МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ОДЕСЬКИЙ ДЕРЖАВНИЙ ЕКОЛОГІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних наук,
управління та адміністрування
Кафедра інформаційних технологій

Бакалаврська кваліфікаційна робота

на тему: Розробка ігрового додатку «Loops» на платформі Android

Виконав студент 4 курсу групи К-41
Напрям 6.050101 2 комп'ютерні науки,
Коперсак Владислав Андрыйович
_____ _

Керівник асистент
Штефан Наталія Зінов'ївна

Консультант: к.т.н., доцент
Великодний Станіслав Сергійович

Рецензент: к.георг.н., доцент
Бояринцев Євген Львович

Одеса 2019

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

CSS – Cascading Style Sheets

Java EE – Java Enterprise Edition

JS – JavaScript

HTML – HyperText Markup Language

GWT – Google Web Toolkit

SQL – Structured Query Language

UML – Unified Modeling Language

Терміни

ОС – операційна система

Android – операційна система для мобільних пристроїв

Гаджет – (англ. gadget – пристрій)

Геймплей – ігровий процес

Квест – стиль гри

Рефакторинг – це контрольований процес поліпшення коду, без написання нової функціональності

Спрайт – графічний об'єкт в комп'ютерній графіці

Фреймворк – програмне забезпечення, що полегшує розробку і об'єднання різних модулів програмного проекту

Action – дія на діаграмі UML

Activity Diagram – діаграма активністі UML

Android – операційна система

AngularJS – JavaScript-фреймворк з відкритим вихідним кодом

Connection – стиль логічної гри

C++ – компільована, статично типізована мова програмування загального призначення

IntelliJ IDEA – інтегроване середовище розробки програмного забезпечення для багатьох мов програмування,

Google Play – магазин додатків, ігор компанії Google і інших компаній

Grails – фреймворк для створення веб-додатків, написаний на скриптовій мовою Groovy, який в свою чергу заснований на Java

Java – мова програмування

JavaFX – платформа на основі Java для створення додатків з насиченим графічним інтерфейсом

JIT-компілятор – Just-in-time Compilation

Material design – стиль графіки у іграх-головоломках

Node.js – програмна платформа, заснована на движку V8

LibGdx – фреймворк для створення відеоігор

Paint.NET – графічний редактор фірми Microsoft

Private – закритий атрибут класу

Protected – захищений атрибут

Public – відкритий атрібут класу

SceneBuilder – додаток для визначення призначеного для користувача інтерфейсу додатку JavaFX

Spring Framework – універсальний фреймворк с відкритим кодом для Java-платформи

Static Structure diagram – діаграма класів системи

Sun Microsystems – американська компанія, виробник програмного і апаратного забезпечення

Tomcat – контейнер сервлетів з відкритим вихідним кодом, що розробляється Apache Software Foundation

TomEE – проект верхнього рівня Apache, сервер додатків базується на специфікації стека Java EE 6 Web Profile

ВСТУП

Мобільні ігри буквально увірвалися в життя кожної людини, у якого є смартфон або будь-який інший гаджет з великим екраном. Кожен день розробники і видавці публікують в Google Play (раніше Play Market) нові ігри, придумують щось цікаве і вносять особливі фішки, щоб привернути увагу користувачів [1][1)].

Ігор на Android так багато, що в них можна розбиратися годинами в спробах знайти щось дійсно варте уваги, без вірусів і нав'язливої реклами, з цікавим сюжетом і режимами гри без інтернету і по мережі.

Мозок можна і потрібно тренувати – читанням, завданнями та розвиваючими іграми. Аналогічно розвитку м'язів, розум людини буде вимагати більшої складності.

Логічні на Android ніколи не втратять популярності: якщо знищення монстрів або прогулянки в "пісочниці" можуть з часом приїстися, то робота розуму ніколи не зупиняється і не приїдається. Розробники ігор знають: думати людині так само природно, як і дихати. А це означає, що для Android-пристрої сьогодні розроблені тисячі логічних ігор – зламати віртуальну систему Пентагону, розгадати слова російською або англійською мовами, пройти квест [2][2)].

Головоломки – ігри, які мають чимало спільного з логічними іграми, допомагають розвивати інтуїтивні здібності і стратегічне мислення. Іноді завдання здаються неймовірно складними, так як їх рішення вимагає послідовного застосування певних прийомів і їх поєднань.

З всього різноманіття асортименту логічних ігор для Android слід виділити основні жанри:

- головоломки;

[1)] [1] Топ 30 лучших игр на Андроид. URL: http://gamebizclub.com/rejtingi/luchshie-igry-na-android/. (дата звернення 02.02.2019).

[2)] [2] Infinity Loop: Energy − головоломка с фантастической атмосферой. URL:https://lifehacker.ru/infinty-loop-energy/. (дата звернення 3.02.2019).

- квести;
- шахи і шашки;
- математичні ігри;
- покрокові стратегії;
- кросворди та вікторини.

Об’єктом розробки є комп’ютерна гра у стилі «Connection», яка буде направлена на розвиток логічного мислення і одночасно буде приємним дозвіллям як для дорослих, так і для дітей.

Загальні характеристики кваліфікаційної роботи:

- повний обсяг сторінок пояснювальної записки – 51;
- кількість рисунків – 26;
- кількість таблиць – 0;
- кількість посилань – 20.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Характеристика об’єкта розробки

Об’єктом розробки є комп’ютерна гра типу «Connection», але на відміну від більшості представників даного жанру ігор, де необхідно пов'язати дві або більше об'єктів між собою за допомогою зв’язків, у даному випадку необхідно просто створити закритий цикл зв'язків між собою. Гра, хоч і налаштована на логіку, все ж можна вважати розслаблюючу.

Отже, гравець перед собою бачить набір ланок, кожен з яких необхідно виставити в положення так, щоб вся їх сукупність змогла утворити єдиний цикл або набір окремих дрібних циклів, причому, жодна ланка не повинна залишитися вільною.

Виконавши все правильно, гравець переходить на наступний рівень і так до нескінченності за рахунок генератора ланок. Взагалі, гра оформлена у гармонічній кольоровій гамі, є музичний супровід.

1.2 Опис предметної області

На даний час ігри-головоломки набирають стрімкої популярності. Існує безліч чудових головоломок для любителів Андроїд ігор на роздуми і комбінування.

Чудова графіка який дозволяє насолодитися додатком як на маленьких, так і великих планшетах, а також смартфонах, в тому числі і не дуже нових. Приємний музичний і звуковий супровід, вібрація і привабливий інтерфейс порадують кожного гравця.

Успіх завжди буде залежати від використання просторового мислення, вміння моделювати і бажання продумувати всі варіанти.

Більшість ігрових додатків створені так, що, цілком можливо, результат буде досягнутий вже після 1-2 простих маніпуляцій, до того ж майже очевидних [3][1)].

Все це підкреслює актуальність обраної теми для об’єкту розробки, а саме – створення логічної гри для розвитку мислення.

1.3 Аналіз існуючих аналогів

На першій стадії проектування слід розглянути декілька аналогів гри для подальшої чіткої постановки завдань і вимог до об’єкту розробки. Першим аналогом і головним прототипом є популярна для Android гра «Loop», або «Петля» (рис. 1):

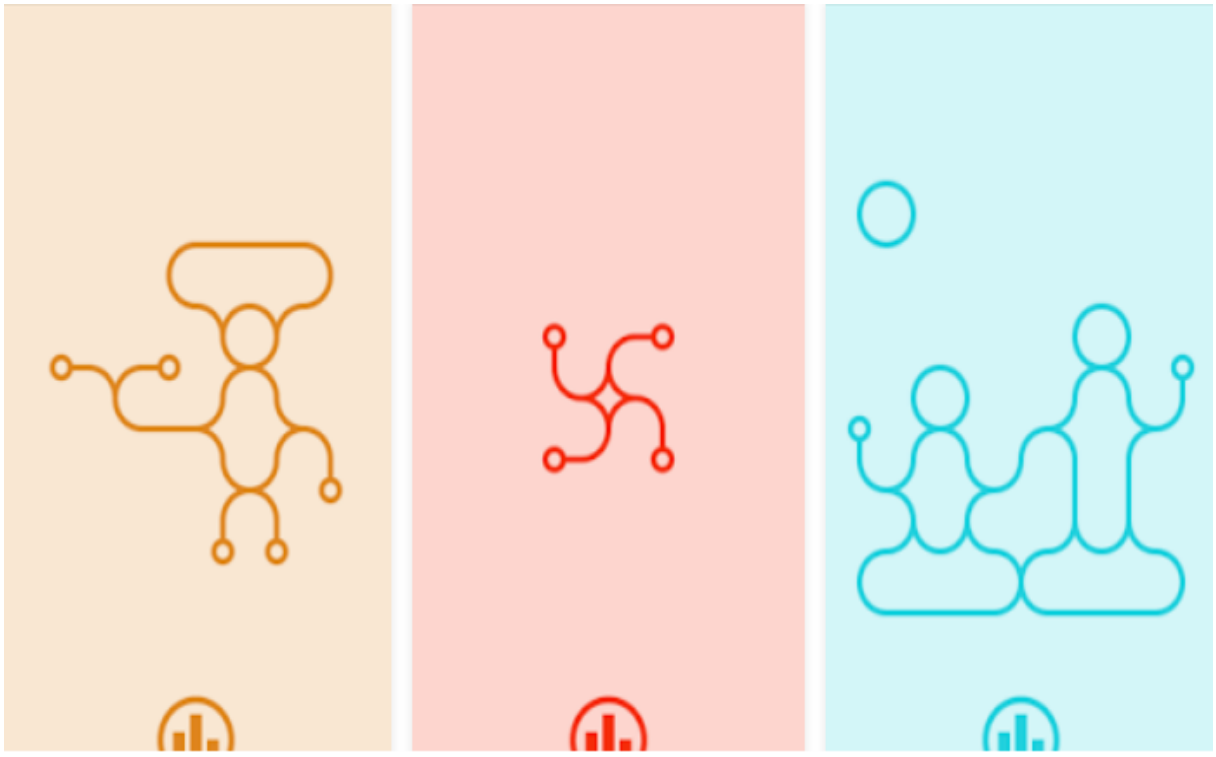

Рисунок 1 – Скріншот мобільної гри для Android «Loop»

[1)] [3] Игры для Android. URL: https://androidinsider.ru/igry. (дата звернення 5.02.2019).

«Петля» – це пазл, метою якого є створення хитромудрих узорів, що утворюють петлі, або це просто додаток, що використовує нескладну ідею «з'єднання між собою безлічі речей» заради розваги. Це проста, розслаблююча, нескінченна гра [4][1)].

Мета даної гри – розташувати всі форми в правильному порядку. З цих форм ігроку належить сформувати петлю. Кількість петель під час гри буде постійно збільшуватися.

Петлі, пофарбовані в різні кольори, можуть переплітатися між собою. В результаті ігрок отримаєте складні композиції. Саме просто управління і мінімалістична графіка. Гра включає більше 100 унікальних рівнів.

Переваги «Loop»:

- графіка в стилі «material design»;
- унікальний і приємний геймплей;
- цікаві форми (на зображенні видно як цикли утворюють форми).

До недоліків слід віднести наступне:

- кінцівка рівнів (рівні статичні, немає генератора);
- графіка хоч і в стилі «material design», все через певний час вона тисне на очі;
- музичний супровід занадто коротке і воно лише одне, що з часом набридає;
- відсутність підказок [5][2)].

Після 30 мільйонів завантажень гри «Loop» вийшла нова версія «Loop Hex» (рис. 2).

Ця гра містить в собі цілу нову колекцію нескінченних рівнів для гри в гексагональній дошці.

[1)] [4] Топ-8 самых популярных игр на Android. https://keddr.com/2017/05/top-8-samyih-populyarnyih-igr-na-android/. (дата звернення 12.12.2019).

[2)] [5] Топ игр на андроид: обзор 10 лучших приложений. URL: https://www.fly-phone.ru/notes/igry-dlya-smartfonov/top-10-igr-na-android/. (дата звернення 13.02.2019).

Рисунок 2 – Скріншот мобільної гри для Android «Loop Hex»

«HEX» – гра дозволяє створювати петлі по-новому. Розробники зберегли ту саму структуру гри «Петля»: чиста і проста гра, яка допомагає поліпшити рівень фокусування і уваги.

«Loop Hex» дозволяє створювати моделі замкнутої форми, з'єднуючи всі частини. Це гра – головоломка, але ретельно побудована, щоб забезпечити моменти відпочинку і радості. Вона орієнтована на позбавлення людини від моментів стресу, оскільки немає ніякого тиску для вирішення рівнів і відсутність таймерів. Ідеально підходить для дітей [6][1)].

При розробці гри було прийнято рішення залишити її без таймерів, оскільки розуміло, що кожен має свій власний темп і час не повинен бути мірою інтелекту: здатність вирішувати головоломку в кінцевому підсумку є єдиною мірою здатності та інтелекту.

Основні концепції гри:

- з'єднати всі лінії та кути для створення закритих форм;

[1)] [6] Обзор Infinity Loop: HEX. URL: https://pdalife.ru. (дата звернення 13.02.2019).

– нескінченна кількість рівнів гри.

Наступна гра є класичним представником жанру Connection, її назва «Plumber Game» (рис. 3).

Класична гра «водопровідник». Логічна гра, в якій ігроку треба побудувати трубопровід і пустити по ньому воду. Елементи труби можна повертати відносно основної точки її положення, поєднуючи її в єдине ціле.

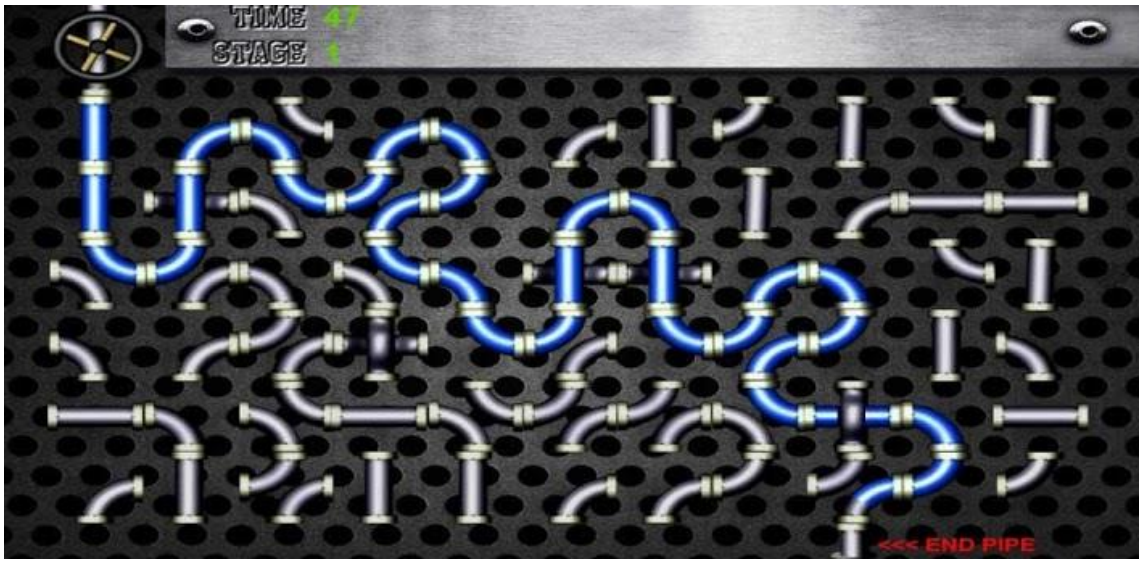

Рисунок 3 – Інтерфейс гри «Plumber Game»

До від’ємносте даної гри можна віднести:

– класичний підхід;
– оригінальні рівні;
– наявність підказок.

Недоліки:

– графіка дуже стара відносно сучасних тенденцій розвитку графічного дизайну таких додатків;
– рівні статичні, генератори відсутні [7][1)].

[1)] [7] Описание игры Plumber 10k для смартфонов с ОС Android. URL: https://plumber10k.wordpress.com/2012/09/24/opisanie-igry-plumber-10k-dlya-smartfonov-s-os-android/. (дата звернення 23.02.2019).

Наступним аналогом є ще одна версія класичного підходу, це "Pipe Puzzle" (рис. 4).

Рисунок 4 – Приклад генерованого рівня гри «Pipe Puzzle»

З переваг даного ігрового додатку можна виділити:

- вдала візуалізація;
- сучасна стилізація;
- зручний інтерфейс.

Недоліком є те, що класичний підхід не є актуальним (гра як мінімум з 70-х) і втрачає популярність.

Останнім аналогом слід розглянути «Infinity Loop: Energy» (рис. 5).

Це головоломка з мінімалістичним дизайном і фантастичною атмосферою. Суть проста: гравцю треба з'єднати джерело енергії з усіма лампочками на кожному етапі гри [2][1)].

На самому початку для передачі енергії будуть доступні тільки прямі і вигнуті лінії, а завдання будуть вирішуватися за лічені секунди.

[1)] [2] Infinity Loop: Energy – головоломка с фантастической атмосферой. URL:https://lifehacker.ru/infinty-loop-energy/. (дата звернення 3.02.2019).

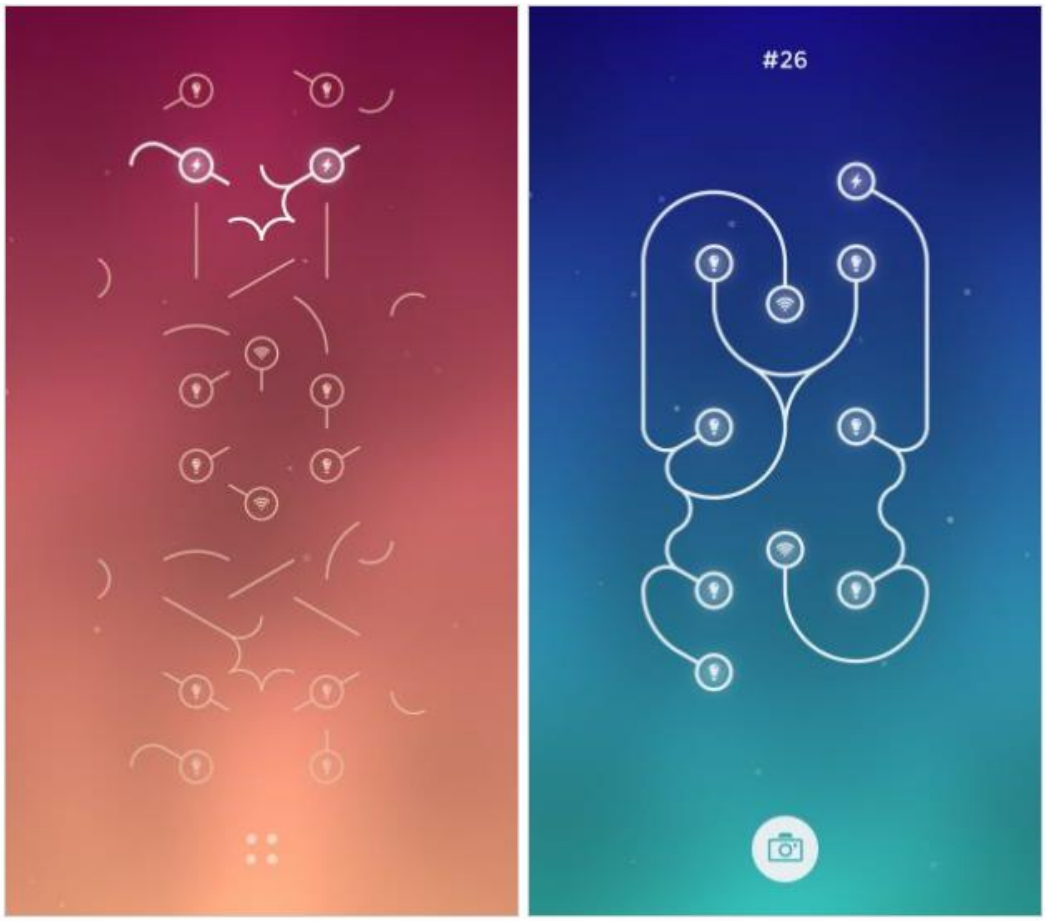

Рисунок 5 – Скріншот мобільної гри «Infinity Loop: Energy»

Щодо недоліків даної гри: іноді трапляються рівні, в яких доводиться подумати, але це швидше виняток. В основному можна інтуїтивно здогадатися, як повинні розташовуватися елементи. Єдине, що дійсно засмучує, – це реклама. У грі вона не дуже нав'язлива, але враження псує [5][1)].

Поступово будуть з'являтися більш складні елементи, які з'єднуються в химерні комбінації. Але навіть незважаючи на постійно зростаючу складність, гра не затримає вас на рівні довше декількох хвилин. «Infinity

[1)] [5] Топ игр на андроид: обзор 10 лучших приложений. URL: https://www.fly-phone.ru/notes/igry-dlya-smartfonov/top-10-igr-na-android/. (дата звернення 13.02.2019).

Loop: Energy» не намагається кинути гравцеві виклик. Вона не створена для того, щоб змусити вас посилено думати над вирішенням завдання [3][2)].

Це медитативна і заспокійлива гра, що поєднує в собі приємний візуальний стиль і легку електронну музику. Приклад завершеного рівня представлено на рис.6.

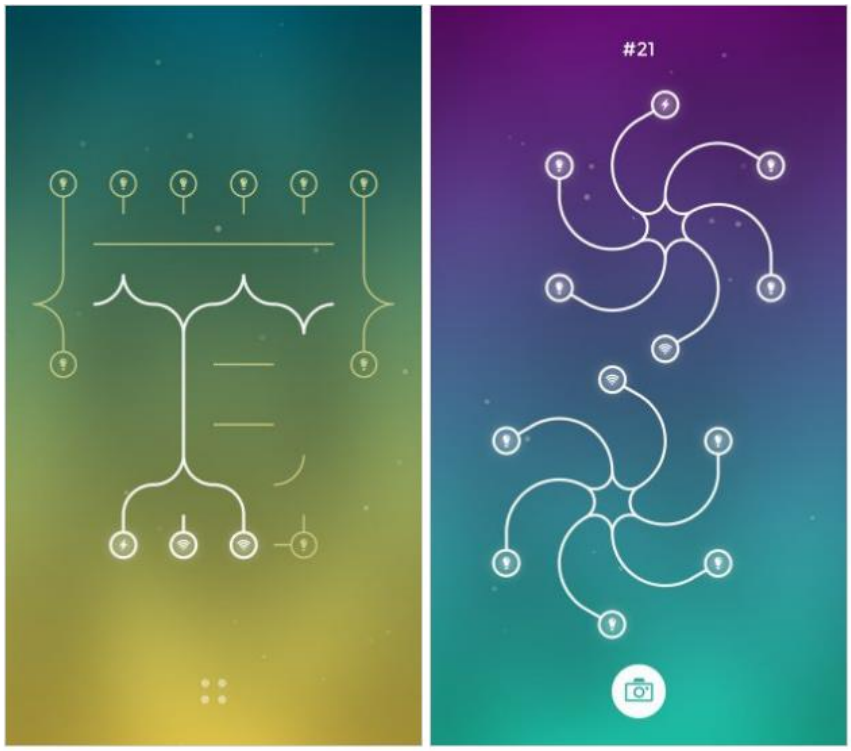

Рисунок 6 – Приклад завершеного рівня «Infinity Loop: Energy»

Загальна проблема всіх аналогів в тому, що вони все нативні тільки під ОС Android, а установлювати для дрібної гри емулятор Андроїда дуже і дуже затратно. А також однотипність ланок, що означає, що кожна ланка, що має якийсь тип зв'язку, має лише один спрайт.

Ідея створити проект, який буде максимально мати все вище перераховані переваги, а так само по максимуму позбутися від недоліків. Тому гра буде в більшій мірі копіювати перший аналог, так як він найбільш

[2)] [3] Игры для Android. URL: https://androidinsider.ru/igry. (дата звернення 5.02.2019).

перспективний з усіх, але і особливості інших так само будуть ураховуватися при проектуванні об’єкту розробки.

В цілому, буде приділена увага до стилю графіки, принципу гри, простоті управління і підказки, але разом з цим поставлена ціль вирішити проблему з музикою, однорідністю графіки, статичністю рівнів і однотипністю зв'язків. Гра буде орієнтованою як на дітей, так і на дорослих користувачів.

1.4 Аналіз засобів розробки

Так як виробництво відеоігор є в першу чергу джерелом прибутку, компанії-розробники шукають більше можливостей для поширення своїх продуктів і збільшення аудиторії прихильників. В цьому їм допомагають кросплатформені кошти розробки додатків.

В цьому є маса переваг, як для самих розробників, так і для користувачів. Гравці отримують можливість користуватися додатками з різних пристроїв, а модель хмарних обчислень дозволяє в будь-якому місці мати доступ до своїх даних або до всіляких веб-сервісів [6][1)].

Це збільшує продажі додатків, за рахунок чого компанія отримує більший прибуток і розширює присутність бренду на ігровому ринку. Крім того, кросплатформені кошти розробки мають масу переваг, які значно спрощують роботу розробників при написанні ігор. Наприклад, економія часу на написання коду, економія ресурсів і т.д.

Детальніше дані переваги будуть розглянуті далі.

1.4.1 Середа разробки IntelliJ IDEA

[1)] [6] Обзор Infinity Loop: HEX. URL: https://pdalife.ru. (дата звернення 13.02.2019).

Найрозумніша і зручне середовище розробки для Java, що включає підтримку всіх останніх технологій і фреймворків. IntelliJ IDEA (рис. 7) надає інструменти для продуктивної роботи і ідеально підходить для створення комерційних, мобільних і веб-додатків [8][2)].

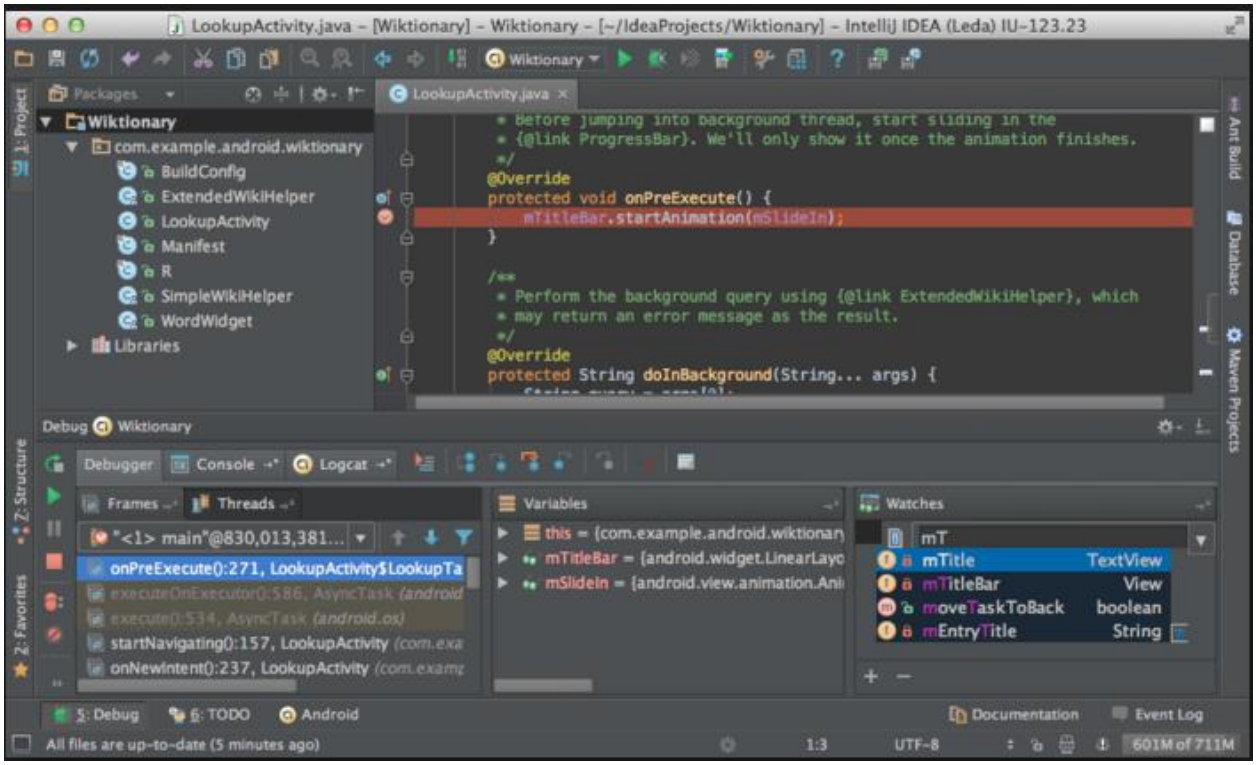

Рисунок 7 – Середа розробкки IntelliJ IDEA

До повної версії середи розробки включено:

- розумне автодоповнення, інструменти для аналізу якості коду, зручна навігація, розширені рефакторингом і форматування для Java, HTML, CSS, JS і багатьох інших мов;
- підтримка всіх популярних фреймворків і платформ, включаючи Java EE, Spring Framework, Grails, GWT, Node.js, Android, і т.д;
- інтеграція з серверами додатків, включаючи Tomcat, TomEE та ін.;

[2)] [8] IntelliJ IDEA. URL:https://jetbrains.ru/products/idea/. (дата звернення 14.02.2019).

- інструменти для роботи з базами даних і SQL файлами, включаючи зручний клієнт і редактор для схеми бази даних;
- інструменти для запуску тестів і аналізу покриття коду, включаючи підтримку всіх популярних фреймворків для тестування [9][1)].

Безкоштовна версія дає такі можливості:

- розумне автодоповнення, інструменти для аналізу якості коду, зручна навігація, розширені рефакторингом і форматування для Java та інше;
- професійний набір інструментів для розробки Android-додатків;
- підтримка JavaFX 2.0, інтеграція з SceneBuilder; інтеграція з автоматизованими інструментами побудови та управління проектом [8][1)].

IntelliJ IDEA середовище розробки, що було обрано для розробки дипломного проекту.

Переваг у цієї програми доволі багато, але важливо те, що саме це середовище, незважаючи на те, що воно здатне підтримати багато мов програмування, зосереджено саме під розробку на мові Java [10][2)].

Дуже вигідним є доволі гнучке налаштування стилізації програмного тексту. Інші середовища не мають настільки зручного налаштування.

1.4.2 Мова програмування Java

Java – сильно типізована об’єктно-орієнтована мова програмування, розроблена компанією Sun Microsystems (в подальшому вона була придбана

[1)] [9] Навчальні матеріали з інформатики » Програмування мовою Java. URL: https://www.ua5.org/java/. (дата звернення 14.02.2019).

[1)] [8] IntelliJ IDEA. URL: https://jetbrains.ru/products/idea/. (дата звернення 14.02.2019).

[2)] [10] Інформаційний блог з програмування. URL: https://habr.com/ru/hub/java/. (дата звернення 05.03.2019).

компанією Oracle). Програми Java зазвичай транслюються в спеціальний байт-код, тому вони можуть працювати на будь-якій комп'ютерній архітектурі, за допомогою віртуальної Java-машини [11][3)].

Байт-код виконується віртуальною машиною Java (JVM) – програмою, що обробляє байтовий код та передає інструкції обладнанню як інтерпретатор. Перевагою подібного способу виконання програм є повна незалежність байт-коду від операційної системи та устаткування, що дозволяє виконувати Java-програми на будь-якому пристрої, для якого існує відповідна віртуальна машина.

Іншою важливою особливістю технології Java є гнучка система безпеки, в рамках якої виконання програми повністю контролюється віртуальною машиною.

Будь-які операції, які перевищують встановлені повноваження програми (наприклад, спроба несанкціонованого доступу до даних або з'єднання з іншим комп'ютером), викликають негайне переривання. Програми, написані на Java, мають репутацію більш повільних та займають більше оперативної пам'яті, ніж написані на мові C.

Проте, швидкість виконання програм, написаних на мові Java, була істотно поліпшена з випуском в 1997-1998 роках так званого JIT-компілятора в версії 1.1 на додаток до інших особливостей мови для підтримки кращого аналізу. Станом на лютий 2012 року, код Java 7 приблизно в 1.8 рази повільніше коду, написаного на мові C [12][1)].

1.4.3 Бібліотека LibGdx

LibGDX – багатоплатформовий фреймворк для розробки ігор і візуалізації, заснований на мові програмування Java з деякими

[3)] [11] Java − объектно-ориентированный язык программирования. URL: https://infoblog1.ru/teach/java/. (дата звернення 01.03.2019).

[1)] [12] Подробнее о технологии Java. URL: https://www.java.com/ru/about/. (дата звернення 01.03.2019).

компонентами, написаними на C і C ++ для підвищення продуктивності певного коду. В даний час підтримує різні операційні системи і HTML5 як цільові платформи [13][2)].

LibGdx дозволяє написати код одного разу і потім розгортати гру або додаток на декількох платформах без модифікації. Є можливість розробляти додаток на основному комп'ютері і отримувати величезну вигоду швидкої розробки, замість того, щоб чекати, коли останні зміни буду впроваджені і встановлені на пристрій і будуть скомпільовані в HTML5 (рис. 8).

Рисунок 8 – LibGdx

LibGDX дозволяє розробнику створювати, тестувати та налагоджувати код на власному комп'ютері, і потім переносити його на інші ОС. Для цього фреймворк використовує окремі модулі для збірки додатку під кожну платформу, а також незалежний модуль, який містить основний код програми.

Гнучкість та збільшення можливостей досягається за рахунок розширень. Можна підключити фізичний двигун для роботи з об’єктами та фізикою реального світу, додати підтримку шрифтів або працювати з 3D об’єктами. [14][1)].

[2)] [13] Обзор игрового движка LibGDX. https://gamedevmania.ru/engines/libgdx. (дата звернення 02.03.2019).

[1)] [14] libGDX − фреймворк для разработки игр. URL: http://www.libgdx.ru/. (дата звернення 04.03.2019).

При необхідності, libGDX може перейти від Java до нативному коду, щоб зосередитися на самій кращому і можливої продуктивності. Весь цей функціонал ховається за Java API, так що ви не повинні турбуватися про крос-компіляції нативного коду на всіх платформах.

Багато частин LibGDX знають специфіку платформи і вам не потрібно з ними стикатися. LibGDX зосереджений на тому, щоб бути фрейворком, ніж движком, визнаючи, що немає універсального рішення.

1.4.4 Додаток PaintDotNet

Мобільні ігри сильно обмежені в дозволі і розмірах екрану, проте саме графік гри надає користувачеві перше враження про гру в цілому. Ігровий процес є основним критерієм оцінки того, наскільки цікава сама гра. Слабка графіка здатна «вбити» саму захоплюючу гру.

Тому для розробника необхідно витратити певний час на створення для гри графіки та анімації, які зможуть привернути увагу користувачів. Обмеження в графіку мобільних ігор роблять процес її створення більш простим у порівнянні з графікою для PC або консольних ігор [15][1)].

Paint.NET (рис. 9), графічний редактор фірми Microsoft, часто позиціонується як заміна MS Paint, з додатком, що входить до складу операційних систем Windows.

[1)] [15] Обзор бесплатного графического редактора Paint.NET. URL: https://www.ixbt.com/soft/paint-net.shtml. (дата звернення 21.03.2019).

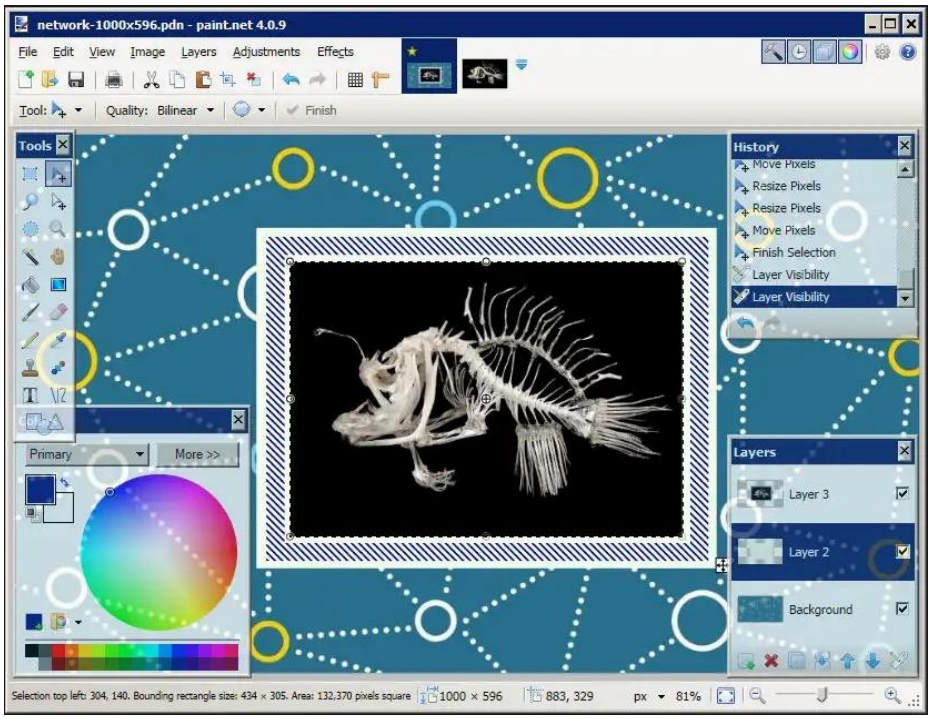

Рисунок 9 – Інтерфейс Paint.NET

В результаті може скластися враження, що їх відмінності мінімальні, що цільова аудиторія обох продуктів багато в чому збігається.

MS Paint – примітивний інструмент, призначений, скоріше, для розваг, ніж для серйозної роботи. Paint.NET є повноцінним графічним редактором, без будь-яких застережень. Він володіє широкими можливостями обробки фотографій. Більшість аналогічних програм мають досить високу ціну, що обмежує їх доступність.

Paint.NET, навпаки, безкоштовний. Ви можете вільно встановити програму і використовувати її необмежену кількість часу.

Графічний редактор може перемикатися між вісьмома мовами інтерфейсу, серед яких, на жаль, відсутня російська локалізація. Paint.NET швидко робить перший старт і без завантажених в пам'ять фотографій займає близько 33 МБ оперативної пам'яті.

Перед тим. як почати створення графіки, слід визначити, якого результату потрібно досягти. Гра повинна мати повністю сформувалася концепцію. Далі необхідно зібрати воєдино всю інформацію по грі і вибрати

графічні об'єкти, які зможуть реалізувати головні ідеї проекту. Для цього слід вибрати графічні елементи і додати їх в програму [15][1)].

2 ПРОЕКТНА ЧАСТИНА

2.1 Вимоги до функціональних характеристик і сумістності гри

Вимоги до функціональних характеристик проекту розробки наступні:

- вихідний код необхідно реалізувати на об'єктно-орієнтованій мові програмування Java;
- ігрова механіка повинна бути інтуїтивно зрозуміла користувачеві і мати внутрішньо ігрову інструкцію, а управління здійснюватися через сенсорні системи введення для мобільних пристроїв;
- зручний і лаконічний інтерфейс, що не перевантажують ігровий процес;
- естетична візуалізація для залучення уваги користувачів;
- обов’язкове звукове впровадження у ігровий процес, так як вона спрямована на довгий часовий період;
- користувачеві повинна бути надана можливість збереження ігрового прогресу для подальшого завантаження з можливістю вибору збереження.

[1)] [15] Інформаційний портал. URL: Обзор бесплатного графического редактора Paint.NET. URL: https://www.ixbt.com/soft/paint-net.shtml. (дата звернення 21.03.2019).

Умови інформаційної та програмної сумісності для проекту: гра повинна бути виконана на об'єктно-орієнтованої мові програмування Java і бути сумісною з такими ОС як Windows і Android [16][1)].

Необхідно використовувати такі конструкції мови Java:

- закриті відкриті члени класів;
- спадкування;
- динамічне створення об'єктів;
- конструктори з параметрами;
- конструктори без параметрів;
- інтерфейси.

2.1 Діаграма діяльності UML

UML (англ. Unified Modeling Language – уніфікована мова моделювання) – мова графічного опису для об'єктного моделювання в області розробки програмного забезпечення, моделювання бізнес-процесів, системного проектування та відображення організаційних структур. UML є мовою широкого профілю – це відкритий стандарт, який використовує графічні позначення для створення абстрактної моделі системи, так званої UML-моделі [17][1)].

UML була створена для визначення, візуалізації, проектування та документування, в основному, програмних систем. UML не є мовою програмування, але на підставі UML-моделей можлива генерація коду.

До переваг UML відносять:

- UML об'єктно-орієнтований, в результаті чого методи опису результатів аналізу та проектування семантично близькі до методів програмування на сучасних об'єктно-орієнтованих мовах;

[1)] [16] Вимоги до функціональних характеристик. URL: https://studopedia.su/13_68404_vimogi-do-funktsionalnih-harakteristik.html. (дата звернення 11.03.2019).

[1)] [17] Ведення в UML 2.0. URL: http://bourabai.kz/dbt/uml/index.htm. (дата звернення 11.03.2019).

- UML дозволяє описати систему практично з усіх можливих точок зору та різні аспекти поведінки системи;
- діаграми UML порівняно прості для читання після досить швидкого ознайомлення з його синтаксисом;
- UML розширює та дозволяє вводити власні текстові та графічні стереотипи, що сприяє його застосування не тільки в сфері програмної інженерії;
- UML набув широкого поширення й динамічно розвивається [18][2)].

Діаграма діяльності або діаграма активності (англ. Activity Diagram) – UML-діаграма, на якій показані дії, стану яких описано на діаграмі станів. Під діяльністю (англ. Activity) розуміється специфікація виконуваної поведінки у вигляді координованого послідовного та паралельного виконання підлеглих елементів – вкладених видів діяльності та окремих дій (англ. action), з'єднаних між собою потоками, що йдуть від виходів одного вузла до входів іншого.

Діаграма виглядає найбільш простою, оскільки нагадує звичну всім блок-схему. Насправді ж діаграма активності – це щось більше, ніж блок-схема, хоча цілі у них схожі – обидві вони відображають певний алгоритм [19][1)]. Нотація UML пропонує п'ять видів системи:

- вид системи з точки зору прецедентів;
- вид з точки зору проектування;
- вид з точки зору процесів;
- вид з точки зору розгортання;
- вид з точки зору реалізації.

При цьому кожен з перерахованих способів подання системи може містити послідовності дій, які можуть бути описані за допомогою алгоритмів. Ось тут, так би мовити, і виходять на сцену діаграми діяльностей.

---

2) [18] Моделювання на UML. URL: http://book.uml3.ru/. (дата звернення 11.03.2019).

1) [19] Виды диаграмм UML. URL: https://studizba.com/lectures/10-informatika-i-programmirovanie/273-uml/3457-2-vidy-diagramm-uml.html. (дата звернення 11.03.2019).

Взагалі кажучи, будь-який елемент моделі, що має динамічну поведінку, може бути доповнений діаграмою діяльності – саме для уточнення цієї самої динаміки.

Гарно підібраний по контексту приклад це можливість застосування діаграм активності для опису бізнес-процесів. Саме на діаграмі діяльності представлені переходи потоку управління від однієї діяльності до іншої.

Це, по суті, різновид діаграми станів, де всі або більша частина станів є деякими діяльностями, а всі або більшість переходів спрацьовують при завершенні певної діяльності та дозволяють перейти до виконання наступної. Діаграма діяльності може бути приєднана до будь-якого елементу моделі, що має динамічну поведінку.

До речі, логічніше говорити не «діаграма діяльності», а «діаграма діяльностей» – у множині.

Приклад діаграми діяльності для відображення роботи алгоритму «CoverScreen» представлено на рис. 10.

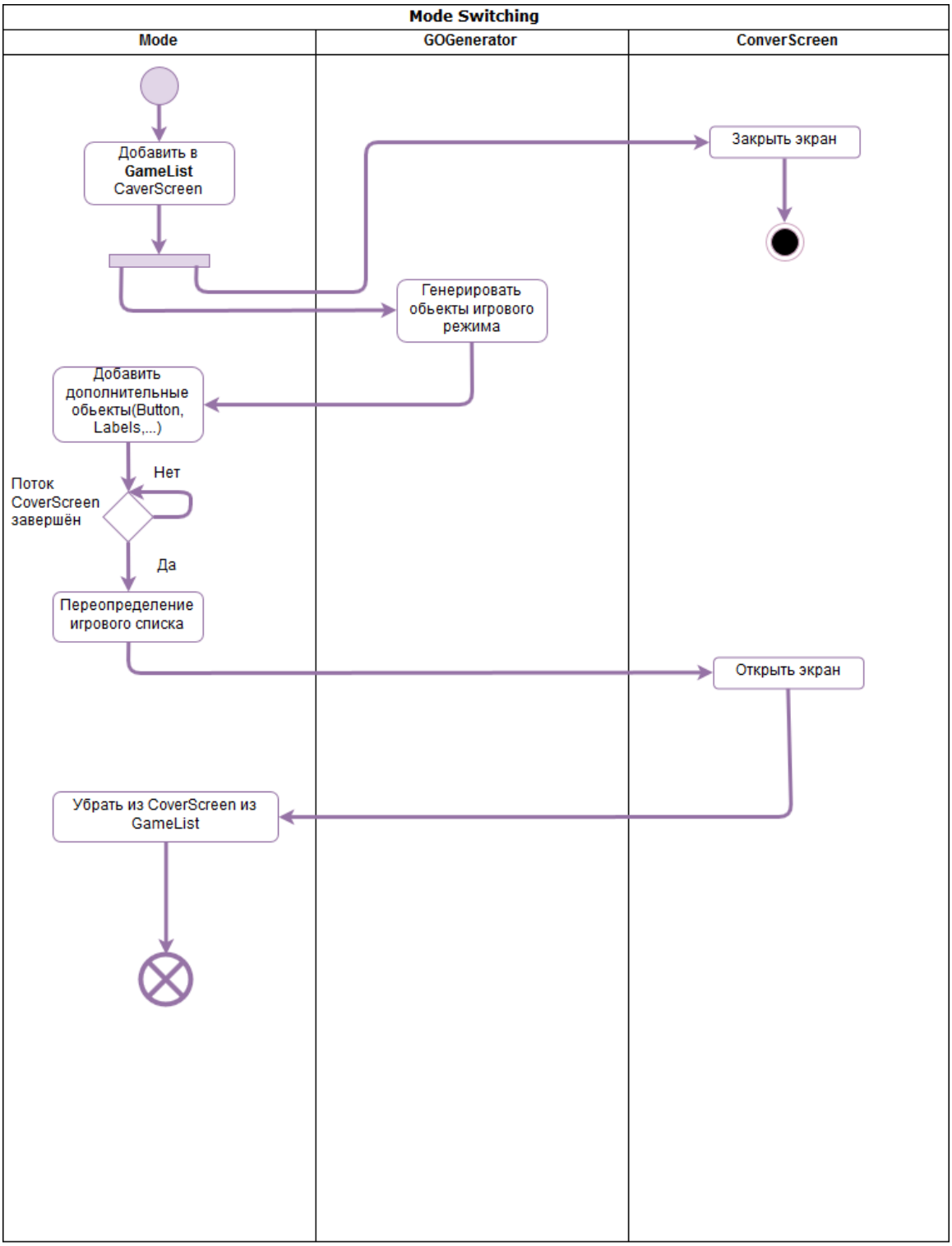

Рисунок 10 – Діаграма діяльності роботи алгоритму «CoverScreen»

2.2 Діаграма класів

Діаграма класів (англ. Static Structure diagram) – діаграма, що демонструє класи системи, їх атрибути, методи і взаємозв'язку між ними. Являється однією з основних UML-діаграм.

Діаграма класів займає центральне місце в проектуванні об'єктно-орієнтованої системи. Нотація класів використовується на різних етапах проектування і будується з різним ступенем деталізації. Мова UML застосовується не тільки для проектування, але і з метою документування.

Це набір статичних, декларативних елементів моделі. Діаграми класів можуть застосовуватися й при прямому проектуванні, тобто в процесі розробки нової системи, та при зворотному проектуванні – описі існуючих та використовуваних систем [20][1)].

Інформація з діаграми класів безпосередньо відображається в вихідному коді програми – в більшості існуючих інструментів UML-моделювання можлива кодогенерація для певної мови програмування (зазвичай Java або C++). Таким чином, діаграма класів – кінцевий результат проектування та відправна точка процесу розробки [17][2)].

Діаграма класів займає центральне місце в проектуванні об'єктно-орієнтованої системи. Нотація класів використовується на різних етапах проектування та будується з різним ступенем деталізації. Мова UML застосовується не тільки для проектування, але і з метою документування, а також створення ескізів проекту.

Основними елементами є класи і зв'язку між ними. Класи характеризуються за допомогою атрибутів і операцій.

Атрибути описують властивості об'єктів класу. Більшість об'єктів в класі отримують свою індивідуальність через відмінності в їх атрибутах і

---

1) [20] UML-диаграммы классов. URL: https://prog-cpp.ru/uml-classes/. (дата звернення 11.03.2019).

2) [17] Ведення в UML 2.0. URL: http://bourabai.kz/dbt/uml/index.htm. дата звернення 14.03.2019).

взаємозв'язку з іншими об'єктами. Ім'я атрибута повинно бути унікально в межах класу. За ім'ям атрибута може слідувати його тип і значення за замовчуванням.

Операція є функція або перетворення. Операція може мати параметри і повертати значення. Види зв'язків: асоціація, агрегації, успадкування.

Асоціація (association) − являє собою відносини між екземплярами класів. Кожен кінець асоціації володіє кратністю, яка показує, скільки об'єктів, розташованих з відповідного кінця асоціації, може брати участь в даному відношенні. Асоціації може бути присвоєно ім'я. Також, на кінцях асоціації під кратністю може вказуватися ім'я ролі, тобто яку роль виконують об'єкти, що знаходяться з даного кінця асоціації.

Композиція (composition) − це така агрегація, де об'єкти-частини не можуть існувати самі по собі і знищуються при знищенні об'єкта агрегує класу. Композиція зображується так само, як асоціація, тільки ромб зафарбований.

Спадкування (inheritance) − це відношення типу «загальне-приватне». Дозволяє визначити таке відношення між класами, коли один клас володіє поведінкою і структурою ряду інших класів. При створенні похідного класу на основі базового виникає ієрархія наслідування.

Реалізація принципів спадкування є ключовою передумовою можливості повторного використання коду, оскільки це основний інструмент досягнення поліморфізму.

У будь-якому об'єктно-орієнтованому процесі проектування діаграма класів є результатом, тому що вона є моделлю, найбільш близькою до реалізації, тобто коду [17][1)].

---

[17] Ведення в UML 2.0. URL: http://bourabai.kz/dbt/uml/index.htm. (дата звернення 11.03.2019).

Існують інструменти, що здатні перетворити діаграму класів в код – такий процес називається кодогенерацією та підтримується безліччю IDE та засобів проектування.

Діаграма класів для проекту розробки представлено на рис. 11-15:

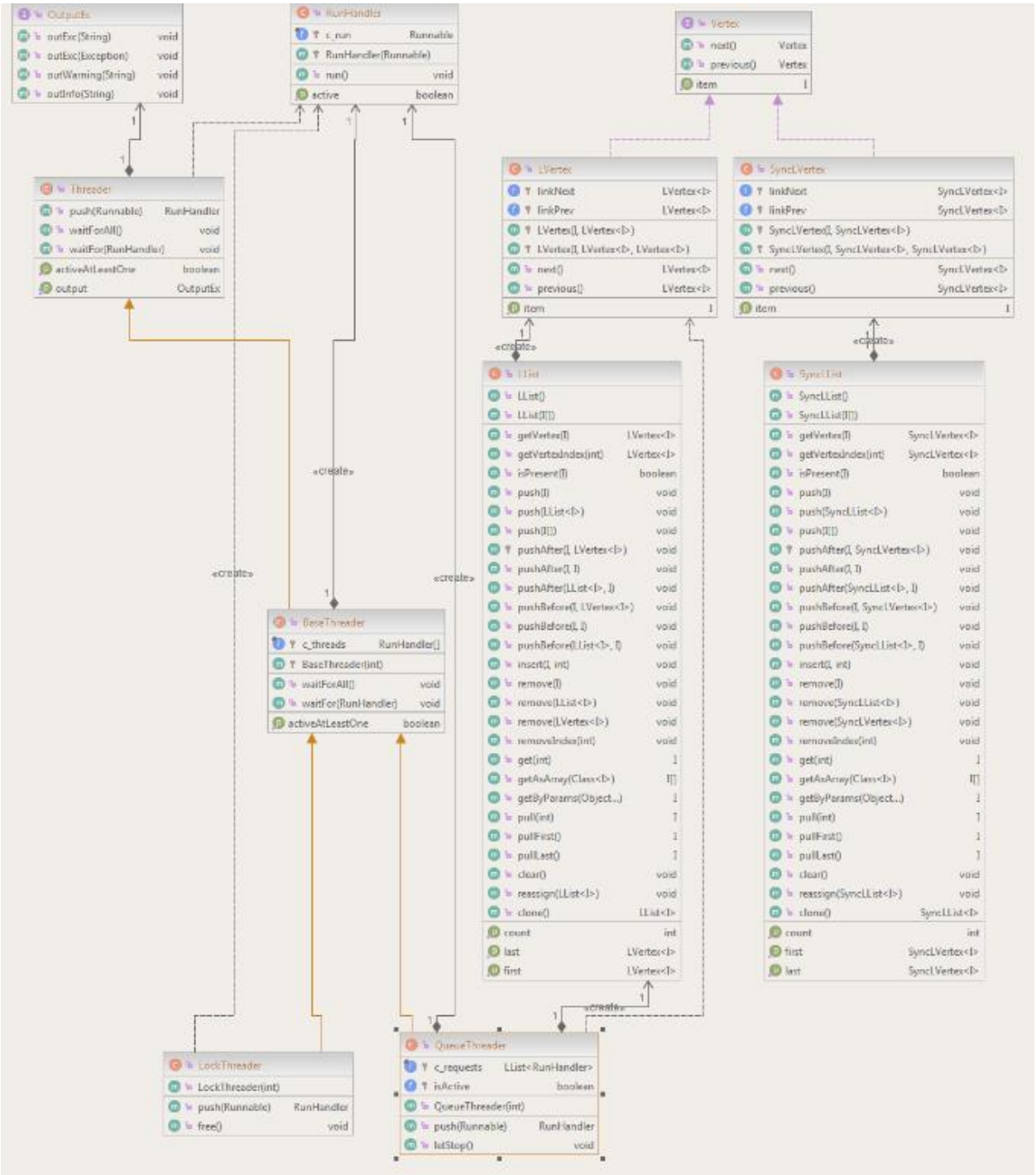

Рисунок 11 – Діаграма класів для двигуна гри

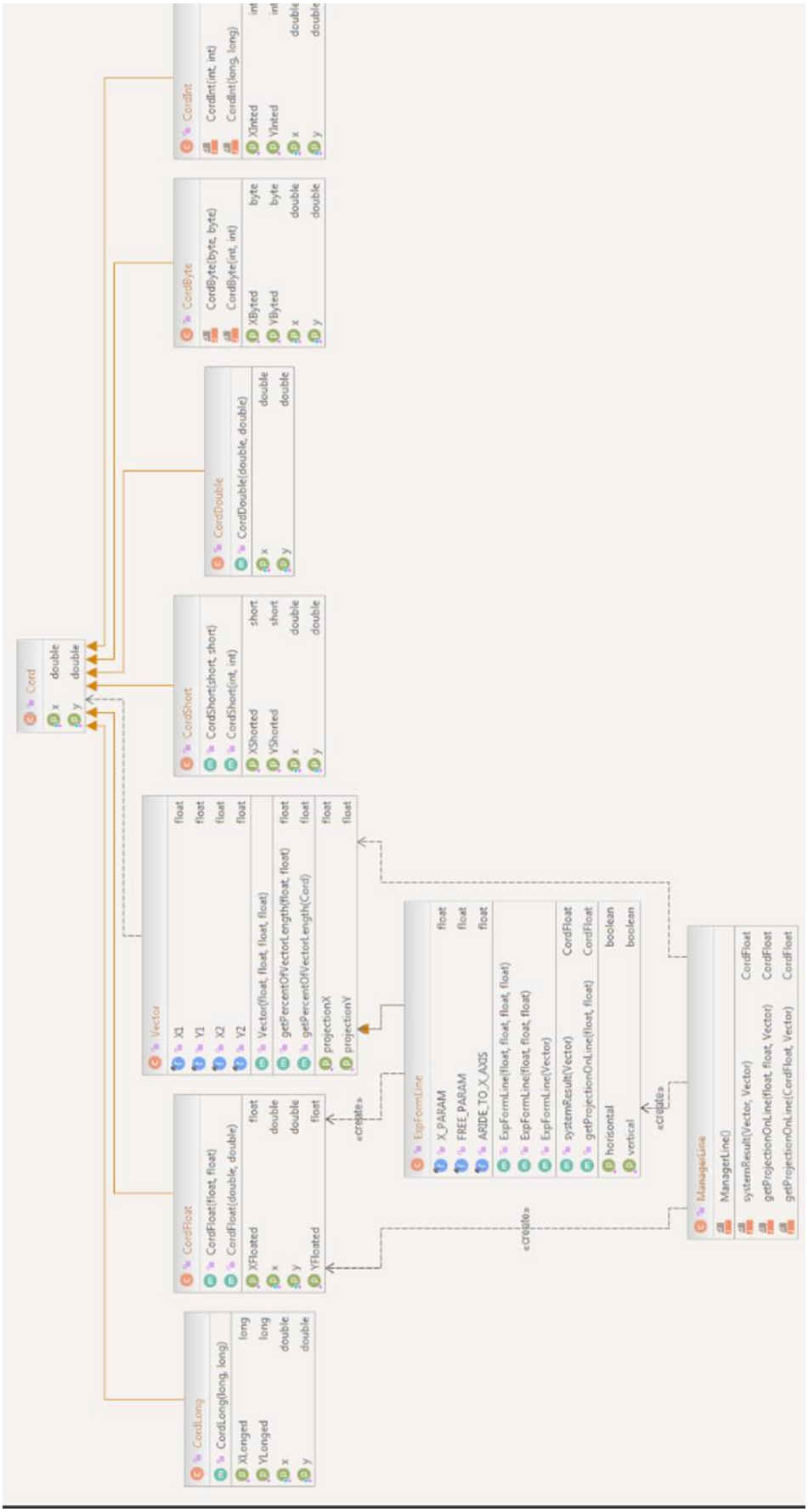

Рисунок 12 – Реалізація класу Cord

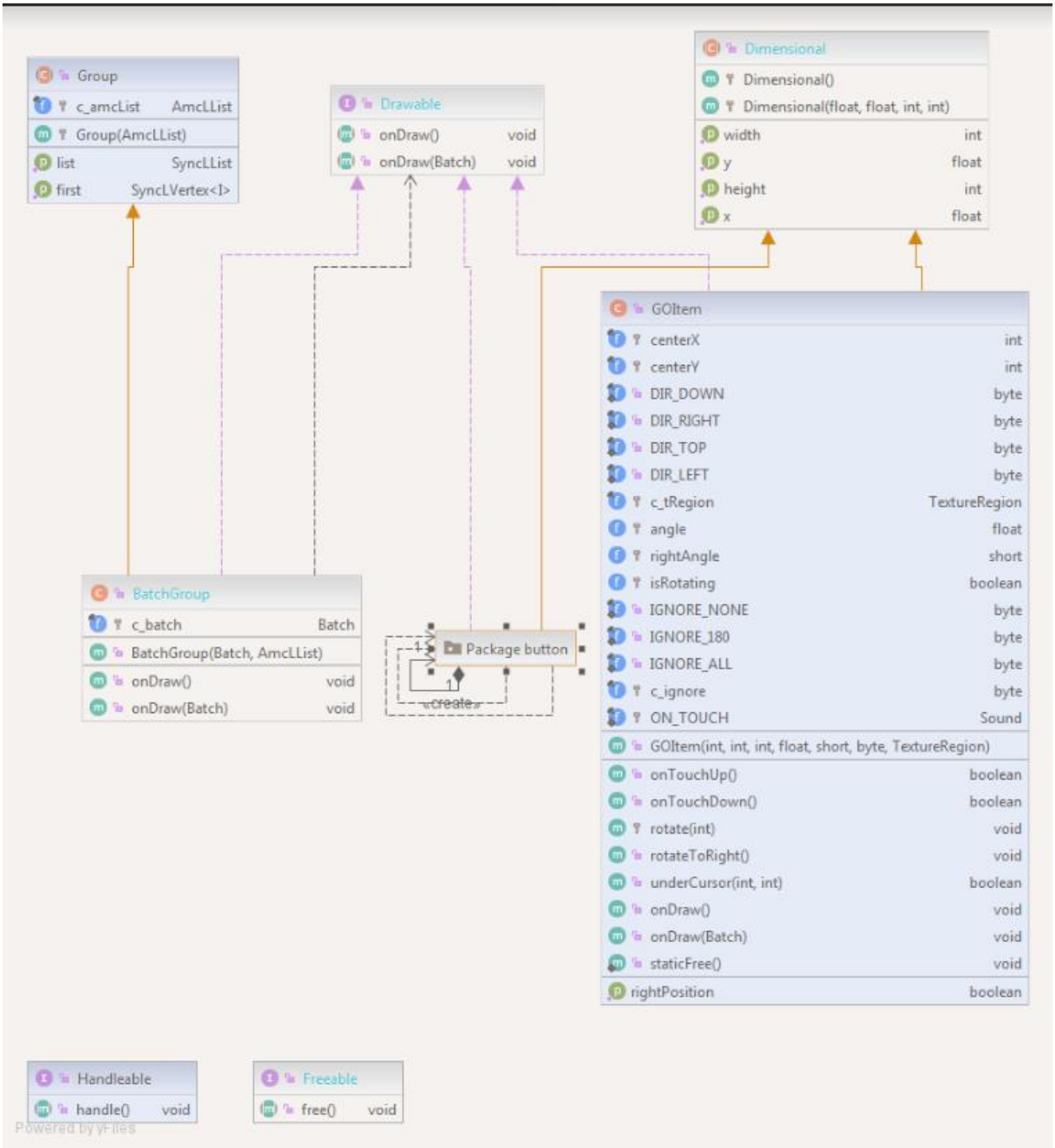

Рисунок 13 – Діаграма класів для Package button

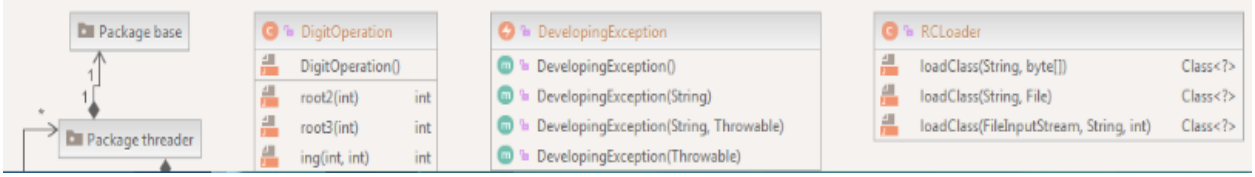

Рисунок 14 – Діаграма класів для Package base, Package threador

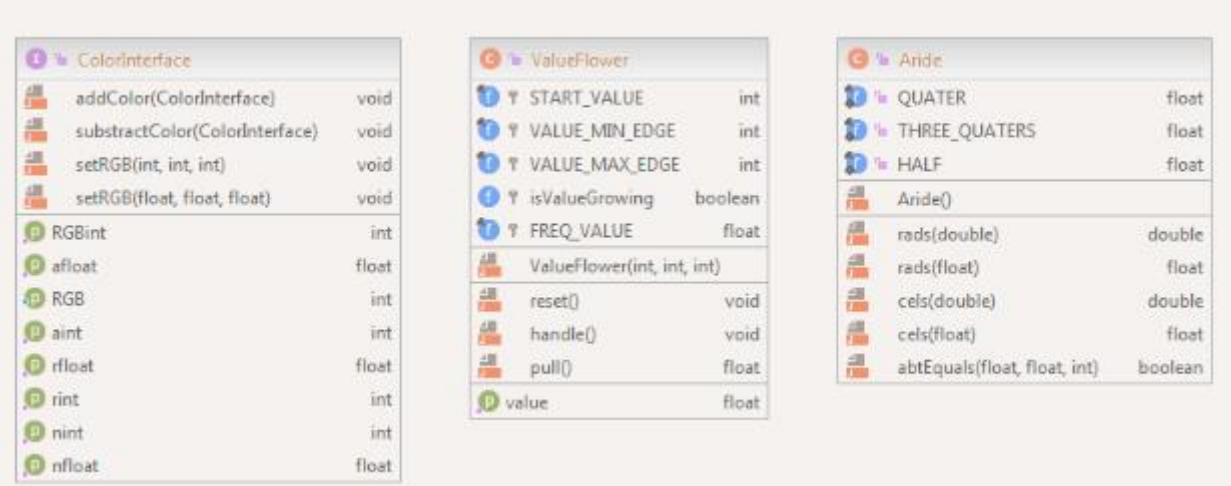

Рисунок 15 – Діаграма класів: допоміжні елементи системи

2.3 Оцінка можливості подальшої реалізації відеоігри

Під час проектування слід оглянути подальші можливості проекту розробки. Так, проект «Loops» у разі успішного завершення дозволить вийти на ринки поширення ігрових додатків, зокрема такі як Google Play, за рахунок використання в своїй основі кроссплатформенного фреймворка libGDX, що підтримує всі основні мобільні і стаціонарні платформи (MacOS, Windows, Linux, Android , ios, HTML5).

Використання libGDX і поширення написаних з його допомогою відеоігор є абсолютно безкоштовним і не зажадає ніяких відрахувань надалі. А за рахунок впровадження в них рекламних банерів, за показ яких будуть нараховуватися кошти від рекламодавців, або за рахунок введення в гру різних платних косметичних функцій або платного контенту, розробник може отримати певний прибуток, яка залежить від якості продукту і інтересу користувачів [14][1)].

Також слід враховувати такий аспект, як фреймворк постійно розвивається, у розробників з'являються нові можливості для поліпшення свого продукту. Однак, внаслідок зміни документації деякі старі функції

[1)] [14] libGDX − фреймворк для разработки игр. URL: http://www.libgdx.ru/. (дата звернення 04.03.2019).

перестають працювати або починають працювати з помилками на нових версій. Тому необхідно постійна підтримка програми.

Рівень розвитку засобів розробки кроссплатформенних додатків, дозволяє досить швидко, просто і без істотних економічних витрат на ресурси створювати якісні ігрові продукти, затребувані на сучасному ринку відеоігор.

Тому зараз є всі можливості, необхідні для реалізації подібної відеоігри з необхідними характеристиками при існуючих обмеженнях для всіх кроссплатформеннх ігрових проектів [16][1)].

[1)] [15] Обзор бесплатного графического редактора Paint.NET. URL: https://www.ixbt.com/soft/paint-net.shtml. (дата звернення 21.03.2019).

3 ОПИС РЕАЛІЗАЦІЇ

3.1 Графіка і спрайти

При розробці графічних додатків часто виникає ситуація, коли є безліч файлів малюнків невеликих розмірів, які файли спрайтів анімації, які було б доцільно помістити в один великий графічний файл, так званий "атлас". Атлас – це велике зображення, що містить деякий безліч зображень меншого розміру, які можна вибирати методом кліппінга прямокутного регіону.

До переваг використання спрайтів можна віднести наступне:

- скорочення кількості файлів малюнків прискорює завантаження програми.
- якщо зображення містять прозорі області, то їх можна видалити, заощадивши тим самим деяку кількість відеопам'яті за рахунок більш щільної упаковки елементів графіки (актуально для ігор і анімації за участю спрайтів різних розмірів) [21][1)].

Будь-яке двовимірне зображення можна розбити на базові елементи, тобто створення двовимірної графіки як об'єднання цих елементів.

Спрайт – це графічний об'єкт в комп'ютерній графіці. Найчастіше – растрове зображення, яке можна відобразити на екрані.

Атлас спрайтів є растровим зображенням, об'єднуючим спрайт в набір розкадрувань анімацій, або іншим комплектом спрайтів.

Для проекту створення спрайтів вироблялося в програмі під назвою PaintDotNet. Приклад видів всіх видів ланок приведено на рис. 16.

Тут в грі присутній 5 кольорів – червоний, синій, зелений, фіолетовий і сірий. Так само, на стадії створення спрайтів була вирішена перша проблема – однотипність зв'язків.

[1)] [21] В каком разрешении рисовать спрайты для мобильной 2D игры? URL: https://progressor-blog.ru/gamedev/v-kakom-razreshenii-risovat-sprajty-dlya-mobilnoj-2d-igry/. (дата звернення 11.03.2019).

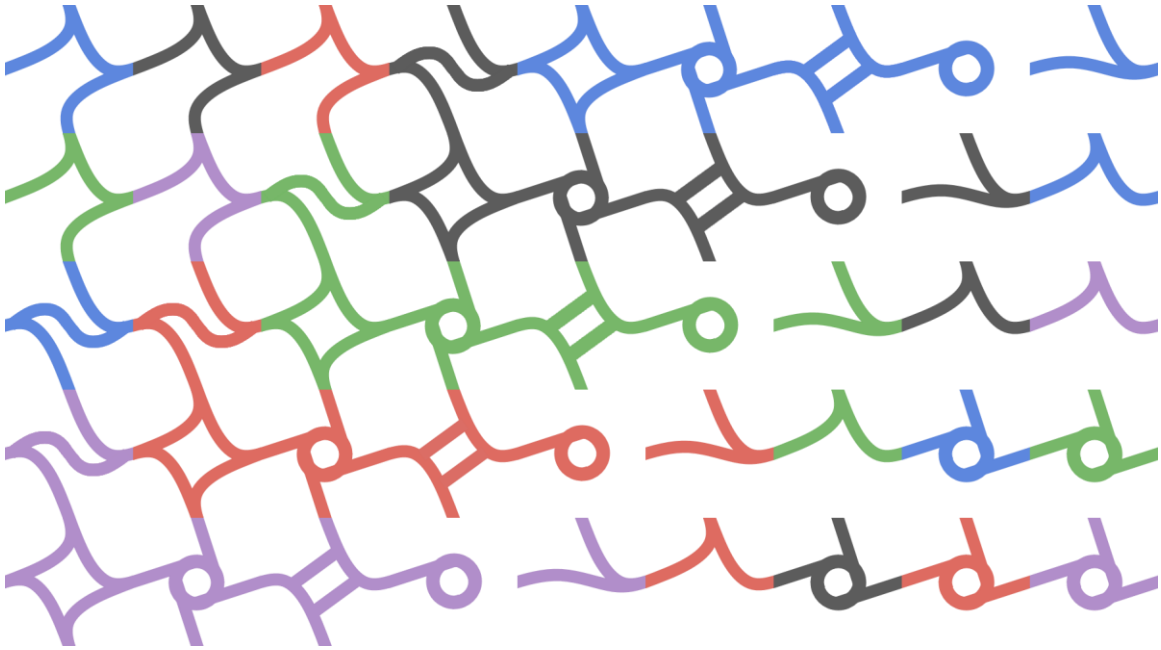

Рисунок 16 – Всі спрайти ланок в одному атласі

Як видно вище з малюнка в грі присутній 5 кольорів – червоний, синій, зелений, фіолетовий і сірий. Так само, на стадії створення спрайтів була вирішена перша проблема – однотипність зв'язків. Тепер, наприклад, зв'язок, що має 4 контакту має не один спрайт, а п'ять (рис. 17). Решта також мають кілька спрайтові уявлень.

Рисунок 17 – Види спрайтів звена одного типу

Для простого управління в грі досить шести кнопок (рис. 18), а саме «Нова гра», «Налаштування», «Оцінити» і «Вихід» в меню, «Підказка» і «Назад в меню» безпосередньо в грі.

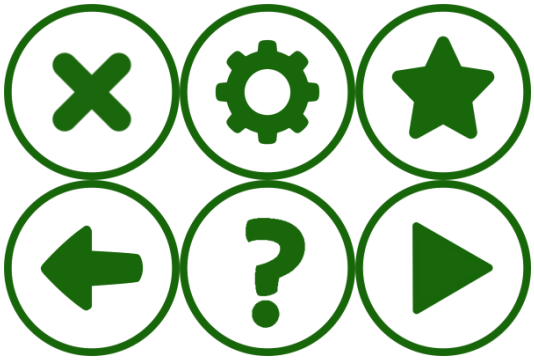

Рисунок 18 – Спрайти кнопок

Фон являє собою одну з головних задач при проектуванні. Як було сказано раніше у огляді аналогів гри, графіка хоч і приваблива, але з часом тисне на очі за рахунок однотонності фону (рис. 19). Тому було прийнято рішення позбавитись від однорідності і створити власний фон.

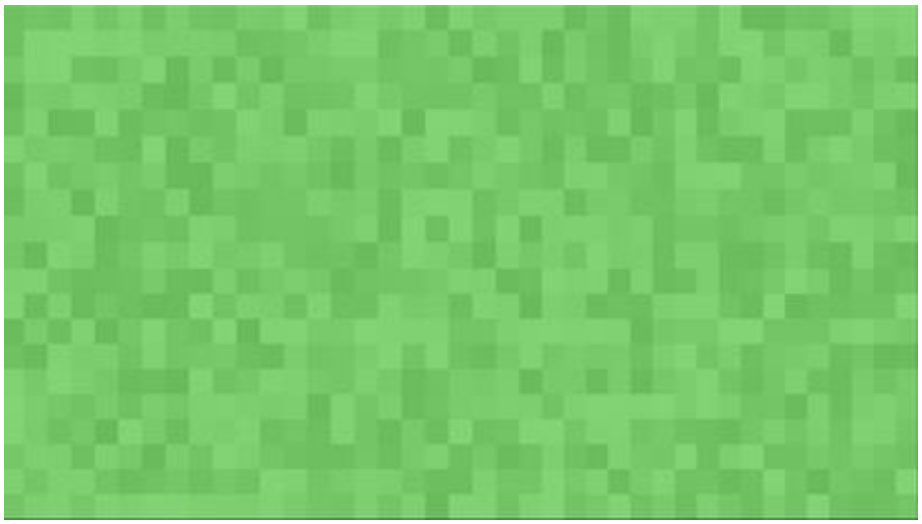

Рисунок 19 – Однорідний фон, розділений на квадрати

Для цього був розроблений алгоритм, який розбивав весь екран на квадрати, які в свою чергу повільно змінювали свій відтінок від світлішого до більш темного і назад. Такий ефект зменшить навантаження на очі під час гри.

Алгоритм реалізації такого ефекту наведено нижче:

```
for (int x = 0; x < countX; ++x) {
                int cordX = ofsX;
                int sizeX = squaresSizeX;

       if (l_pixelsOverX > 0) {
                      ++sizeX;  --l_pixelsOverX;
                }

squares[y][x] = new SquareGdx (cordX, cordY, sizeX, sizeY, new
ValueFlower (0, 1, (int) (random.nextFloat() * DIFF_FREQ +
MIN_FREQ)));
}
```

Це дало значний успіх, а саме, дійсно знизило тиск на очі за рахунок плавної динаміки фону, і при цьому він не нав'язливий і не відволікає від гри. Але було прийняте рішення наступним кроком реалізували градієнтний розпад, що лише додало сили вже і так гарному ефекту.

Суть в тому, що десь була велика концентрація світлого відтінку, а десь більш темна, і квадратики це сприймали, роблячи на перший погляд хаосну картину, але на загальному плані градиентну. Код реалізації має наступний вигляд:

```
float COEFF_X = modelWidth /(float) Gdx.graphics.getWidth();
float COEFF_Y = modelHeight /(float) Gdx.graphics.getHeight();

for (int y = 0; y < LEN_SQR_Y; ++y)

     for (int x = 0; x < LEN_SQR_X; ++x)
square.resetColor (modelMatrix [(int) ((square.getY() +
(square.getHeight() >> 1)) * COEFF_Y)][(int) ((square.getX() +
(square.getWidth() >> 1)) * COEFF_X)]);
```

3.2 Реалізація алгоритму «CoverScreen»

Так як в грі передбачався генератор рівнів, то не можна було визначити скільки часу займе генерація нових об'єктів і очищення старих, і не можна було допускати щоб гравець при переході з рівня на рівень просто чекав поки поле оновиться. Ну і звичайно ж постало питання, як саме буде візуально оновлюватися поле.

Для цього був розроблений підхід «CoverScreen». Його суть в тому, щоб, перед тим, як видаляти ланки з поля, закрити все перехідним фоном, потім видалити старі ланки, згенерувати нові і розкрити CoverScreen.

І це спрацювало, але є маленький момент – коли оновлення відбуватися дуже швидко, то з'являвся ефект мерехтіння від того, що CoverScreen не встигнувши з'явився вже зникає. Для цього йому була виведена затримка, яка відраховується в паралельному потоці. Алгоритм довелося трохи ускладнити за рахунок синхронізації потоків, але результат вартий того.

«CoverScreen» є спадкоємцем класу «Dimensional» і реалізатором інтерфейсу «Drawable», що дозволяє йому відображатися на екрані, але при цьому не бути ігровим об'єктом. Перед очищенням ігрового поля і його

перегенерація створюється екземпляр класу «CoverScreen» з довільним типом покриття. Покриття визначається через псевдослучайную генерацію.

Далі, викликається метод hide(), який в паралельному потоці покриває весь екран, не даючи бачити нічого під ним, тобто ні кнопок ні зв'язків, нічого. Це потрібно для того, щоб під час покриття паралельно проводити генерацію, економлячи час. Тому створюємо екземпляр класу «AmcList», який буде помістити вже нові ігрові об'єкти.

Визначаємо нову колірну гамму, на її підставі генеруємо фон і зв'язку. Після викликаючи метод waitTillInactive(), який блокуватиме потік до тих пір, поки екран повністю не заповнить екран.

Як тільки екран повністю стане покритий, блокування знімається і викликаються методи clear() для очищення всього ігрових обьетов і reassign(), який переносить всі новий ігрові об'єкти з локального списку в глобальний.

Після цього викликаємо метод unhide() для того щоб розкрити екран і знову блокуємо потік, але вже за допомогою waitTillActive(), щоб зачекати повного розкриття екрану. Коли екран польностью розкритий, можна включати введення.

Для більш зручного та гнучкого використання потоків в даній задачі, був розроблений клас «ThreadManager», який здійснює всі необхідні перевірки і управління потоками в цілому.

Ось приклади того, як працюють вище описані методи:

Першим методом розберемо hide(). Спочатку ми в синхронізованою секції визначаємо є екран вже покриває. Якщо немає, то виставляємо його в режим покриває, інакше просто виходимо з методу:

```
synchronized (this){
                if ( ! v_isActive) v_isActive = true;
                else return;
          }
```

У разі, якщо ми тільки що виставили режим покриває, то відразу ж з методу не виходимо і створюємо новий потік, в якому визначаємо сам процес покриття. Сам процес описаний в лямбда-функції, тому спочатку визначимо алгоритм, а лише потім створимо потік і закинемо в якості параметра лямбда.

Алгоритм дуже простий, ми кожну ітерацію додаємо до альфа-каналу зображення порцію (PORTION), і, якщо альфа перевищує значення 1, то обриваємо цикл в потоці.

```
Runnable lambda = () ->
{
alpha += PORTION;
               if (alpha >= 1) {
                    c_batch.setColor(1, 1, 1, 1);
                    break;
               }
               c_batch.setColor(1, 1, 1, alpha);
          }
```

Далі створюється поток:

```
Resource.getInstance().getThreadManager().put(lambda);
```

Методи waitTillInactive() і waitTillActive() практично ідентичні, відрізняються лише перевіркою на рівність з активністю, тобто або true або false:

```
do{
     synchronized (this){
          if ( ! v_isActive)
          return; }
     }
while (true);
```

Наступним слід розглянути приклад створення фону. Щодо згенерованого псевдовипадковим алгоритмом числа визначаємо колірну гамму. Для цього це число передаємо в switch і вибираємо файл, відповідний певної гамі:

```
switch (colorType){
     case 0 :
          fileHandle=Gdx.files.internal("greenModel.data");
```

```
        break;
        //....
        default :
        throw new IllegalArgumentException("There's no
                                        such color style with
this id : '" + colorType + '\'');
}
```

3.4 Реалізація генератору ланок у грі

Однією з основних особливостей даного проекту над аналогами є наявність генератора. Його перевага в тому, що рівні тепер не статичні, кожен раз зайшовши в гру або просто перейшовши на новий рівень, він буде унікально згенерований.

Друга перевага – це нескінченна кількість рівнів. Розробники аналогів викручуються тим, що час від часу вони оновлювали гру, тим самим додаючи нових рівнів раз по раз. НДаному проекту це не потрібно. Генератор має кілька підходів до створення:

- відображення;
- шаблони;
- вибірка елементу.

Отже, спочатку визначається рівень генерації – все поле, половина або чверть. Якщо генерирувалась половина або чверть, то інші частини просто віддзеркалює щодо згенерованого шматка поля, що, до речі, дало гарні симетричні форми не гірше статичних рівнів.

Наступним етапом є саме генерація, де поле засівають рандомно вибраними з бази даних шаблонами. Даний код демонструє код одного з шаблонів:

```
bmap = new byte[1][3];
        bmap[0][0] = DIR_RIGHT;
        bmap[0][1] = DIR_RIGHT | DIR_LEFT;
        bmap[0][2] = DIR_LEFT;
```

```
lstLoops.push(new Pattern(bmap));
```

Розглянемо роботу генератору більш детально. Отже, є поле, яке повинно бути всіяне ланками, для цього обходимо кожну клітинку поля:

```
for (int yi = 0; yi < cellsY; ++yi)
                for (int xi = 0; xi < cellsX; ++xi)
```

Клонуємо список шаблонів і отримуємо їх кількість:

```
LList<Pattern> last = c_lstLoops.clone();
          int count = last.getCount();
```

Потім обходимо весь список і перевіряємо на валідність шаблону на поточному осередку і, якщо шаблон не валиден, то він видаляється зі списку.

```
Pattern map = last.get(i);

      if ( ! map.isValid(xi, yi, cellsX, cellsY)) {
                last.remove(map);
                --count;  --i;
           }
```

Далі потрібно перевірірити кількість шаблонів в списку після відсіювання невалидность, і якщо контактів немає, то пропускаємо ітерацію, оскільки генерувати нічого.

```
if (last.getCount() == 0)
                continue;
```

І все ж якщо перелік не порожній, то рандомно вибираємо шаблон і поміщаємо на поточну комірку:

```
Pattern map = last.get(random.nextInt(last.getCount()));
          map.pushPattern(xi, yi, bmap);
```

По закінченню генератор повертає карту ігрового поля, яка є масивом типу byte. Кожним елементом масиву є напрямком ланки. Потім клас

«AmcMode», отримавши від генератора карту, по ній створює вже реальні ланки на поле.

Створюючи ланки, «AmcMode» вибирає рандомно спрайт типу ланки, яке розглядається.

Ось приклад визначення ланки по карті:

```
if ((directions & DIR_DOWN) == DIR_DOWN)
     wasDown = true;
           else wasDown = false;

if ((directions & DIR_TOP) == DIR_TOP)
      wasTop = true;
           else wasTop = false;

if (wasDown ^ wasTop) {
               if (wasDown) {
                    directions &= ~DIR_DOWN;
                    directions |= DIR_TOP; }
else{
                    directions &= ~DIR_TOP;
                    directions |= DIR_DOWN;
}}
```

Наступний код демонструє знаходження спрайту у зв’язках:

```
switch (count) {
     case 1 :  strBuilder.append("one");
appendItemLevel(0, strBuilder);
}

switch (colorType) {
     case 0 :  strBuilder.append("green");
}

return Resource.getInstance().getTRegion(strBuilder.toString());
```

3.5 Реалізація руху елементів

Наступним кроком необхідно реалізувати інтерфейс взаємодії гравця з ланками, а саме після натискання обертати ланки за годинниковою стрілкою. Отже, для початку визначимо методи onTouchDown() і underCursor() з інтерфейсу «Touchable». Наступний блок коду демонструє визначення натискання на ланка:

```
@Override
     public boolean underCursor(int x, int y)    {
          return(x > this.x && x < this.x + width && y > this.y
&& y < this.y + height);
     }
```

Тепер необхідно визначити дію натискання на ланка. Спочатку в синхронізація блоці перевіряємо не обертається чи ланка, тому що якщо так, то проворот робити не треба (захист від постійного натискання на ланка):

```
synchronized (this) {

 if (isRotating)
     return true;
 isRotating = true;
}
```

І, якщо ланка не обертається, то програти звук і провернути елемент на 90 градусів за годинниковою стрілкою:

```
ON_TOUCH.play(0.7f);
rotate((int)angle + 90);
```

3.6 Перевірка рішення рівня гри та підказки

Після кожного натискання необхідно здійснювати перевірку на завершення рівня. Для цього необхідно рекурсивно обійти поле, шукаючи при цьому цикли, якщо всі елементи є частиною циклів, то рівень пройдений. У теорії, обхід починається з першої ланки. Перевіряється, на які сусідні осередки спрямовані його контакти.

Якщо на цих сусідніх осередках знаходяться інші ланки і їх контакти спрямовані в бік поточного ланки, то обхід перескакує на сусіднє ланка, і так виробляється пошук циклу. Якщо ж контакти спрямовані на сусіднє ланка, яке своїми контактами не спрямоване до поточного або сусід зовсім відсутня, то це призводить до обриву рекурсії і перевірки в цілому.

Наступний код відображаю роботу алгоритму:

```
for (SyncLVertex<GOItem> vertex = itemGroup.getFirst();
    vertex != null;
    vertex = vertex.next())
if ( ! vertex.getItem().isRightPosition())
    return;
     Lcynte.mode.nextLevel();
```

Підказки є останнім нововведенням, яке відсутнє в аналогах. Підказка просто рандомно визначає три ланки, які знаходяться в неправильній позиції і провертає їх на правильну. А влаштована підказка наступним чином: спочатку клонується карта:

```
SyncLList list = itemGroup.getList().clone();
```

Наступним кроком проводиться обхід циклу на три ланки (саме стільки використовується за одне використання підказки):

```
final int HINT_ITEMS = 3;
     for (int i = 0; i < HINT_ITEMS;)
```

Всередині циклу ми перевіряємо чи є в нашому списку ланки, і якщо ні, значить все ланки стоять в правильному положенні:

```
int count = list.getCount();
if (count == 0)  break;
```

Інакше, необхідно рандомно витягнути ланка зі списку. І в даному випадку витягнути значить, що ланка віддаляється зі списку, для цього ми і клонували карту. А видаляємо ланка зі списку, щоб при наступній ітерації циклу при черговому виборі ланки ми не потрапили на раніше вбрання.

```
item = (GOItem) list.pull(random.nextInt(count));
```

Перевіряємо, чи знаходиться елемент в потрібній позиції, і якщо так, то пропускаємо ітерацію циклу без інкрементація лічильника, адже інакше ми підкажемо вже не три ланки, а менше. А якщо все таки ланка варто в неправильній позиції, то провертає на правильну.

```
if (item.isRightPosition()) continue;
item.rotateToRight();
     ++i;
```

Кінцевий результат розробки гри представлено на рис. 20-21:

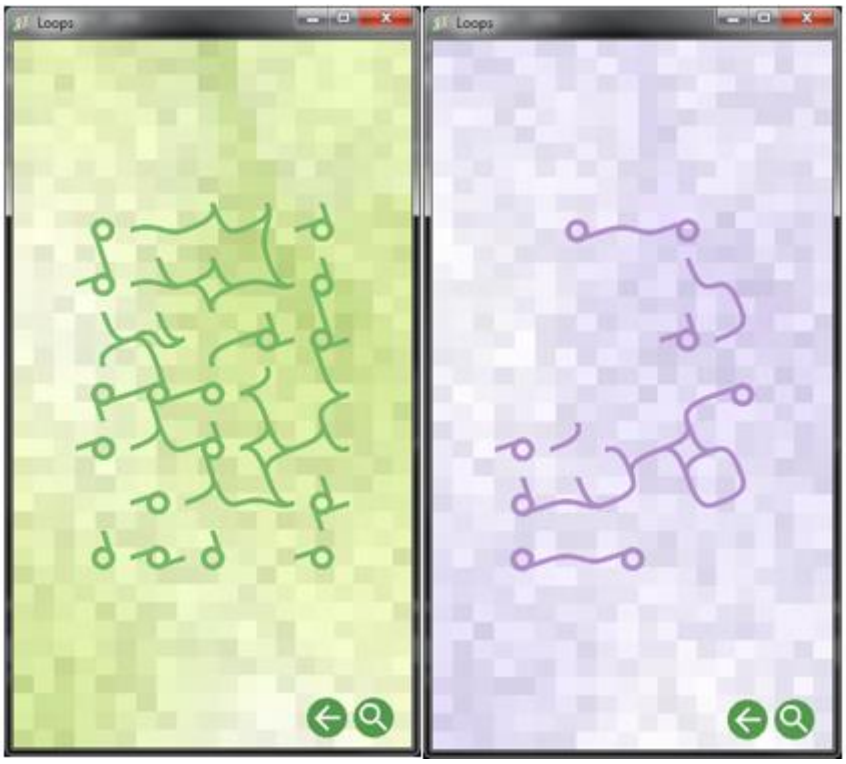

Рисунок 20 – Робота ігрового додатку

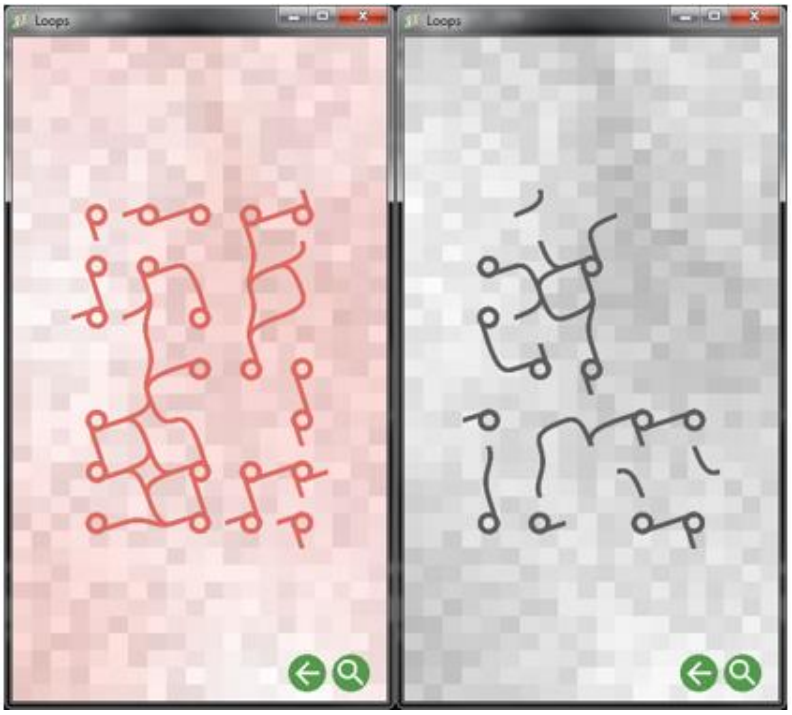

Рисунок 21 – Приклад рівнів гри

ВИСНОВКИ

Android – ідеальна ОС для початківців розробників ігрових додатків. Вона користується популярністю у широкої аудиторії користувачів мобільних пристроїв і має досить потужні інструменти програмування. Розробка мобільних додатків може принести вагомий прибуток навіть при невеликих витратах на ресурси. І навіть незважаючи на високу конкуренцію, багатьом розробникам вдається досягти певного успіху, так як запити користувачів мобільних додатків ростуть, а старі додатки з часом втрачають свою актуальність, або зовсім перестають отримувати підтримку.

Логічні ігри на Android ніколи не втратять популярності. Головоломки – ігри, які мають чимало спільного з логічними іграми, допомагають розвивати інтуїтивні здібності і стратегічне мислення. Іноді завдання здаються неймовірно складними, так як їх рішення вимагає послідовного застосування певних прийомів і їх поєднань.

За результатами проектної частини диплому було зроблено:

- аналітичний огляд предметної області, а саме створення програмних додатків під мобільні пристрої на Android;
- зроблено аналіз аналогів об’єкту розробки, на основі якого поставлені вимоги і сформовані мета роботи;
- обрані програмні засобі для реалізації проекту;
- за допомогою UML-засобів побудовані діаграми діяльності для алгоритму «CoverScreen», а також діаграма класів для всіх елементів системи розробки.

За результатами практичної частини виконано:

- створені графічні об’єкти для ігрового додатку, включаючи спрайти всіх елементів;
- розроблено алгоритм вирівнювання відтінків для фонових зображень.

- реалізовано функцію генератору ланок для рандомного формування рівнів гри;
- створено алгоритм взаємодії з елементами у єдину систему.

В рамках подальшого розвитку програмного продукту стане тестування і розширення функціоналу для комерційного використання.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Топ 30 лучших игр на Андроид. URL: http://gamebizclub.com/rejtingi/luchshie-igry-na-android/. (дата звернення 02.02.2019).
2. Infinity Loop: Energy − головоломка с фантастической атмосферой. URL:https://lifehacker.ru/infinty-loop-energy/. (дата звернення 3.02.2019).
3. Игры для Android. URL: https://androidinsider.ru/igry. (дата звернення 5.02.2019).
4. Топ-8 самых популярных игр на Android. https://keddr.com/2017/05/top-8-samyih-populyarnyih-igr-na-android/. (дата звернення 12.12.2019).
5. Топ игр на андроид: обзор 10 лучших приложений. URL: https://www.fly-phone.ru/notes/igry-dlya-smartfonov/top-10-igr-na-android/. (дата звернення 13.02.2019).
6. Обзор Infinity Loop: HEX. URL: https://pdalife.ru. (дата звернення 13.02.2019).
7. Описание игры Plumber 10k для смартфонов с ОС Android. URL: https://plumber10k.wordpress.com/2012/09/24/opisanie-igry-plumber-10k-dlya-smartfonov-s-os-android/. (дата звернення 23.02.2019).
8. IntelliJ IDEA. URL:https://jetbrains.ru/products/idea/. (дата звернення 14.02.2019).
9. Навчальні матеріали з інформатики » Програмування мовою Java. URL: https://www.ua5.org/java/. (дата звернення 14.02.2019).
10. Інформаційний блог з програмування. URL: https://habr.com/ru/hub/java/. (дата звернення 05.03.2019).
11. Java − объектно-ориентированный язык программирования. URL: https://infoblog1.ru/teach/java/. (дата звернення 01.03.2019).

12. Подробнее о технологии Java. URL: https://www.java.com/ru/about/. (дата звернення 01.03.2019).
13. Обзор игрового движка LibGDX. https://gamedevmania.ru/engines/libgdx. (дата звернення 02.03.2019).
14. libGDX − фреймворк для разработки игр. URL: http://www.libgdx.ru/. (дата звернення 04.03.2019).
15. Обзор бесплатного графического редактора Paint.NET. URL: https://www.ixbt.com/soft/paint-net.shtml. (дата звернення 21.03.2019).
16. Вимоги до функціональних характеристик. URL: https://studopedia.su/13_68404_vimogi-do-funktsionalnih-harakteristik.html. (дата звернення 11.03.2019).
17. Ведення в UML 2.0. URL: http://bourabai.kz/dbt/uml/index.htm. (дата звернення 11.03.2019).
18. Моделювання на UML. URL: http://book.uml3.ru/. (дата звернення 11.03.2019).
19. Виды диаграмм UML. URL: https://studizba.com/lectures/10-informatika-i-programmirovanie/273-uml/3457-2-vidy-diagramm-uml.html. (дата звернення 11.03.2019).
20. UML-диаграммы классов. URL: https://prog-cpp.ru/uml-classes/. (дата звернення 11.03.2019).