Bubble shooter

Bavdaz, Luka 4228561
Clark, Liam 4303423
Gmelig Meyling, Jan-Willem 4305167
Hoek, Leon 4021606
Smulders, Sam 4225007

October 26, 2014

1 Requirements analysis

We will develop a game like Frozen Bubble, with both 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.

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 won't have) and a unique number.

2.1 Must have

- `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 (red, green, blue, purple, yellow and cyan), the colours are randomly distributed over the canvas
  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 mouse
- `M-117` The interface should show the colours for the next two bubbles to shoot (we will also refer to this as ammunition)
- 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 a bubble is popped, a pop animation should be played

7. `M-124` When you do not succeed in popping bubbles, the player gets a penalty
8. `M-125` 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
9. `M-126` When a row of bubbles reaches the bottom of the canvas, the game is over
10. `M-127` When all bubbles have popped, the game is finished

- There are two different modes to play:
  1. `M-128` Singleplayer mode
     (a) `M-129` A player wins the game when all bubbles have popped
     (b) `M-130` A player loses the game when a bubble reaches the bottom of the canvas
     (c) `M-131` When a player wins the game, a new game level (another canvas of bubbles) is presented
     (d) `M-132` When a player loses the game, the score will be stored in the highscores and a highscore dialog will appear
     (e) `M-133` When the player clicks the highscore button he should be able to see a list of highscores
  2. `M-140` Multiplayer mode
     (a) `M-141` In the multiplayer mode, each player has his own canvas with a bubble gun and the bubbles to shoot (split screen)
     (b) `M-142` Multiplayer mode can be started through the game menu
     (c) `M-143` Another player can join a multiplayer game through the menu
     (d) `M-144` A player wins the game when all bubbles have popped
     (e) `M-145` A player loses the game when a bubble reaches the bottom of the canvas
     (f) `M-146` When a player wins the game, the other player automatically loses
     (g) `M-147` When a player loses the game, the other player automatically wins
     (h) `M-148` When a player exits or disconnects from a multiplayer game, the other player automatically wins
     (i) `M-149` Competitive multiplayer. When a player manage to pop a few bubbles, a new bubble is inserted in the other players screen

- `M-160` Every once in a while a player should get a power-up bubble loaded up in the cannon which has special abilities:
  1. `M-161` A `JokerBubble` should pop with a bubble of any color. If it is adjacent to less than two bubbles, the joker receives the colour of the bubble it collided with
  2. `M-162` A `BombBubble` should pop all bubbles in a certain radius around it after colliding
  3. `M-163` A `StoneBubble` can't be popped directly but only by popping the bubbles that connect it to the top
  4. `M-164` A `DrunkBubble` should move in a more difficult to predict way, so the user will have more difficulty accurately aiming the bubble
  5. `M-165` Sound effects shoud play when power-up bobbles pop or snap into the grid
  6. `M-166` Different power-up bubbles should have different pop animations
  7. `M-167` The `DrunkBubble` can be combined with any other `ColouredBubble` or power-up bubble

- `M-170` The game should have logging functionalty on which game events are logged

  The logger should have the following functionality:
  1. `M-171` Show a timestamp at which the event happend
  2. `M-172` Show a log priority [Apache, 2014] of the event (*DEBUG*, *INFO*, *WARN*, *ERROR*)
  3. `M-173` Show in which class and at which line number the log is called
  4. `M-174` Ability to pass objects to a format string [Oracle, 2014a]. For example: `log("called on %s", this)`.

5. `M-174` Ability to pass throwables to the log function, which causes the Throwables stacktrace to be printed to the outputstream [Oracle, 2014b]
6. `M-175` Append log to system outputstream

The following events should be logged:

1. `M-181` When a bubble is shot, along with its direction
2. `M-182` When a bubble is popped, along with which type of bubble it is
3. `M-183` When a row of bubbles is inserted
4. `M-184` When a ammo bubble is created
5. `M-185` When points are awarded to the player
6. `M-186` When looking for a multiplayer room
7. `M-187` When connected to a multiplayer room
8. `M-188` When disconnected from a multiplayer room
9. `M-189` When an exception is thrown

- `M-190` A player should be able to choose between different game modes in singleplayer and multiplayer modes

  1. `M-191` Classic mode. No power-up bubbles (see `M-160`)
  2. `M-192` Powerup mode. Power-up bubbles are loaded into the cannon at a small chance
  3. `M-193` Timed mode. All bubbles should be popped in a certain time
  4. `M-194` Drunk mode. All bubbles are drunk bubbles

- `M-200` The game modes should be available in the multiplayer (see `M-140`)
- `M-201` Multiple levels with different maps should be attached to each game mode (this is more advanced than `M-131`, which will remain used for game modes that do not provide levels)

2.2 Won't have

- `W-150` Beginners tutorial. The controls and the goal of the game are explained when a player selects a button in the menu
- `W-210` Game level editor. Give players the ability to design their own maps for the game
- `W-176` Append log to a text file

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 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 Continuous 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.

Apache. Apace commons logging, october 2014. URL http://commons.apache.org/proper/commons-logging/guide.h

Oracle. Formatter (java platform se7), october 2014a. URL http://docs.oracle.com/javase/7/docs/api/java/util/Formatter.html.

Oracle. Throwable (java platform se7), october 2014b. URL http://docs.oracle.com/javase/7/docs/api/java/lang/Throwable.html.