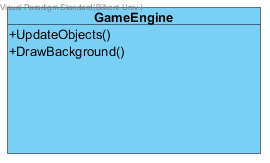
**GameEngine Class**

****

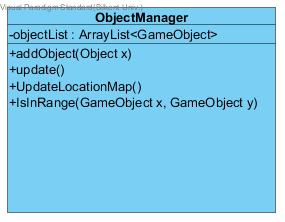
* Every game needs its engine and so does ours. This class will provide everything needed to play the game in the correct manner and it will connect the other main subclasses. This class is updates GameManager class which then takes care of the other things.
* There will be no attributes in this class, but maybe we can add some if we need them in the future.

**Methods:**

**public void UpdateObjects():** This method will update every game view in every second depending on the user choices or the automatic movements that will take place during the execution of the program.

**public void DrawBackground()**: This method is just going to draw the background based on what stage the game is. By that I mean, if the user is currently playing, is in the main menu, is in the pause menu or is restarting the game.

**ObjectManager Class**



ObjectManager is also one of the important classes of our project since every object that will be displayed throughout the whole execution time of the program will be controlled and stored the ObjectList this class will have. It will basically function as an ArrayList of GameObject’s.

**Attributes:**

**private objectList:** This attribute will be an ArrayList of GameObjects. GameObject is everything that will be displayed in the screen, the castle, the enemies, the hero or its allies.

**Methods:**

**public void addObject(GameObject x):** This method will be used to add gameObjects to the ArrayList mentioned above.

**public void update()**: This method is just going to update the list every moment and it is going to update every GameObject.

**public void updateLocationMap():** Here we will update every GameObject’s location on the map of the game, that means the frame of java application, because everything is simply a bunch of paintings that makes the view look like a game.

**public boolean isInRange(GameObject x,y):** The method will be used to check the location of each object. This Boolean method will be helpful for the update methods above.

**InputManager Class**



**Attributes:**

**Private ControlButtons:** This arrayList is enough to store the buttons that the player can use during his game. The buttons will then be provide to the GameManager class.

**Methods:**

**public ArrayList getButtons():** This method will be accessed from the GameManager class we previously mentioned in order to get the buttons.