Designing Interactive Systems II - Assignment 03

Group 12

- Vinoth Pandian Sermuga Pandian - 373445
- Arijit Gupta - 373982
- Vincentius Renaldi - 374050

3. Testing Your Understanding

1. **Design choices in your `WindowSystem` class:**

- We have created a class called `DecoratedWindow` which stores an instance of `SimpleWindow` object, and an instance of `WindowDecoration` (title bar, border etc.) object.
- In the `WindowSystem` class we maintain the list of all `DecoratedWindow` in a `LinkedList`.
- `WindowSystem` adds or removes simple windows.
- Window system receives the events from GEL and demultiplexes it find which window is being affected and passes the events to `WindowManager` for handling.

2. **Design choices in your `WindowManager` class:**

- This class adds the decoration e.g. title bar, border etc. to the `SimpleWindows` using by creating a decorated window object using `createDecoratedWindow`.
- It also has the methods to close, drag, bring a window (`DecoratedWindow`) to the front. (Expert question)

Non-obvious things in the code:

We created several additional classes called `Dimension`, `Object`, `Rectangle`, `Button` and `Label`.

- `Dimension` class keeps track of an object starting coordinate and its width and height. We also use this class to convert abstract coordinate to the window system coordinate and vice versa.
- `Object` class sets the properties (starting coordinate and size) of an object and is inherited by `Rectangle` and `Button` class.
- `Rectangle` class has color property and a method to determine whether a mouse event happening inside it or not.
- `Button` class has color and text properties. It is used to define the close button of a window.
- `Label` class is used to create the title of a window. It also has color and text properties.

Design decision for optimized code:

- We have tried to separate the functionality between layers according to the concepts (Window System Architecture) explained in the lecture.
- Late refinement is implemented as window decoration class can be replaced or the colors can be changed easily to created different designs without changing any code.
- We have coded in such a way that only one instance of `WindowManager` and `WindowSystem` can be created. This eliminates the possibility of having multiple instances of `WindowManager` and `WindowSystem` so the interaction between them in the window system architecture is seamless.