Ethan Behar, Christian Dummer, Raghunadham Gattu, Saurabh Gulati
9/19/2021
Professor Rawlins
P532: Group 2 Assignment 3

Architecture Description

The architecture is based off of Group1 Week2 and Group3 Week2's architecture. We extracted and combined the strengths of both to serve as the base. The architecture now starts with a BreakoutApplication class. This class handles creating the high-level objects. These include: SaveLoadManager, LayoutManager, PasuableGameEngine, GameManager, CommandInvoker, ObjectPooler, CollisionHandler2D, and Renderer. It is also responsible for handling the functionality for the Save and Load buttons. The RootSavable, CommandInvoker, ObjectPooler, MacroCommands, Commands, and DrawObject[1] all implement the Savable interface. This is the Composite Pattern in action. The SaveLoadManager triggers the RootSavable by calling save or load. Save trickles down to the Commands and DrawObjects. They return their Json representations and allow the higher objects to finish building their Json representation as each recursive call returns. Loading is similar, a JsonObject trickles down to the Commands and DrawObjects returning themselves to the higher objects. RootSavable then takes all the created objects and ensures references are reconnected between the DrawObjects and Commands. The ObjectPooler is a class that makes this possible. Any DrawObject that is created is done so by the ObjectPooler. When the ObjectPooler creates any DrawObject it assigns that object and id. This id is how references can be rebuilt when loading a saved file. The LayoutManager is another new addition that implements the Composite Pattern. The objects that implement that Layable interface are LayoutManager, LayableCanvas, Layable Pane and LayableButtons. LayoutManager is also responsible for handling the Change Layout button functionality. PasuableGameEnginer is an observable that manages the tick cycles. It is also responsible for handling the functionality for the following buttons: Resume, Reset, Pause, Undo, Undo All, Redo, Redo All, Replay, and Random Color. GameManager, facilitates the loading of DrawObjects via the ObjectPooler and ensuring the PausableGameEngine is made aware of these newly created objects. The CommandInvoker receives and manages commands. The PausableGameEngine tells the CommandInvoker when to execute commands as well as when to undo and redo commands. The CollisionHandler2D's and Renderer's responsibilities are straightforward. Respectively, they check and notify objects of collision and draw the DrawObjects.

[1] When we use the term DrawObject this can be any of the classes that extend DrawObject. They are GameObject, Text, DigitalTimer, Ball, Paddle, and Brick.