CS 571 Mobile Computing Software Architecture and Development

Batch - Spring 2016

Group Team PBS Class Project "Red Remover" Report

Benjamin Barrett

Rakesh K. Swarankar

Tanvi Ravindra Pawale

1. Game Description

a. What is the game:

This game is called "Red Remover", and the concept for this game was taken directly from an online game (Creator Gaz, Red Remover, *thegamehomepage.com*, http://www.thegamehomepage.com/play/red-remover). The game consisted of shapes (rectangles, squares, and circles) with different color like dark red, light red, blue, green etc. Each level had different kinds of motion in different directions based on the online game's level concepts. Check **concept drawing** for more details.

b. How it is played:

Each level begins with a collection of red, and/or green, and/or blue objects arranged in some configuration.

The goal is to remove red by using blue objects if required, in such a manner so as to cause all of the red objects to fall while leaving the green shapes on the screen perched on other green objects or blue objects.

Blue objects are neutral and thus it is acceptable to allow blue shapes to fall, or stay on the screen, while working towards the overall goal of causing all of the red objects to fall off of the screen. Blue objects can be removed directly by clicking on them.

Red objects come in two shades; light red and dark red. The light red shapes can be removed by clicking on them, but the dark red shapes cannot be removed by clicking on them.

If box has a face, then it shows facial expressions depending on what motion it is experiencing and what color it is. If the object is green and has a face it shows anxiety when motion occurs, while red boxes have a frown until they begin to move in anticipation of being removed, at which point is begins to smile. There were nine levels taken from the online game that we implemented.

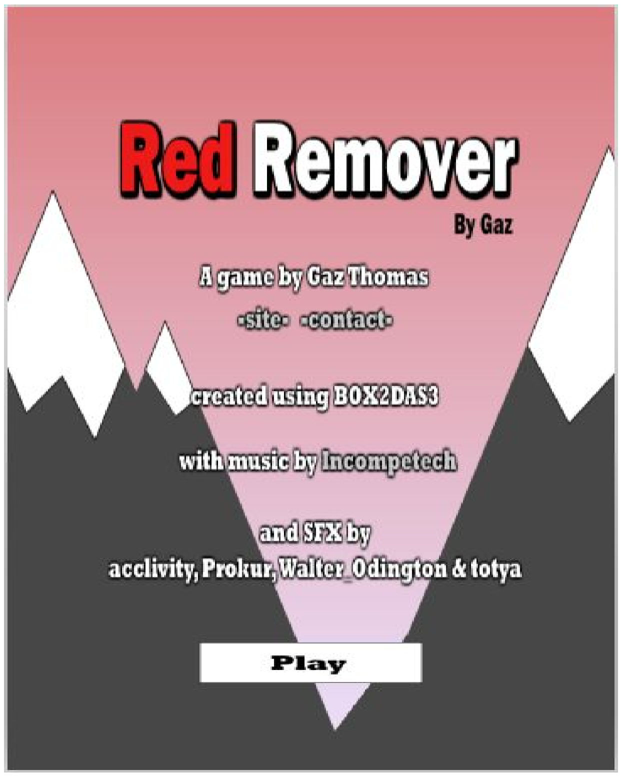
c. Target Platform:

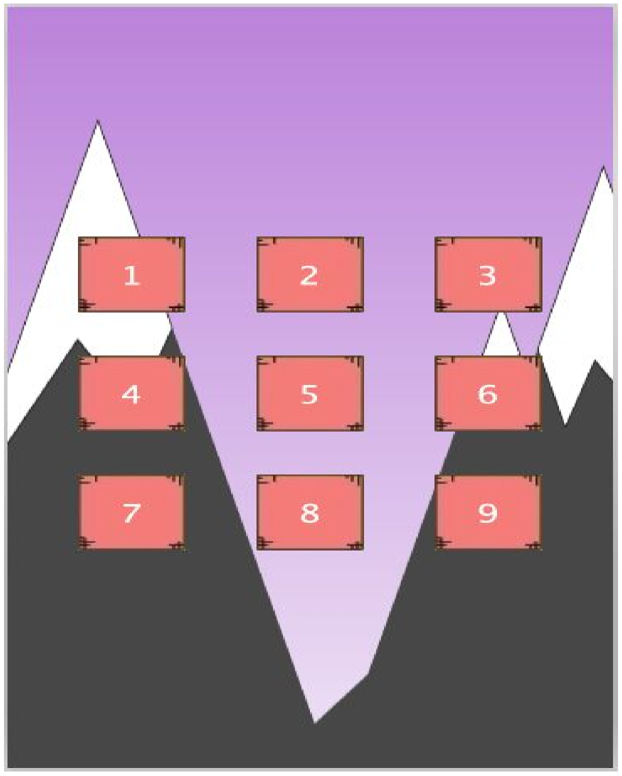
We designed and developed this game on simulator by using Samsung Galaxy S3 settings.

d. Intended Audience

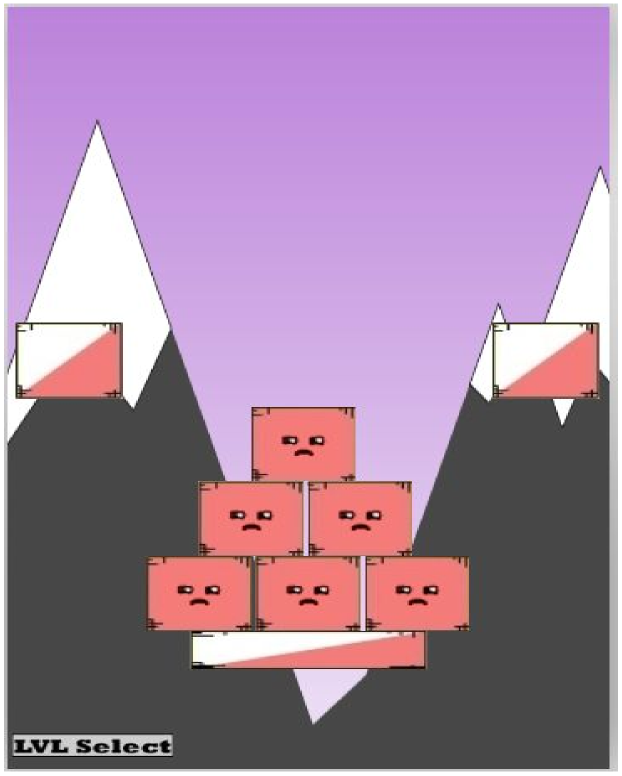
Dr. Kim, students in class CS571 and people who like to solve puzzle games.

2. **Screen Shots:**

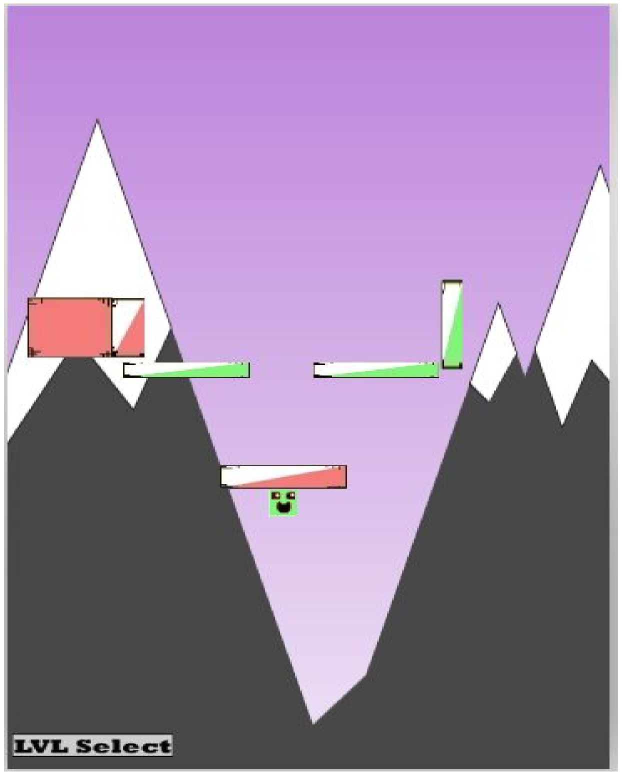




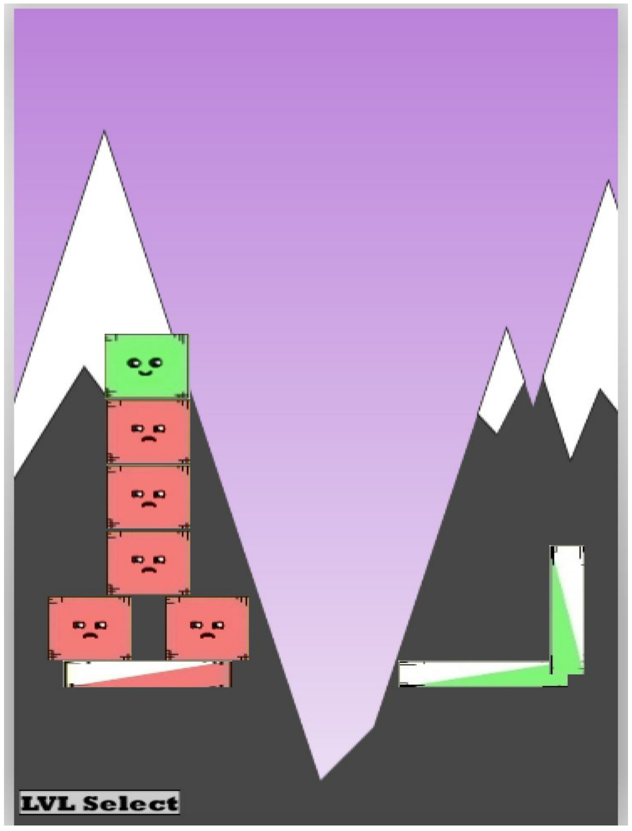
Main Screen Level Screen



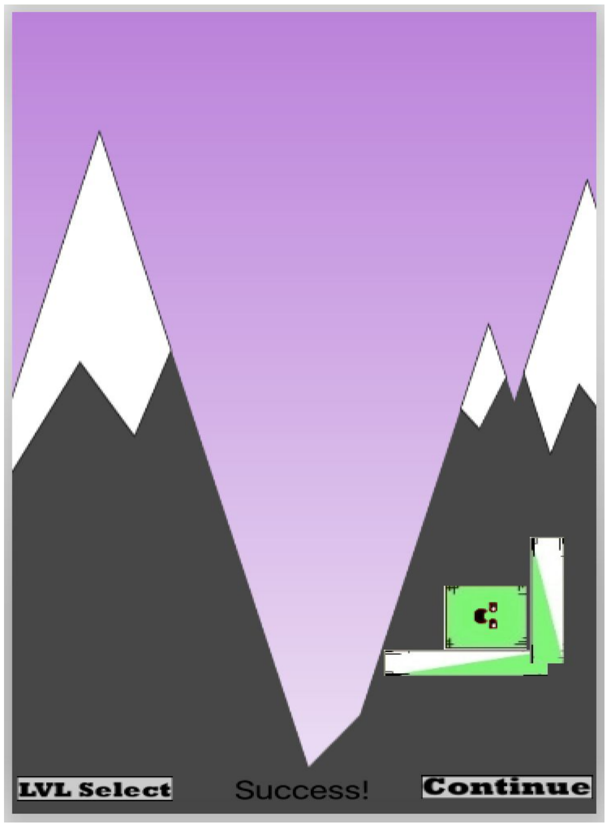
Level 1 Screen



Level 2 Screen

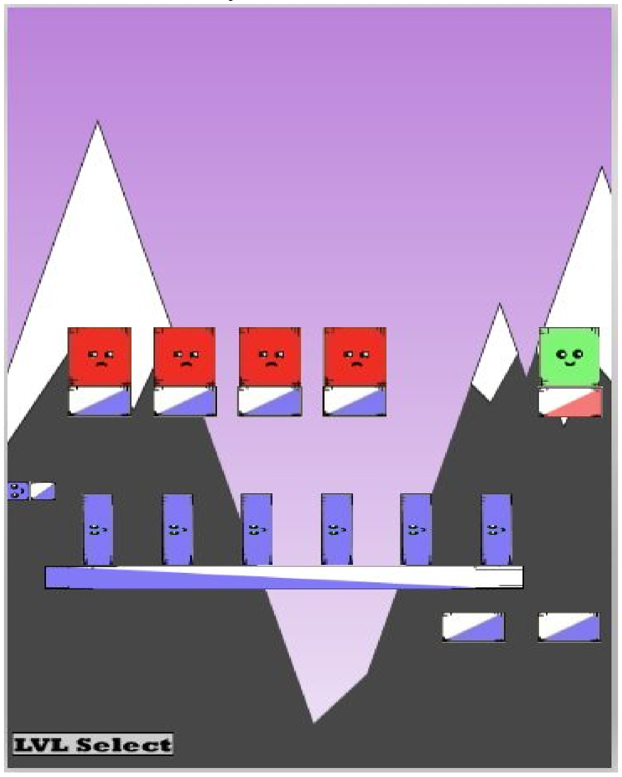


Level 3 Screen

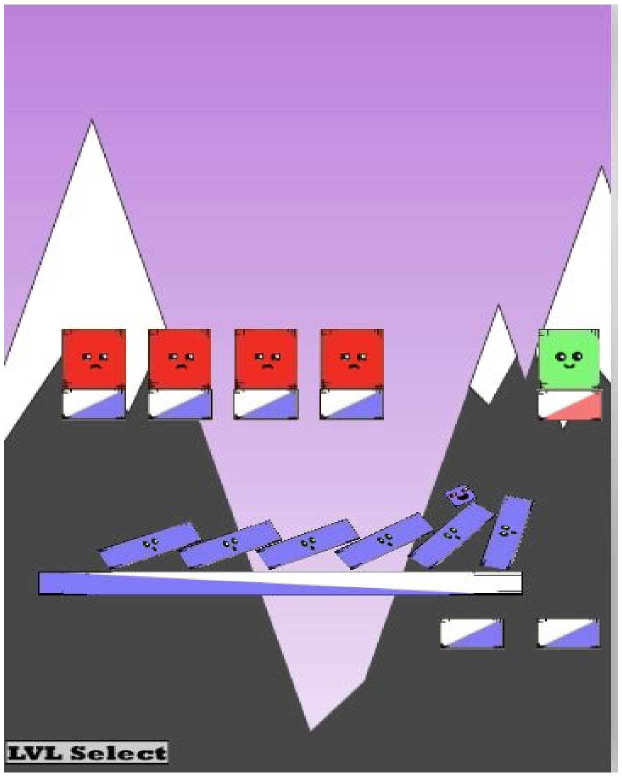


Level 3 Success Screen

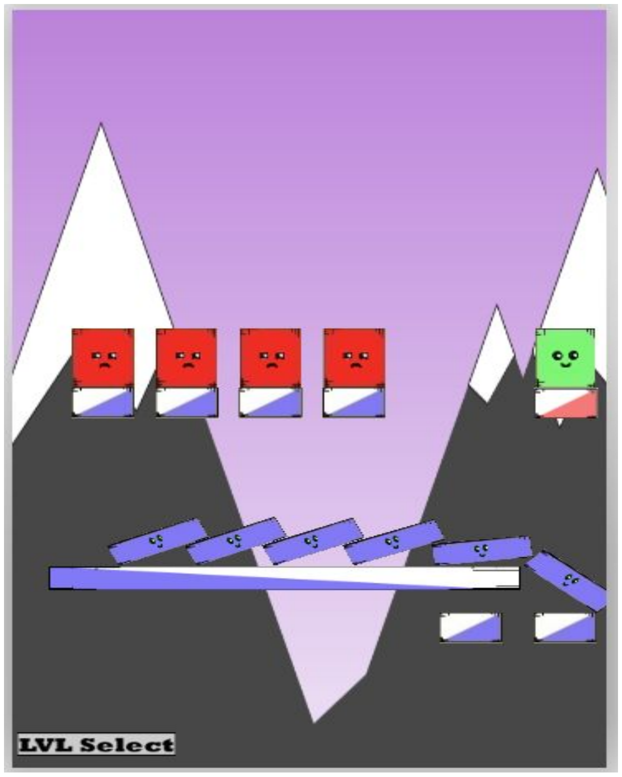
Level 4 – Move by Move



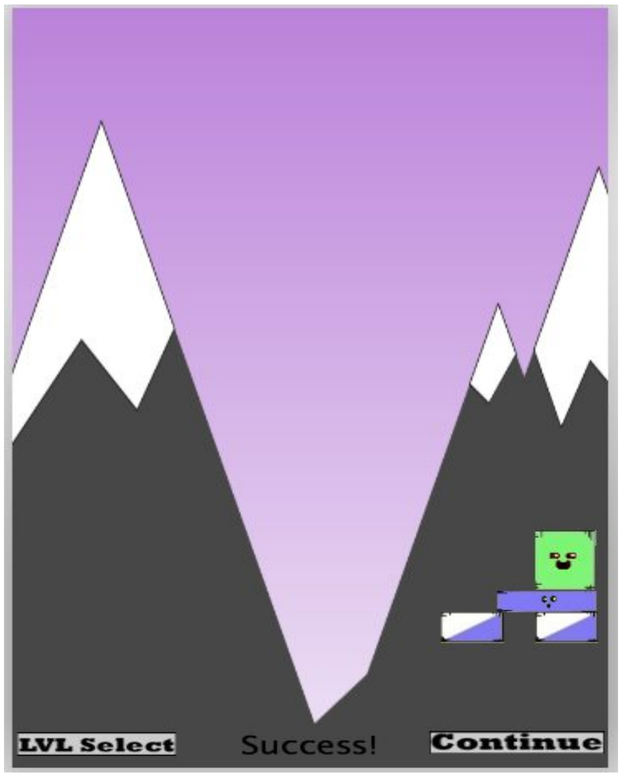
Level 4 Screen



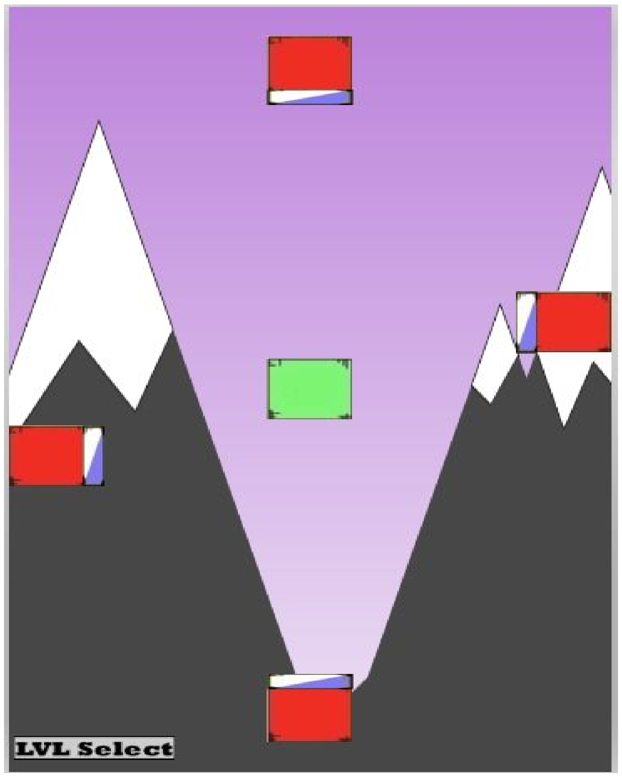
Level 4 Screen Move-1



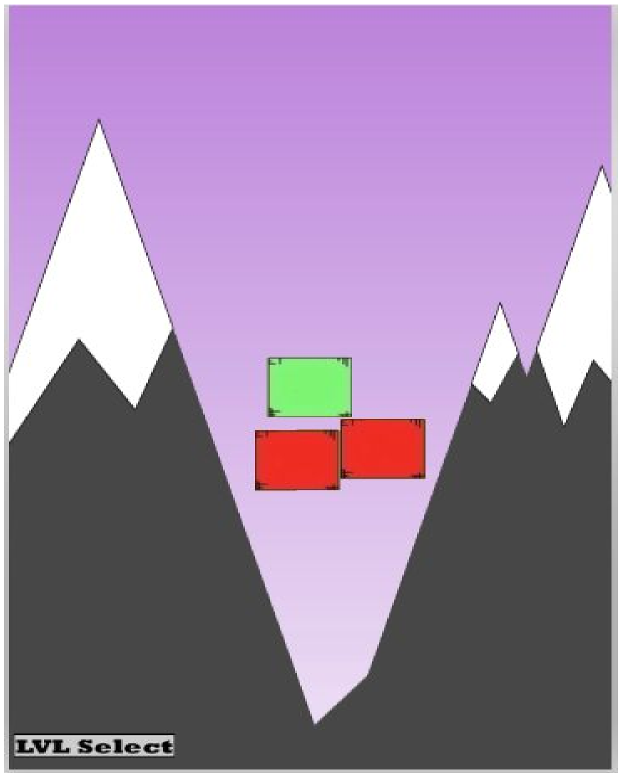
Level 4 Screen Move-2



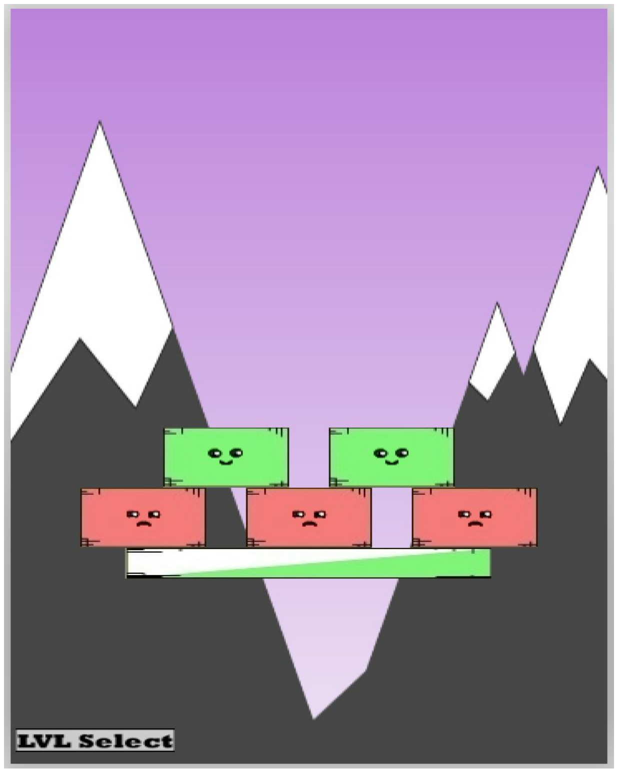
Level 4 Screen Success.



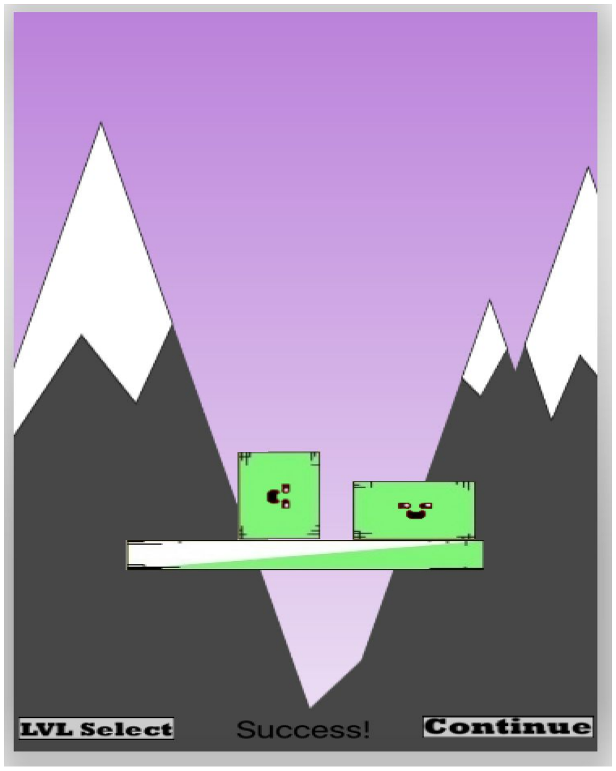
Level 5 Screen



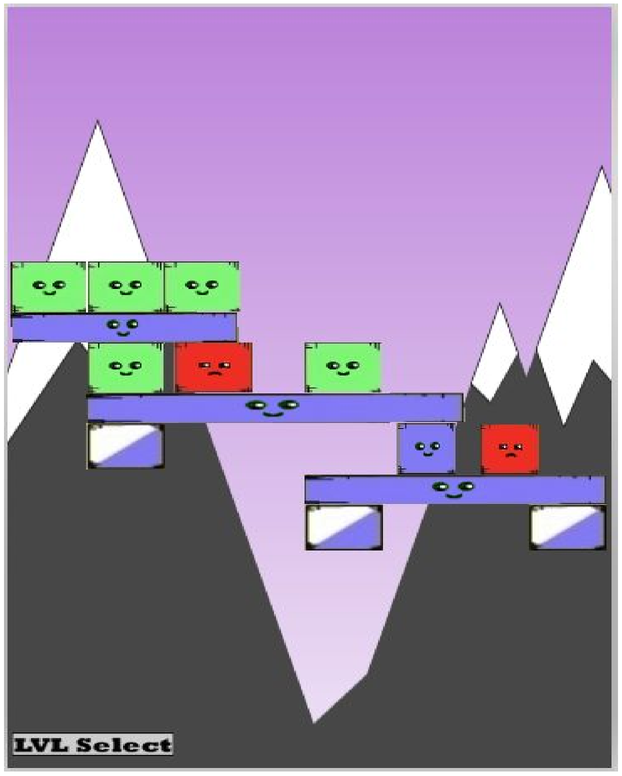
Level 5 Screen Move-1



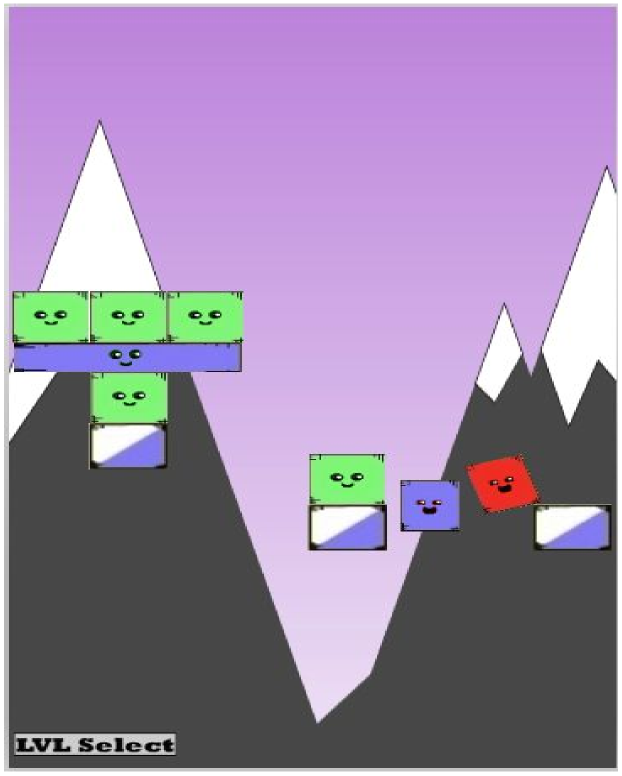
Level 6 Screen



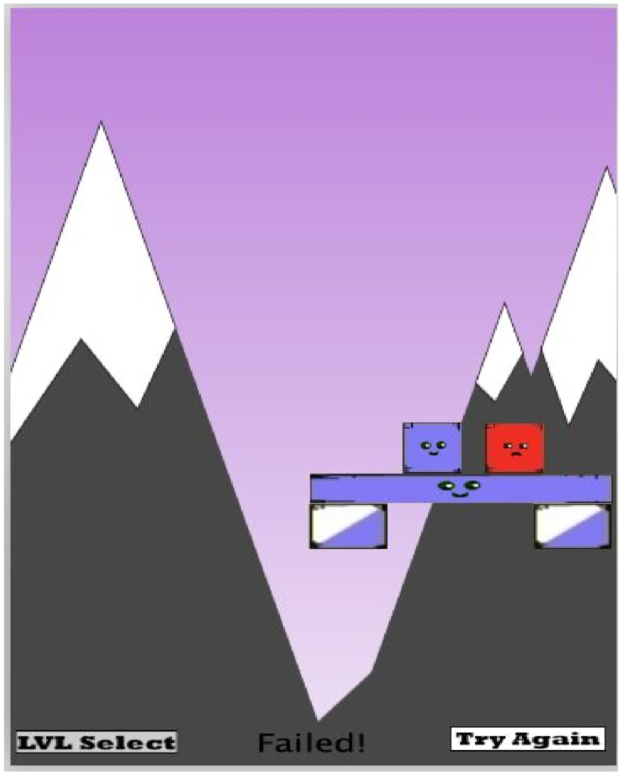
Level 6 Success Screen



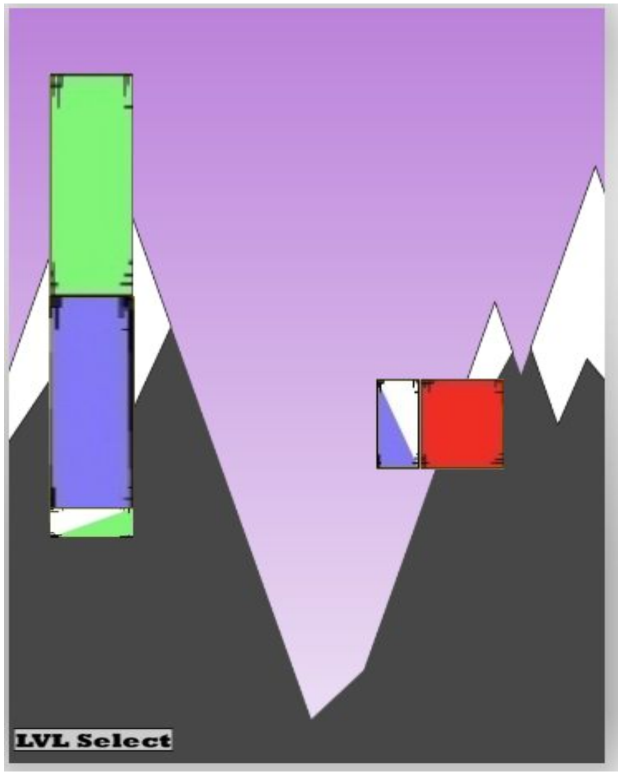
Level 7 Screen



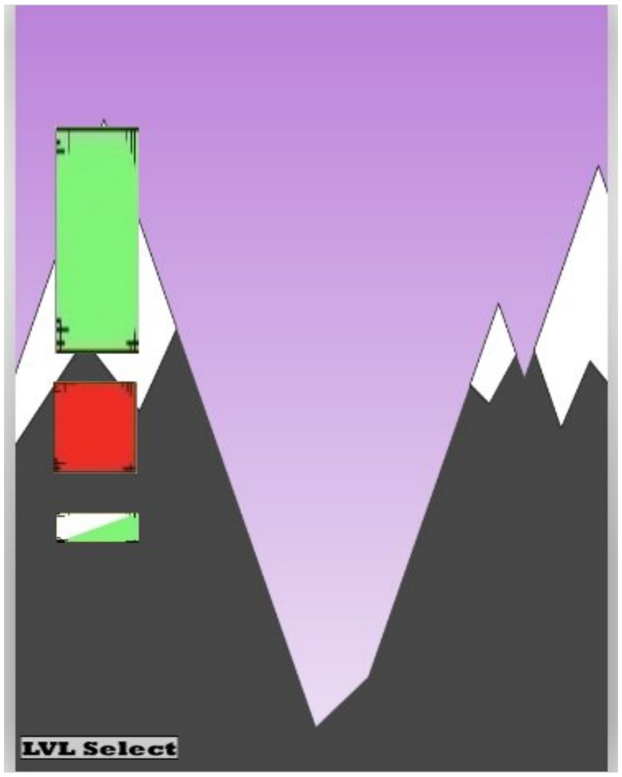
Level 7 Screen Move -1



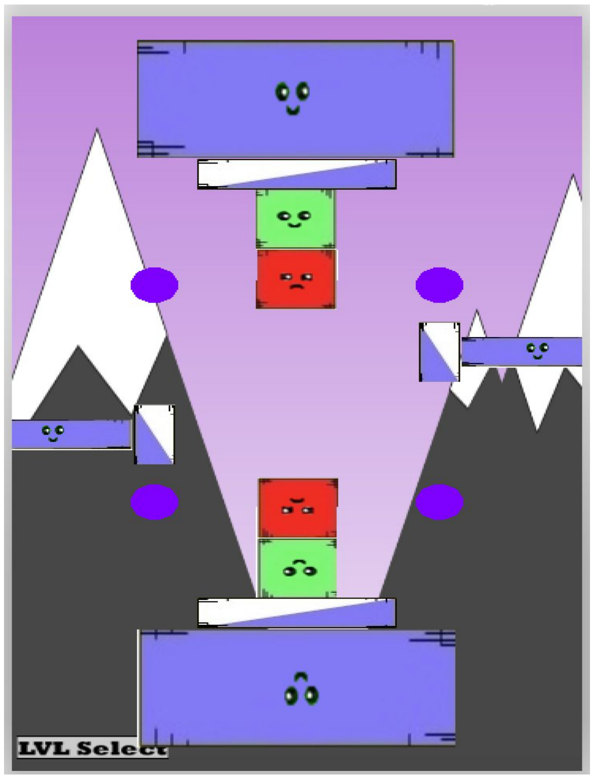
Level 7 Screen – Failed.



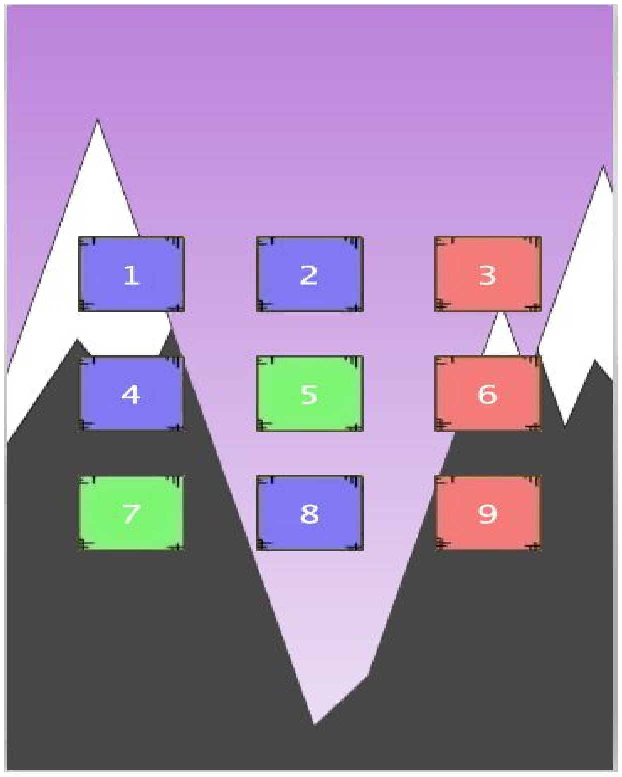
Level 8 Screen



Level 8 Screen Move



Level 9 Screen



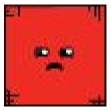
Level Screen: Red- not attempted, Blue-Completed, Green- Attempted.

End of Game Screen ("Well Done")
(Appears only if 9 levels are completed successfully)

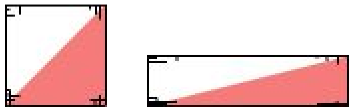
3. Software Design Plan

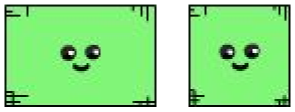
a. **Diagram of Components with Explanation:**

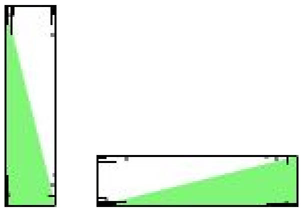
Light Red Square (Clickably Removeable)

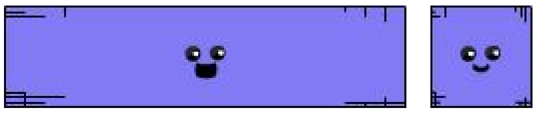
Dark Red Square (Not Clickably Removable) 

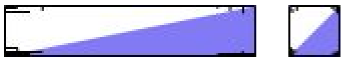
Light Red Rectangle (Clickably Removable)

Light Red Square/Rectangular Platforms (Clickably Removable) 

Green Squares/Rectangles (Objects to Save from Falling/Unclickable) 

Green Rectangular Platforms (Unclickable and Fixed) 

Blue Rectangles/Squares (Clickably Removable) 

Blue Rectangular/Square Platforms (Clickably Removable) 

Green Circle (Object to Save from Falling)
There are more drawings, which we used in our program.

Clickable: Light red objects and blue objects are removable by clicking on them.

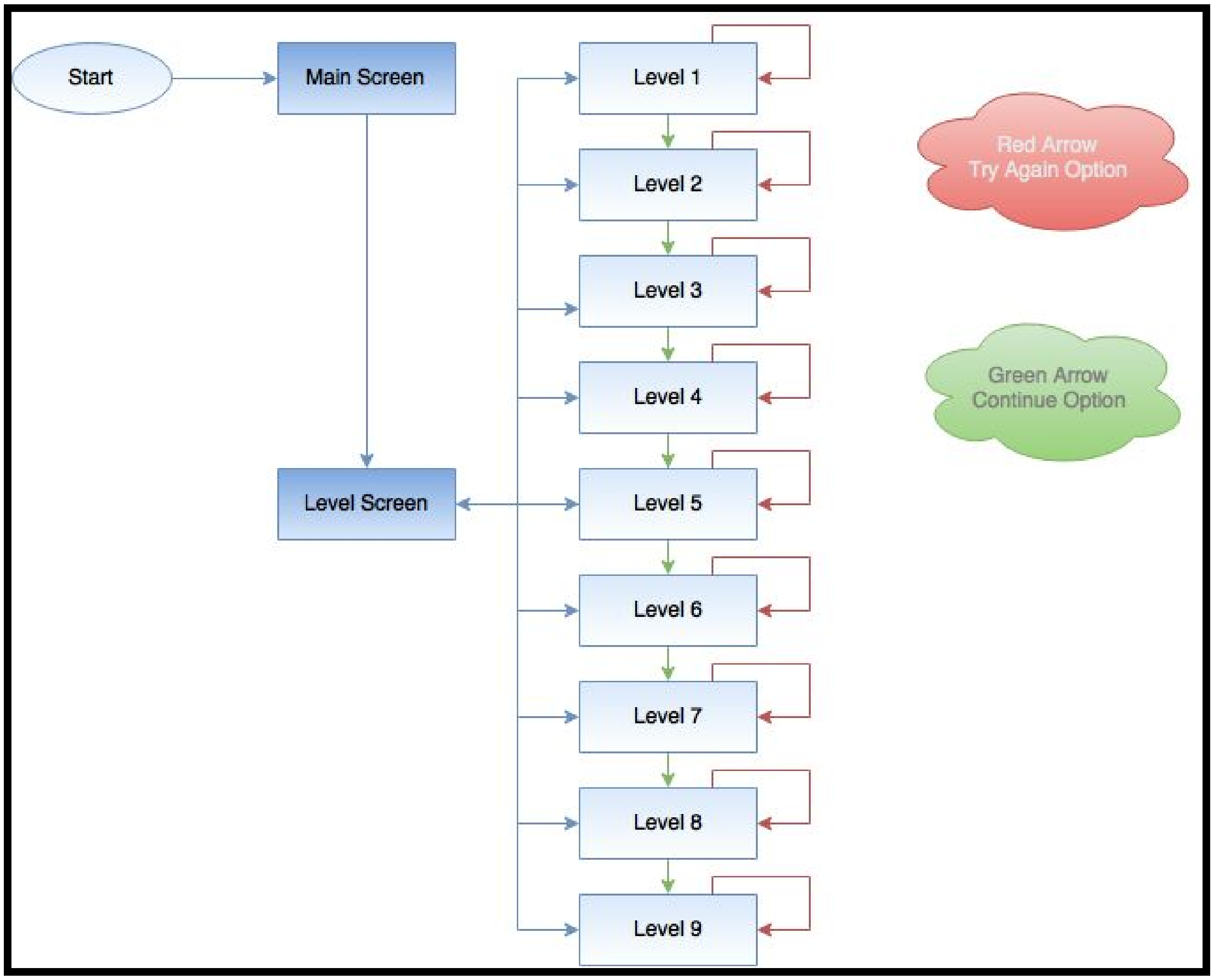
Fixed: All platforms and faceless blue objects are fixed, and do not move. Platforms are used to support other objects.

Not Clickable: All green objects and dark red objects are not clickably removable.

Moveable: Most objects can experience motion. The only ones that don't move are objects that are diagonally half colored.

The user has to click removable objects in such a way so as to cause all red boxes to be removed from screen, while preserving all green objects. Blue objects are neutral. So, it does not matter of they are removed or stay on screen. Blue objects are used to aid the user in removing red objects, and to help keep green objects on the screen.

4. Architecture



When the user starts the game and he/she will see the main screen in which are citations to the original authors of the online game. On the main screen there is a play option which takes the user to the level screen.

The level screen displays all the levels as squares with different colors (Red- not attempted, Blue-Completed, Green- Attempted.)

Level-x: Contains actual objects with different layouts. User always has the option for go back to the level screen. Once a level is attempted and the player succeeds in completing the level or not a ‘success’ or ‘failed’ message is displayed along with a ‘continue’ and ‘try again’ option. The user can choose from these options after attempting a level and move ahead to the next level, or try again depending on if they successfully completed the level or not.

5. Work Division

In this game we have a total 9 levels. Each group member took 3 levels and coded the scene for that level.

Benjamin:

1. Implemented level 1,4 and 7.
2. Prepared Project template for whole game.
3. Prepared PPT for Presentation.
4. Reviewed Report.

Tanvi:

1. Implemented level 3,6 and 9.
2. Reviewed report.
3. Suggested the game idea that was decided on and proposed levels to be implemented by other members of the group.
4. Final integration and submission.

Rakesh:

1. Implemented level 2,5 and 8.
2. Prepared final report.
3. Prepared project proposal.
4. Fixed bugs in project template.

Each member reviewed the code written by other team members and helped other members fix bugs in their levels.

6. Challenges and Solution:

All members faced many common issues along with some specific to the level difficulty. We all found solutions to the various challenges and tried to implement the best solutions in our levels.

1. Faced how to give motion to objects:
   Default gravity was creating problems for us, so we set the gravity to 0 and used linear velocity API functions to set objects’ specific motion in some of the levels.
2. Scaling issue:
   We enlarged the object size by scaling them, but it was creating problems for use in collision handling by the Box2D physics engine. So we used either newImageRect(width,height) or calculated the new width and height based on the scaling factors and passed in a shape table to the physics engine’s “add body” function.

3. Checking Status: Failed or Success

    Each level has its own layout and the status check logic depends upon the layout of the level. If there are only red boxes then we use some counter and decrease that count variable if it was clicked or moved out of the screen.

    In second scenario when we have green box along with red box then we used 2 counters variable one for mandatory removable objects and another one is for non-removable objects. If any green box is out of the screen then we failed the level. If all green boxes present on screen and red boxes are removed then level status as 'success'.

    In both the scenarios we used timers for checking the level status.

4. Timer Issue:

    We used timers to check our level status, but timers were causing problems. After level completion sometimes timers were left running, which was an issue. So we killed all the timers and started the timers according to specific conditions. Also we used flags that were set at the right time to indicate that the level was complete (success or failed) at which point all of the timers were canceled.

5. When we have to add objects into physics:

    When we create all moveable objects as dynamic and all non-movable objects static. When we clicked on static removable object then associated dynamic object with this static object, moved in another direction. For resolve this issue we create one table and associate affected object number and x-force and y-force for source objects and add affected moveable object into physics when some event generated like tap or collision and applied force.

6. Objects were not removed from screen once level complete:

    In hide function we removed all the objects, which were created in the scenes' create events or in the scenes' show method, and even outside these methods. The hide scene event worked fine if the object was created inside the scene's create or show method and associated with the scene group. To resolve this problem we created all objects inside these methods, made them invisible and put them in the scene group. After that the hide function worked fine and no objects were left alive upon level completion, which was having the effect of the objects from a previous level being seen in the current one.

7. Managing the objects:

    We did not use any object-oriented approach, so for managing all the objects, we used tables in our program. Also we saved object's related information in standard tables like motionFlag, mandatoryFlag, remoableFlag, which contained lists of object properties.

Note: There were other issues in the development of this game, but these were the major ones.

7. Reference:

Red Remover: http://www.silvergames.com/red-remover (Links to an external site.)
http://www.thegamehomepage.com/play/red-remover