

Politechnika Śląska
Wydział Automatyki, Elektroniki i Informatyki

Computer Programming 3

Bomb Mario

author	Amadeusz Drożdż
instructor	mgr inż. Krzysztof Pasterak
year	2019/2020
lab group	Thursday, 10:00 – 12:00
date	2020-01-31

TABLE OF CONTENTS

First page.....	1
1 Project's objective	3
2 Game Overview	4
2.1 Controls	6
3 Program structure	7
3.1 Class diagram.....	8
3.2 Testing	8

1 PROJECT'S OBJECTIVE

The project's objective was to design and implement a program exploiting object-oriented programming paradigm and its associated tools.

Platform game engine was chosen as a topic, as it meets the requirements and provides the possibility of practicing proper game architecture creation.

2 GAME OVERVIEW



Game was written using SFML. Theme of the project is the turn based arcade game. The player controls the character using arrows. The main goal of the game is to avoid bombs and collect apples, which fall from the top of the screen with random speed and direction. The initial number of lives is three. If a player is hit by a bomb, one life is taken. Player can gain extra lives by collecting falling hearts. The maximum of lives that a player can gain is also three. The player can collect points in the game – by catching falling apples, one apple adds 100 points to score. The game is divided into several levels, each level increases the difficulty of the game:

BEGINNER:

- Level duration: 30s
- Frequency of bombs dropping: random value in the range from 0,6s to 1,4s
- Speed (velocity) of bomb dropping: random value in the range from 150 pixels per second to 360 pixels per second
- Frequency of apple (points) dropping: random value in the range from 6s to 15s
- Frequency of heart (new life) dropping: random value in the range from 8s to 17s

NORMAL:

- Level duration: 45s
- Frequency of bombs dropping: random value in the range from 0,3s to 0,9s
- Speed (velocity) of bomb dropping: random value in the range from 180 pixels per second to 500 pixels per second
- Frequency of apple (points) dropping: random value in the range from 6s to 12s
- Frequency of heart (new life) dropping: random value in the range from 8s to 17s

ADVANCED:

- Level duration: 60s
- Frequency of bombs dropping: random value in the range from 0,2s to 0,65s
- Speed (velocity) of bomb dropping: random value in the range from 200 pixels per second to 560 pixels per second
- Frequency of apple (points) dropping: random value in the range from 5s to 10s
- Frequency of heart (new life) dropping: random value in the range from 10s to 17s

EXPERT:

- Level duration: 90s
- Frequency of bombs dropping: random value in the range form 0,2s to 0,45s
- Speed (velocity) of bomb dropping: random value in the range form 220 pixels per second to 630 pixels per second
- Frequency of apple (points) dropping: random value in the range form 5s to 9s
- Frequency of heart (new life) dropping: random value in the range form 10s to 17s

LEGENDARY:

- Level duration: 90s
- Frequency of bombs dropping: random value in the range form 0,15s to 0,35s
- Speed (velocity) of bomb dropping: random value in the range form 230 pixels per second to 660 pixels per second
- Frequency of apple (points) dropping: random value in the range form 7s to 10s
- Frequency of heart (new life) dropping: random value in the range form 9s to 17s

If the player loses all lives, the game ends. The number of points will be displayed for 5 seconds, and after that a new game will start.

2.1 CONTROLS

Program is limited to game view, which uses controls listed below:

- Right arrow – go right
- Left arrow – go left
- Spacebar – jump

If the player loses all lives, the game ends. The number of points will be displayed for 5 seconds, and after that a new game will start.

3 PROGRAM STRUCTURE

Project contains of eight main classes, each class is divided into separate source and header file. Methods have public access, attributes have private or protected access. Objects are constructed using constructors and deleted after finishing the game. Polymorphism and overloaded operators were used many times. More specific description of types and functions is moved to the appendix.

- **Game class** - Controls all game objects, events and rendering
- **GameObject class** - Simple static object in game
- **AbstractRigidbodyGameObject abstract class** - Moving game object in game inherits GameObject class - every moving object also has a static object properties
- **Game object manager class** – List that holds game objects.
- **Player class** - Defines player animations and behavior
- **Animation class** - Animates a game object
- **Ridigbody (moving) classes** – all objects that inherits AbstractRigidbodyGameObject abstract class – moving objects in the game, withs its own physics:
 - Bomb - missile that falls form the sky
 - Apples – points that falls form the sky
 - Life – new life
 - Cloud - moving clouds in the background
- **Utils class** - Useful tool (such as algorithms for calculating mathematical formulas) used in many classes

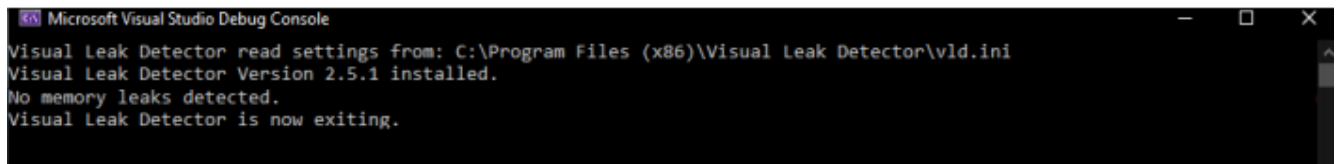
3.1 CLASS DIAGRAM

Detailed class diagram due to its size was moved to separate file *"detailed class diagram.pdf"*

3.2 TESTING

The program has been tested many times provided memory leak testing program and no memory leak was found. All dynamically allocated objects are deleted.

No memory leaks were detected by Visual Leak Detector.

A screenshot of the Microsoft Visual Studio Debug Console window. The window has a title bar with the text "Microsoft Visual Studio Debug Console" and standard window controls (minimize, maximize, close). The console output is as follows:

```
Visual Leak Detector read settings from: C:\Program Files (x86)\Visual Leak Detector\vld.ini  
Visual Leak Detector Version 2.5.1 installed.  
No memory leaks detected.  
Visual Leak Detector is now exiting.
```