МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ УЧРЕЖДЕНИЕ

КАЛИНИНГРАДСКОЙ ОБЛАСТИ

ПРОФЕССИОНАЛЬНАЯ ОБРАЗОВАТЕЛЬНАЯ ОРГАНИЗАЦИЯ

«КОЛЛЕДЖ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СТРОИТЕЛЬСТВА»

(ГБУ КО ПОО «КИТиС»)

|  |  |  |
| --- | --- | --- |
| Курсовой проект  допущен к защите  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (дата)  Зам.директора по УМР  \_\_\_\_\_\_\_\_\_\_\_\_\_Павленко Г.Я.  (подпись) |  | Курсовой проект  защищен с оценкой  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (оценка)  Руководитель проект  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись, дата) |

Пояснительная записка к курсовому проекту

по дисциплине: МДК 01.01 Разработка программных модулей

Тема: Разработка игры “Прыгающая Птичка”

Специальность 09.02.07 «Информационные системы и программирование»

|  |  |
| --- | --- |
| Выполнил: | студент 3 курса,  группы ИСп19-2к  Глушков Роман Александрович  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |
| Руководитель: | Большакова-Стрекалова Анна Викторовна, преподаватель ГБУ КО ПОО КИТиС  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) |

Калининград

2022

Содержание

[Введение](#_Toc102414295) 2

[1 Описание предметной области](#_Toc102414296) 3

[1.1 Аналоги разрабатываемого приложения](#_Toc102414297) 3-4

[1.2 Техническое задание](#_Toc102414298) 4-5

[1.3 Описание структуры приложения](#_Toc102414299) 5

[2 Описание разработки приложения](#_Toc102414300) 6

[2.1 Обоснование средств разработки](#_Toc102414301) 6-7

[2*.*2 Разработка интерфейса](#_Toc102414302) 8

[2.3 Разработка логики работы приложения. Схема взаимодействия компонентов проекта](#_Toc102414303) 11

[2.4 Описание переменных, компонентов, классов и подпрограмм](#_Toc102414304) 13

[3 Тестирование и установка приложения](#_Toc102414305) 12-13

[Заключение](#_Toc102414306) 13

[Список использованной литературы](#_Toc102414307) 14

[Приложения (листинг рабочего кода с подробными комментариями)](#_Toc102414308) 15-24

Введение

Самостоятельная разработка курсового проекта на тему игры – “Flappy Bird”.

Цель выполнения курсового проекта:

* Систематизация и закрепление полученных теоретических знаний и практических умений по общепрофессиональным и специальным дисциплинам специальности «Информационные системы и программирование»;
* Углубление теоретических знаний в соответствии с заданной темой;
* Формирование умений применять теоретические знания при решении поставленных вопросов;
* Формирование умений использовать справочную, нормативную и правовую документацию;
* Развитие творческой инициативы, самостоятельности, ответственности и организованности;

Задачи курсового проекта:

* Изучение особенностей конкретной предметной области, относящихся к теме курсового проекта;
* Анализ возможных подходов и методов решения с обоснованием выбранного метода;
* Выбор или разработка модели (математической, структурной, информационной и т.д.), необходимой для достижения цели;
* Выбор эффективных алгоритмов с учётом их точности, устойчивости, сходимости и т.д.;
* Разработка приложения под ОС Windows;
* Анализ полученных результатов.

1. Описание предметной области

Вся суть игры заключается в том, чтобы играя за птичку облетать препятствия не сталкиваясь с ними. В игре имеется один и единственный одиночный режим в котором ведется подсчёт вашего рекорда.

* 1. Аналоги разрабатываемого приложения

Как пример оформления и функционала рассмотреть Flappy Bird Classic

****

Рисунок 1 - Flappy Bird Classic

Это приложение обычный Time Killer, который не имеет замысловатого интерфейса и какого-то интерфейса. Весь смысл игры заключается в том, чтобы в роли птички облетать препятствия.

В самом приложении можно посмотреть свой рекорд и прогресс того, куда вы дошли в процессе игры.

Таблица 1. Таблица плюсов и минусов

|  |  |
| --- | --- |
| Плюсы | Минусы |
| Игра более оптимизирована в точки зрения игрового процесса | Нет рекорда с другими пользователями сети |
| Красивое оформление |  |
| Для игры не требуется подключение к интернету |  |
| В игре предусмотрено получение медалей в процессе игрового процесса |  |

## Техническое задание

Игра «Прыгающая Птичка» для Windows. Данный программный продукт представляет из себя приложение для пользователей Windows.

1. Область применения

Использование рядовыми пользователями компьютеров на базе Windows в развлекательных, а так же в развивающих целях.

1. Основание для разработки

Игра «Прыгающая Птичка» для Windows предназначена для игрового времяпрепровождения.

Функциями приложения являются:

* Игровое времяпрепровождение;
* Предоставление инструкций к игре;
* Статистические данные;
* Развитие навыков реакции и логики.

1. Требования к программе

Игра «Прыгающая Птичка» для Windows должна выполнять следующие функции:

* Вести статистику рекордов;
* Останавливать игру при соприкосновении с трубами;
* Развивать реакцию пользователя.

1. Функциональные требования:

* Возможность управления птичкой нажатиями на экран;
* Работать на базе ОС Windows;

## Реализовать вывод рекорда/ прогресса

## Описание структуры приложения

В соответствии ТЗ нужно было сделать игру простого типа, так скажем обычный Time Killer. В его функционал входит:

1. Кликабельная кнопка начала игры
2. Кликабельная кнопка повтора игры
3. Кликабельная кнопка для просмотра своего рекорда

## 2 Описание разработки приложения

## 2.1 Обоснование средств разработки

Выбор операционной системы:

Выбор ОС пал на всеми известный Windows. Так как почти каждый человек имеет дома компьютер с подобной ОС, то можно охватить большее количество аудитории.

Преимущества для разработки на Windows:

1. Язык программирования Java может быть использован при написании не только игр, но и таких как веб-приложения, мобильные приложения и тд
2. Среда разработки анализирует код и может автоматически формировать некоторые его части
3. Хороший фреймворк можно найти готовые модули под самые разные цели
4. Платформу можно адаптировать под любую специализированную задачу
5. Публикация приложений в GitHub и прочие площадки для скачиваний приложений происходит практически мгновенно

Недостатки для разработки на Windows:

1. Многообразие устройств и их параметров создает почву для багов, неодинаковой работы приложения на разных компьютерах
2. Не все обучающие материалы и ответы на форумах актуальны: система существует давно, нюансы разработки меняются

Выбор среды разработки:

**IntelliJ IDEA**

**IntelliJ IDEA** — [интегрированная среда разработки программного обеспечения](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D0%B3%D1%80%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D1%80%D0%B5%D0%B4%D0%B0_%D1%80%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%BA%D0%B8) для многих [языков программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), в частности [Java](https://ru.wikipedia.org/wiki/Java" \o "Java), [JavaScript](https://ru.wikipedia.org/wiki/JavaScript" \o "JavaScript), [Python](https://ru.wikipedia.org/wiki/Python" \o "Python), разработанная компанией [JetBrains](https://ru.wikipedia.org/wiki/JetBrains" \o "JetBrains).

Достоинства данной среды разработки:

* 1. Легкий старт
  2. Сочетания клавиш для любых действий
  3. Стандартные и пользовательские темы
  4. Специальные возможности
  5. Плагины

Какую среду разработки выбрать:

IntelliJ IDEA. Она отлично подойдёт если:

Разработка ведётся на нескольких языках программирования;

Компьютер достаточно мощный (минимум 2 ГБ оперативной памяти).

Выбор языка программирования:

Java — язык программирования общего назначения. Относится к объектно-ориентированным языкам программирования, к языкам с сильной типизацией.

Может сейчас Java слегка и потерял свою популярность среди новых пользователей сети, но это все еще самый популярный язык программирования буквально по всем параметрам

## 2.2 Разработка интерфейса

Пользовательский интерфейс - это средства общения пользователя с программой. которые могут включать в себя изображения, звуки и текст. Ориентируясь на среднего пользователя интерфейс должен быть простым и удобным. Это снизит вероятность ошибок.

Всего в игре имеется 5 компонентов:

1. Задний фон игры
2. Птичка в состояние активации
3. Птичка в состояние падение(не активного состояния)
4. Труба в верхнем положении
5. Труба в нижнем положении



Рисунок 2 – Задний фон



Рисунок 3 – Птичка в неактивном положении



Рисунок 4 – Птичка в активном положении

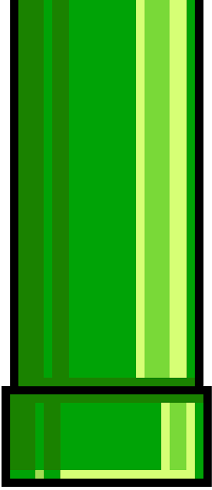


Рисунок 5 – Труба в верхнем положении

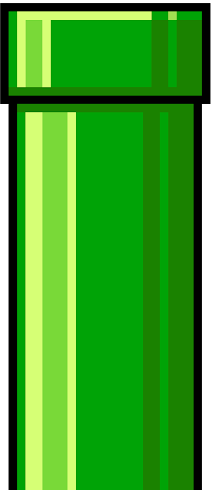


Рисунок 6 – Труба в нижнем положении

## 2.3 Разработка логики работы приложения. Схема взаимодействия компонентов проекта

## 

## Рисунок 7 – Блок схема работы игры

2.4 Описание переменных, компонентов, классов и подпрограмм

Таблица 2 - Методы

|  |  |
| --- | --- |
| Методы | Описание |
| Draw | Прорисовка холста. |
| onLoadComplete | Загрузка звука. |
| gameOver | Окончание игры. |
| reset | Перезапуск игры. |
| dialogScore | Форма рекорда |
| onCreate | Привязка к активностям. |
| Update | Обновление до последний версии java |

Таблица 3 - Классы

|  |  |
| --- | --- |
| Методы | Описание |
| public class Flappy | Класс Птички |
| public class Chrono | Класс игры |
| public class Score | Класс рекорда |

1. Тестирование и установка приложения

При запуске приложения у вас на экране загорается кнопка начала игры



Рисунок 8 – Кнопка начала игры

После смерти птички у вас на экране будет высвечен ваш рекорд и кнопка повтора игры

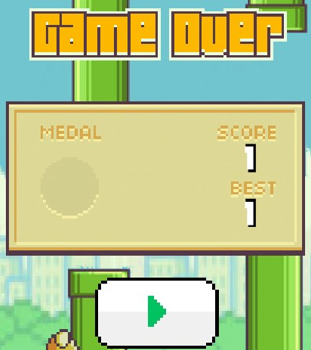


Рисунок 9 – Окно повтора игры и вывод вашего рекорда

Заключение

В ходе выполнения данной курсовой работы на основе различных источников данных была проанализирована заданная предметная область.

В процессе кодирования была изучена Java в части пользовательского интерфейса и технологий отображения графики.

В результате выполнения проекта была полностью реализована игра «Прыгающая Птичка».

Были выполнены следующие задачи:

* При нажатии на экран птичка поднимается выше;
* После прохождения припятствия набирание рекорда;
* Смерть при соприкосновении птички с стенами;
* При смерти птички вылезала форма с очками набранными во время игры.

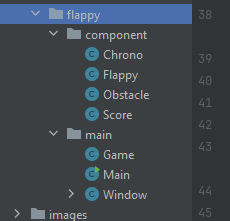
В процессе тестирования приложения ошибок не обнаружено.

Список использованной литературы

1. Герберт Шилдт «Java: Полное руководство»
2. Герберт Шилдт - Java 8 Полное руководство
3. Кэти Сьерра и Берт Бейтс - Изучаем Java
4. Кей С. Хорстманн - Java SE 8 Вводный курс
5. Майк МакГрат | Программирование на Java для начинающих (2016)

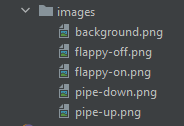
# Приложение А. Листинг кода

## Код разделён на классы:



Картинка 8. Классы игры

Картинки имеющиеся в игре:



Картинка 9. Картинки игры

Класс Main:

Main.class

package com.flappy.main;  
  
public class Main {  
 public Main() {  
 }  
  
 public static void main(String[] args) {  
 new Window();  
 }  
}

Класс Game:

Game.class

package com.flappy.main;  
  
import com.flappy.component.Chrono;  
import com.flappy.component.Flappy;  
import com.flappy.component.Obstacle;  
import com.flappy.component.Score;  
import java.awt.Color;  
import java.awt.Font;  
import java.awt.Graphics;  
import java.awt.Graphics2D;  
import java.awt.Image;  
import java.awt.image.ImageObserver;  
import java.io.IOException;  
import java.util.ArrayList;  
import java.util.Iterator;  
import java.util.List;  
import javax.imageio.ImageIO;  
import javax.swing.JPanel;  
  
public class Game extends JPanel {  
 private final int windowsWidth = 434;  
 private final int windowsHeight = 483;  
 private Chrono chrono = new Chrono(this);  
 private Flappy flappy = new Flappy();  
 private List<Obstacle> obstacles = new ArrayList();  
 private int frame = 0;  
 private Score score;  
 private Font fontScore;  
 private Image backgroundImage;  
 private String backgroundImagePath = "/images/background.png";  
 private Image pipeUpImage;  
 private String pipeUpImagePath = "/images/pipe-up.png";  
 private Image pipeDownImage;  
 private String pipeDownImagePath = "/images/pipe-down.png";  
  
 public Game() {  
 this.obstacles.add(new Obstacle());  
 this.score = new Score();  
 this.fontScore = new Font("Arial", 1, 25);  
 this.setBackground(Color.black);  
  
 try {  
 this.backgroundImage = ImageIO.read(this.getClass().getResource(this.backgroundImagePath));  
 this.pipeUpImage = ImageIO.read(this.getClass().getResource(this.pipeUpImagePath));  
 this.pipeDownImage = ImageIO.read(this.getClass().getResource(this.pipeDownImagePath));  
 } catch (IOException var2) {  
 var2.printStackTrace();  
 }  
  
 }  
  
 public void paintComponent(Graphics g) {  
 super.paintComponent(g);  
 Graphics2D g2 = (Graphics2D)g;  
 g2.drawImage(this.backgroundImage, 0, 0, 434, 483, (ImageObserver)null);  
 Iterator var4 = this.obstacles.iterator();  
  
 while(var4.hasNext()) {  
 Obstacle obs = (Obstacle)var4.next();  
 g2.drawImage(this.pipeDownImage, (int)obs.getPosX(), 0, (int)obs.getWidth(), (int)obs.getTop(), (ImageObserver)null);  
 g2.drawImage(this.pipeUpImage, (int)obs.getPosX(), (int)(483.0 - obs.getBottom()), (int)obs.getWidth(), (int)obs.getBottom(), (ImageObserver)null);  
 if (!this.isDeath()) {  
 obs.update();  
 }  
 }  
  
 g2.drawImage(this.flappy.fly(50), (int)this.flappy.getPosX(), (int)this.flappy.getPosY(), (int)this.flappy.getWidth(), (int)this.flappy.getHeight(), (ImageObserver)null);  
 if (!this.isDeath()) {  
 this.flappy.update();  
 } else {  
 this.chrono.setRun(false);  
 }  
  
 this.updateObstaclesList();  
 g2.setColor(Color.white);  
 g.setFont(this.fontScore);  
 g2.drawString(this.score.getMessage(), 324, 20);  
 }  
  
 public void keyPressed() {  
 this.flappy.jump();  
 this.repaint();  
 }  
  
 private void updateObstaclesList() {  
 for(int i = 0; i < this.obstacles.size(); ++i) {  
 if (((Obstacle)this.obstacles.get(i)).getPosX() < -((Obstacle)this.obstacles.get(i)).getWidth()) {  
 this.obstacles.remove(i);  
 this.score.setScore(this.score.getScore() + 1);  
 }  
 }  
  
 ++this.frame;  
 if (this.frame >= 250) {  
 this.obstacles.add(new Obstacle());  
 this.frame = 0;  
 }  
  
 }  
  
 private boolean isDeath() {  
 if (this.flappy.getPosY() >= 483.0 - (this.flappy.getWidth() - 15.0)) {  
 this.flappy.setPosY(483.0 - (this.flappy.getWidth() - 15.0));  
 this.flappy.setVelocity(0.0);  
 return true;  
 } else {  
 Iterator var2 = this.obstacles.iterator();  
  
 while(var2.hasNext()) {  
 Obstacle obs = (Obstacle)var2.next();  
 if (this.flappy.getPosX() + this.flappy.getWidth() >= obs.getPosX() && this.flappy.getPosX() <= obs.getPosX() + obs.getWidth()) {  
 if (this.flappy.getPosY() < obs.getTop()) {  
 return true;  
 }  
  
 if (this.flappy.getPosY() + this.flappy.getHeight() > 483.0 - obs.getBottom()) {  
 return true;  
 }  
 }  
 }  
  
 return false;  
 }  
 }  
}

Код Окна игрового процесса:

Window.class

package com.flappy.main;  
  
import java.awt.Component;  
import java.awt.event.KeyEvent;  
import java.awt.event.KeyListener;  
import javax.swing.JFrame;  
  
public class Window extends JFrame {  
 private Game game;  
 private final int windowsWidth = 450;  
 private final int windowsHeight = 522;  
  
 public Window() {  
 this.initParts();  
 this.initWindow();  
 this.initListener();  
 this.setVisible(true);  
 }  
  
 private void initParts() {  
 this.game = new Game();  
 }  
  
 private void initWindow() {  
 this.setTitle("Flappy Bird");  
 this.setSize(450, 522);  
 this.setDefaultCloseOperation(3);  
 this.setResizable(false);  
 this.setAlwaysOnTop(false);  
 this.setLocationRelativeTo((Component)null);  
 this.add(this.game, "Center");  
 }  
  
 private void initListener() {  
 KeyboardListener keyboardListener = new KeyboardListener();  
 this.addKeyListener(keyboardListener);  
 }  
  
 private void pause() {  
 try {  
 Thread.sleep(10L);  
 } catch (InterruptedException var2) {  
 var2.printStackTrace();  
 }  
  
 }  
  
 class KeyboardListener implements KeyListener {  
 KeyboardListener() {  
 }  
  
 public void keyPressed(KeyEvent event) {  
 }  
  
 public void keyReleased(KeyEvent event) {  
 if (event.getKeyCode() == 32) {  
 Window.this.game.keyPressed();  
 }  
  
 Window.this.pause();  
 }  
  
 public void keyTyped(KeyEvent event) {  
 }  
 }  
}

Код Рекорда игры:

package com.flappy.component;  
  
public class Score {  
 private int score = 0;  
  
 public Score() {  
 }  
  
 public int getScore() {  
 return this.score;  
 }  
  
 public String getMessage() {  
 return "score : " + this.score;  
 }  
  
 public void setScore(int score) {  
 this.score = score;  
 }  
}

# Приложение Б. Руководство пользователя

Руководство пользователя мобильного приложения для устройств на ОС Windows

При запуске игры у вас на экране будет кнопка начала игры:



Рисунок 10 – Кнопка начала игры

После запуска игры весь игровой процесс заключается в том, что ты нажимаете на пробел тем самым заставляете птички подлетать выше

Ваша задача не врезаться в препятствия

Если вы врезались птичкой в препятствие, то будет высвечена окно с вашим счетом и кнопка повтора игры:

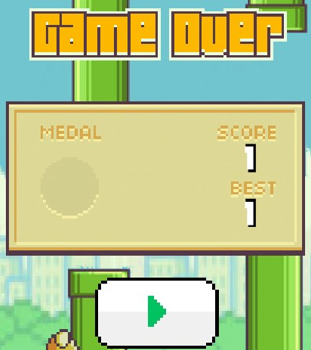


Рисунок 11 – Окно вашего рекорда и кнопка повтора игры

Рецензия

В ходе работы над игрой “Flappy Bird” была проделана работа под ОС Windows. Проект состоит из введения, трёх глав, заключения, списка использованных источников и приложения А.В Ведение рассказано немного о игре , а также отмечены: цель, задачи. В первой главе были проанализированы приложения- конкуренты, разработано техническое задание. Вторая глава посвящена практической части, где обусловлен выбор среды разработки и языка программирования. А в третьей главе описывается тестирование приложения В заключении данного курсового проекта развернуто подытожена работа.

Можно сделать вывод, что в целом, автор выполнил работу по всем требованиям. Каждая часть проекта рассматривалась с применением необходимой профессиональной терминологией. Студент излагал работу с логичной последовательностью, все названия глав полностью соответствуют тексту.

Ссылка на Github: https://github.com/Bloofq/My-first-game.git