Report

Entities:

First I started with making my entity hierarchy which you can see in Figure 1. My Entity class together with the object class are an interface for my other entities which I use in my project. I have 2 types of bonuses, so 2 classes inherit from my class Bonus. We have 3 special types of Platforms, so 3 classes inherit from my class Platform. Player and BG_Tile are just classes that inherit from Object.

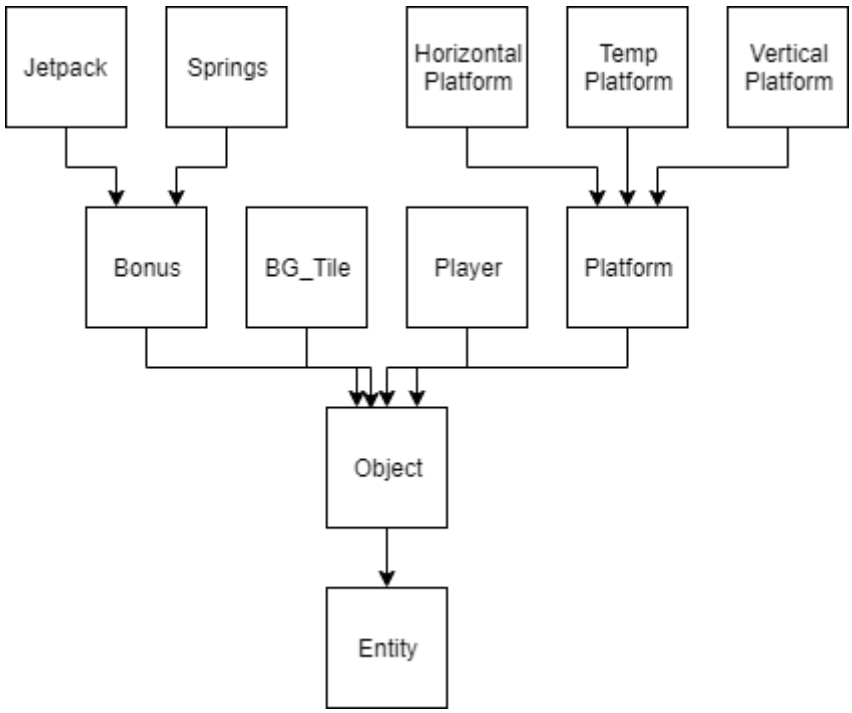

Figure 1

Observer

I have 2 classes for this pattern Observable and Observer. Object inherits from Observable and an Observable has a vector of Observers. This means that all my entities have a vector of Observers that they can notify. I made a class ObserverEvent which specifies what the Observer should update.

World

The world contains sets of entities and makes sure all entities are updated. It also contains all the observers for the specific types of entities. On top of that, it makes new entities, deletes entities and handles collisions.

SFMLEntities

The SFMLEntities update the visualization on the screen. They inherit from Observer so they handle the event that they get from the Observables.

Random

Random class generates all the random numbers. It makes sure my bonuses have a random percentage of spawning and platforms are not always on the same position.

Stopwatch

Stopwatch is a class Clock. It keeps the current tick and the previous tick and makes sure the game logic runs at the same speed on every device.

Camera

The Camera class follows the player and makes sure it doesn't go above the middle of the screen. It also transforms logic coordinates to pixel coordinates

Abstract Factory

Abstract Factory is an abstract class in the logic library. A concrete implementation inherits from it in the representation code. Then game provides a pointer to world of this concrete factory so that each entity has the correct view attached.

Model-View-Controller

In this pattern, the World is my Entity Controller. Then by using the Observer pattern we update the View when the Model state changes. In this case my View is SFMLEntities and the Model are the Entities.

Score

Score class inherits from the observer. It keeps the score for the player and updates it when the player reaches new height, jumps on platform and when it takes bonuses.

Game

The Game class contains the main game loop and draws everything on the window. It also draws the end screen when the player loses. It manages how the highest score is saved and displayed again on the window.