Государственное Образовательное Учреждение

“Коми Республиканский Лицей при Сыктывкарском Государственном Университете”,

“Детский технопарк Кванториум г. Сыктывкар”

**СЕРВИС ДЛЯ РАЗРАБОТКИ И ОЦЕНКИ ПРОГРАММНОГО КОДА НА ПЛАТФОРМЕ ANDROID ДЛЯ ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ И ИЗУЧЕНИЯ АЛГОРИТМОВ С ИСПОЛЬЗОВАНИЕМ ВОЗМОЖНОСТЕЙ ГЕЙМИФИКАЦИИ ПРОЦЕССА ПРАКТИЧЕСКОГО ПРИМЕНЕНИЯ НАВЫКОВ, РАЗРАБОТАННЫЙ НА ЯЗЫКАХ ПРОГРАММИРОВАНИЯ JAVA И PYTHON-3**

Исполнитель:

Размыслов К.В.

11 класс (51группа)

Руководители:

Мишарин Н.Д.

преподаватель курсов Sumsung IT School (Детский Технопарк Кванториум)

Дуркин О.Л.

преподаватель информатики (ГОУ “КРЛ при СГУ”)

Оглавление

[Введение 2](#_Toc59400696)

[Процесс разработки и принцип работы 2](#_Toc59400697)

[Концепция 2](#_Toc59400698)

[Состав клиент-серверного приложения 3](#_Toc59400699)

[Работа Серверной части проекта. Разработка 4](#_Toc59400700)

[Работа Клиентской части проекта. Разработка 5](#_Toc59400701)

[Игровой процесс 5](#_Toc59400702)

[Итог Разработки 6](#_Toc59400703)

[Перспективы Развития 6](#_Toc59400704)

[Источники 7](#_Toc59400705)

Введение

Системы и способы геймификации образования в наши дни претерпевают стремительные изменения, причиной же тому является различия психологии детей и подростков, в особенности учеников средней и старшей школ. Сама по себе игровая форма образования удобна и вполне применима при обучении дошкольников и младшеклассников – она позволяет увлечь детей, не позволят им заскучать, однако на более старших это уже не работает – подростки не любят чувствовать себя ”детьми”, а сама форма примитивной игры их часто раздражает, преподаватели же и вовсе не считают данную форму передачи знаний и навыков средне- и старшеклассникам эффективной. Поэтому идея о геймификации образования во время своего появления была воспринята как учителями, так и учащимися весьма негативно, особенно в свете постоянно усложняющихся ЕГЭ и ОГЭ.

Решением же такой проблемы, может служить непосредственно частичная геймификация, как мне кажется, более продвинутая – такой способ уже используется: например, на уроках иностранных языков мы часто используем сервис “Kahoot!“.

Примеры использования игр для изучения алгоритмов и программирования уже существуют, практически все из них имеют исключительно web-характер. Одним из крупнейших примеров является сервис CodinGame ([www.codingame.com](http://www.codingame.com)), поддерживающий целый ряд языков программирования (на выбор пользователя). Сервис позволяет пользователю выбрать любую игру из широкого списка и отослать свое “решение” – программный код, исполняемый в качестве ИИ для аватара пользователя. Имеется так же и рейтинг участников. Сервис постоянно развивается, добавляются новые режимы и т.д.

Процесс разработки и принцип работы

Концепция

Концепция проекта такова: пользователь Android-устройства имеет возможность выбрать некую задачу-игру и “решить” ее; решением же является программный код (в данной версии проекта только на языке Python, что является важным направлением будущего расширения возможностей сервиса в пользу мультиязыковой платформы с возможностью выбора компилятора и со-отв. языка вне зависимости от игры; о возможностях улучшения и расширения будет сказано в конце документа). Сам код отправляется на сервер и привязывается к сущности “Match” (матч), по которой можно будет отследить результат решения “игры”. Каждый матч проигрывается и определенным образом логируется в JSON-документ так, что последующее воспроизведение матча в набор изображений будет однозначным и соответствующим действиям каждого игрока, участвовавшего в матче. После вычисления лога пользователь имеет возможность получить данный лог, который преобразуется в набор изображений, количество которых соответствует числу проведенных ходов до завершения игры.

В свою очередь, все матчи привязаны к сущностям “Hub” (хаб), каждый хаб имеет в себе некое число пользователей, которые и между собой соревнуются. (игры могут быть как на одного игрока, так и на несколько – но все пользовательские решения привязаны к хабам) Соревнование характеризуется разным числом очков, начисляемых в зависимости от качества предложенного пользователем решения.

Состав клиент-серверного приложения

Сервис представляет собой сервер и клиент. Сервер имеет в себе следующие составляющие:

* Базу данных (sqlite3)
* Сервис аунтефикации пользователя и передачи ему ответов на запросы, имеет следующие задачи и возможности:
  + Аунтефикация
  + Отправка информации об играх и матчах
  + Получение пользовательских решений
  + Генерация матчей
* Сервис оценки и обработки игровых матчей
* Сервис загрузки игр, который предоставляет основные модули создателям игр и позволяющий отправить их на проверку администрации

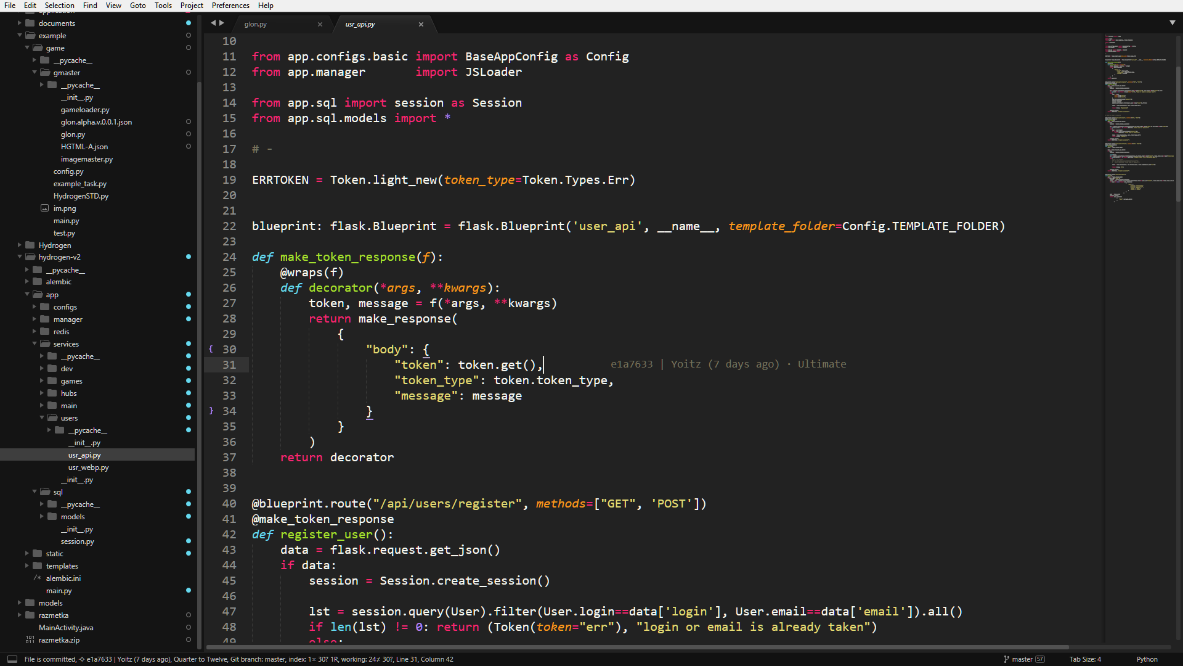
Серверный код написан на языке Python 3 с использованием микрофреймворка Flask и системы управления базами данных (фреймворка) SQLAlchemy. Выбор был сделан из-за простоты использования, установки и поддержки. “Песочница” – так называется программа, запускающая код других программ с ограничением их возможностей (в нашем случае сервис по просчету матчей), так же реализована на Python 3.

Клиент же обладает следующими функциями и составляющими:

* Формы регистрации и авторизации
* Автоматический вход пользователя и запоминание регистрации (входа) через систему Auth- и Refresh- токенов (сущность Token – в простейшем виде является парой из строки 16-ричных символов определенной длины и строки имени типа, тип определяет предназначение токена и его время жизни на сервере)
* Формы выбора и создания хаба и матча
* Локальный редактор кода
* Визуализатор матчей

Android-приложение (клиент) написан на языке программирования Java в IDE Android Studio. Выбор как языка, так и среды разработки был обоснован их распространённостью и наличием достаточного количества учебных материалов по ним.

Таким образом, основной идеей является создание клиент-серверного приложения, которое позволяет пользователям практиковаться в программировании и использовать алгоритмы на практике в игровой форме, путем написания своего “ИИ” (код пользователя, призванный решить задачу, так же именуется “контроллером” (“controller”) ).

Работа Серверной части проекта. Разработка

Сервер реализован на микрофреймворке Flask, что подразумевает собой особую форму написания сервера. Все сервисы приложения имеют свою папку и свой blueprint – список функций, привязанных к определенным URL: т.е. мы имеем разделение разных частей кода, что позволяет удобно расширять и модифицировать приложение без сильных изменений уже имеющегося кода. Так же это позволяет разместить сервисы на разных машинах (хостингах), при таковой необходимости. Регистрация URL и их привязка к соответствующим функциям происходит при помощи Python-декораторов.

Рисунок 1. фрагмент кода сервера

Каждый сервис имеет возможность обращения к БД. Обращения, как и вся система работы сервера с БД, основаны на SQLAlchemy – все таблицы и зависимости описаны в соответствующих классах-моделях, а возможности SQLAlchemy позволяют избежать прямых SQL-запросов. Так же они позволяют достаточно просто сменить локальную sqlite-базу данных на более продвинутую, например на PostgreSQL или MySQL, конечно, при желании перехода на нереляционные БД (Redis, MongoDB) необходимы достаточно серьезные изменения в структуре (по этой причине была отвергнута идея о использовании в проекте Redis для хранения некоторой части данных (в том числе токенов) – система из нескольких разных БД громоздка и неудобна для учебного проекта).

Для написания кода сервера был использован редактор кода Sublime Text 3.

О принципе работы игровой системы можно узнать из приложенной к документу схемы №1 (game\_schematic.pdf)

Работа Клиентской части проекта. Разработка

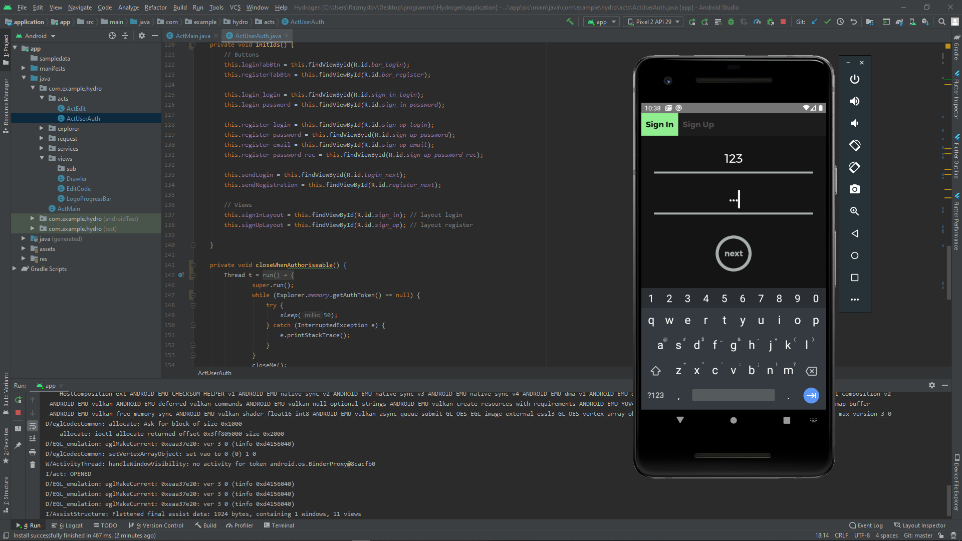


Рисунок 2. логотип и иконка приложения

Рисунок 3. процесс авторизации в приложении при использовании эмулятора

Приложение включает в себя несколько Activity, а так же классы, отвечающие за запросы к серверу и работу с Shared Preferences (способ хранения данных приложения в ОС Android в закрытом от других приложений виде, в данном случае используется для хранения токенов). После ввода логина и пароля клиент получает Refresh-токен – он и хранится в постоянно памяти и имеет определенный срок жизни без обновления, после чего получает временные Login-токены, которые и используются как “цифровая подпись” каждого пользователя.

Интерфейс имеет верхнее меню, которое позволяет переключаться между вкладками одной активности, что позволяет упростить работу с ними (вместо нескольких мы создаем лишь одну). После входа, пользователь может либо создать свой хаб, либо уже тот, в который пользователь вошел, либо войти в новый. Так же доступен список игр для создания своих хабов. Хаб может быть публичным и приватным; приватный хаб потребует пароль перед входом. (вход характеризуется созданием элемента в таблице Connections в БД на сервере и появлением соответствующего пункта в меню приложения). Отображение списков реализовано через RecycleView и ViewHolder.

После написания кода игрок отправляет его на сервер и генерирует матч, после чего получает лог игры и имеет возможность посмотреть ход матча пошагово.

Игровой процесс

Игры представляют собой задачи, имеющие возможность графической интерпретации в виде 2D – карты, состоящей из полей, каждое из которых имеет свою текстуру. Каждый матч содержит в себе N тиков – количество ходов, ровно столько карт можно будет посмотреть после окончания матча.

Карта имеет следующую структуру: имеется 3-х мерный массив полей, назовем первые два его измерения – сторонами карты, а третье – слоем. Во время генерации графического представления карты слои накладываются друг на друга, учитывая возможную прозрачность текстур. На данный момент не предусмотрены сущности эффектов или иных анимаций, однако их можно симулировать, используя слоистую структуру карты. Соответственно, во время проецирования 3-х мерная карта полей превращается в 2-х мерную карту пикселей (во время проецирования берутся текстуры каждого поля и накладываются соответственно на место клетки поля), которая может беспрепятственно преобразоваться в изображение. Вместо передачи даже такой карты, строится сигнатура – специальным образом сгенерированный json-файл, хранящий в себе карту на нулевом тике, а также все последующие изменения пикселей в соответствующие тики.

Итог Разработки

Были частично разработаны как android-приложение, так и сервер. Общая разработка находится на финальной части: не реализован только непосредственный просчет матчей, при наличии отдельно функционирующей системы игры (на данный момент нет системы создания матчей и их просчета), а также не реализован графический менеджер для Android-приложения, который будет преобразовывать сигнатуры матчей в изображения.

Были изучены технологии web- и android- программирования, системы организации проектов, их грамотной структуризации.

Перспективы Развития

Во-первых, конечно, необходимо закончить текущую стадию проекта, после чего возможны следующие расширения:

* Добавление статистики в хабе
* Возможность игроков создания матчей с конкретными игроками
* Введение общеигровой статистики
* Улучшения внешнего вида интерфейса Android-приложения
* Оптимизация приложения, переход на систему Callback`ов для отправки запросов и получения от них ответа
* Создание web-интерфейса для разработчиков игр
* Создание пакета библиотек, для разработчиков игр

Источники

Eine lange Schlange riegelzich um eine lange Stange