CSCI – P – 532 – Object Oriented Software Development

Team 5: Ankit Sadana, Harini Rangarajan, Jash Jhaveri and Vivek Patani

Project Report Review on Assignment 4 – Team 5

GENERAL OVERVIEW:

- **Ease of Understanding:**
  - Nomenclature of the code was **concrete**.
  - Functions were named in apt manner adhering to what they did.
  - Decoupling the elements most of the times brought down complexity of the code, making it much easier for the programmer to read the code and comprehend the overall idea.

- **General Idea:**
  - Some classes in the programme gave the idea directly while some were tightly coupled abstracting the bigger picture of what that function is contributing to the project.
  - Magic Numbers were not absolutely absent, but there were a few present.

- **Documentation:**
  - Comments were not always there to give the idea of what the function is trying to do making it more time consuming and difficult for the programmer to understand the flow of the project.

- **Implementing Patterns/Tests:**
  - Excellent use of the Observer Pattern and Command Pattern.
  - Observer pattern was thoroughly used for what it is meant updating all the related components.
  - Command Pattern was also correctly used indicating the flow and processing of each sprite, action and events.
  - The strategy pattern was only implicitly used and not explicitly decoupling elements.
  - 7 Test cases were written and were really specific hence reducing the code coverage.

- **Following Style Guidelines:**

- The team violated the style guidelines in sections. We could see that different parts were developed by different individuals and only a few followed the guidelines strictly.

- **Coding Standards:**
  - Most of the times coding standards were maintained such as all caps constants, starting a class with capitals, function naming.
  - Try Catch blocks were aptly used.
  - Encoding/Decoding in JSON made it all the more useful and helpful.

- **Architecture:**
  - The Architecture followed, at specific points tightly coupled which made development of deltas much more difficult even to make a minor change but otherwise the overall architecture was satisfactory.
  - There was no God classes which was the upside.
  - The package structure was also cogent enough, there very few or no misleading function/package names.
  - The use of access specifiers was correct.

SPECIFIC DETAILS:

Sr. No.	Issue Description	Comments/Review	Solution
1.	Delete and Detach Function	The classes were empty, which made it more difficult for one to design a delete function for the already existing architecture	Selecting the Sprite and deleting the instance along with corresponding actions and events.
2.	Absence of log	The logger for all the bugs and issues was absent causing us overhead in finding bugs	Added the log4j library and adding the logger at required places.
3.	Drag and Drop Functionality		We tried several alternatives but could not get it to work.
4.	Comments and Description	Comments were missing at places of complex functions.	Added the relevant comments.
5.	Unused and not required classes	Empty classes were present for future implementations.	Removed them as we had a different vision from that team.
6.	String Comparisons	To detect a certain action, String comparisons were used	Made it more contextual from where the data was coming making more generic.