Requirements and Analysis Document for the (insert project name here) project (RAD)

Version: 1.0
Date: some date
Authors: Eric Arnebäck, Johan Gerdin, Andreas Wahlström, Linnea Andersson
This version overrides all previous versions.

1 Introduction

This is a physics game where you as a black hole float around in space and eat planets and other objects.

1.1 Purpose of application

The project aims to create a physics based eat-em-up game that provides entertainment for the user.

1.2 General characteristics of application

The game is a physics based android game, controllable by touch controllers. It is a single player.

The player moves a black hole swallowing objects until he dies. The black hole dies when it is eaten by another black hole. If the black hole dies, it is respawned and a life is used up. This goes on until the player has run out lives, then it's game over.These other black holes are controlled by the computer. There are no time constraints.

Once the player has eaten enough objects, the next level will start.

1.3 Scope of application

There is only one player, and enemies are controlled by the computer. The application saves the high scores. It is not possible for the player to edit and make their own levels. There are instructions for the game and the settings of the game can be changed.

1.4 Objectives and success criteria of the project

It should be possible to play the game for as long as the player is still alive, there is no "final level". Stages are generated dynamically if necessary to keep the game going. The game should be playable on an android mobile/pad of any size.

1.5 Definitions, acronyms and abbreviations

Eat-em-up game - A game where the main goal for the player is to swallow objects in the level.

2 Requirements

2.1 Functional requirements

The user should be able to:

1. Start a new game
2. Play the game

    3.2 Move around
    3.3 Collide with other objects

3. When the game is over, the user should be able to save high score.
4. View highscore.
5. View instructions.
6. Change the settings of the game.
    5.1 Music
    5.2 Mirrored movement
    5.3 Have enemies

2.2 Non-functional requirements

2.2.1 Usability

As the application is a game it should be easy to understand how to use and not take long to start playing. An instructions option should be accessible from the main menu. Tests should be performed to verify and ensure the applications usability.

2.2.2 Reliability

NA.

2.2.3 Performance

The players touch should receive immediate feedback from the system and in worst case response time should not exceed 1 sec. Longer response time would mean a high risk of the game not functioning properly(in the meaning that lag will make it impossible to avoid enemies).

2.2.4 Supportability

The application should be implemented in a way that supports android platforms. Also, the game should be implemented in a way that makes it easy to locate reported faults and fix them.

2.2.5 Implementation

The application is written in the java programming language for use in an android

environment. This makes it easy to run it on any android device. Devices that uses other languages than java won't be able to run the application.

2.2.6 Packaging and installation

The application will be available on github. There you can download the APK-file and install it. See SecondHand/Twirl.apk in the repo.

2.2.7 Legal

NA

2.3 Application models

2.3.1 Use case model

See the other files in the directory for the use cases.

2.3.2 Use case Priority

High: Move
PowerUp
Collision
SwallowEntity

Medium: GameOver
PlayerSwallowed
Enemy Move
Game Play

Low: NextLevel
Main Menu

2.3.3 Analysis model

See Appendix: Analysis model

2.3.4 User interface

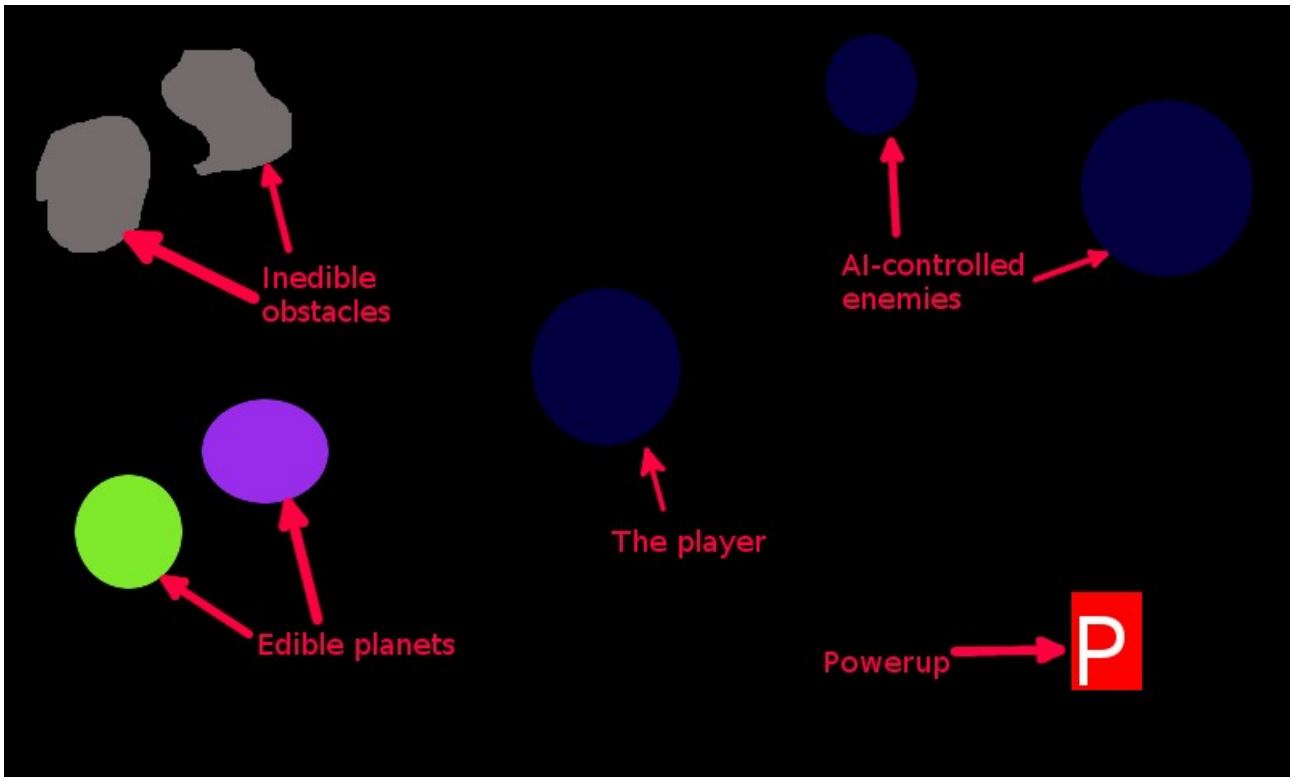
As this game is made for use on android devices, it needs to be playable on a wide variation of screen sizes. The applications GUI will be fixed (you can't change the skin or theme).

Appendix

Use cases

See the odt files in the Document branch of the repo. Their names all start with “user_case”.

GUI Sketch



Analysis Model