Requirements and Analysis Document for MiniMiner

TDA367

Gabriel Wallin, Omar Oueidat, Erik Strid, Olof Enström

May 28, 2017

Version

Version: Final
Date: May 28, 2017
Author: Revised by Omar Oueidat, Erik Strid, Gabriel Wallin, Olof Enström

This version overrides all previous versions.

Contents

1 Introduction

1.1 MiniMiner

The main reason behind the creation of miniMiner is the common interest of platform games. All four members of the group liked the idea of a platform-digging game, and therefore, the concept of miniMiner emerged. The essential goal of the application is entertainment and allowance of "micro-breaks".

1.2 Definition, acronyms and abbreviations

- Miner - The player
- Hull - The miner's health, which decreases upon impact with another object
- Fuel - The gas which the miner uses to be able to move around
- LibGDX - The library used to create the game
- MVC - Model-View-Controller, a design pattern that organizes the program in a way that avoids high coupling and dependencies
- Drilling - The act of the miner moving into the ground and removing a ground object
- Gear - A collective term for the miner's different attributes. Currently Fuel and Hull.

1.3 Scope of application

The application will not be able to save any data or restore any previously used data when terminated. Game controllers will only include a touchpad and a drill button, this means that the game will not be optimal for desktop usage. The user will not be able to modify the map or store, it is a non-dynamical world built once. Furthermore, sound effects can be implemented into the game.

The game is developed with extensibility in mind. More extensions and complex features should be easily added to the final product.

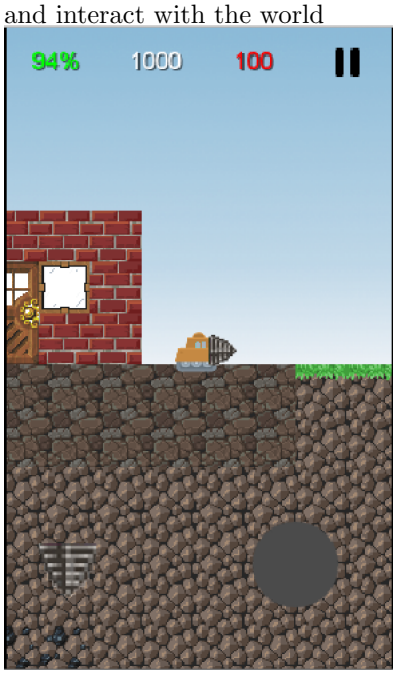
2 Requirements

2.1 User interface

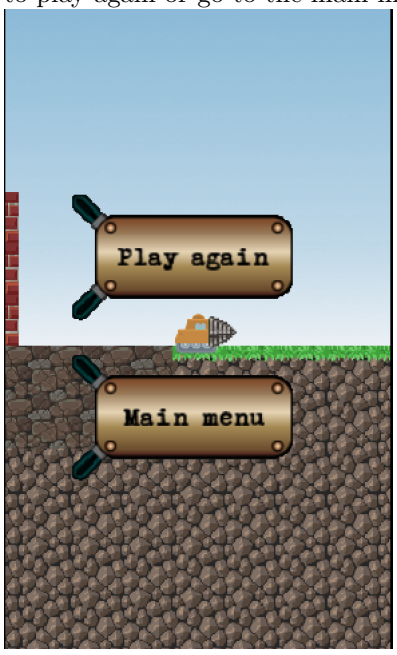
The application is based on a 2D-platform GUI, with the user digging downwards into a dirt-like environment. There will be 3 different screens

- Main menu - the screen which initially greets the user and provides the option to start the game.

- Play screen - the screen where you can play the game, control the miner and interact with the world



- Endgame screen - Where the game tells you it's over and you can choose to play again or go to the main menu.



The main menu only contains two buttons, one for starting the game and one for exiting the application entirely

The UI for the play screen will contain a joystick in the bottom right corner that controls the miner. The UI will also include a drill button in the bottom left corner that allows for digging. A shop will also be added to the game which interacts with the user's drilling machine. The shop will include a refill button and a repair button for the fuel and hull respectively. Additionally, a workshop that can upgrade the power of the Miner will be included in the world. On the top of the screen there will be a label for the amount of fuel, a label for the score and a label for the hull, as well as a pause button so the user can pause the game at any time. The pause button will give you a choice to go to the main menu or resume the game.

The interface for the endgame screen also contains two buttons, one for exiting the game and one for starting the game again.

2.2 Functional requirements

The user will be able to move, dig and fly until the fuel gauge is empty. On the surface, the user is allowed to sell materials, purchase new equipment, as well as refuel and repair the vehicle. The world in which the player will move in is gravity-based, so there will always be a downward-facing force dragging the vehicle back down.

The user should be able to

1. Start a game or change options in the starting menu
    (a) There should be a starting game screen in which the player can chose to start the game or see through options.
2. Play through the game which involves
    (a) Moving around the map
        i. Moving the touchpad should move the miner in the desired direction
    (b) Digging through the ground
        i. Using a button, the miner will dig through the ground to collect minerals
    (c) Gather minerals
        i. Sell minerals
    (d) Upgrade fuel and hull and drill parts

i. Using cash the miner gets from minerals, the player will be able to upgrade the gear to improve its efficiency

(e) Take damage

i. Hitting anything in high speed will result in a decrease of hull (health) if this is 0, the game is over

(f) Fill fuel and fix the drill

i. The miner will be able to fill the fuel or repair the hull. If either of these values reach zero, the game over screen will be shown

2.3 Non-functional requirements

2.3.1 Usability

The game focuses greatly on usability. The game language is English.

The user will get a brief rundown of how the program is used and further knowledge will demand some intuition from the user.

2.3.2 Performance

The game will be constructed in a way which will reduce the amount of memory used to process the game. This is very important for a mobile phone user. Input from the user should be almost instantaneous with very little delay (if any at all), and will respond immediately graphically as well. Since it is a offline game, there will be no network and therefore no latency.

2.3.3 Supportability

The program is constructed for mobile use but can also be played on a desktop computer.

The touchpad however is not optimal for desktop use. The arrow keys would be more suitable for moving the miner.

2.3.4 Implementation

The program will use the Java Environment in which the user running the program will be required to have a device that can run Java programs.

3 Use cases

3.1 Use case listing

High priority:

- Move
- Drill
- Start game
- Quit game

Medium priority:

- Show shop
- Shop: Refill fuel
- Shop: Repair hull

Low priority:

- Pause game
- Resume game
- Show game over menu
- Play again
- Go to main menu
- Show upgrades
- Upgrades: Upgrade fuel
- Upgrades: Upgrade hull

See Use Case PDF under documents.

3.2 Use case UML

4 Domain model