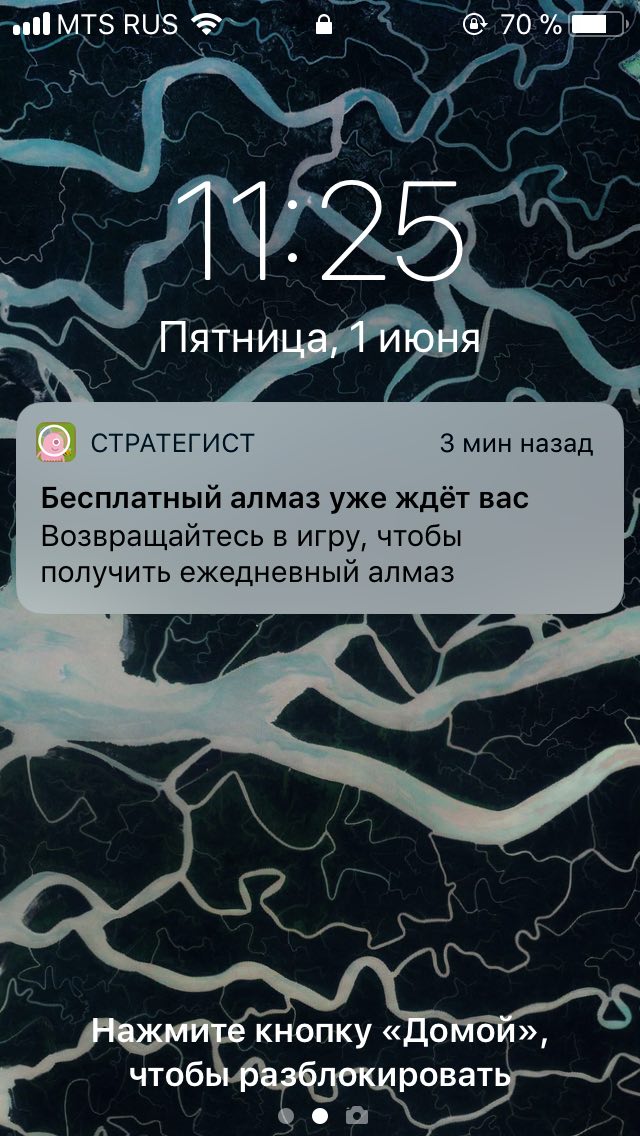


Рисунок 16 — Скриншот push-уведомления

# Реализация и тестирование

* За время выполнения работы было написано строк кода: 8100 (400 Кб);
* Количество экранов: 4 (достижения, уровни, игровое поле, меню);
* Количество сохранений (commit) в GIT: 90;
* https://github.com/bezgodov/thrup;
* В период с 1 марта по 16 июня приложение установило 347 пользователей;

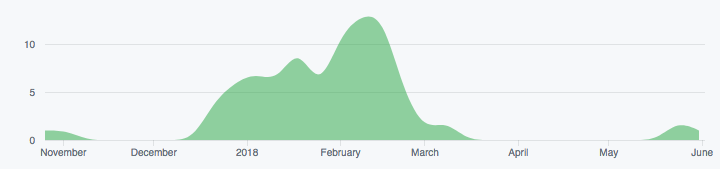


Рисунок 17 — Частота написания коммитов в git

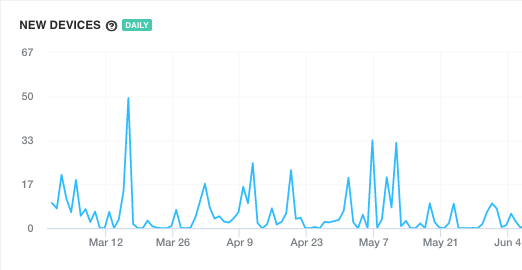


Рисунок 18 — Загрузка копий приложения по дням

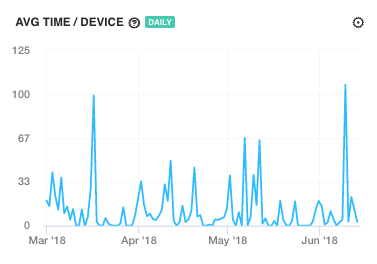


Рисунок 19 — Среднее время на устройство по дням

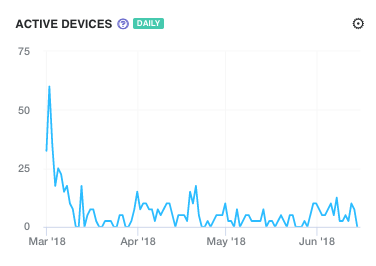


Рисунок 20 — Количество активных устройств по дням

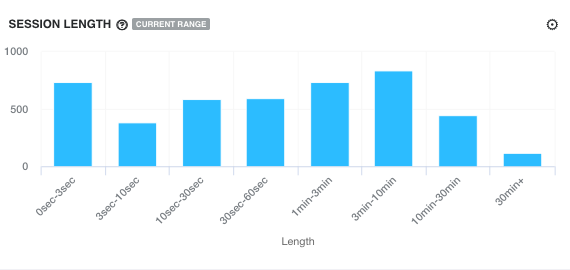


Рисунок 21 — Длина сессии

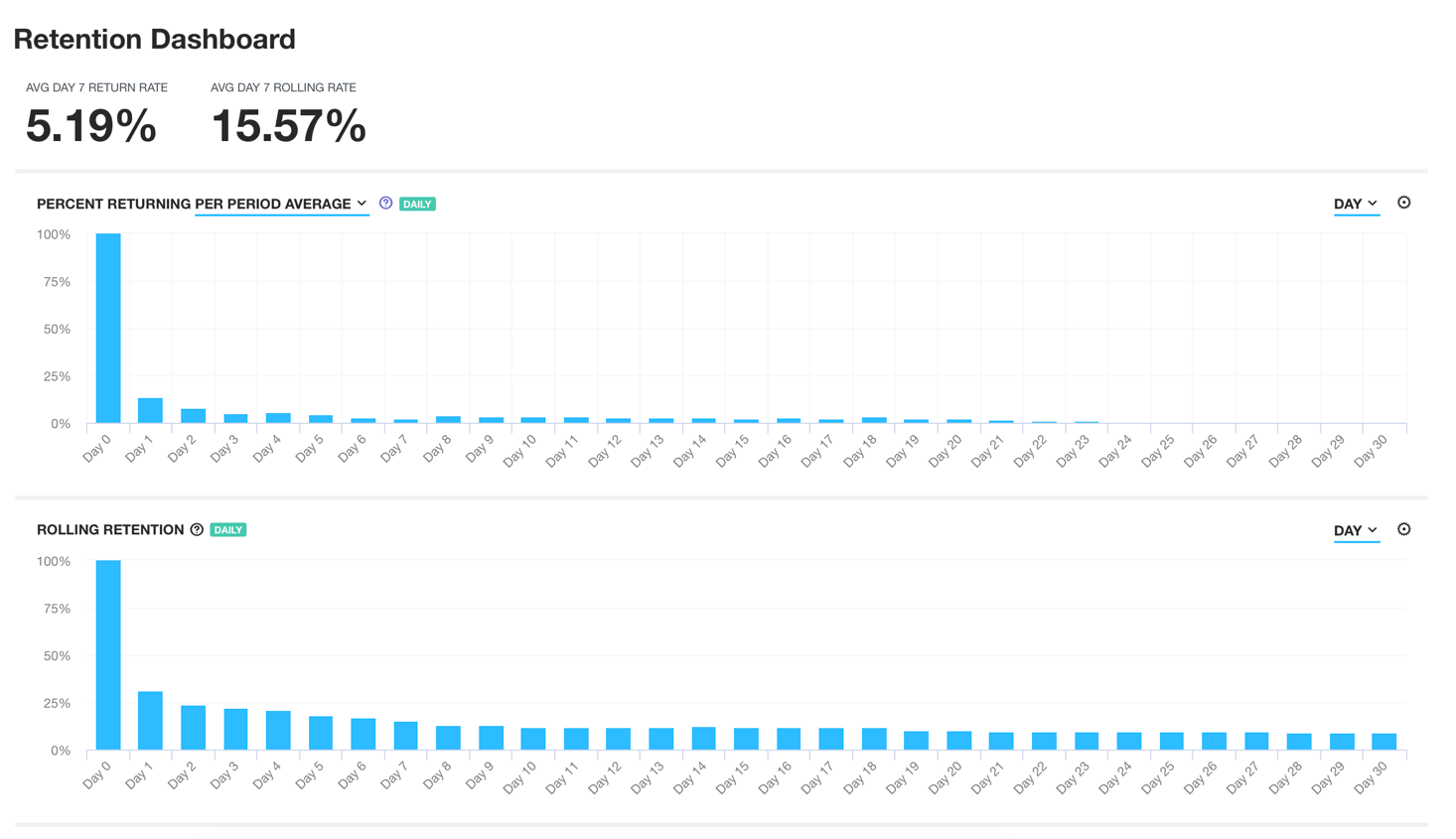


Рисунок 22 — Показатель ретенции

# Заключение

Таким образом, в процессе выпускной квалификационной работы были углублены общие знания разработки мобильных игр. Были углублены знания языка программирования *Swift*, интегрированной среды разработки *Xcode*. Были изучены методы получение и анализа статистических данных из мобильных игр. Были изучены методики монетизирования мобильных игр. Были углублены знания работы с системой контроля версий *GitHub*. Были изучены методики привлечения и удержания пользователей в мобильных играх.

Мобильная игра «Стратегист» была выложена в интернет-магазин приложений *AppStore* 1 марта 2018 года. За промежуток с 1 марта по 16 июня 347 человек установили игру. Каждый 17 пользователь заходил в игру, как минимум, дважды в течении 7 дней. Было выпущено 9 обновлений. Средняя оценка в *AppStore* – 4.9.

На данный момент в игре 38 уровней. В дальнейшем могут быть добавлены новые локации, уровни, а также новые объекты. Новые локации могут получить новый дизайн, отличный от текущего, чтобы разнообразить пользовательский интерфейс. Могут быть добавлены новые способы монетизации – разблокировка новых локаций, новых персонажей.

# Список литературы

1. **Хохлова Дарья** Введение в геймдизайн: Разработка и дизайн головоломок в видеоиграх [В Интернете] // vc.ru. - 21 07 2016 г.. - https://vc.ru/17028-gamedesign-challenges-3.
2. **Annie App** В 1-м кв. 2018 г. вновь побиты мировые рекорды магазинов приложений [В Интернете] // App Annie. - 9 Апреля 2018 г.. - https://www.appannie.com/ru/insights/market-data/q1-2018-apps-record-downloads-spend/.
3. **Annie App** 10 главных прогнозов для индустрии приложений на 2018 год [В Интернете] // App Annie. - 05 12 2017 г.. - https://www.appannie.com/ru/insights/market-data/predictions-app-economy-2018.
4. **[App2Top** Главные прорывы мобильного рынка: март 2018 [В Интернете] // App2Top. - 18 04 2018 г.. - http://app2top.ru/industry/glavny-e-prory-vy-mobil-nogo-ry-nka-mart-2018-119259.html.
5. **AppTractor** App Annie: обзор игровой индустрии 2017 [В Интернете] // AppTractor . - 13 Марта 2018 г.. - https://apptractor.ru/info/analytics/app-annie-obzor-igrovoy-industrii-2017.html.
6. **Churylov Maksym** Swift vs. Objective-C Comparison. What To Choose? [В Интернете] // MindK. - 06 06 2018 г.. - https://www.mindk.com/blog/swift-vs-objectivec.
7. **Kasabulatov Akim** App Store: Chanquan [В Интернете] // Apple (Россия) – Официальный сайт. - 06 06 2018 г.. - https://itunes.apple.com/ru/app/chanquan/id1362776234.
8. **LLC Big Duck Games** App Store: Flow Free [В Интернете] // Apple (Россия) – Официальный сайт. - 06 06 2018 г.. - https://itunes.apple.com/ru/app/flow-free/id526641427.
9. **nezabyvaemo** Обзор рынка мобильных приложений: стоит ли игра свеч? [В Интернете] // Habr. - 04 06 2015 г.. - https://habr.com/post/291238.
10. **Playdots Inc.** App Store: Two Dots [В Интернете] // Apple (Россия) – Официальный сайт. - 06 06 2018 г.. - https://itunes.apple.com/ru/app/two-dots/id880178264.
11. **Quora** What are the major differences between Swift and Objective C? [В Интернете] // Quora. - 12 08 2017 г.. - https://www.quora.com/What-are-the-major-differences-between-Swift-and-Objective-C.
12. **Simple Machine LLC** App Store: Smove [В Интернете] // Apple (Россия) – Официальный сайт. - 06 06 2018 г.. - https://itunes.apple.com/us/app/smove/id968818637.
13. **Soft Creative** App Store: Водопровод [В Интернете] // Apple (Россия) – Официальный сайт. - 06 06 2018 г.. - https://itunes.apple.com/ru/app/water-pipes-puzzle-game/id1341680606.
14. **Wijman Tom** Мобильные игры занимают более половины всего игрового рынка [В Интернете] // IXBT.com. - 02 05 2018 г.. - https://www.ixbt.com/news/2018/05/02/mobilnye-igry-zanimajut-bolee-poloviny-vsego-igrovogo-rynka.html.
15. **Боголюбов Леонид** Средний пользователь iPhone в 2017 году потратил на покупки в приложениях $58 [В Интернете] // AppTractor. - 16 Апреля 2018 г.. - https://apptractor.ru/measure/app-store-analytics/sredniy-polzovatel-iphone-v-2017-godu-potratil-na-pokupki-v-prilozheniyah-58.html.