Engine Change Recommendation Report

Thoughts on Game Engine

In general, the game engine was relatively easy to understand and was also very enjoyable to use. When starting the project there is a very large learning curve needed for the engine, and it took many attempts to truly grasp what each class and module did. However, after this period and practise with the engine, the overriding of methods and extensions of classes provided easy modification for code without the need to rewrite major functionalities. Whilst saying this, there were some recommendations that would create a more friendly coding environment for the programmers.

Recommendations

1) Changing menu

One recommendation would be to change the menu to allow for greater flexibility. Currently the menu only utilizes actions. When creating the sandbox and challenge game modes. A new menu class had to be created due to the menu class only working with actions.

The proposed solution would generalise the menu option so that the programmer can use it for other functionality instead of only using it for actions. The advantage of this would be to reduce the repetition of code. In our program another additional menu class had to be created to allow the user to input his choice of challenge or sandbox mode.

The menu class can also only have no more than 26 options. This is due to the game not being able to have letters to represent the actions. As such the solution would be to use numbers or symbols in addition.

The advantage of this would be more actions in the menu but it would result in a very unreadable game display with so many options.

2) Having two actors on the same location

Currently two actors cannot be in the same location on the map. They can only be adjacent to each other. This does not take into account the longitude of the actors. For example, in our game the Pterodactyls can fly over a stegosaurus however due to the restricted game engine it cannot do so. As such a proposed idea is to enable certain actors to be in the same location. This will be useful when taking into account actors that are in the sky.

The advantage of this would be the greater logic of the game. The disadvantage would be that if two actors are on the same spot, it will only show the character symbol of one of the actors, hiding the other's character.

3) Game Map can select all actors more efficiently
The current application when searching through all the actors on the game map, the user must go through in a loop to check each location spot and see whether there is an actor on the spot or not. This is because the game map is not iterable and cannot access the ActorLocation class which has methods for iteration in the constructor to find the collection of mapping between actors and location.

A possible solution is to use the GameMap class to implement the Iterable interface and implement the method iterator from the method and have the method return the iterator by calling the method in ActorLocation. This would allow us to iterate through every actor without needing to go through each location on the GameMap.

4) Improving Javadoc of engine code
The lack of detail in the engine code made it more difficult to understand what individual classes do. By having a more detailed Javadoc it would allow for greater readability, ultimately making it more efficient for programmers.

Positives of Engine

One of the best parts of the engine is how each class is strongly related and focused on its responsibilities. Each class has a set function in the engine and there is very little repetition of code needed due to the ability to extend and also override methods. As such this engine has very high cohesion and also relatively low couplings. There are dependencies but they are loosely coupled.

Another positive of this engine is the use of encapsulation boundaries. Most of the methods and variables are protected as opposed to being private, which allows it to inherit classes and therefore easier to modify code, reducing dependencies. As such this was a great way to reduce connascence through the use of encapsulation.

Through the use of multiple interfaces in the engine it allows for better abstraction. This is because the public interface allows the ability to add methods to the classes with modification of code references in the game.

The tick method used by both the Item and ground was a very useful method of allowing the tracking of the certain age of the things.

The items class was also very well implemented within the game. Each item had its own capabilities and such as being able to be portable or not. This allowed adding to a player's inventory much easier.