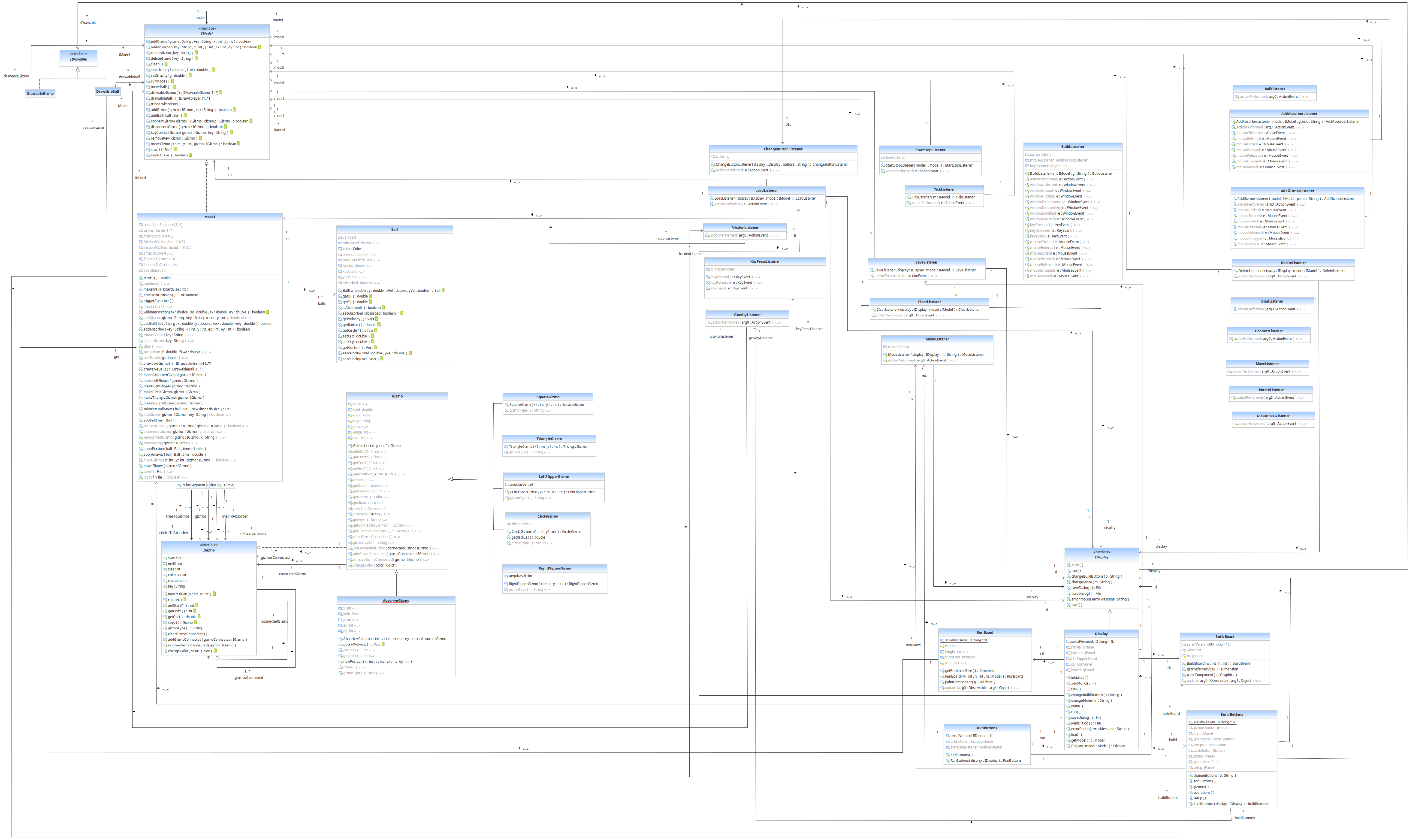
Deliverable 2 | **Gizmoball**

Group JS5

Contents

Statement: This submission is entirely work of the group.


«interface» IModel
addGizmo (gizmo : String , key : String , x : int , y : int) : boolean
addAbsorber (key : String , x : int , y : int , ex : int , ey : int) : boolean
rotateGizmo (key : String)
deleteGizmo (key : String)
clear ( )
setFriction (f : double , fTwo : double)
setGravity (g : double)
runMode ( )
moveBalls ( )
drawableGizmo ( ) : IDrawableGizmo [1..*]
drawableBall ( ) : IDrawableBall [1..*]
triggerAbsorber ( )
addGizmo (gizmo : IGizmo , key : String) : boolean
addBall (ball : Ball)
connectGizmo (gizmo1 : IGizmo , gizmo2 : IGizmo) : boolean
disconnectGizmo (gizmo : IGizmo) : boolean
keyConnectGizmo (gizmo : IGizmo , key : String)
removeKey (gizmo : IGizmo)
moveGizmo (x : int , y : int , gizmo : IGizmo) : boolean
save (f : File)
load (f : File) : boolean
«interface» IDrawable
DrawableGizmo
DrawableBall
Model
Ball
Gizmo
«interface» IGizmo
AbsorberGizmo
SquareGizmo
TriangleGizmo
LeftFlipperGizmo
CircleGizmo
RightFlipperGizmo
ChangeButtonListener
LoadListener
FrictionListener
KeyPressListener
GravityListener
StartStopListener
TickListener
SaveListener
ClearListener
ModelListener
BuildListener
BallListener
AddAbsorberListener
AddGizmosListener
DeleteListener
BindListener
ConnectListener
MoveListener
RotateListener
DisconnectListener
«interface» IDisplay
RunBoard
Display
BuildBoard
RunButtons
BuildButtons
model
iModel
drawableGizmo
drawableBall
balls
linesToGizmos
gizmos
linesToAbsorber
circlesToAbsorber
circlesToGizmos
gizmosConnected
connectedGizmo
connectedGizmo
frictionListener
keyPressListener
gravityListener
display
runBoard
bB
build
buildBoard
buildButtons
run

Gizmo Ball Design

Changes

A number of groups of listeners were merge into a single listener as they code was repeated in each one. Run listener was removed as the inheriting classes would not use the methods they gain from it. A number of the methods no longer inherit the build listener for the same reason. Listeners were added from save, load and tick which were missed in the original design. A keypress listener was added to check for when keys are pressed.
Absorber is now covered under the gizmo interface as it shared many of the same methods.

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

Ball:

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

IDrawable

An interface between the model and view to allow the view access to only necessary information to draw the gizmo, without the ability to modify them.

DrawableGizmo

A class that will allow the view to read information about where to draw the Gizmo.

DrawableBall

A class that will allow the view to read information about where to draw 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
BuildButtons:
This class will create the buttons in building mode
RunButtons:
This class will create the buttons in running mode

Controller

BuildListener:
An interface for controlling all the mouse inputs during build mode
AddGizmoListener:
Activated by square,triangle, circle, or flipper button, inherits BuildListener. The process of adding a gizmo will then begin then be displayed on the GUI
MoveListener:
Activated by the move button, inherits BuildListener. The process of moving a gizmo on the board will then begin.
BindListener:
Activated by the bind button inherits BuildListener. The process to bind a key to a gizmo will then take place.
DeleteListener:
Activated by the delete button inherits BuildListener. The process for deleting a gizmo or ball then takes place.
RotateListener:
Activated by the rotate button inherits BuildListener. The process for rotating a gizmo will then begin.
ConnectListener:
Activated by the connect button, inherits BuildListener. The process for connecting one gizmo to another will then take place.
DisconnectListener:
Activated by the disconnect button inherits BuildListener. The process for disconnecting a gizmo from another gizmo and bound keys begins.
AddAbsorberListener:
Activated by the absorber button inherits BuildListener. An absorber object will then be created in the model and displayed in the gui.
BallListener:

Activated by the ball button inherits BuildListener. A ball object will then be created in the model and displayed in the gui.

FrictionListener:

Activated by the friction button. The value for friction will then be updated in the model.

GravityListener:

Activated by the gravity button. The value for gravity is updated in the model.

ClearListener:

Activated by the clear button. This will clear the board of all objects leaving the board completely empty.

ChangeButtonsListener

Activated by the add gizmo, setup, operations and back buttons. This will change the buttons displayed on the GUI

ModeListener:

Activated by the Run mode or build mode button. This will switch the view between build and run mode and vice versa.

StartStopListener:

Activated by the start or stop button. This will switch between the game running and the game stopped

LoadListener

Activated by the load menu option. Will start the process of loading a file into the board

SaveListener

Activated by the save menu option. Will start the process of saving the current board state to file

KeyListener

Checks for whether the key bound to a gizmo has been pressed if so it activates the action mapped to that key

Validation Testing Strategy

Add Gizmo:
Test Number: 1
Purpose: to check if a new gizmo is placed correctly on a clear board.
Test Inputs: To execute the test, while in build mode press the “Add Gizmo” button and then pick one of the gizmo shapes (e.g “Square”). In the end, press on the position on the board where you want to place the new gizmo.
Expected Outputs: If after following the instructions above the correct gizmo shape is placed on the board on the desired position, then the test has passed. Otherwise, if there is no gizmo placed on the board, it means that adding a gizmo does not work correctly.

Add Overlapping Gizmo
Test Number: 2
Purpose: to check if a gizmo will be added when the user selects to add it on a position of the board where another gizmo was already placed.
Test Inputs: To execute the test, while in build mode press the “Add Gizmo” button and then pick one of these options of gizmo shapes (Circle,Triangle or Square). After that, press on the position on the board where you want to place the new gizmo. Then, pick one of these options of gizmo shapes (Left Flipper or Right Flipper) to add another gizmo and select the position of the board in which you placed the previous gizmo.
Expected Outputs: If after following the instructions above an error message appears that there is already another gizmo placed at this position and the second gizmo is not added on the board on top of the first gizmo then the test has passed. Otherwise, this means that there is an issue when adding a gizmo on a position where another gizmo is already placed.

Add Gizmo Outside Build Area
Test Number: 3
Purpose: to check that a new gizmo is not added to the board when the user selects a position out of the build area.
Test Inputs: To execute the test, while in build mode press the “Add Gizmo” button and then pick one of the gizmo shapes (e.g “Square”). After that, try to click somewhere outside the build area to place the gizmo.
Expected Outputs: If after following the instructions above an error message that says that the position selected is invalid appears and no gizmo is added on the board, then the test has passed. Otherwise, this means that the test has failed and that adding a gizmo does not work correct in that case.

Rotate Gizmo
Test Number: 4
Purpose: To test the rotation of the standard gizmos at 90 degree angles.
Test Inputs: While in build mode, click on the operations button, click on rotate gizmo button to highlight it. Then click on an existing gizmo that is rotatable.
Expected Outputs: The gizmo that was clicked on will be rotated 90 degrees clockwise. If an empty space is clicked; nothing will happen. If an absorber is clicked a message will say that absorbers cannot be rotated.

Move Gizmo
Test Number: 5
Purpose: To test successfully moving a gizmo from one position to another.
Test Inputs: While in build mode, click on the operations button, click on move gizmo button to highlight it. Then click on an existing gizmo to select it. Finally click on an empty space on the grid.
Expected Outputs: The gizmo selected will change position from its initial position to the empty space selected on the grid. If the initial position is clicked instead of and empty space nothing will happen.

Move Overlapping Gizmo
Test Number: 6
Purpose: To test that moving a gizmo to an area which is already occupied won't move.
Test Inputs: While in build mode, click on the operations button, click on move gizmo button to highlight it. Then click on an existing gizmo to select it. Finally click on a space on the grid that is already occupied by a gizmo.
Expected Outputs: The gizmo selected will stay in its initial position and an error message will display saying that a gizmo already exists in the location selected.

Move Gizmo Outside Build Area
Test Number: 7
Purpose: To test that trying to move a gizmo from outwith the build area will not in fact move the gizmo.
Test Inputs: While in build mode, click on the operations button, click on move gizmo button to highlight it. Then click on an existing gizmo to select it. Finally click on an area that is not within the build area.
Expected Outputs: The gizmo selected will stay in its initial position and an error message will display saying that the area selected is not within the build area.

Connect Gizmos
Test Number: 8
Purpose: To test successfully connecting an action between two gizmos.
Test Inputs: While in build mode, click on the operations button, click on the connect gizmo button to highlight it. Then click on an existing gizmo to highlight it. Then click on another existing gizmo. Switch to run mode.
Expected Outputs: A message will appear saying that the two gizmos are connected. Nothing will change within the build area. However, in run mode when the program is run when the ball hits the first selected gizmo the second selected gizmo will perform a visible action.

Connect Gizmos to Empty Spaces
Test Number: 9
Purpose: To test that no connection should happen between a gizmo and an empty space.
Test Inputs: While in build mode, click on the operations button, click on an existing gizmo then an empty space or the other way round.

Expected Outputs: An error message will display when an empty part of the grid is clicked saying that you cannot connect an action to an empty space. In run mode Gizmoball should play without any triggering actions..

Add Absorber
Test Number: 10
Purpose: To test that an absorber is added to the board successfully.
Test Inputs: While in build mode, click on the add gizmos button, click on the add absorber button to highlight it. Then click and drag on an empty area on the board to create the absorber.
Expected Outputs: An absorber will be added to the board successfully with the correct size from where the user first clicked to where they dragged and let go of the mouse.

Add Overlapping Absorber
Test Number: 11
Purpose: To test that an absorber won't be added if it overlaps with existing gizmos.
Test Inputs: While in build mode, click on the add gizmos button, click on the add absorber button to highlight it. Then click and drag on an area on the board where gizmos already exist.
Expected Outputs: No absorber will be added to the board and an error message will appear saying that it cannot be added to that location due to overlapping with existing gizmos.

Add Absorber Outside Build Area
Test Number: 12
Purpose: To test that an absorber won't be added if any of the dragged area is outwith the build area.
Test Inputs: While in build mode, click on the add gizmos button, click on the add absorber button to highlight it. Either click a space outwith the build area and drag or click a space within and drag outside the build area.
Expected Outputs: No absorber will be added to the board and an error message will appear saying that it cannot be added to that location due to overlapping with existing borders.

Connect Keys With Gizmos
Test Number: 13
Purpose: To test that a keyboard key press is getting connected to a gizmo.
Test Inputs: While in build mode and a gizmo is already on the board, click on the "Operations" option and then "Bind Key". After that, you select one gizmo on the board and press the key that you want to connect a gizmo's action with it. Then you press the "Back" button, "Run Mode" and finally "Start".
Expected Outputs: If after following the instructions above an informative message that the gizmo has been connected appears and when the game is started the specific gizmo's action is triggered by the keypress, then the test has passed. In any other case it means that the binding of gizmo with the key was unsuccessful.

Connect Keys with Empty Spaces
Test Number: 14

Purpose: To test that a keyboard key press is not getting connected to a gizmo when the user chooses a position where no gizmo exists.
Test Inputs: While in build mode, click on the “Operations” option and then “Bind Key”. After that, you select an empty square on the board and press the key that you want to connect a gizmo’s action with it.
Expected Outputs: If after following the instructions above an error message that there is no gizmo selected appears then the test has passed. In any other case it means that the binding of a gizmo that does not really exist on the board with the key was successful.

Disconnect Gizmos
Test Number: 15
Purpose: To test successfully disconnecting an action between two gizmos.
Test Inputs: While in build mode and having at least two gizmos on the board, click on the operations button, click on the “Disconnect” gizmo button to highlight it. Then click on an existing gizmo to highlight it. Then click on another existing gizmo. Switch to run mode.
Expected Outputs: A message will appear saying that the two gizmos were disconnected. Nothing will change within the build area. However, in run mode when the program is run when the ball hits the first selected gizmo there won’t be any actions from the second selected gizmo. If after following the instructions above an informative message that the gizmos were disconnected appears, then the test has passed. Otherwise, the disconnection of the two gizmos was unsuccessful.

Disconnect Gizmos from Empty Spaces
Test Number: 16
Purpose: To test that no disconnection should happen between a gizmo and an empty space.
Test Inputs: While in build mode, click on the operations button and then press the “Disconnect” button. Afterwards, click on an existing gizmo and then an empty space or the other way round.
Expected Outputs: If the test is successful, an error message will display when an empty part of the grid is clicked saying that you cannot disconnect an action to an empty space. In run mode Gizmoball should play without any triggering actions. In any other case, this means that the test has failed and the disconnection was not successful.

Disconnect Key from Gizmo
Test Number: 17
Purpose: To test that a gizmo is being successfully disconnected from a key.
Test Inputs: While in build mode, click on the “Operations” button and then press the “Disconnect Key” button. Afterwards, click on an existing gizmo. Then you press the “Back” button, “Run Mode” and finally “Start”.
Expected Outputs: A message will appear saying that the gizmo has been disconnected. Nothing will change within the build area. However, in run mode when the program is run and the user presses the key, this should not trigger any action on the selected gizmo anymore. In any other case, this means that the test has failed and the disconnection was not successful.

Disconnect Key from Empty Spaces
Test Number: 18
Purpose: To test that no disconnection should happen between a key and an empty space.
Test Inputs: While in build mode, click on the “Operations” button and then press the “Disconnect Key” button. Afterwards, click on an empty space.
Expected Outputs: If the test is successful, an error message will display when an empty part of the grid is clicked saying that you cannot disconnect an action of a key to an empty space. In any other case, this means that the test has failed and the disconnection was not successful.

Delete Gizmo
Test Number: 19
Purpose: To test removing a gizmo from the board.
Test Inputs: While in build mode, click on the delete gizmos button to highlight it. Then click an existing gizmo on the board.
Expected Outputs: The gizmo selected will be removed from the board.

Delete Empty Slot
Test Number: 20
Purpose: To test trying to remove an empty slot on the board.
Test Inputs: While in build mode, click on the delete gizmos button to highlight it. Then click an empty area with no gizmos.
Expected Outputs: Nothing will happen on the build board and a message will display saying that there’s nothing to be deleted.

Add Ball
Test Number: 21
Purpose: To test trying to successfully add a ball to the board.
Test Inputs: While in build mode, click on the setup button. Then type the coordinates of the ball which is an empty space and the ball velocity in the provided input boxes. Then place add ball.
Expected Outputs: The ball will appear on the screen at the selected position and will move at the specified velocity when run.

Add Overlapping Ball
Test Number: 22
Purpose: To test that a ball won’t be added if there is a gizmo in the specified position.
Test Inputs: While in build mode, click on the setup button. Then type the coordinates of the ball where a gizmo already exists and the ball velocity in the provided input boxes. Then place add ball.
Expected Outputs: The ball will not appear on the screen and a message will be displayed specifying that the ball will overlap with an existing gizmo.

Add Ball Outside Build Area
Test Number: 23

Purpose: To test that the ball won't be added when the specified position is outwith the build area.
Test Inputs: While in build mode, click on the setup button. Then type the coordinates of the ball where it doesn't exist on the build board and the ball velocity in the provided input boxes. Then place add ball.
Expected Outputs: The ball will not appear on the screen and a message will be displayed specifying that the coordinates are outside the build area.

Add Ball With Null or Incorrect Inputs
Test Number: 24
Purpose: To try and add a ball with no coordinates or velocity values.
Test Inputs: While in build mode, click on the setup button. Then leave both input boxes blank or type a non valid input and click place ball.
Expected Outputs: The ball will not be added to the board and a message will be displayed that specifies that the inputs are not valid.

Modify Gravity
Test Number: 25
Purpose: To check that a valid gravity value has been given
Test Inputs: While in build mode, click on the setup button. Then type a numeric value for the gravity value field and then press the gravity button.
Expected Outputs: An informative message will appear saying that the gravity has been updated. No any other visible changes in build mode.

Modify Gravity With Null or Incorrect Inputs
Test Number: 26
Purpose: To check that the gravity value will not be updated when incorrect inputs are given from the user.
Test Inputs: While in build mode, click on the setup button. Then type a value other than a number or type nothing at all on the gravity value field and then press the gravity button.
Expected Outputs: A message will be displayed saying that an invalid value was given for the gravity and the gravity value will not be updated. No any other visible changes in build mode.

Modify Friction
Test Number: 27
Purpose: To check that a valid friction value has been given
Test Inputs: While in build mode, click on the setup button. Then type a numeric value for the friction value field and then press the friction button.
Expected Outputs: An informative message will appear saying that the friction has been updated. No any other visible changes in build mode.

Modify Friction With Null or Incorrect Inputs
Test Number: 28
Purpose: To check that the friction value will not be updated when incorrect inputs are given from the user.

Test Inputs: While in build mode, click on the setup button. Then type a value other than a number or type nothing at all on the friction value field and then press the friction button.
Expected Outputs: A message will be displayed saying that an invalid value was given for the friction and the friction value will not be updated. No any other visible changes in build mode.

Switch to Run Mode
Test Number: 29
Purpose: To test switching from build mode to run mode.
Test Inputs: While in build mode, click the run mode button.
Expected Outputs: Run mode will be displayed to the user.

Switch to Build Mode
Test Number: 30
Purpose: To test switching from run mode to build mode.
Test Inputs: While in run mode, click the build mode button.
Expected Outputs: Build mode will be displayed to the user.

Triggering an Action
Test Number: 31
Purpose: To test that actions are performed on gizmos when a key is pressed while running.
Test Inputs: While in run mode and a key is connected to a gizmo; eg spacebar to flipper. Click start and then press the connected key.
Expected Outputs: Watch the connected gizmo perform an action; eg flipper moves.

Save Game
Test Number: 32
Purpose: To test saving a file of the current game board.
Test Inputs: In any mode, click the file tab on top then click save. Name the file.
Expected Outputs: A file will be created within the filesystem with the extension .gizmo.

Save Game with Same File Name
Test Number: 33
Purpose: To test saving a file of the current game board with an already existing file name.
Test Inputs: In any mode, click the file tab on top then click save. Name the file the same as one already existing.
Expected Outputs: A message will be displayed asking the user if they want to overwrite the existing file or not.

Load Game
Test Number: 34
Purpose: To test that loading a file actually loads the saved file on the board.
Test Inputs: In any mode, click the file tab on top and then click load. Choose the .gizmo file to be loaded.
Expected Outputs: The game previously saved is loaded on the board. In any other case, it could be that the load functionality is not working as it should have.

Load Game with Invalid file
Test Number: 35
Purpose: To test that when an invalid type file is loaded then no load occurs.
Test Inputs: In any mode, click the file tab on top then click load. Choose to load a file with an extension different than .gizmo.
Expected Outputs: An error message will be displayed saying that an invalid file was tried to be loaded. Nothing more is loaded on the board.

Start Game
Test Number: 36
Purpose: To test starting the game.
Test Inputs: While in run mode, click start.
Expected Outputs: The simulation will start running.

Stop Game
Test Number: 37
Purpose: To test stopping a current game already running.
Test Inputs: While in run mode and game has already started, click stop.
Expected Outputs: The ball will stop moving and the simulation will stop at the current state.

Quit Game
Test Number: 38
Purpose: To test quitting the game.
Test Inputs: From anywhere click the file tab at the top and then click exit.
Expected Outputs: The window will close terminating the program.

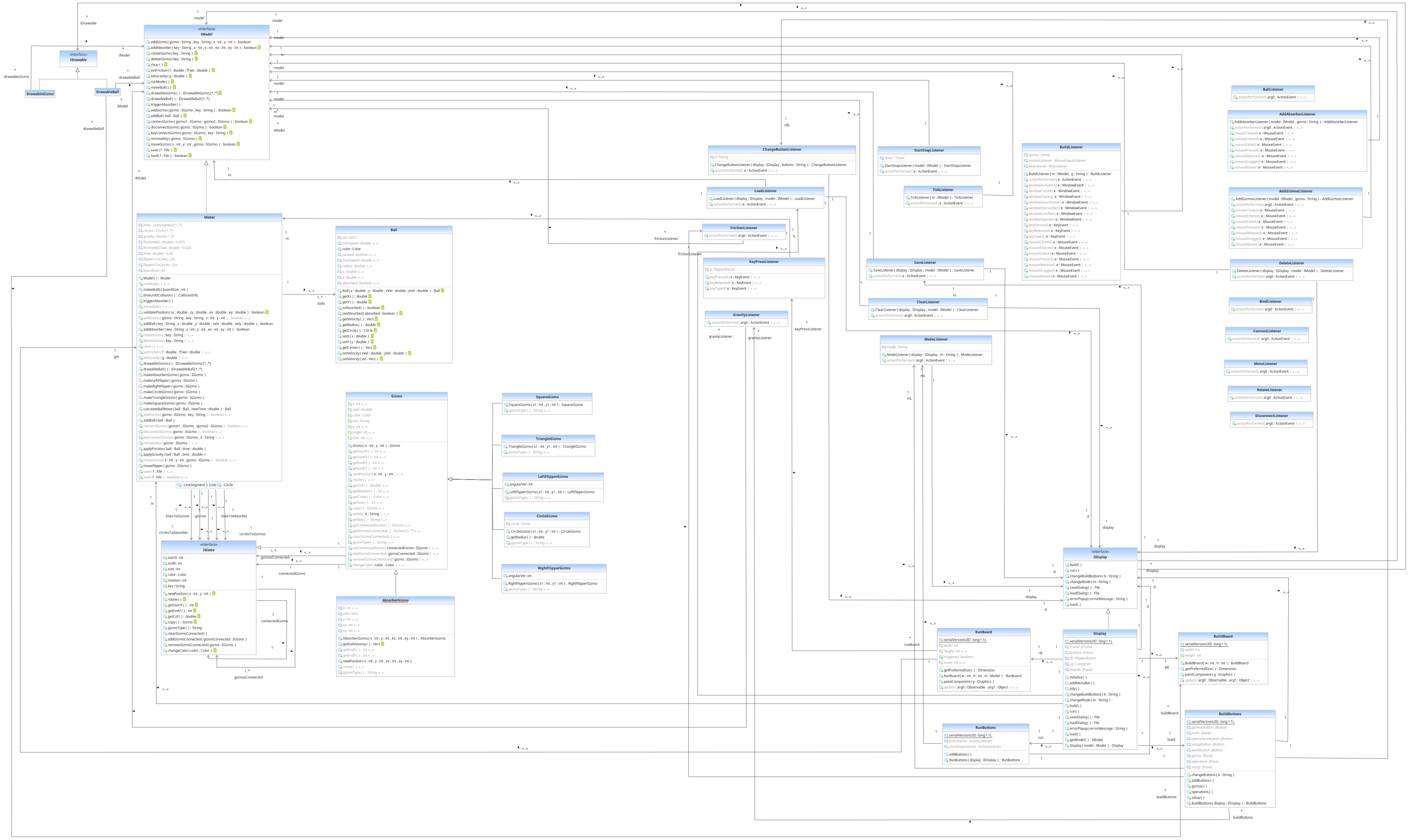
New Game
Test Number: 39
Purpose: To test starting a new game.
Test Inputs: From anywhere click the file tab at the top and then click new.
Expected Outputs: A new window will open containing a blank canvas.

Clear Board
Test Number: 40
Purpose: To test that clearing the board works correctly.
Test Inputs: While in build mode and having some gizmos on the board, click on the operations button. Then click on the clear button.
Expected Outputs: The board will be cleared from gizmos and will be empty.

JUnit Testing Strategy

Identification of JUnit Test Cases
To identify any potential JUnit test cases we will try to find the parts of the code that may cause bugs in functionality. This is why most of our tests will be written for methods within the model. More specifically, we will write test cases in which we will perform boundary and exception tests. We will identify some more tests that will show us the expected behaviour of the system's functionality is correct.

JUnit Group Approach
As a group we decided that we cover thoroughly all of the methods associated with the model except from getters and setters. We will aim for above 80% code coverage on the model as sections of the model aren't directly related to the use cases. Tests that are written will be grouped together based on the functionality of method being tested allowing the tests to be organized while making it easier to detect where the bugs in the code are. We will aim to write the JUnit tests in parallel with the creation of the methods. We won't write any more code until all of the tests written pass. As a group we will be all writing tests for the model to make sure the system's functionality is intact.


«interface» IModel
addGizmo (gizmo : String , key : String , x : int , y : int) : boolean
addAbsorber (key : String , x : int , y : int , ex : int , ey : int) : boolean
rotateGizmo (key : String)
deleteGizmo (key : String)
clear ( )
setFriction (f : double , fTwo : double)
setGravity (g : double)
runMode ( )
moveBalls ( )
drawableGizmo ( ) : IDrawableGizmo [1..*]
drawableBall ( ) : IDrawableBall [1..*]
triggerAbsorber ( )
addGizmo (gizmo : IGizmo , key : String) : boolean
addBall (ball : Ball)
connectGizmo (gizmo1 : IGizmo , gizmo2 : IGizmo) : boolean
disconnectGizmo (gizmo : IGizmo) : boolean
keyConnectGizmo (gizmo : IGizmo , key : String)
removeKey (gizmo : IGizmo)
moveGizmo (x : int , y : int , gizmo : IGizmo) : boolean
save (f : File)
load (f : File) : boolean
«interface» IDrawable
DrawableGizmo
DrawableBall
Model
Ball
Gizmo
«interface» IGizmo
AbsorberGizmo
SquareGizmo
TriangleGizmo
LeftFlipperGizmo
CircleGizmo
RightFlipperGizmo
ChangeButtonListener
LoadListener
FrictionListener
KeyPressListener
GravityListener
StartStopListener
TickListener
SaveListener
ClearListener
ModelListener
BuildListener
BallListener
AddAbsorberListener
AddGizmosListener
DeleteListener
BindListener
ConnectListener
MoveListener
RotateListener
DisconnectListener
«interface» IDisplay
build ( )
run ( )
changeBuildButtons (b : String)
changeMode (m : String)
saveDialog ( ) : File
loadDialog ( ) : File
errorPopup (errorMessage : String)
load ( )
RunBoard
Display
BuildBoard
RunButtons
BuildButtons