A black and red logo

Description automatically generatedA green and black background with triangles

Description automatically generated

Name: Abu Hasib Shanewaz

Student Number: 4987179

Change

Report

**Refactoring JabberPoint**

Refactoring JabberPoint code is necessary to maintain cleanliness, readability, and scalability. It improves maintainability, enhances performance, and facilitates bug fixing while promoting code reusability and adaptability to changing requirements. Below I will mention the class name and changes that I have done:

**JabberPoint**

* **Method Extraction:** Broke down the **main** method into smaller, specific methods (**loadPresentation**, **loadDemoPresentation**, **loadFromXMLFile**, **showErrorMessage**) for better modularity.
* **Centralized Error Handling:** Consolidated IOException handling into a single method to reduce duplication.
* **Reduced Redundancy:** Removed unnecessary object instantiations, such as the direct creation of **SlideViewerFrame** without using its reference.

**DemoPresentation**

* Removed the inheritance from **Accessor** as it was no longer necessary with the new interface-based design.
* Implemented the **PresentationLoader** interface.
* Refined the **load** method to create and append slides to the presentation.
* Removed the **loadFile** and **saveFile** methods since they were abstract methods from **Accessor** and now handled by interfaces.

**XMLPresentationLoader**

* Implement the **PresentationLoader** interface.
* Beaking down the **load** method into smaller, more focused methods for readability and maintainability.
* Create utility methods for creating slides and slide items from XML elements.
* Use **XMLUtils** class methods for getting element text content and parsing attributes.

**XMLPresentationSaver**

* Implemented the **PresentationSaver** interface.
* Broke down the **save** method into smaller methods to handle writing presentations, slides, and slide items.
* Introduced **escapeXml** for proper handling of special characters in XML content.

**XMLUtils**

* Improved error handling, such as throwing exceptions when elements or attributes are not found (if this aligns with the application's error-handling strategy).
* Added Javadoc comments for better documentation and readability.

**Accessor (Discussion of Refactoring)**

* Removed abstract methods **loadFile** and **saveFile** as they were no longer used, in favor of using the new **PresentationLoader** and **PresentationSaver** interfaces.
* Turned **Accessor** into a factory-like class and renamed it to **PresentationAccessorFactory**.
* **getDemoAccessor** return an instance of **PresentationLoader** directly.

**PresentationLoader and PresentationSaver Interfaces**

* Defined **load** and **save** methods respectively.

**ActionHandler**

* Refactored to use individual methods for each case in the **switch** statement of the **performAction** method. This helped in making the code cleaner and more maintainable.
* Added a try-catch block to handle any exceptions that may occur during action handling.

**KeyController**

* Introduced separate methods for each key event action to replace the **switch** statement in the **keyPressed** method. This organized the handling of different key events and improved readability.

**MenuBarBuilder**

* Broke down the menu creation into separate methods for each menu (**createFileMenu**, **createViewMenu**, and **createHelpMenu**) instead of having all the menu item creation in one method. This modularized the code and made it easier to manage.

**MenuController**

* This class was organized to only set up and integrate **MenuBarBuilder**, **ActionHandler**, and **PresentationManager**. It initializes these classes and adds the constructed menu bar to the **JFrame**.

**BitmapItem**

* **Encapsulation**: Changed the **imageName** to use **Optional<String>** to explicitly handle potential **null** values.
* **Image Loading**: Introduced a private method **loadImage()** to encapsulate the logic for loading images.
* **Exception Handling**: Improved error output for missing image files, using a clearer error message.
* **API Adjustments**: Modified **getName()** to return **Optional<String>** and adjusted usage in other classes accordingly.

**SlideItem**

* **Documentation**: Added JavaDoc comments to each method, providing clear descriptions of their purpose and usage.
* **Constructor Overload**: Provided an additional constructor for default instantiation without parameters.
* **Code Clean-up**: Ensured consistent formatting and naming conventions throughout the class for better readability.

**TextItem**

* **Separation of Concerns**: The separation of text layout and drawing responsibilities into helper classes (**TextLayoutHelper** and **TextRenderer**), reducing the complexity of the **TextItem** class.
* **Modularity**: This change aimed to make the class more modular, improving maintainability and testability by isolating specific functionalities.

**TextRenderer**

* **Type Parameter Fix**: Fix the method signature to use a generic **List<TextLayout>**, which resolved compilation errors related to generics.