МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ   
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

КАФЕДРА ТЕХНОЛОГИЙ ЗАЩИТЫ ИНФОРМАЦИИ

КУРСОВАЯ РАБОТА   
ЗАЩИЩЕНА С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Жиданов К.А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ПОЯСНИТЕЛЬНАЯ ЗАПИСКА  К КУРСОВОЙ РАБОТЕ  Приложение на ОС Android основанное на функции распознавания лиц в доступном обзоре фронтальной камеры устройства.  по дисциплине: Технологии и методы программирования |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | 3645 |  |  |  | Сомов Р.А. |
|  | 3645 |  |  |  | Таран Г.С. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2018

**Оглавление**

[1. Введение 3](#_Toc515978040)

[2. Репозиторий 5](#_Toc515978041)

[2.1 Выбор репозитория 5](#_Toc515978042)

[2.2 Создание репозитория 5](#_Toc515978043)

[3. Task tracker 6](#_Toc515978044)

[4. Техническое задание 6](#_Toc515978045)

[5. Выбор игрового движка 6](#_Toc515978046)

[6. Планирование спринтов 6](#_Toc515978047)

[7. Спринт Ревью 8](#_Toc515978048)

[8. Анализ результатов 9](#_Toc515978049)

[9. Вывод 10](#_Toc515978050)

**1. Введение**

Целью работы является использование гибкой методологии разработки (англ. Agile software development) в разработке программного обеспечения.

Agile-методология – это серия подходов к разработке программного обеспечения, ориентированных на использование итеративной разработки, динамическое формирование требований и обеспечение их реализации в результате постоянного взаимодействия внутри самоорганизующихся рабочих групп, состоящих из специалистов различного профиля.

Применяется как эффективная практика организации труда небольших групп (которые делают однородную творческую работу) в объединении с управлением ими комбинированным (либеральным и демократическим) методом.

Основные идеи:

1) Люди и их взаимодействие важнее процессов и инструментов;

2) Работающий продукт важнее исчерпывающей документации;

3) Сотрудничество с заказчиком важнее согласования условий контракта;

4) Готовность к изменениям важнее первоначального плана;

Работа спроектирована по каскадной модели Waterfall (модель «Водопад», в которой процесс выглядит как поток последовательно проходящей фазы анализа требований, проектной реализации, тестирования интеграции и поддержки. Коммуникация между участниками проекта производилась по гибкой методологии Agile.

Для классической модели разработки программного обеспечения выделяют следующие этапы:

1. Анализ требований проекта. Определяются программные требования для информационной предметной области системы.
2. Проектирование. Разрабатывается и формулируется логически последовательная техническая характеристика программной системы. Детализация системы.
3. Реализация ПО. Воплощение полноценного проекта.
4. Тестирование продукта. Тестовая эксплуатация продукта
5. Интеграция системы. Включает установку и официальную приёмку продукта.
6. Поддержка. Предоставление технической помощи по продукту после запуска в коммерческую эксплуатацию.

Эта модель подразумевает строго последовательное и однократное выполнение каждой фазы проекта. Переход от одной фазы к другой возможен только после успешного завершения предыдущего этапа.

Сама методология Agile является более совершенным способом организации групп людей, работающих над одним проектом. Но для того, чтобы полностью перейти к Agile нам нужно:

1. иметь проект, времени на который будет выделено больше 4-недель;
2. проект должен быть важным, но если вдруг проект не будет готов к пилоту, то это не будет проблемой;
3. группа людей, в которой будет много разных специалистов, таких как: Product Owner, Product Owner, Product Owner;

Данным требованиям Agile в данном проекте не подходит 3 пункт –большая команда. В нашей команде имеется лишь 2 человека, оба выполняют все функции большой полноценной специализированной команды. Поэтому полностью перейти к методологии Agile в данном проекте не представляется возможным. Но для маленьких команд и для такого не крупномасштабного проекта Waterfall достаточно полезен и подходит. В нашей работе мы разрабатываем приложение для распознавания лица в форме 2D игры. Игра представляет из себя наклоны головой к объективу камеры и наоборот, отвод головы из доступного обзора камере. Зачем? Это весело.

**2. Репозиторий**

2.1 Выбор репозитория

Если есть несколько разработчиков, работающих с репозиторием, то у каждого на локальной машине будет полная копия этого репозитория. Конечно, где-то есть и центральная машина, с которой можно клонировать репозиторий. Это напоминает SVN. Как плюс идёт то, что если вдруг у вас нет доступа к интернету, сохраняется возможность работать с репозиторием. Останется только один раз сделать синхронизацию, и все остальные разработчики получат полную историю.

GIT распределяется, а SVN - нет. GIT сохраняет метаданные изменений, а SVN целые файлы. Это экономит место и время. Система создания branches, versions и прочее в GIT и SVN отличаются значительно.

GIT удобнее, позволяет сделать больше и проще всякого разного и интересненького. Например, реорганизовать порядок и содержание коммитов. Работа с ветками удобнее и быстрее, через git stash удобно проверять перед коммитом что не забыл добавить вновь созданный файл. Удобнее переносить свои наработки в разные ветки - создать локально ветку, в ней вести разработку, потом просто сделать merge ветки, в svn же нужно просматривать лог на предмет нужных коммитов и переносить их ручками.

Так же распределённость git может быть полезна при восстановлении данных. Жёсткие диски любят выходить из строя в неподходящий момент, всё можно восстановить благодаря локальным копиям.

2.2 Создание репозитория

На сайте GitHub.com создаём репозиторий с вложенным файлом README.md (В котором позже будет находиться ТЗ). Выбираем формат репозитория Public, т.к. он бесплатный.

**3. Task tracker**

В данной работе мы выбрали task tracker github, потому что не придётся использовать сторонние ПО, используя репозитории github.

**4. Техническое задание**

Необходимо разработать кроссплатформенное приложение, используя онлайн-хостинг для совместной разработки и контроля версий. Приложение будет ловить в объективе фронтальной камеры лицо и оповещать пользователю об этом: будет реализован фон приложения и функция распознавания лица с возможностью оповещения.

**5. Выбор игрового движка**

Задача должна быть реализована с помощью движка Java через интегрированную среду разработки Android studio для работы с данной платформой. Этот движок был выбран по нескольким причинам: расширенный редактор макетов: WYSIWYG, способность работать с UI компонентами при помощи Drag-and-Drop, функция предпросмотра макета на нескольких конфигурациях экрана. Сборка приложений, основанная на Gradle. различные виды сборок и генерация нескольких .apk файлов, рефакторинг кода, статический анализатор кода (Lint), позволяющий находить проблемы производительности, несовместимости версий и другое и шаблоны основных макетов и компонентов Android.

Также использовалась среда разработки Microsoft Visual Studio (использование высокоуровневого языка С# облегчает разработку за счет уже реализованных элементов).

В качестве онлайн-хостинга должен быть использован GitHub. Он прост в освоении и использовании, имеет удобную систему контроля версий, а также необходимый для разработки функционал бесплатный. Приложение должно управлять с помощью фронтальной камеры. Перед запуском самой игры идёт интро с названием игры. Функционал игры весьма прост: после запуска приложения на экране устройства начинают копиться черные шарики, увеличиваться. Происходит это до появления лица в поле видимости фронтальной камеры - шарики начинают «всматриваться» при фокусировке камеры и вовсе разбегаются при четко определённом контуре лица.

**6. Планирование спринтов**

У планирования спринта две основные задачи: выбор элементов журнала пожеланий для реализации и их декомпозиция. Планирование проводится в самом начале спринта и обычно занимает не более дня для спринта длиной в неделю.

Для того, чтобы спланировать спринт команде нужно знать:

* Объем работ по каждой истории, которую команда собирается реализовать в спринте
* Скорость команды
* Количество часов, которое есть у команды на спринт

В нашем случае, на всю реализацию у нас было 6 недель. Команда состоит из 2 человек, значит, время работы для команды сильно ограничено, но и объем работ в проекте небольшой.

Ниже на рисунке 1 представлены спринты, размещенные на сайте github.com

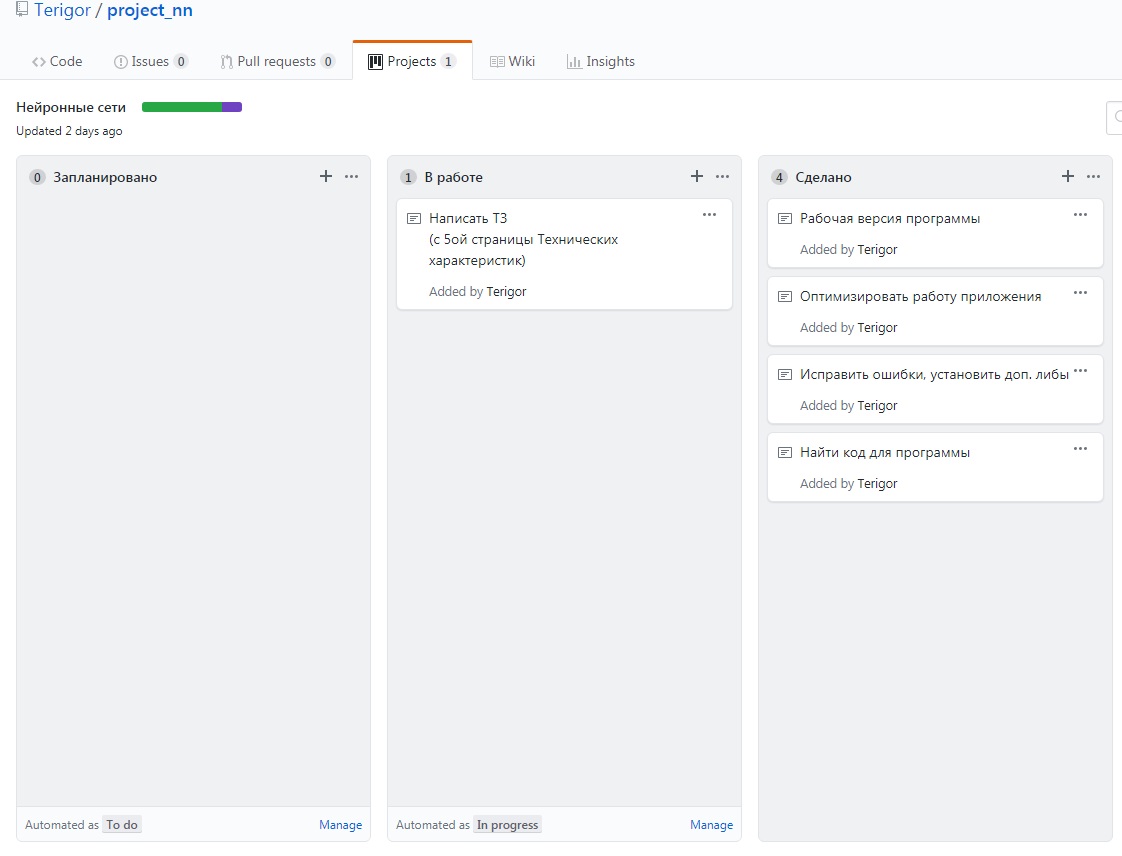


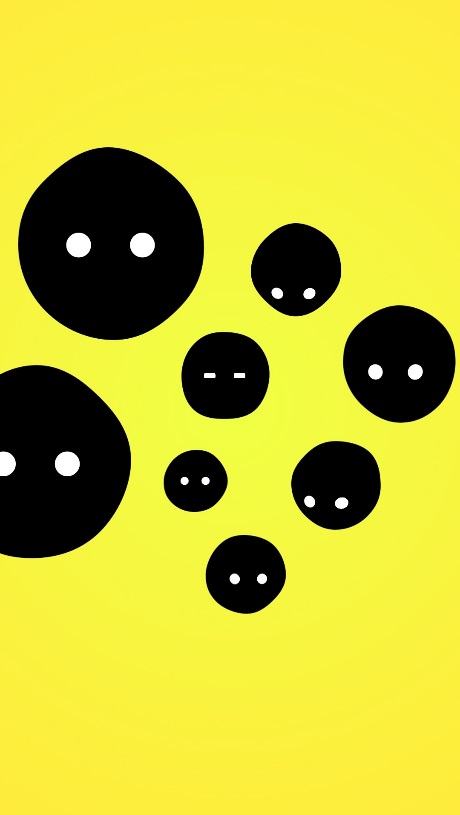
Рисунок 1 – список задач

**7. Спринт Ревью**

Окончание каждого спринта знаменуется значительным приростом функционала продукта. Более того, это означает, что команда полностью написала код, провела полноценное тестирование и выдала готовую к употреблению часть программного продукта, или целый продукт.

Sprint Review Meeting проводится в конце каждого спринта и носит обзорный характер. На встрече команда оценивает то, что из спринтов выполнено и чаще всего это выглядит в виде демонстрации новых возможностей.

Не стоит относится к Sprint Review Meeting как к формальной четко поставленной встрече с развернутыми докладами. Sprint Review Meeting это всё же просто логическое завершение Sprint.

**8. Анализ результатов**

Игра работает корректно.

На рисунке 2 представление общего вида игры. Лицо смотрит в камеру, камера на этапе фокусировки на лице.   
 На рисунке 3 камера определила лицо и черные пятна «разбежались».

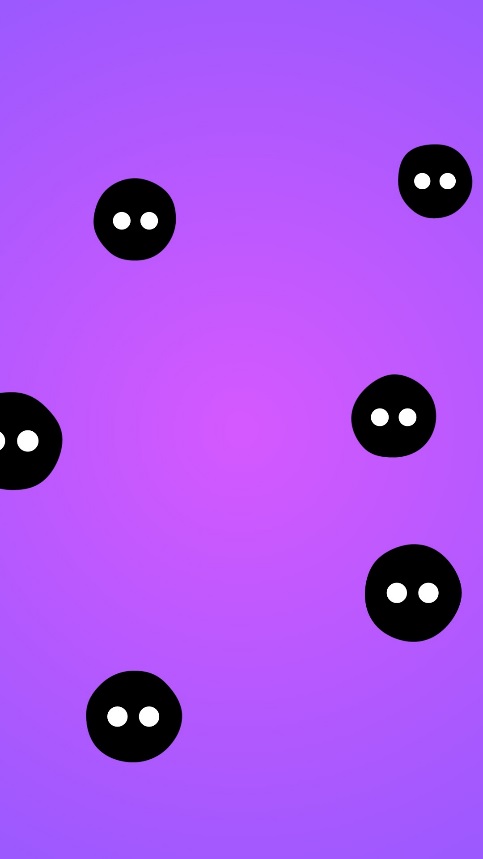


Рисунок 2 – общий вид игры

Рисунок 3 – «рабочая составляющая игры»

**9. Листинги**

***Листинг цвета фона***

<?xml version="1.0" encoding="utf-8"?>

<shape xmlns:android="http://schemas.android.com/apk/res/android";

android:shape="rectangle" >

<gradient

android:startColor="@color/background\_pink\_start"

android:endColor="@color/background\_pink\_end"

android:gradientRadius="400"

android:type="radial"

>

</gradient>

</shape>

***Листинг спектра цвета для фона***

<?xml version="1.0" encoding="utf-8"?>

<resources>

<color name="background\_pink\_start">#c566b9</color>

<color name="background\_pink\_end">#f5148a</color>

<color name="background\_blue\_start">#00c0ff</color>

<color name="background\_blue\_end">#007dff</color>

<color name="background\_green\_start">#19d2c7</color>

<color name="background\_green\_end">#19d29c</color>

<color name="background\_orange\_start">#ffc62d</color>

<color name="background\_orange\_end">#ff9b2d</color>

<color name="background\_purple\_start">#d658ff</color>

<color name="background\_purple\_end">#9d58ff</color>

<color name="background\_red\_start">#ff9561</color>

<color name="background\_red\_end">#ff5442</color>

<color name="background\_teal\_start">#00d5c3</color>

<color name="background\_teal\_end">#00bcd5</color>

<color name="background\_yellow\_start">#f0ff44</color>

<color name="background\_yellow\_end">#ffeb3c</color>

<color name="body\_color">#000000</color>

<color name="eye\_color">#ffffff</color>

</resources>

***Листинг активности шариков***

<?xml version="1.0" encoding="utf-8"?>

<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android";

xmlns:app="http://schemas.android.com/apk/res-auto";

android:layout\_width="match\_parent"

android:layout\_height="match\_parent">

<SurfaceView

android:id="@+id/preview"

android:layout\_width="@dimen/preview\_width"

android:layout\_height="144dp"

android:layout\_alignParentRight="true"

android:layout\_marginTop="@dimen/margin"

android:layout\_marginRight="@dimen/margin"

/>

<com.larvalabs.boo.RadialGradientView

android:id="@+id/background"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

app:centerX=".5"

app:centerY=".5"

app:startColor="@color/background\_pink\_start"

app:endColor="@color/background\_pink\_end"

app:gradientRadiusWidthPercent="1"

/>

<com.larvalabs.boo.CreaturesView

android:id="@+id/intro"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

/>

<com.larvalabs.boo.CreaturesView

android:id="@+id/creatures"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:visibility="gone"

/>

<ImageView

android:id="@+id/title"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_alignParentBottom="true"

android:layout\_centerHorizontal="true"

android:layout\_marginBottom="@dimen/large\_margin"

android:src="@drawable/android\_experiment"

/>

</RelativeLayout>

Листинг кода идентификации лица

* package [com.larvalabs.boo](https://vk.com/away.php?to=http%3A%2F%2Fcom.larvalabs.boo&cc_key=" \t "_blank);  
    
  import android.app.Activity;  
  import [android.hardware.Camera](https://vk.com/away.php?to=http%3A%2F%2Fandroid.hardware.Camera&cc_key=" \t "_blank);  
  import android.os.Bundle;  
  import android.view.SurfaceHolder;  
  import android.view.SurfaceView;  
  import android.view.View;  
  import android.view.WindowManager;  
    
  import java.io.IOException;  
    
  import static com.larvalabs.boo.Util.log;  
    
  public class BooActivity extends Activity implements SurfaceHolder.Callback {  
    
  private static final int[][] BACKGROUNDS = {  
  {R.color.background\_pink\_start, R.color.background\_pink\_end},  
  {R.color.background\_yellow\_start, R.color.background\_yellow\_end},  
  {R.color.background\_blue\_start, R.color.background\_blue\_end},  
  {R.color.background\_green\_start, R.color.background\_green\_end},  
  {R.color.background\_orange\_start, R.color.background\_orange\_end},  
  {R.color.background\_purple\_start, R.color.background\_purple\_end},  
  {R.color.background\_red\_start, R.color.background\_red\_end},  
  {R.color.background\_teal\_start, R.color.background\_teal\_end},  
  };  
    
  // Turn this off when testing the creature behavior so you can actually look at the phone  
  private static final boolean HIDE\_FROM\_FACE = true;  
    
  // Don't hide right away after intro  
  private static final long FIRST\_HIDE\_BUFFER = 2500;  
    
  // After hiding, don't hide again too quickly  
  private static final long HIDE\_BACKOFF\_TIME = 1500;  
    
  private Camera camera;  
  private SurfaceView surfaceView;  
  private SurfaceHolder surfaceHolder;  
  private boolean preview = false;  
  private boolean foundFaces = false;  
    
  private long lastHideTime = 0;  
    
  private CreaturesView introView;  
  private CreaturesView creaturesView;  
    
  private RadialGradientView gradientView;  
    
  private int backgroundIndex;  
    
  private boolean introRunning = true;  
  private long startTime = 0;  
    
  [@Override](https://vk.com/override)  
  protected void onCreate(Bundle savedInstanceState) {  
  super.onCreate(savedInstanceState);  
  getWindow().addFlags(WindowManager.LayoutParams.FLAG\_KEEP\_SCREEN\_ON);  
  setContentView(R.layout.activity\_boo);  
  surfaceView = (SurfaceView)findViewById(R.id.preview);  
  surfaceHolder = surfaceView.getHolder();  
  surfaceHolder.addCallback(this);  
  introView = (CreaturesView) findViewById(R.id.intro);  
  introView.setIntroMode(true);  
  creaturesView = (CreaturesView) findViewById(R.id.creatures);  
  gradientView = (RadialGradientView) findViewById(R.id.background);  
  backgroundIndex = 0;  
  }  
    
  Camera.FaceDetectionListener faceDetectionListener = new Camera.FaceDetectionListener(){  
    
  [@Override](https://vk.com/override)  
  public void onFaceDetection(Camera.Face[] faces, Camera camera) {  
  // Don't do this until the intro is done, and after intro wait a bit before detecting faces  
  if (introRunning) {  
  return;  
  } else if (System.currentTimeMillis() - startTime < FIRST\_HIDE\_BUFFER) {  
  return;  
  }  
  boolean found = faces.length >= 1;  
  if (foundFaces != found && HIDE\_FROM\_FACE) {  
  long time = System.currentTimeMillis();  
  if (found) {  
  log(" - Found a face!");  
  boolean hiding = creaturesView.setFaceVisible(true);  
  if (hiding) {  
  creaturesView.postDelayed(new Runnable() {  
  [@Override](https://vk.com/override)  
  public void run() {  
  changeBackground();  
  }  
  }, 500);  
  }  
  foundFaces = true;  
  lastHideTime = time;  
  } else if (time - lastHideTime > HIDE\_BACKOFF\_TIME) {  
  log(" - No faces!");  
  creaturesView.setFaceVisible(false);  
  foundFaces = false;  
  } else {  
  long delay = HIDE\_BACKOFF\_TIME - (time - lastHideTime);  
  creaturesView.postDelayed(new Runnable() {  
  [@Override](https://vk.com/override)  
  public void run() {  
  if (foundFaces == true) {  
  log(" - No faces (delayed)!");  
  creaturesView.setFaceVisible(false);  
  foundFaces = false;  
  }  
  }  
  }, delay);  
  }  
  }  
  }  
  };  
    
  [@Override](https://vk.com/override)  
  public void surfaceChanged(SurfaceHolder holder, int format, int width,int height) {  
  if(preview){  
  camera.stopFaceDetection();  
  camera.stopPreview();  
  preview = false;  
  }  
    
  if (camera != null){  
  try {  
  camera.setPreviewDisplay(surfaceHolder);  
  camera.setDisplayOrientation(90);  
  camera.startPreview();  
  camera.startFaceDetection();  
  preview = true;  
  } catch (IOException e) {  
  e.printStackTrace();  
  }  
  }  
  }  
    
  [@Override](https://vk.com/override)  
  public void surfaceCreated(SurfaceHolder holder) {  
  int n = Camera.getNumberOfCameras();  
  Camera.CameraInfo cameraInfo = new Camera.CameraInfo();  
  for (int i = 0;
* i < n; i++) {  
  Camera.getCameraInfo(i, cameraInfo);  
  if (cameraInfo.facing == [Camera.CameraInfo.CAMERA](https://vk.com/away.php?to=http%3A%2F%2FCamera.CameraInfo.CAMERA&cc_key=" \t "_blank)\_FACING\_FRONT) {  
  camera = [Camera.open](https://vk.com/away.php?to=http%3A%2F%2FCamera.open&cc_key=" \t "_blank)(i);  
  camera.setFaceDetectionListener(faceDetectionListener);  
  return;  
  }  
  }  
  Util.error("Could not open front-facing camera!");  
  }  
    
  [@Override](https://vk.com/override)  
  public void surfaceDestroyed(SurfaceHolder holder) {  
  try {  
  camera.stopFaceDetection();  
  camera.stopPreview();  
  camera.release();  
  camera = null;  
  preview = false;  
  } catch (Exception e) {  
  e.printStackTrace();  
  }  
  }  
    
  public void endIntro() {  
  // Fade out the experiment title view  
  final View title = findViewById(R.id.title);  
  Util.log("ENDING INTRO");  
  title.animate().alpha(0).setDuration(600).setStartDelay(1000).start();  
  introView.postDelayed(new Runnable() {  
  [@Override](https://vk.com/override)  
  public void run() {  
  changeBackground();  
  }  
  }, 500);  
  introView.postDelayed(new Runnable() {  
  [@Override](https://vk.com/override)  
  public void run() {  
  creaturesView.setVisibility(View.VISIBLE);  
  introView.setVisibility(View.GONE);  
  introRunning = false;  
  startTime = System.currentTimeMillis();  
  }  
  }, 2000);  
  }  
    
  private void changeBackground() {  
  backgroundIndex++;  
  if (backgroundIndex == BACKGROUNDS.length) {  
  backgroundIndex = 0;  
  }  
  gradientView.postDelayed(new Runnable() {  
  [@Override](https://vk.com/override)  
  public void run() {  
  int startColor = getResources().getColor(BACKGROUNDS[backgroundIndex][0]);  
  int endColor = getResources().getColor(BACKGROUNDS[backgroundIndex][1]);  
  gradientView.changeColor(startColor, endColor);  
  }  
  }, 500);  
  }  
    
  }

**10. Вывод**

Выполняя курсовую работу, мы учились работать по правилам методики разработки. В результате, скоординированных сил двух исполнителей, был реализован проект приложения для мобильного устройства на популярной ОС Android.