Criterion B: Design

Design Methodologies

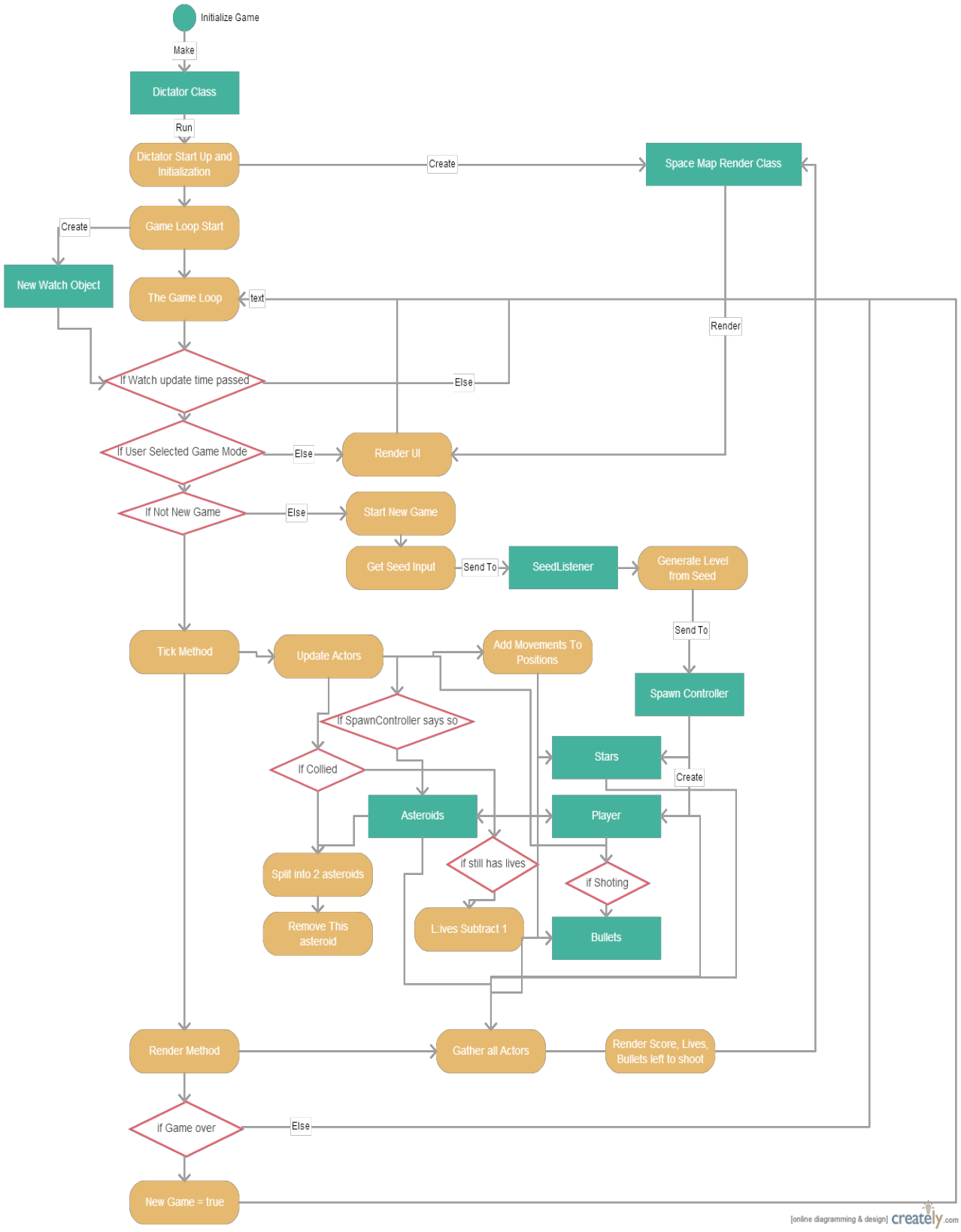
The Game loop is going to be a crucial part of this program. A game loop is basically a loop that continues to run while the game is in session. Inside the game loop are two main methods that are present in almost every game. They include the tick or update method, and the render method. The tick and update methods in my game are dedicated to all non-graphical computation. While the render method deals directly with rendering all objects that need to be rendered.

Basic Components overview

- Actor Abstract Object
  - Basic abstract which the Asteroid, Bullet, Player all inherit. Actors are any entity that has to be rendered on the board (besides stars).
- Asteroid Objects
  - Inherits from Actor, this object is a random polygon that is displayed onto the screen, that when the player touches, will cause a life to be taken away. Points are also awarded for destroying these objects with Bullets shot from the player.
- Bullet Objects
  - Inherits from Actor, this small object that travels in a straight line from the player, depending on where the player is pointing. If it comes in contact with an Asteroid, the Asteroid will split, and points will be awarded.
- Dictator Class
  - The main class of this game. The dictator class manages all other classes, and also controls the game loop. The Dictator also controls music, user input, and UI.
- Mouse Object
  - Basic object that keeps track of where the Mouse of the user is on the screen.
- Movement Abstract
  - This abstract is a velocity vector for any Actor, where upon every game tick, the movement values will be added to the position values.
- Player Object
  - Inherits from Actor, this object is the triangle like spaceship the user controls, which can move around the space with the “wasd” keys, and click for shooting. It also automatically orients itself based on the location of the mouse.
- Position Abstract
  - Similar to the Movement Abstract, this object keeps track of where every actor is currently located, and also where to be rendered.
- SeedListener Object
  - This object takes the input form the User and sends a list of commands to the SpawnController, which gives a time stamp of when each Asteroid should be spawned.
- SpaceMap Object
  - This object is for rendering. It takes all the actors from the game, and renders their location from their position object. It also manages displaying the UI and menu's.
- SpawnController Object
  - This object reads a file and appropriately adds Asteroids to the Dictator class at certain times, when the SeedListener told it too.

- Star Object
  - These are ambient twinkling dots in the background of the screen.
- Watch Object
  - This object keeps track of system time, so that the SpawnController can know what time it is, and when to spawn more asteroids. This object also manages the game-loop inside the Dictator class.

Class Structure Diagram



Testing Methods:

The Test	Method
Start Menus and other UI during game work properly	Create booleans and selects to test User input, for UI. And make sure that the R, P, and Escape keys function properly during the game.
Killing an Asteroid gives score.	Shoot an asteroid of all sizes, and make sure they all give back a score.
The player can move.	Test acceleration in all 4 directions
The player can aim and shot the blasters.	Shoot and move mouse around the screen..
Asteroids spawn in a way that require players to move or interact.	Play the game and observe rate at which asteroids hit player
The game finished eventually, leaving a game won indication.	Play the game a few times, and win.
The game is seed based.	Only one outcome for one seed
The game keeps track of bullets shot by the character.	Follow the small indicator on the bottom right.
The game keeps track of lives that player has.	Test the amount of lives that are being displayed, confirm that they go down as the player gets hit.