*Deliverable 1* | **Gizmoball**

Group JS5

**Contents**

1. **Revised Specification**

2. **Use Cases**

3. **Class Diagram**

4.**Screenshots of the GUI**

5. Descriptions of:

a. The**Physics Loop**

b. The **Triggering System**

6. **Project Plan**

Statement: This submission is entirely work of the group.

**1. Revised Specification**

There are two modes in gizmo ball, build in which the user can add and edit gizmos and run in which gizmos and the ball move and interact.

**Playing Area:**

* L is the standard distance unit and is equal to one side of a square bumper
* Playing area minimum size 20L wide and 20L high (400 square bumpers)
* The upper left corner is the starting coordinate of 0,0 and increase going right and down

**In building mode:**

* Using the gizmos toolbar user should be able to

-   Create bumpers (square, circular, triangular)

-   Create flippers

- Create absorbers

* From the operations toolbar user should be able to

-   Edit gizmos (delete, rotate, move)

-   Connect gizmos allowing one to trigger the action of another

* Connect the action of bumpers and flippers to triggers (e.g. button pressed)
* Connect gizmo trigger to gizmo action (can’t chain)
* Save and load a game to and from a file named by the user
* Gizmos must snap onto a grid (0,1) (0,2) (0,3) ... (19,19) and can only be place on grid locations
* Should be able to move previously placed gizmos.
* Gizmos should not be able to be placed if overlapping with existing gizmo.
* Rotate gizmos by 90 degrees clockwise
* Add a ball to the playing area (specify position and velocity) (can’t overlap gizmos or boundaries of playing area)
* Switch to running mode
* Quit application

**In running mode:**

* The user should be able to play and stop the game
* Ball velocity must range from 0.01L to 200L per second
* Preferably 20 frames per second
* Ball must be redrawn at least every 0.05L by 0.05L for smooth animation
* Reasonable interactions between ball and gizmos (expected direction and velocity when ball bounces)
* Gravity should affect the ball (25 L/sec2)
* Friction should affect the ball (Vnew = Vold \* (1 - mu \* delta\_t - mu2 \* |Vold| \* delta\_t)(mu = 0.025 / s)(mu2 = 0.025 / L)
* User be able to press keys that could trigger gizmos
* Save and load a game to and from a file named by the user
* Switch to building mode
* Quit application

**In both:**

* Have a menu toolbar, where the user has the options to save or load a game
* Ball must have a diameter of 0.5L

**Gizmos**

Square

* Edge length 1L
* Coefficient of reflection: 1.0

Circle

* Diameter 1L
* Coefficient of reflection: 1.0

Triangle

* right-triangular shape with sides of length 1L and hypotenuse of length Sqrt(2)L
* Coefficient of reflection: 1.0

Flipper (left & right)

* rectangular rotating shape with bounding box of size 2Lx2L
* Rotates 90 degrees when triggered
* Coefficient of reflection: 0.95
* During run mode, a flipper should never extend outside its bounding box.
* When triggered it has an angular velocity of 1080 degrees per second

Absorber

* A rectangle with integral-length sides
* Shoots out stored ball when triggered
* When a ball hits an absorber, the absorber holds it in the bottom right-hand corner
* When triggered the captured ball will shoot upwards with a velocity of 50L/s
* Cannot be rotated

Outer Walls

* Impermeable barriers surrounding the playfield.
* Coefficient of reflection: 1.0
* The user cannot move, delete, or rotate the outer walls.
* The outer walls lie just outside the playing area.

**File Format**

* Each line in the file is a command
* Command consist of operation code and zero or more arguments
* Names must be unique
* Names must be declared before calling methods on
* Deleted items cannot have methods called on

**2. Use Cases**

Add Gizmo

Preconditions: Be in building mode

Triggers: Press Add Gizmo button

Scenario:

1.          The system prompts the user to select a type of gizmo

2.          The user selects the type of gizmo

3.          The system prompts the user to choose the location where the gizmo will be placed 4.          The user chooses the location

5.          The system verifies that the location is valid

6.          The gizmo is placed

Alternative path:

5.1 There is already a gizmo in the location. The system informs that the location is not valid

5.2 The location is outside the build area. The system informs that the location is not valid

6    Go back to step 3 (main path)

Post Condition: The gizmo is added to the game board

Rotate Gizmo

Preconditions: Be in building mode, having a gizmo already selected

Triggers: Press Rotate button

Scenario:

1. The gizmo is rotated in the game board

Alternative path:

1.1 This type of gizmo is not rotatable. The game board is unchanged

Post Condition: The gizmo is rotated in the game board

Move Gizmo

Preconditions: Be in building mode, having a gizmo already selected

Triggers: Press Move button

Scenario:

1. The system prompts the user to choose the location where the gizmo will be moved to

2. The user chooses the location

3. The system verifies that the location is valid

4. The gizmo is moved

Alternative path:

3.1     There is already a gizmo in the location. The system informs that the location is not valid

3.2     The location is outside the build area. The system informs that the location is not valid

3.3    The location is the current location. The system informs that the location is not valid 4.      Go back to step 1 (main path)

Post Condition: the gizmo is moved to another position in the game board

Connect Gizmos

Preconditions: Be in building mode

Triggers: Press Connect button

Scenario:

1. The system prompts the user to choose the gizmo that will trigger an action to another gizmo when is hit

2. The user chooses the gizmo

3. The system prompts the user to choose the second gizmo which will make an action when the first gizmo’s trigger is activated

4. The user chooses the second gizmo

Alternative path:

2.1, 4.1 The user chooses a position where no gizmo exists. The system informs that the location is not valid

5.   Go back to step 1 (main path)

Post Condition: The two selected gizmos are connected to each other

Add Absorber

Precondition: Be in building mode

Triggers: Press place absorber button

Scenario:

1. The system prompts the user to choose the location for the top left corner

2. The user chooses the location

3. The system verifies that the location is valid

4. The system prompts the user to choose the location for the bottom right corner

5. The user chooses the location

6. The system verifies that the location is valid

7. The absorber is placed

Alternative path:

3.1     A gizmo exists at this location; the system informs that the location is not valid

4.        Go back to step 1 (main path)

5.1     A gizmo exists at this location; the system informs that the location is not valid

5.2     A gizmo exists in the absorber area; the system informs that the location is not valid

5.3     The location select is above or left of the first location. The system informs that the location is not valid

6.       Go back to step 1 (main path)

Connect Keys with Gizmos

Preconditions: Be in building mode, having a gizmo already selected

Triggers: Press Connect Key button

Scenario:

1. The system prompts the user to choose the keyboard key to that will trigger a gizmo when is hit

2. The user chooses the keyboard key

3. The system prompts the user to choose the gizmo which will make an action when the key is pressed or released

4. The user chooses the gizmo

Alternative path:

4.1     The user chooses a position where no gizmo exists, the system informs that the location is not valid

5.   Go back to step 1 (main path)

Post Condition: The keyboard key is connected to a gizmo

Disconnect Gizmos

Preconditions: Be in building mode

Triggers: Press Disconnect button

Scenario:

1. The system prompts the user to choose the gizmo to disconnect
2. The user chooses the gizmo
3. The system informs you of remove connections

Alternative path:

2.1     The user chooses a position where no gizmo exists the system informs that the location is not valid

3.   Go back to step 1 (main path)

Post Condition: The selected gizmo has no connections

Delete Gizmo

Preconditions: Be in building mode, having a gizmo already selected

Triggers: Press Delete button

Scenario:

1. The system prompts the user to choose the location of the gizmo which will be deleted

2. The user chooses the location

3. The system verifies that the location is valid

4. The gizmo is deleted

Alternative path:

3.1     There is no gizmo to be deleted at this position the system informs that the location is not valid and that

4.    Go back to step 1 (main path)

Post Condition: The gizmo is deleted from the game board

Add ball

Preconditions: Be in building mode, no ball already placed in the playing area

Triggers: Press Add Ball button

Scenario:

1. The system prompts the user to choose the position where the ball will be placed

2. The user chooses the position

3. The system prompts the user to choose the velocity of the ball

4. The user chooses the velocity

Alternative paths:

2.1     The location contains a gizmo. The system informs that the location is not valid

3.       Go back to step 1 (main path)

3.1     The velocity value is not a valid number. The system informs that the velocity is invalid

Post Condition: The ball is added in the playing area

Modify Gravity

Preconditions: Be in running mode, give a value to the gravity field

Triggers: Press Gravity button

Scenario:

1. The system checks if the value in the gravity field is in range.

2. The gravity is modified.

Alternative Path:

2.1     The gravity value is not a valid number. The system informs that the gravity is invalid

Modify Friction

Preconditions: Be in running mode, give a value to the friction field

Triggers: Press Friction button

Scenario:

1. The system checks if the value in the friction field is in range.

2. The friction is modified.

Alternative Path:

2.1     The friction value is not a valid number. The system informs that the friction is invalid

Switch to Run mode

Preconditions: In Build mode

Triggers: Press Run button

Post Condition: The system switches the mode from build mode to running mode

Switch to Build mode

Preconditions: In run mode

Triggers: Press Switch button

Post Condition: The system switches the mode from running mode to build mode

Press key connected to gizmo

Preconditions: Be in running mode, have the keys connected to actions of gizmos

Triggers: Press a key associated with an action

Post Condition: The gizmos associated with the key moves

Save Game

Preconditions: None

Triggers: Press Save option

Scenario

1. The game pauses

2. The system prompts user to select save details

3. The system saves the current state

Alternative path

3.1    The file already exists; the system ask user if they want to overwrite

Post Condition: The current state is saved to file

Load Game

Preconditions: Have a save file already stored

Triggers: Press Load option

Scenario

1. The user selects a saved file

2. The system loads the saved state

Alternative Path

2.1 The file is invalid, the system notifies the user

2.2 The file does not exist, the system notifies the user

Post Condition: The game selected is loaded to the board

Start Game

Preconditions: Be in running mode

Triggers: Press Start button

Post Condition: Simulation starts

Stop

Preconditions: Be in running mode, the ball has to be moving

Triggers: Press Stop button

Post Condition: The simulation stops

Quit Game

Preconditions: None

Triggers: Press Quit button

Post Condition: Game closes

New Game

Preconditions: None

Triggers: Press New Game button

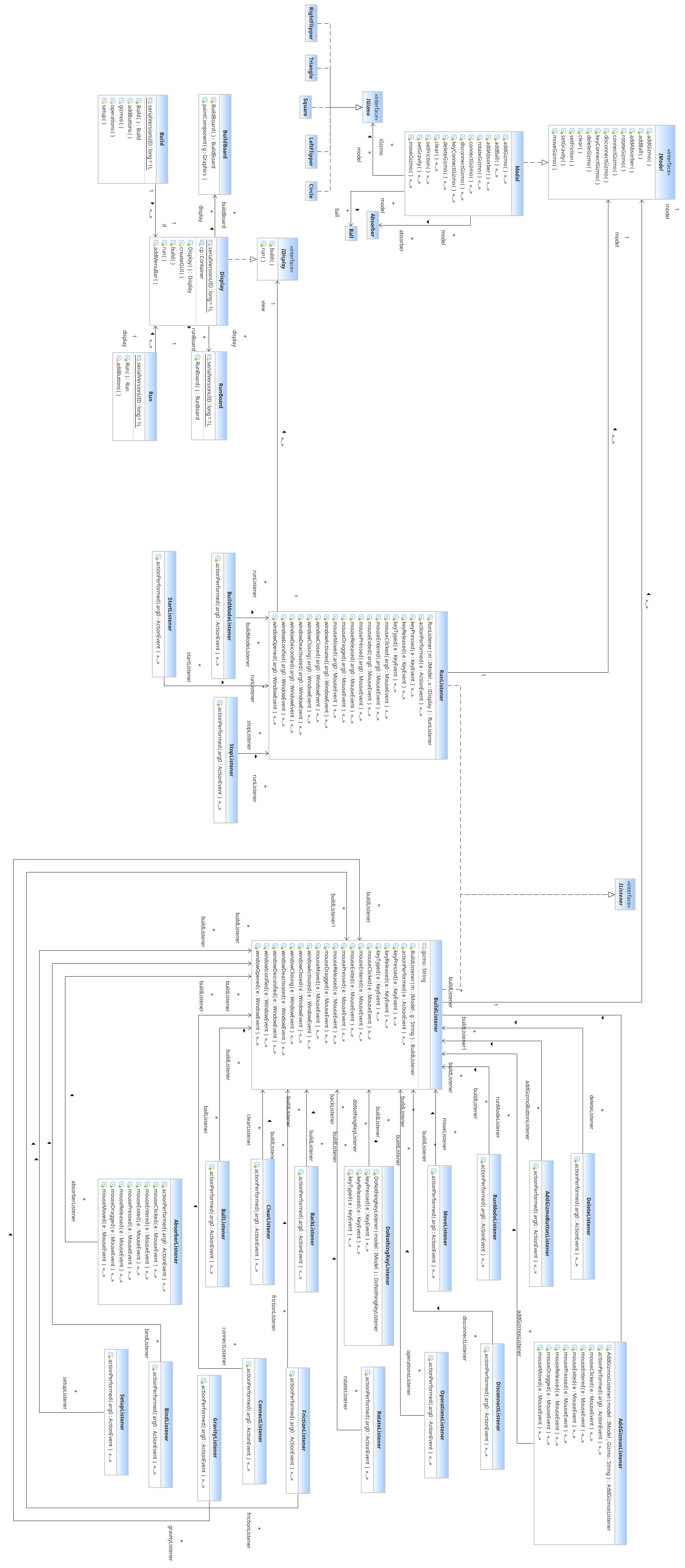
Post Condition: A new game starts

Clear

Preconditions: Be in build mode

Triggers: Press Clear button

Post Condition: Board is cleared



**3. Class Diagram & Descriptions**

**Model**

IModel: This is the interface for the model which contains all the abstract methods which deal with gizmos and the physics.

Model: Model is the class which implements the IModel class.

IGizmo: This is the interface class for making objects of the gizmos listed below.

-Circle

-Triangle

-Right Flipper

-Square

-Left Flipper

Absorber: This is the class for the gizmo absorber. This class does not implement the IGizmo interface because the absorber’s properties vary too much from the other gizmos.

Ball: This is the class for making an object of the ball.

**Display**

IDisplay: This is the interface for the view part of the system.

Display: Display is the class which implements the IDisplay class so this class will create, display and redisplay the gui in all its various states.

BuildBoard: This class will create the board in building mode

RunBoard: This class will create the board in running mode

Build: This class will create the buttons in building mode

Run: This class will create the buttons in running mode

**Controller**

IListener: This is the interface for the controller part of the system.

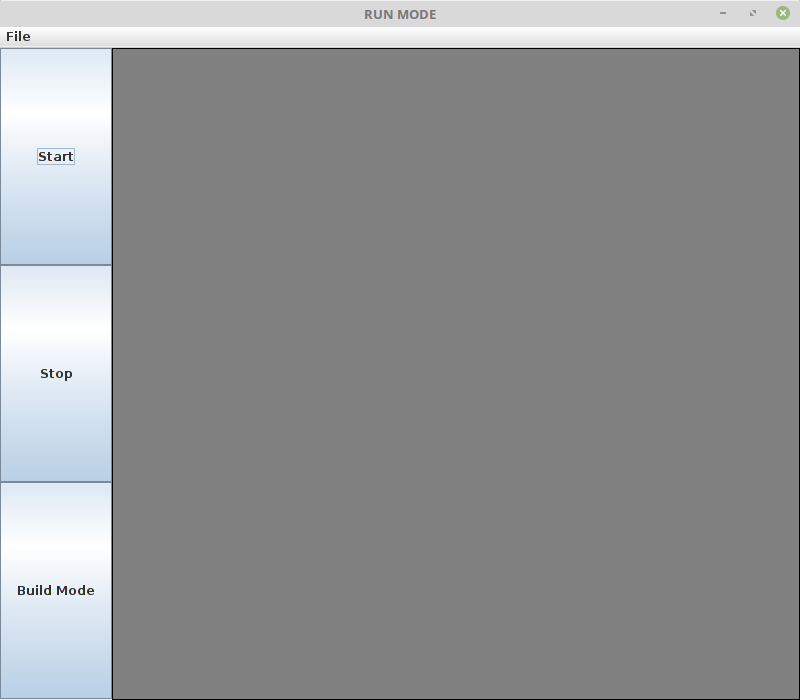
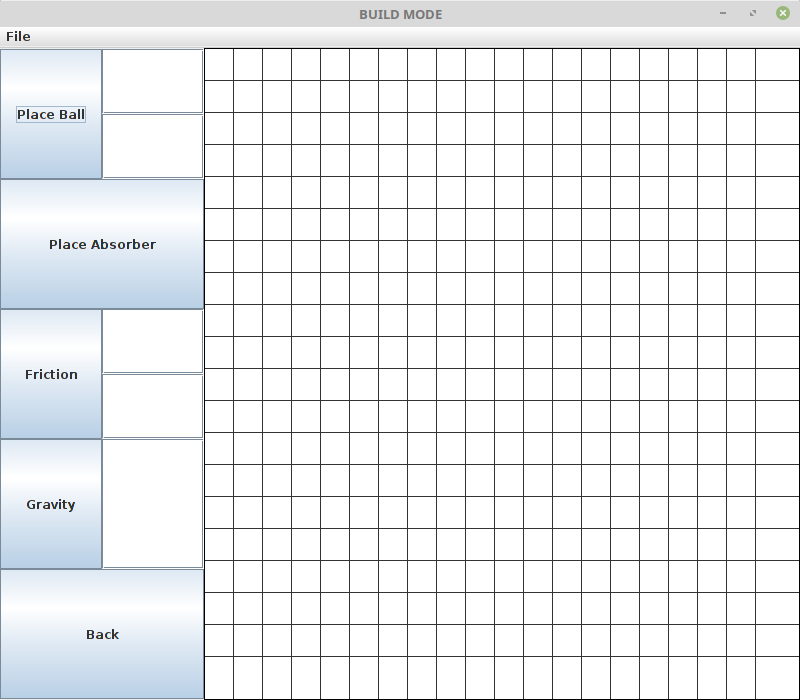
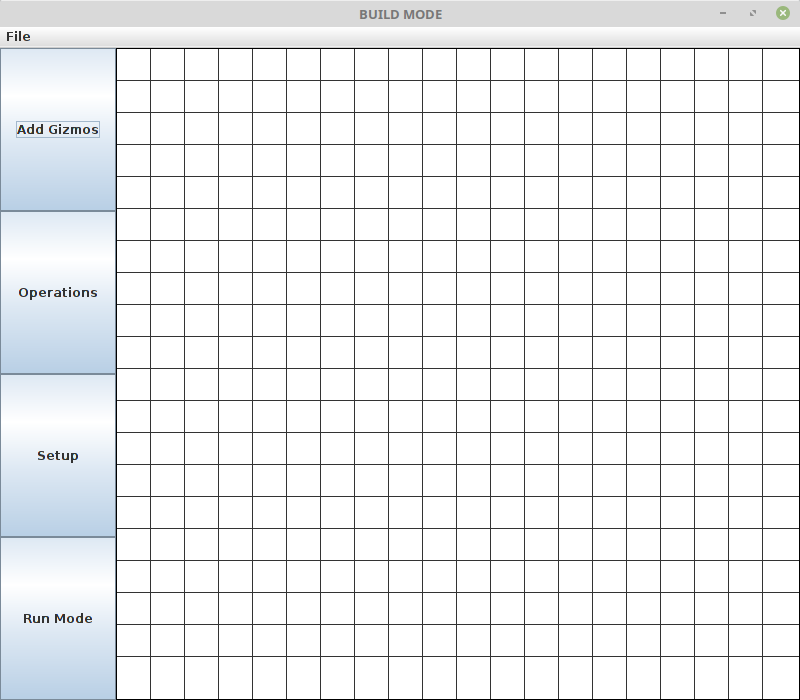
BuildListener: This is one of the two classes which implements the IListener interface. This listener reacts to any key or mouse input and calls the specific listener linked to that action. All the Listeners linked to the Build Listener are as follows:

* AddGizmoListener: Activated by square,triangle, circle, or flipper button
* MoveListener: Activated by the move button. The process of moving a gizmo on the board will then begin.
* FrictionListener: Activated by the friction button. The value for friction will then be updated in the model.
* BindListener: Activated by the bind button. The process to bind a key to a gizmo will then take place.
* DeleteListener: Activated by the delete button. The process for deleting a gizmo or ball then takes place.
* RotateListener: Activated by the rotate button. The process for rotating a gizmo will then begin.
* ConnectListener: Activated by the connect button. The process for connecting one gizmo to another will then take place.
* DisconnectListener: Activated by the disconnect button. The process for disconnecting a gizmo from another gizmo begins.
* SetupListener: Activated by the setup button. This will bring up the options add a ball, absorber, edit gravity and friction
* OperationsListener: Activated by the operations button. This will bring up operations which can be used on gizmos such as delete, move, rotate, etc.
* GravityListener: Activated by the gravity button. The value for gravity is updated in the model.
* TriangleListener: Activated by the triangle button. A triangle object will then be created in the model and displayed in the gui.
* BackListener: Activated by the back button. This will return to the main build mode button menu
* ClearListener: Activated by the clear button. This will clear the board of all objects leaving the board completely empty.
* RunModeListener: Activated by the Run mode button. This will switch the view from building mode to running mode.
* DoNothingKeyListener: Activated when no key is pressed
* AbsorberListener: Activated by the absorber button. An absorber object will then be created in the model and displayed in the gui.
* AddGizmoButtonListener: Activated by the add gizmos button. This will bring up the options to add all types of gizmos.

RunListener: This is the second out of 2 classes to implement the IListener interface. This listener reacts to any key or mouse input and calls the specific listener linked to that action. The listeners in run mode are:

* BuildModeListener: Activated by the build mode button. This will switch the view from running mode to building mode.
* StartListener: Activated by the start button. This will start the gizmo ball moving and the game will run.
* StopListener: Activated by the stop button. This will stop all movement in the game, essentially freezing the board.

**4. Screenshots of the GUI**



**5. Descriptions of:**

a. The Physics Loop

The ball moves a set number of times per second (20 by default). In this case the ball moves every 0.05 seconds. Every time the ball moves this is called a tick. At the start of every tick the program calculates the time until next collision for every ball/moving gizmo. If there is a collision before the end of the tick then the point of collision has to be redrawn in the GUI, this includes the triggering of any gizmos due to the collision. The ball’s velocity will be recalculated. After this the rest of the tick will elapse looking for further collisions and resolving them. So the ball will at the very least be redrawn every tick and in addition will be redrawn for every collision. The ball’s velocity will be recalculated with regards to gravity/friction after every tick.

b. The Triggering System

While in build mode a connect button will be provided that allows the user to add a trigger to a gizmo when another gizmo is hit to allow it to perform an action. The user will choose two gizmos and then select an action like to change colour; this action will be added to the second gizmo when the first is hit. For all gizmos the trigger occurs whenever the ball hits it. Chaining triggers are not allowed for example when the ball hits a gizmo it cannot trigger another gizmo which will in turn trigger another gizmo afterwards.

Alternatively, a trigger can be connected between a key and a gizmo. For example, select a flipper and assign it to spacebar to allow the user to press the spacebar to move the flipper. To handle keypresses java.awt.event.KeyEvent can be used to see what keypresses have occurred. To decide if a key has been pressed ‘KEY\_RELEASED’ is better to use than ‘KEY\_PRESSED’. This is because if a key is held in even for a short amount of time KeyEvent will return ‘KEY\_PRESSED’ multiple times therefore triggering the action multiple times. Each trigger can be connected to many gizmos.

**6. Project Plan**

**Task Allocation:**

Everyone – Class Diagram, Testing, Gantt Chart

Douglas – UI, Listeners, Build mode

Stuart – Running Mode(Physics), Interfaces

Lee – Running Mode(Physics), Interfaces

Ross – Animation, Documentation

Christos – Animation, Documentation

**Week 3**

-        Revised specification

-        Class diagram

-        Screenshots

-        Project Plan

-        Triggering system

-        Physics loop

-        Use cases

**Week 4**

-         Interfaces

-         Controller (listeners)

-         Ball velocity and collision

-         Basic build mode

**Week 5**

-         Flippers

-         Key presses

-         Animations

-         Loading and saving game states

**Week 6**

-         Final design

-   Revised class diagram w/ explanation of changes

-   Updated project plan

-         Validation Testing Strategy

-         Junit testing strategy

-         Executables (Absorber, collisions, key press for flipper, save/load)

-         Gantt chart

**Week 7**

-         Combining build and running mode to interchangeable user interfaces

-         Modify velocities, gravity, friction

-         Pausing

**Week 8**

-         Rotating gizmos

-         Connecting gizmos

-         Disconnecting gizmos

-         Clear

**Week 9**

-         Perfect collisions with the flippers

**Week 10**

-         Finalising all components for final demonstration