НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра автоматизованих систем обробки інформації і управління

**КУРСОВА РОБОТА**

з дисципліни «Технології створення програмних продуктів для мобільних платформ»

на тему: «ConquerorApp»

Студента 3 курсу

ФІОТ ІП-41

напряму підготовки 050103

«Програмна інженерія»

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Керівник: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Національна оцінка: \_\_\_\_\_\_

Кількість балів: \_\_\_\_\_\_\_\_\_\_

Оцінка ECTS: \_\_\_\_\_\_\_\_\_\_\_\_

Члени комісії

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(підпис) (вчене звання, науковий ступінь, прізвище та ініціали)

Київ, 2017

Національний технічний університет України

“Київський політехнічний інститут”

Факультет (інститут) інформатики та обчислювальної техніки

Кафедра автоматизованих систем обрабки інформації і управління

Освітньо-кваліфікаційний рівень бакалавр  
Напрям підготовки 6.050103 «Програмна інженерія»

***ЗАВДАННЯ***

НА КУРСОВУ РОБОТУ СТУДЕНТУ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(прізвище, ім’я, по батькові)*

1. Тема роботи: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Керівник роботи: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)*

1. Строк подання студентом роботи: \_\_\_\_\_\_\_\_\_\_\_\_\_\_.
2. Вхідні дані до роботи  
   - засоби розробки для платформи (назва платформи)  
   - мова програмування: (мова програмування)  
   - бібліотеки: (перерахувати використані бібліотеки / SDK)
3. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно   
   - огляд засобів розробки застосунку для платформи (назва платформи), бібліотек (перерахувати бібліотеки)  
   - розробка і тестування розробленого застосунку (назва застостунку)
4. Дата видачі завдання: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.

*КАЛЕНДАРНИЙ ПЛАН*

| № з/п | Назва етапу виконання КР | Строк виконання етапу КР |
| --- | --- | --- |
| 1 | Виконання розділу 1 |  |
| 2 | Виконання розділу 2 |  |
| 3 | Виконання розділу 3 |  |
| 4 | Оформлення КР |  |
| 5 | Перевірка КР викладачем |  |
| 6 | Захист КР |  |

**Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*(підпис) (прізвище та ініціали)*

**Керівник роботи \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

*(підпис) (прізвище та ініціали)*

# ЗМІСТ

ВСТУП…………………………………………………………………………………..…3

# РОЗДІЛ 1. ЗАПИТИ ЗАЦІКАВЛЕНИХ ОСІБ………………………………………….5

1.1. Вступ……………………………………………………………………………..5

1.2. Короткий огляд продукту………………………………………………………5

1.3. Ділові правила…………………………………………………………………...5

1.4. Сценарії використання……………………………………………………...…..6

1.5. Функціональність………………………………………………………….……7

1.6. Надійність………………………………………………………………….…….8

1.7. Цільова аудиторія……………………………………………………………….8

1.8. Використані під час розробки технології……………………………….……..8

# РОЗДІЛ 2. РОЗРОБКА ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ……………………...9

2.1. Загальна схема прецедентів……………………………………………….……9

2.2. Опис основних прецедентів………………………………………………...…10

2.3. Діаграма бізнес-сутностей……………………………………………….……13

# РОЗДІЛ 3. РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ…………………………….....15

### 3.1. Структура бази даних………………………………………………………….15

### 3.2. Спеціфікація DAO класів…………………………………………………..….19

### 3.3. Класи activity………………………………………….…………………..……24

3.4. Допоміжні класи…………………………………..……………...……………30

# РОЗДІЛ 4. ІЛЮСТРАЦІЯ РОБОТИ ПРОГРАМИ……………….…………….………34

4.1. Запуск за стосунку……………………………………………………….…….34

4.2. Пошук закладів харчування………………………….………………..………35

4.3. Перегляд детальної інформації……………………………..…….....………..39

ВИСНОВКИ………………………………………………………..…….………....……41

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ……………………………..………....….43

# ДОДАТКИ…………………………………………………………………………..……44

# ВСТУП

Людські потреби бути завжди в курсі справ виводять інформаційні технології на пряму лінію по створенню все нових девайсів і гаджетів. Незручність експлуатації комп'ютерів і ноутбуків зумовлює появу різних міні-комп'ютерів, смартфонів і комунікаторів, в основі яких лежить все та ж операційна система. Лідируючі позиції на сьогоднішній день займають платформи Android і iPhone. За новинами сучасного ринку мобільних телефонів дуже тяжко слідкувати, тому що швидкість створення нових технологій на розробок досягла рекордних показників.

Очевидно, що мобільний телефон вже давно перестав бути просто засобом спілкування. Звичайний розмова по телефону поступово стає другорядною функцією, зникаючи в величезному наборі функцій, що реалізуються мобільним телефоном. Слухати музику, фотографувати, грати, працювати, спілкуватися за допомогою месенджерів... Цей список можна продовжувати нескінченно. Мобільний телефон став багатофункціональним пристроєм, що дозволяє людині користуватися практично всіма сучасними технологіями.

Таким чином, варто визнати, що майбутнє ПК – у портативній, легкій та функціональній техніці – планшетах, електронних книжках, нетбуках і смартфонах – і все це в основному працює саме на операційній системі Android. Згідно зі статистикою, саме Android лідирує зараз на ринку смартфонів, займаючи на ньому більше половини всього обсягу продажів. Таким чином розробка додатків під платформу Android на сьогоднішній день вкрай актуальна.

Android - операційна система для комунікаторів, планшетних комп'ютерів, цифрових програвачів, цифрових фоторамок, наручних годинників, нетбуків і смартбуків, що базується на ядрі Linux. Основною мовою для розробки служить Java.

У компанії Google виділяють кілька основних переваг, що відрізняють пристрої на базі платформи Android від аналогічних продуктів:

- Відкритість - Android дозволяє отримати доступ до основних функцій мобільних пристроїв за допомогою стандартних викликів API;

- Руйнування кордонів - можна об'єднувати інформацію з інтернету з даними телефону, наприклад контактною інформацією або даними про географічне положення, щоб отримати нові можливості;

- Рівноправність додатків - для Android немає різниці між основними додатками телефону і стороннім програмним забезпеченням - можна змінити навіть програму для набору номера або заставку екрана;

- Швидка і легка розробка - в SDK є все, що потрібно для створення і запуску додатків Android, включаючи імітатор справжнього приладу і розширені інструменти відладки;

Мобільний додаток «Система пошуку закладів харчування» виконує функцію визначення найбільш оптимальних закладів харчування за критеріями, що були визначені користувачем.

Метою даної роботи є розробка мобільного за стосунку, що являє собою гру, в якій перемога досягається шляхом відповідей на питання з різних сфер.

Темою даної роботи є розробка програмного забезпечення для планформи Android.

Актуальність даної роботи полгялає у необхідності створення за стосунку, що дозволить весело та користно провести вілний час.

# РОЗДІЛ 1

# ЗАПИТИ ЗАЦІКАВЛЕНИХ ОСІБ

### Вступ

У даному документі описуються запити зацікавлених осіб, якими є: гравці(користувачі, що пройшли реєстрацуію), які будуть гарти.

**1.1.1. Мета**

Метою даного документу є визначення основних вимог до функціональності, продуктивності, експлуатаційної придатності системи, а також визначення бізнес-правил і технологічних обмежень, що висуваються до предмету розробки.

**1.1.2. Контекст**

Перелік вимог у даному документі є основою технічного завдання для розробки гри.

### Короткий огляд продукту

Дана система виконує функцію розважання користувача.

Доступ до системи мають:

- Користувач (людина, що пройшла реєстрацію);

- Адміністратор (фізична особа, що обслуговує систему).

### Ділові правила

#### Призначення системи

Основними призначеннями системи є розважання користувача.

#### Політика взаємовідносин з клієнтом

Клієнтом, що використовує систему може бути фізична особа, яка може використовувати основні функціональні здатності системи – весело та корисно проводити час.

#### Характеристика ділового процесу

Керівництво системою здійснюється командою адміністраторів, що можуть виконувати наступні дії:

- додавати нові запитання

### Сценарії використання

**1.4.1. Бізнес-сценарій участі в грі**

Учасники: Користувач, система.

Передумова: Користувач увійшов до застосунку.

1. Користувач обирає функцію “Нова гра”.

2. Система надаю користувачу гру, коли система знайде необхіну кількість гравців.

3. Користувач чекає свого кроку.

4. Користувач обирає напрямок атаки.

5. Система надає питання задля успішної атаки.

6. Якщо користувач відповів правильно, система надає йому територію. Крок переходить до іншого гравця.

7. Якщо всі походили по сім разів, то гра завершуеться, система надає кожному гравцю результати бою, та відповідно змінює їх рейтинг; якщо не уінець гри, то повернутись до кроку 3.

Результат: Гра закінчена, рейтинг гравців змінився.

**1.4.2. Бізнес-сценарій реєстрування через обліковий запис «Вконтакте»**

Учасники: Користувач, система.

Передумова: Користувач не увійшов до застосунку.

1. Користувач натискає кнопку “авторизація”.

2. Користувач натискає кнопку “реєстрація”.

3. Користувач натискає кнопку “через ВК”.

4. Система запрошує у користувача данні аккаунту ВК.

5. Користувач заповнює необхідні поля та дозволяє застосунку використовувати його данні.

6. Користувач створює пароль.

7. Система реєструє нового користувача.

Результат: Система реєструє нового користувача.

### Функціональність

Головні вимоги, пред’явлені зацікавленими особами, щодо функціональності, відносяться до однієї категорії:

- Користувач;

*5. 1.Можливості для користувача*

Користувачу система надає свою основну функціональну здатність, а саме:

- грати в гру.

- реєстрування черех «Вконтакте».

### Надійність

Система спроектована так, щоб зловмистник ніяк не міг змінити ігрову ситуацію на пристроях інших користувачів.

**1.7. Цільова аудиторія**

Цільовою аудиторією даного за стосунку є будь-яка людина.

**1.8. Використані під час розробки технології**

Під час розробки мобільного додатку було використано наступні технології:

- Android SDK;

- MySQl;

- VK API.

# РОЗДІЛ 2

# РОЗРОБКА ІНФОРМАЦІЙНОГО ЗАБЕЗПЕЧЕННЯ

**2.1. Загальна схема прецедентів**

Загальна схема прецедентів (рисунок 1) показує можливі дії для різних акторів.

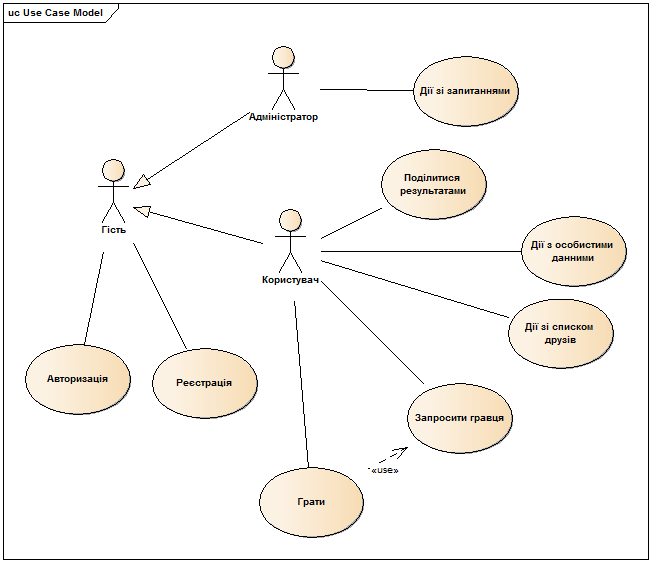


Рисунок 2.1 – Схема прецедентів

**2.2. Опис основних прецедентів**

Нижче описані прецеденти для акторів з вказаними передумовами, результатом, виключними ситуаціями, сутностями та детальним описом послідовності дій.

Таблиця 2.1

Прецедент №1. Почати нову гру

|  |  |  |  |
| --- | --- | --- | --- |
| ID | UC001 | | |
| Назва | Почати нову гру | | |
| Учасники | Користувач, система | | |
| Передумова | Користувач зайшов до програмного застосунку | | |
| Результат | Користувач весело проводить час, система змінює рейтенг користувача | | |
| Основний сценарій |  | Дії користувача | Дії системи |
| 1. | Користувач натискає кнопку «Нова гра» |  |
| 2. |  | Система знаходить вільних гравців, та починаю гру |
| 3. | Користувач починає участь в грі(обирає тереторію, та атакує) |  |
| 6. |  | Система надає користувачу питання та можливі відповіді дл них |
| 7. | Користувач обирає відповідь |  |
|  |  | Система надає користувачу тереторію, якщо він правильно відповів |
| 8. | Користувач, чекає свого кроку. |  |
|  |  | Ситема завершує гру, коли всі гравці зробили по сім кроків, та змінює рейтинг кожного гравця. |

Таблиця 2.2

Прецедент №2. Реєстрація, використовуючи аккаунт ВК

|  |  |  |  |
| --- | --- | --- | --- |
| ID | UC002 | | |
| Назва | Реєстрація, використовуючи аккаунт ВК | | |
| Учасники | Користувач, система | | |
| Передумова | Користувач має аккаунт ВК | | |
| Результат | Система реєструє користувача. | | |
| Основний сценарій |  | Дії користувача | Дії системи |
| 1. | Користувач натискає кнопку «авторизація» |  |
| 2. |  | Система відображає форму авторизації |
| 3. | Користувач натискає кнопку «реєстрація» |  |
| 4. |  | Система відображає форму реєстрації |
| 5. | Користувач натискає кнопку «через ВК» |  |
| 7. |  | Система відображає форму авторизацї через ВК та запрошує змогу до отримання данних аккаунту |
| 8. | Користувач заповнює необхідні поля та підтверджує можливість отримати системою данні аккаутну |  |
| 9. |  | Система надає форму створення паролю |
| 10. | Користувач створює пароль |  |
| 11. |  | Система реєструє нового користувача |
| Виняткові ситуації | Дані введені користувачем не пройшли валідацію | | |

# 

# РОЗДІЛ 3

**РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ**

### 3.1. Структура бази даних

У даному застосунку використовуються [реляційна система керування базами даних](https://uk.wikipedia.org/wiki/%D0%A0%D0%B5%D0%BB%D1%8F%D1%86%D1%96%D0%B9%D0%BD%D0%B0_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BA%D0%B5%D1%80%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F_%D0%B1%D0%B0%D0%B7%D0%B0%D0%BC%D0%B8_%D0%B4%D0%B0%D0%BD%D0%B8%D1%85) – MySQL. Дана система була обрана, оскільки MySQL - це найпоширеніша повноцінна серверна СУБД. MySQL дуже функціональна, що вільно розповсюджується СУБД, яка успішно працює з різними сайтами і веб додатками. Незважаючи на те, що в ній не реалізований весь SQL функціонал, MySQL пропонує досить багато інструментів для розробки додатків. Так як це серверна СУБД, додатки для доступу до даних, на відміну від SQLite працюють зі службами MySQL.

Також, кілька [процесів](https://uk.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)) або [потоків](https://uk.wikipedia.org/wiki/%D0%9D%D0%B8%D1%82%D1%8C) можуть одночасно без жодних проблем читати дані з однієї бази. Запис в базу можна здійснити тільки в тому випадку, коли жодних інших запитів у цей час не обслуговується; інакше спроба запису закінчується невдачею, і в програму повертається код помилки. Іншим варіантом розвитку подій є автоматичне повторення спроб запису протягом заданого інтервалу часу.

У комплекті постачання йде також функціональна клієнтська частина у вигляді виконуваного файлу sqlite3, за допомогою якого демонструється реалізація функцій основної бібліотеки. Клієнтська частина працює з [командного рядка](https://uk.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D0%BD%D0%B8%D0%B9_%D1%80%D1%8F%D0%B4%D0%BE%D0%BA), і дозволяє звертатися до файлу БД на основі типових функцій [ОС](https://uk.wikipedia.org/wiki/%D0%9E%D0%A1).

Особливості MySQL.

Швидкодія: завдяки внутрішньому механізму багатопоточності швидкодію MySQL вельми високо.

Безпека: досить високий рівень безпеки забезпечується завдяки базі даних mysql, що створюється при установці пакету і містить п'ять таблиць. За допомогою цих таблиць можна описати, який користувач з будь домену з якою таблицею може працювати і які команди він може застосовувати. Паролі, що зберігаються в базі даних, можна зашифрувати за допомогою вбудованої в MySQL функції password ().

Ліцензія: раніше ліцензування MySQL було трохи заплутаним; Зараз ця програма для некомерційних цілей поширюється безкоштовно.

Відкритість коду: завдяки цьому ви зможете самі додавати в пакет потрібні функції, розширюючи його функціональність так, як вам потрібно. До речі, за окрему плату для вас це можуть зробити і самі автори MySQL.

Надійність: творці MySQL потрудилися на славу: наскільки мені відомо, цей пакет досить стабільний і його важко вивести з ладу.

У розроблювальному застосунку схема бази даних має наступний вигляд:

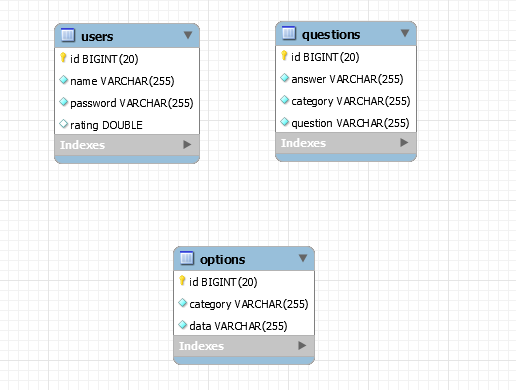


Рис 3.1 – Схема бази даних

Як можна побачити із схеми на рис. 3.1 база даних застосунку містить три табдиці. У даній таблиці зберігаються дані про заклади харчування, що доступні користувачам для пошуку.

Таблиця «users» використовується для зберігання зареєстрованих користувачів

Таблиця 3.1

Таблиця «users»

| **#** | **Ім’я** | **Тип** | **Null** | **Ключ** | **Автоінкремент** | **Опис** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Id | INTEGER | Ні | Так | Так | Ідентифікатор зареєстрованого користувача |
| 2 | Name | TEXT | Ні | Ні | Ні | Унікальний логін користувача |
| 3 | password | TEXT | Ні | Ні | Ні | Пароль, за яким виконується вхід в систему |
| 4 | Rating | NUMBER | Ні | Ні | Ні | Рейтинг користувача, що змінюється в процесі гри |

Таблиця 3.2

Таблиця «question» зберігає питання, що використовуються в грі.

Таблиця «question»

| **#** | **Ім’я** | **Тип** | **Null** | **Ключ** | **Автоінкремент** | **Опис** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Id | INTEGER | Ні | Так | Так | Ідентифікатор питання |
| 2 | Answer | TEXT | Ні | Ні | Ні | Відповідб на питання |
| 3 | Category | TEXT | Ні | Ні | Ні | Категорі питання, за якою будуть знаходитися варіанти відповідей |
| 4 | Question | TEXT | Ні | Ні | Ні | Текст питання |

Таблиця «options» використовується для зберігання варіантів відповідей на запитання.

Таблиця 3.3

Таблиця «users»

| **#** | **Ім’я** | **Тип** | **Null** | **Ключ** | **Автоінкремент** | **Опис** |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Id | INTEGER | Ні | Так | Так | Ідентифікатор можливої відповіді |
| 2 | Category | TEXT | Ні | Ні | Ні | Категорія можливої відповіді |
| 3 | data | TEXT | Ні | Ні | Ні | Можлива відповідь |

### 3.3. Класи activity

Роль контролерів у розроблювальному за стосунку відіграють класи із пакету activities, що відповідають за взаємодію користувача із за стосунком, а також за відображення даних користувачу.

Дія (Activity) являється компонентом програми, яка забезпечує інтерфейс з яким взаємодіє користувач. Вікно зазвичай заповнює весь екран, але може бути меншим і розміщуватись поверх інших вікон.

Програма зазвичай складається з багатьох дій, які слабо зв’язані між собою. Зазвичай, одна дія у програмі вказується як “головна”, яка представляється користувачу під час першого запуску програми. Кожна дія може запустити іншу дію для забезпечення додаткових функцій. Кожного разу, під час запуску нової дії (activity), попередня зупиняється, але система зберігає її у стеку, а нова вставляється зверху даного стеку і отримує фокус користувача. Стек відповідає базовому механізму “перший увійшов – перший вийшов”, отож, коли користувач завершив роботу з поточною дією і натискає кнопку “назад”, вона витягується з стеку і попередня дія продовжує свою роботу. Коли дія зупиняється через створення нової, вона сповіщується цією зміною свого стану через методи життєвого циклу дії.

Наведемо специфікацію класів пакету activities.

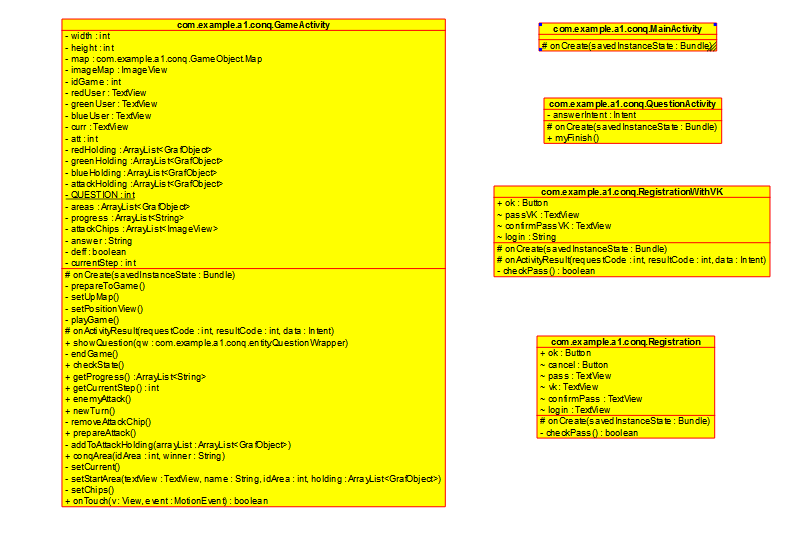


Рисунок 3.3 – Контролери

Наведемо нижче опис кожного із цих класів:

1. MainActivity

Даний клас відповідає за початок роботи за стосунку. Так, він наділяє кнопки обробниками. Завдяки цьому користувачеві надається можливість авторизуватися та почати нову гру. Дана функціональність забезпечуються наступним методом класу:

protected void onCreate(Bundle savedInstanceState) {

…

Button newGame = (Button) findViewById(R.id.newGame);

View.OnClickListener oclNewGame = new View.OnClickListener() {

@Override

public void onClick(View v) {

if (SingletonUser.getSingletonUser().getName()!=null){

new FindGame(MainActivity.this).execute();

}

}

};

newGame.setOnClickListener(oclNewGame);

Button quit = (Button) findViewById(R.id.quit);

View.OnClickListener oclQuit = new View.OnClickListener() {

@Override

public void onClick(View v) {

finish();

}

};

quit.setOnClickListener(oclQuit);

Button regAuto = (Button) findViewById(R.id.regauto);

View.OnClickListener oclRegAuto = new View.OnClickListener() {

@Override

public void onClick(View v) {

AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this);

builder.setTitle("Title");

Context context = MainActivity.this.getBaseContext();

LinearLayout layout = new LinearLayout(context);

layout.setOrientation(LinearLayout.VERTICAL);

final EditText inputLogin = new EditText(MainActivity.this);

final EditText inputPass = new EditText(MainActivity.this);

inputLogin.setInputType(InputType.TYPE\_CLASS\_TEXT | InputType.TYPE\_TEXT\_VARIATION\_PASSWORD);

inputPass.setInputType(InputType.TYPE\_CLASS\_TEXT | InputType.TYPE\_TEXT\_VARIATION\_PASSWORD);

layout.addView(inputLogin);

layout.addView(inputPass);

builder.setView(layout);

builder.setPositiveButton("OK", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

String l=inputLogin.getText().toString();

String p=inputPass.getText().toString();

new Login(l,p,MainActivity.this).execute();

}

});

builder.setNeutralButton("reg", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

Intent intent = new Intent(MainActivity.this, Registration.class);

startActivity(intent);

}

});

builder.setNegativeButton("Cancel", new DialogInterface.OnClickListener() {

@Override

public void onClick(DialogInterface dialog, int which) {

dialog.cancel();

}

});

builder.show();

}

};

regAuto.setOnClickListener(oclRegAuto);

}

# 2. Regestration

Даний клас відповідає за реєстрацію нових користувачів. Встановлює кнопкам обробники, які надають можливість користуватися застосунком. Використувує AsyncClass-и, про які буде написано нижче. Дана функціональність забезпечуються наступним методом класу:

protected void onCreate(Bundle savedInstanceState) {

ok = (Button) findViewById(R.id.ok);

View.OnClickListener oclNewGame = new View.OnClickListener() {

@Override

public void onClick(View v) {

if(checkPass()) {

new MyDownloadTask().execute();

}

}

};

ok.setOnClickListener(oclNewGame);

vk = (Button) findViewById(R.id.vkButton);

View.OnClickListener oclVK = new View.OnClickListener() {

@Override

public void onClick(View v) {

Intent intent = new Intent(Registration.this, RegistrationWithVK.class);

startActivity(intent);

}

};

vk.setOnClickListener(oclVK);

Button cancel = (Button) findViewById(R.id.cancel);

View.OnClickListener oclCancel = new View.OnClickListener() {

@Override

public void onClick(View v) {

finish();

}

};

cancel.setOnClickListener(oclCancel);

pass = (TextView) findViewById(R.id.pass);

confirmPass = (TextView) findViewById(R.id.confirmPass);

login = (TextView) findViewById(R.id.login);

}

Крім цього, даний клас містить службовий метод для перевірки співпадіння паролей:

- private boolean checkPass();

3. RegistrationWithVK

Даний клас відповідає за реєстрацію нових користувачів через сайт «ВКонтакте». Посилає запит до данних аккаунту користувача в «ВКонтакте». Встановлює кнопкам обробники, які надають можливість користуватися застосунком. Використувує AsyncClass-и, про які буде написано нижче. Дана функціональність забезпечуються наступним методом класу:

protected void onCreate(Bundle savedInstanceState) {

…

ok = (Button) findViewById(R.id.okvk);

View.OnClickListener oclNewGame = new View.OnClickListener() {

@Override

public void onClick(View v) {

if(checkPass()) {

new RegistrationWithVK.MyDownloadTask().execute();

}

}

};

ok.setOnClickListener(oclNewGame);

Button cancel = (Button) findViewById(R.id.cancelvk);

View.OnClickListener oclCancel = new View.OnClickListener() {

@Override

public void onClick(View v) {

finish();

}

};

cancel.setOnClickListener(oclCancel);

…

VKSdk.login(this);

}

Крім цього, даний клас містить службовий метод для перевірки співпадіння паролей:

- private boolean checkPass();

4. GameActivity

Данний клас відповідає за процес гри. Він знаходить гравців, відображає прогресс та синхронизує його між гравцями. Задля роботи гри було написано двигун гри, про який пійдеться пізніше. Використувує AsyncClass.

Крім цього, даний клас містить службовий методи, які виеористовуються задля піддержання роботи программи:

private void prepareToGame() – винує підготовку до гри

private void setUpMap() – виконує налаштування карти

private void setPositionView() – надає мождивість «зонам» відображити фішки гравців

private void playGame() – початок гри

protected void onActivityResult() – метод, що чекає відповіді гравця на запитання

public void showQuestion() – відображає питання

private void endGame() – кінець гри

public void checkState() – перевіряє зміни на полі

public void enemyAttack() – чекає хід іншого гравця

public void newTurn() – починаж новий хід

private void removeAttackChip() – прибирає з карти фішки, що позначають можливість атаки на відповідну територію

public void prepareAttack() – відображає фішки атаки

private void addToAttackHolding() – шукає території на які можна напасти

public void conqArea() – захоплення території

private void setStartArea() – визначає початкове місцеположення баз гравців

private void setChips() – відображає фішки гравців

public boolean onTouch() – оброблює натискання на карту

5. QuestionActivity

Даний клас відповідає за відображення питання та обробки відповіді на нього. Використовує AsyncClass-и. Дана функціональність забезпечуються наступним методом класу:

protected void onCreate(Bundle savedInstanceState) {

…

answerIntent = getIntent();

Button quit = (Button) findViewById(R.id.Q);

View.OnClickListener quitAnswer= new View.OnClickListener() {

@Override

public void onClick(View v) {

QuestionActivity.this.finish();

}

};

quit.setOnClickListener(quitAnswer);

final String question = answerIntent.getStringExtra("question");

final String answer1 = answerIntent.getStringExtra("a1");

final String answer2 = answerIntent.getStringExtra("a2");

final String answer3 = answerIntent.getStringExtra("a3");

final String answer4 = answerIntent.getStringExtra("a4");

TextView q = (TextView) findViewById(R.id.question);

q.setText(question);

Button a1 = (Button) findViewById(R.id.answer1);

Button a2 = (Button) findViewById(R.id.answer2);

Button a3 = (Button) findViewById(R.id.answer3);

Button a4 = (Button) findViewById(R.id.answer4);

a1.setText(answer1);

a2.setText(answer2);

a3.setText(answer3);

a4.setText(answer4);

View.OnClickListener sendAnswer= new View.OnClickListener() {

@Override

public void onClick(View v) {

Button b =(Button) findViewById(v.getId());

new CheckAnswer(QuestionActivity.this,b.getText().toString()).execute();

}

};

a1.setOnClickListener(sendAnswer);

a2.setOnClickListener(sendAnswer);

a3.setOnClickListener(sendAnswer);

a4.setOnClickListener(sendAnswer);

}

**3.4. Допоміжні класи**

Допоміжні класи знаходяться у пакеті async, gameObject, vk та відповідають за отримання інформації про стан системи, та за взаємодію із сервісами пристрою. Поговоримо про кожний пакет.

Нижче наведено діаграму класів async пакету:

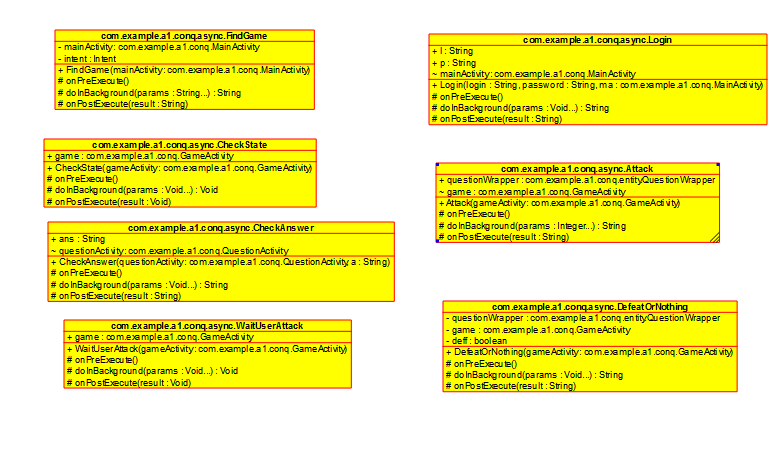


рис. 3.4 Класи пакету async

# Наведемо детальний опис даних класів:

1. Клас Attack

Даний клас при виконні створює новий потік та в ньому надає сереверу інформацію про атаку. Відбувається в методі doInBackground()

protected String doInBackground(Integer... params) {

try {

URL url = new URL("http://10.0.3.2:8080/rest/game/attackTerritory?gameId="+ SingletonUser.getSingletonUser().getCurrentGame()+"&username="+ SingletonUser.getSingletonUser().getName()+"&territoryNumber=" +params[0].intValue());

HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

try {

BufferedReader in = new BufferedReader(

new InputStreamReader(urlConnection.getInputStream()));

String inputLine;

StringBuffer response = new StringBuffer();

while ((inputLine = in.readLine()) != null) {

response.append(inputLine);

}

in.close();

Gson gson = new Gson();

questionWrapper = (gson.fromJson(response.toString(), QuestionWrapper.class));

Log.d("GAME","Attack with question: "+questionWrapper.getQuestion() );

}

finally {

urlConnection.disconnect();

}

}

catch (IOException e) {

e.printStackTrace();

}

return null;

}

2. Клас CheckAnswer

Даний клас при виконні створює новий потік та в ньому перевіряє відповідь користувача. Відбувається в методі doInBackground()

protected String doInBackground(Void... params) {

try {

URL url = new URL("http://10.0.3.2:8080/rest/game/sendAnswer?gameId="+

SingletonUser.getSingletonUser().getCurrentGame()+ "&username="+SingletonUser.getSingletonUser().getName()+"&answer="+ans);

HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

try {

BufferedReader in = new BufferedReader(

new InputStreamReader(urlConnection.getInputStream()));

String inputLine;

StringBuffer response = new StringBuffer();

while ((inputLine = in.readLine()) != null) {

response.append(inputLine);

}

in.close();

Gson gson = new Gson();

boolean b = (gson.fromJson(response.toString(), Boolean.class));

Log.d("GAME","ANSWER " + b);

Intent intent= questionActivity.getIntent();

intent.putExtra("result",b);

}

finally {

urlConnection.disconnect();

}

}

catch (IOException e) {

e.printStackTrace();

}

return null;

}

3. Клас CheckState

Даний клас при виконні створює новий потік та в ньому перевіряє дії інших користувачів і відображає їх. Відбувається в методі doInBackground

@Override

protected Void doInBackground(Void... params) {

try {

URL url = new URL("http://10.0.3.2:8080/rest/game/getLastChanges?gameId="+

SingletonUser.getSingletonUser().getCurrentGame());

HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

try {

BufferedReader in = new BufferedReader(

new InputStreamReader(urlConnection.getInputStream()));

String inputLine;

StringBuffer response = new StringBuffer();

while ((inputLine = in.readLine()) != null) {

response.append(inputLine);

}

in.close();

Gson gson = new Gson();

java.util.Map.Entry<Long, String> entry;

entry = (gson.fromJson(response.toString(), AbstractMap.SimpleEntry.class));

Number key = entry.getKey();

String value = entry.getValue();

if(value!=null){

game.conqArea(key.intValue(),value);

}

}

finally {

urlConnection.disconnect();

}

}

catch (IOException e) {

e.printStackTrace();

}

return null;

}4. Клас DefeatOrNothing

Даний клас при виконні створює новий потік та в ньому перевіряє чи не напав на нього інший гравець. Відбувається в методі doInBackground

# @Override

# protected String doInBackground(Void... params) {

# try {

# Log.d("GAME","execute DefeatOrNothing");

# URL url = new URL("http://10.0.3.2:8080/rest/game/checkMove?gameId="+ SingletonUser.getSingletonUser().getCurrentGame()+"&username="+

# SingletonUser.getSingletonUser().getName());

# HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

# try {

# BufferedReader in = new BufferedReader(

# new InputStreamReader(urlConnection.getInputStream()));

# String inputLine;

# StringBuffer response = new StringBuffer();

# while ((inputLine = in.readLine()) != null) {

# response.append(inputLine);

# }

# in.close();

# Gson gson = new Gson();

# questionWrapper = (gson.fromJson(response.toString(), QuestionWrapper.class));

# if(questionWrapper==null){

# Log.d("GAME","NOTHING");

# }else{

# Log.d("GAME","DEFF" + questionWrapper.getQuestion() );

# deff=true;

# }

# }

# finally {

# urlConnection.disconnect();

# }

# }

# catch (IOException e) {

# e.printStackTrace();

# }

# return null;

# }

5. Клас FindGame

Даний клас при виконні створює новий потік та в ньому винує пошук нової гри. Відбувається в методі doInBackground

# @Override

# protected String doInBackground(String... params) {

# try {

# URL url = new URL("http://10.0.3.2:8080/rest/game/getGame?username="+ SingletonUser.getSingletonUser().getName());

# HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

# try {

# InputStream response = urlConnection.getInputStream();

# Scanner s = new Scanner(response).useDelimiter("\\A");

# String result = s.hasNext() ? s.next() : "";

# Log.d("MAIN","id game:" + result);

# int id = Integer.parseInt(result);

# if(id>=0){

# SingletonUser.getSingletonUser().setCurrentGame(id);

# }

# }

# finally {

# urlConnection.disconnect();

# }

# }

# catch (IOException e) {

# e.printStackTrace();

# }

# return null;

# }

6. Клас Login

Даний клас при виконні створює новий потік та в ньому перевіряє надані користувачем login та password. Відбувається в методі doInBackground

# @Override

# protected String doInBackground(Void... params) {

# try {

# URL url = new URL("http://10.0.3.2:8080/rest/user/check?username="+l+"&password="+p);

# HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

# try {

# InputStream response = urlConnection.getInputStream();

# Scanner s = new Scanner(response).useDelimiter("\\A");

# String result = s.hasNext() ? s.next() : "";

# if(result.equals("true")){

# SingletonUser.getSingletonUser().setName(l);

# Log.d("LogIn",result+" "+l+" "+ p);

# }

# }

# finally {

# urlConnection.disconnect();

# }

# }

# catch (IOException e) {

# e.printStackTrace();

# }

# return null;

# }

7. Клас WaitUserAttack

Даний клас при виконні створює новий потік та в ньому чекає дію гравця. Відбувається в методі doInBackground

# protected Void doInBackground(Void... params) {

# try {

# URL url = new URL("http://10.0.3.2:8080/rest/game/waitUserAttack?gameId="+

# SingletonUser.getSingletonUser().getCurrentGame());

# HttpURLConnection urlConnection = (HttpURLConnection) url.openConnection();

# try {

# BufferedReader in = new BufferedReader(

# new InputStreamReader(urlConnection.getInputStream()));

# String inputLine;

# StringBuffer response = new StringBuffer();

# while ((inputLine = in.readLine()) != null) {

# response.append(inputLine);

# }

# in.close();

# Log.d("GAME","enemy attacked");

# }

# finally {

# urlConnection.disconnect();

# }

# }

# catch (IOException e) {

# e.printStackTrace();

# } return null;

# }

Нижче наведено діаграму класів gameObject пакету:

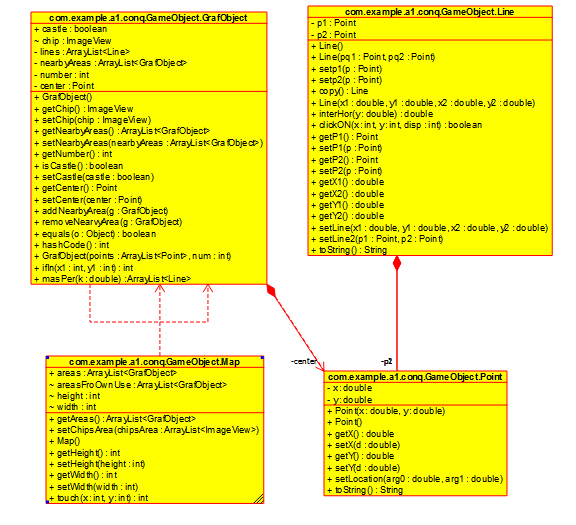


рис. 3.5 Класи пакету gameObject

# Данний пакет відповідає за «двигун» гри. Він знаходить відповідну до натискання вобласть.

# Наведемо детальний опис даних класів:

1. Клас GrafObject

Являє собою программну імітацію області. Складається зі списку ліній

1. Клас Line

Являє собою програмну імітацію лінії. Складається з точок.

1. Map

Являє собою програмну імітацію карти. Складається зі списку обрастей.

1. Point

Являє собою програмну імітацію точки. Складається з пари координат X, Y.

Нижче наведено діаграму класів vk пакету:

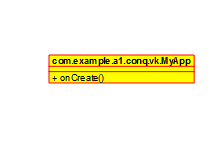


рис. 3.6 Класи пакету vk

# Класс MyApp виконує ініціалізацію частини программи, що праціює з сайто «Вконтакте»

# РОЗДІЛ 4 ІЛЮСТРАЦІЯ РОБОТИ ПРОГРАМИ

**4.1. Запуск застосунку**



Рисунок 4.1 – Головна сторінка застосунку

Після натискання кнопки “A” на головній сторінці, буде відкрито форму авторизації.

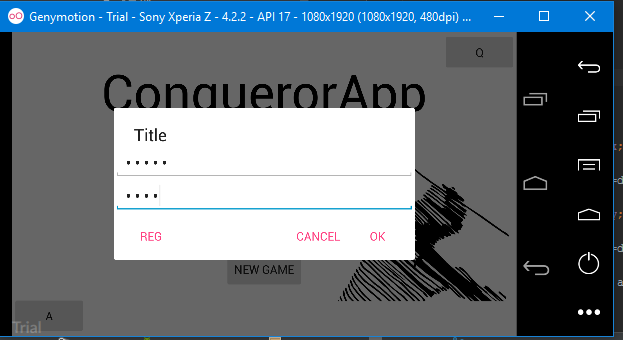


Рисунок 4.2 – авторизація

Якщо користувач зареєстрований та ввів свої данні, при натисканні «ОК» буде виконана авторизація. Про вдачну авторизацію буде сповіщенно «Auto S»



Рисунок 4.3 – вдачна авторизація

Якщо користувач натисне на кнопку «New Game», то система почне пошук гравців. Коли знайде – почнеться гра.

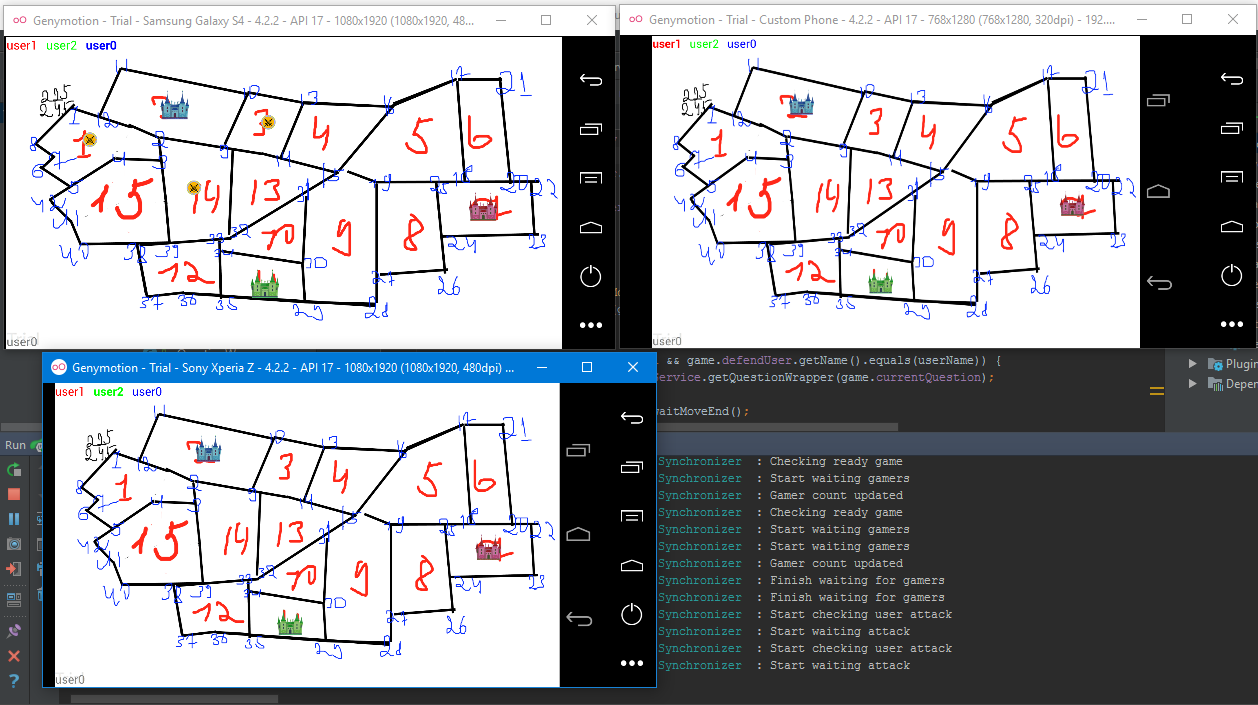


Рисунок 4.4 – початок гри у трьох різних гравців

Якщо логін користувача співпадає з логіном в лівому нижньому кутку(це означає, що зараз крок гравця), то він має обрати теріторію, яку бажає атакувати. При натисканні на обрану територію з’явиться питання.

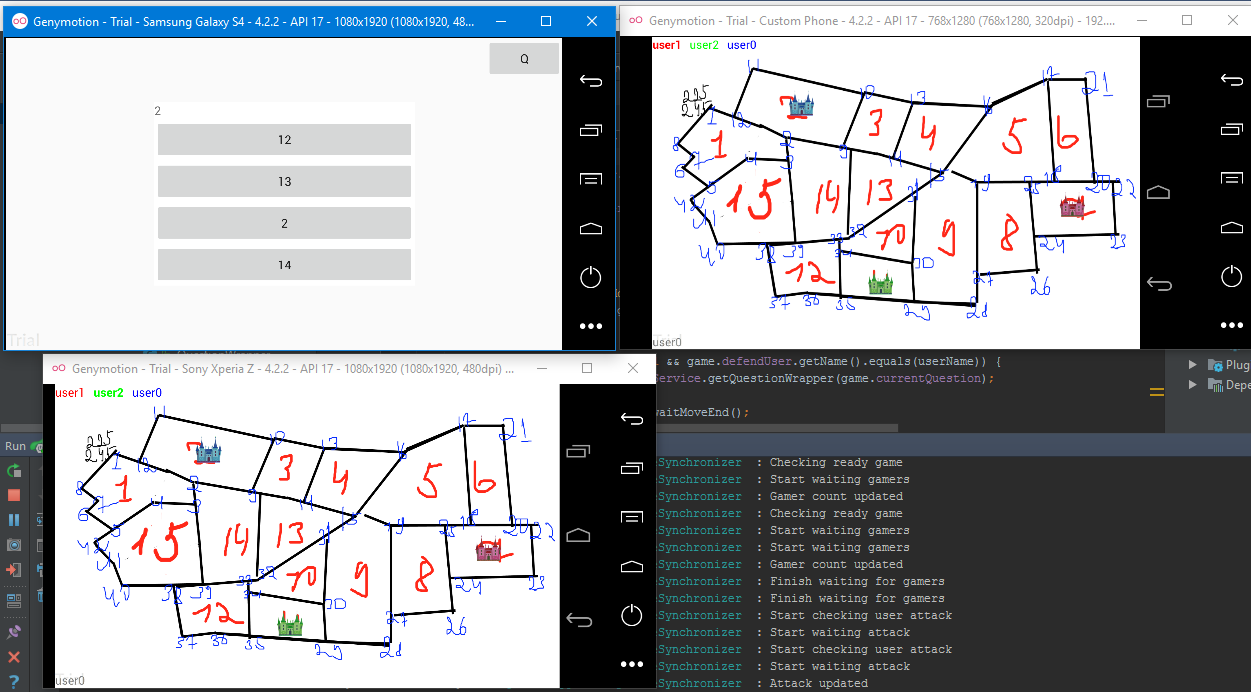


Рисунок 4.5 – початок гри

Якщо користувач відповів правильно, то він захоплює територію. Крок переходить до наступного гравця.

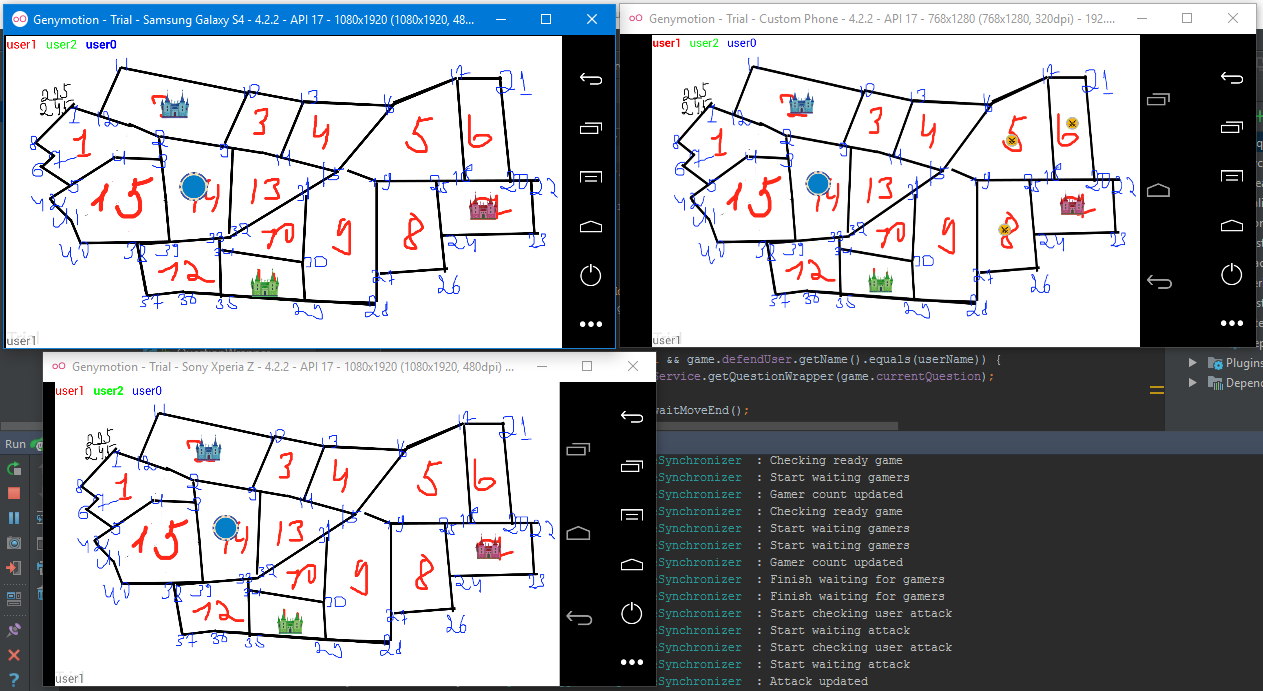


Рисунок 4.5 – правильна відповідь та крок іншого гравця

Якщо користувач атакує іншого гравця, питання відображається у обох. Той, кто відповів правильно – перемагє.

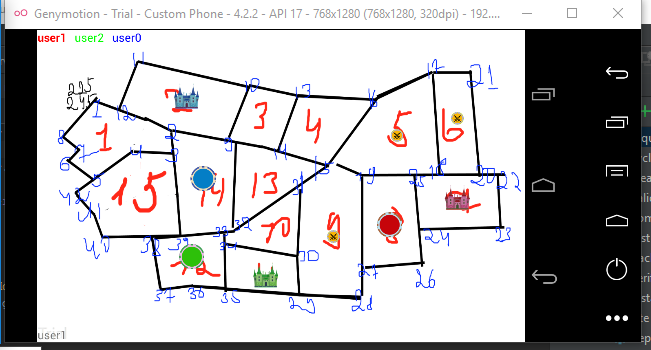


Рисунок 4.6.1 – початкова ситуація

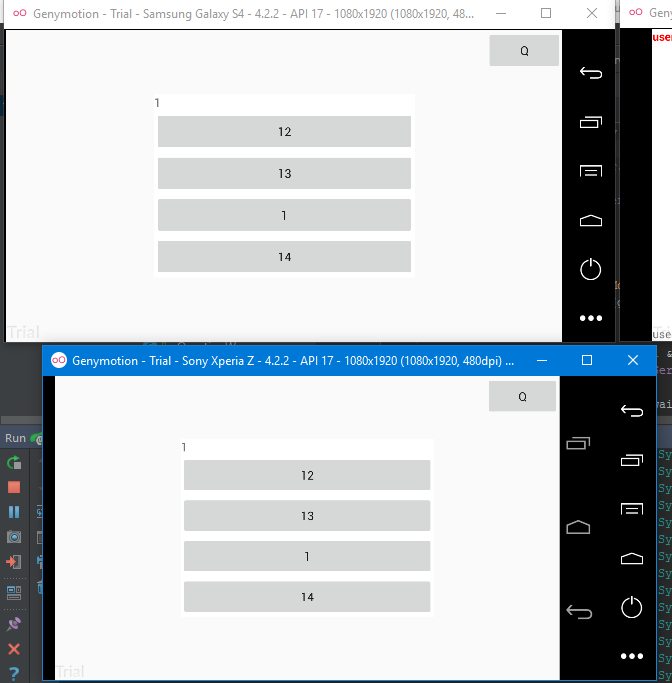


Рисунок 4.6.2 – питання у обох

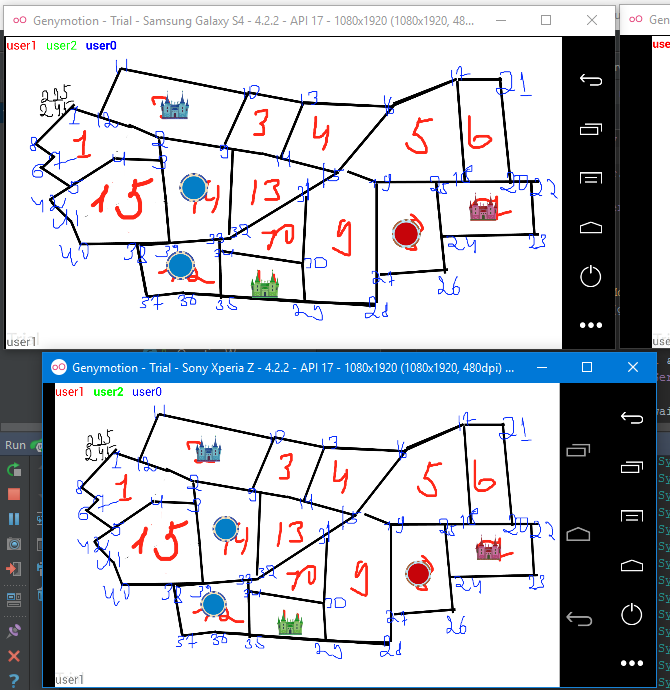


Рисунок 4.6.3 – кінцева ситуація

Гра продовжується до того, когли всі гравці не зроблять по сім кроків. Перемегає той, кто має найбільше територій. Після кінця гри застосунок переходить до головного єкрану.

* 1. **Авторизація через «ВКонтакте»**

Після натискання кнопки авторизації, користувач натискаяє кнопку реєстрації.

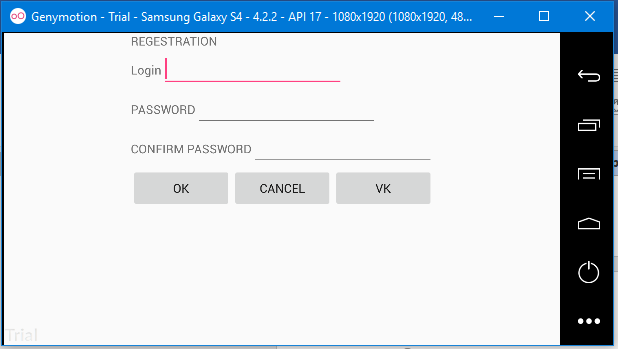


Рисунок 4.7 – вікно реєстрації

Користувач натискає кнопку «VK».

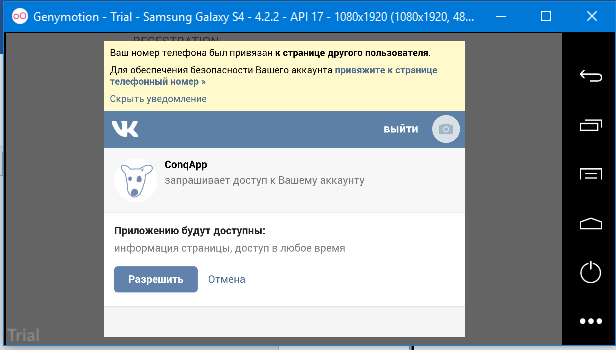


Рисунок 4.7 – застосування запитує, чи можна використовувати данні поточного користувача

Користувач натискає кнопку «Разрешить».

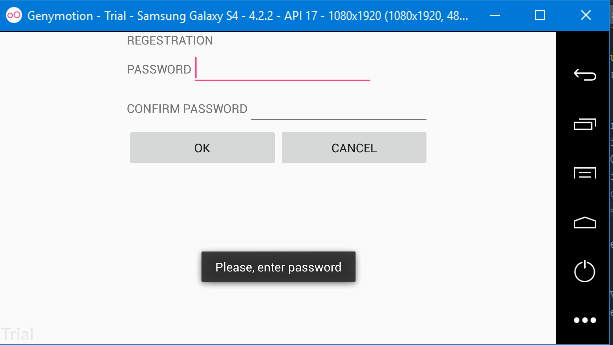


Рисунок 4.8 – форма створення паролю

Користувач натискає кнопку «ОК». Система реєструє нового гравця.

**ВИСНОВКИ**

Розробка мобільних застосувань систем є доволі складним, і тривалим процесом. Одним із найбільш важливих її етапів є збір та аналіз вимог. Окрім цього, додаткової уваги потребую виявлення бізнес процесів та бізнес-сутностей, що будуть наявні у розроблювальній системі.

У даній роботі, для створення програмного за стосунку, використовувалася платформа Android. Наразі дана платформа є однією із найбільш поширених у світі, оскільки, простота розробки, якісна зворотній зв'язок, величезне число розробників по всьому світу, відкритий вихідний код, доступність виходу на ринок - ось шлях успіху в сучасному світі комп'ютерних мобільних технологій.

Важливим питанням, під час розробки застосунків для мобільних платформ є, використання та інтеграція інших технологій, а також підтримка за стосунку на більш старих версія операційної системи.

Під час розробки програмного застосунку, було ми освоїли основи проектування і розробки додатків під Android. Надалі планується не зупинятися на досягнутому, а далі продовжувати займатися розробкою даного за стосунку.

Під час розробка системи, нами було використано наступні технології:

- Android SDK;

- MySQL;

- VK API.

Як кінцевий продукт, ми отримали застосунок, який надає користувачу можливість весело та цікало провести вільний час.

Під час розробки системи було отримано значні навички у аналізі вимог, проектуванні та розробці мобільних застосувань за допомогою технології Android та її окремих складових.

# СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Android – Режим доступу : http://www.3dnews.ru/581873 - Дата доступу : 16.05.2017.

2. Полный обзор изменений в Android M. – Режим доступу : https://www.iguides.ru/main/gadgets/google/android\_m\_review/- Дата доступу : 16.05.2017.

3. Android Fragmentation Vizualized – Режим доступу : http://opensignal.com/reports/2014/androidfragmentation/#android\_version\_timeseries - Дата доступу : 24.05.2017.

13. Android Studio. – Режим доступу : https://developer.android.com/sdk/index.html - Дата доступу : 24.05.2017.

4. SharedPreferences. – Режим доступу :http://developer.android.com/reference/android/content/SharedPreferences.html - Дата доступу : 24.05.2017.

5. AssyncTask. – Режим доступу : http://developer.android.com/reference/android/os/AsyncTask.html - Дата доступу : 28.05.2017.

6. Material Design Introduction. – Режим доступу : https://www.google.com/design/spec/material-design/introduction.html - Дата доступу : 28.05.20137.

7. Основние принципы Material Design. – Режим доступу : http://habrahabr.ru/post/248653/ - Дата доступу : 28.05.2017. 86

8. Работа с Google Maps а Android – Режим доступу : https://developers.google.com/maps/documentation/android-api/code-samples?hl=ru

- Дата доступу : 28.05.2017. 86