Bubble shooter

Bavdaz, Luka 4228561 — Clark, Liam 4303423 — Gmelig Meyling, Jan-Willem 4305167

Hoek, Leon 4021606 — Smulders, Sam 4225007

September 4, 2014

1 Requirements analysis

The stakeholder wants a game like Frozen Bubble, with both the singleplayer and multiplayer modes. Frozen Bubble is a Bubble shooter game, in which players have to shoot colored bubbles with a bubble gun. The bubble gun can be aimed and then shoot a bubble of a specific color. When the shot bubble hits a group of bubbles (two or more) of the same colour, these bubbles will pop and the player will receive an amount of points. When the player fails to hit a bubble of the same colour multiple times, a new row of bubbles will be inserted at the top, pushing the other bubbles. When one of these bubbles reaches the bottom of the screen, or all bubbles are shot, the game is over, and the score will be saved into a highscore board.

In the single player mode, players will face various levels with a different layout of the bubbles.

In the multiplayer mode, each player has his own canvas with a bubble gun and the bubbles to shoot (split screen). The players have different key bindings to rotate and fire the bubble gun.

2 Functional requirements

Each requirement is identified by a requirement identifier. This identifier consists of one or more marks pointing priority and status. For the priority status we use the *MoSCoW Prioritisation* (Clegg, 2014) (`M` for must have, `S` for should have, `C` for could have, `W` for wont have) and a unique number. Requirements that require further concideration are marked with `TBD`.

- `S-111` When the player clicks the highscore button he should be able to see a list of highscores
- `S-112` When the game has ended, the score and name of the player should be saved to the highscores
- `M-113` The player should be able to start a game through the menu
- When the player starts the game...

   1. `M-114` The canvas is filled with bubbles of certain colours (for example: red, green and blue), the colours are distributed depending on the game mode
   2. `M-115` Bubbles in the canvas are snapped to the top of the canvas

- `M-116` The player should be able to aim the bubble cannon (`left` and `right`) using the keyboard.
- `M-117` The interface should show the colours for the upcomming bubbles to shoot
- Behaviour for when a bubble is shot
   1. `M-118` When a bubble hits a border of the canvas, it bounces
   2. `M-119` When a bubble hits another bubble, it snaps to it
   3. `M-120` When a bubble hits a group of bubbles of the same color, these bubbles should pop
   4. `M-121` When a bubble gets isolated from any bubble at the top of the canvas, they pop
   5. `M-122` When bubbles pop, points should be awarded to the player
   6. `M-123` When you do not succeed in popping bubbles, the player gets a penalty
   7. `M-124` For each a to be determined number of penalties a new row of bubbles with the remaining colours is inserted at the top, and other bubbles are shifted down
   8. `M-125` When a row of bubbles reaches the bottom of the canvas, the game is over
   9. `M-126` When all bubbles have popped, the game is finished
- There are two different modes to play:
   1. `M-127` Singleplayer mode
      (a) `M-128` In singleplayer mode, a player will face various levels with a different distribution of bubbles across the canvas
      (b) `M-129` When a player finishes the game, a new level will appear
      (c) `C-130` In the singleplayer mode, the user should be able to use the mouse to aim the cannon, instead of the keyboard bindings
   2. `W-140` Multiplayer mode
      (a) `W-141` In the multiplayer mode, each player has his own canvas with a bubble gun and the bubbles to shoot (split screen)
      (b) `W-142` The players have different key bindings to rotate and fire the bubble gun
      (c) `W-143` When a player shoots a bubble, a new bubble is inserted in the other players screen
      (d) `w-144` In the multiplayer mode, the bubbles are distrubuted randomly
      (e) `TBD-145` The player should be able to play against another player remotely using an internet connection

3. W-146 The user should be able to switch between these modes through a menu

- C-150 Beginners tutorial. The controls and the goal of the game are explained when a player selects a button in thp e menu

3 Non-functional requirements

3.1 Product requirements

- M-231 The game should be able to run on the desktop computers at the TU Delft
- M-232 The game should not crash within 15 minutes of usage
- M-233 It shouldn't take a user more than five minutes to learn the basics of the game

3.2 Organizational requirements

This game should be developed using the Java programming language, using the Maven, Git, Devhub tools for Continious Integration and revision management and JUnit for Test-Driven Development (TDD). Within two weeks a fully functional game should be delivered. We have to work in a team of five.

3.3 External requirements

There are no external requirements at this point of the project.

References

Dai Clegg. Moscow prioritisation, may 2014. URL http://dsdm.org/content/10-moscow-prioritisation.