JabberPoint Refactoring

Contents

[AboutBox.java 1](#_Toc131295911)

[Accessor.java 2](#_Toc131295912)

[BitmapItem.java 2](#_Toc131295913)

[DemoPresentation.java 2](#_Toc131295914)

[KeyController.java 2](#_Toc131295915)

[MenuController.java 2](#_Toc131295916)

[Presentation.java 2](#_Toc131295917)

[Slide.java 3](#_Toc131295918)

[SlideItem.java 3](#_Toc131295919)

[SlideViewerComponent.java 3](#_Toc131295920)

[Style.java 4](#_Toc131295921)

[TextItem.java 4](#_Toc131295922)

[Additions.java 4](#_Toc131295923)

[RESIT CODE CHANGES 5](#_Toc131295924)

[XML Accessor.java 5](#_Toc131295925)

[Accessor.java 6](#_Toc131295926)

[DemoPresentation.java 6](#_Toc131295927)

[JabberPoint.java 6](#_Toc131295928)

[KeyController.java 6](#_Toc131295929)

[MenuController.java 7](#_Toc131295930)

[Presentation.java 7](#_Toc131295931)

[SlideViewerComponent.java 8](#_Toc131295932)

[SlideViewerFrame.java 9](#_Toc131295933)

[Additional Changes 10](#_Toc131295934)

# AboutBox.java

Nothing out of ordinary. Small class, no need for changes

# Accessor.java

Changed to an interface due to the constructor not being used in any way. Removed “abstract” from the methods. Removed two static fields, because they weren’t used.

# BitmapItem.java

Removed second constructor. It wasn’t being used

public BitmapItem() {  
 this(0, null);  
}

# DemoPresentation.java

Unnecessary second object requirements in the constructor (due to the Accessor class extension) – *String unusedFilename*

# KeyController.java

Simplified switch case.

public void keyPressed(KeyEvent keyEvent) {  
 switch (keyEvent.getKeyCode()) {  
 case KeyEvent.VK\_PAGE\_DOWN, KeyEvent.VK\_DOWN, KeyEvent.VK\_ENTER, '+' -> presentation.nextSlide();  
 case KeyEvent.VK\_PAGE\_UP, KeyEvent.VK\_UP, '-' -> presentation.prevSlide();  
 case 'q', 'Q' -> System.*exit*(0);  
 *//Should not be reached* default -> {  
 }  
 }  
}

# MenuController.java

Constructor has a lot of code smell in it (bloaters). Made menuItem, fileMenu, viewMenu, helpMenu private variables outside constructor. Split bits of code responsible for control into smaller methods and put back to constructor.

Created separate class for all static fields.

Removed this variable because of no usage in any of the classes

private static final long serialVersionUID = 227L;

Function moveToSlide – added if statement to forbid from going to the slide number that doesn’t exist. Joption gets the current slide, then its parses it to the int and then I can prevent it from going over the ArrayList size.

# Presentation.java

Removed object type in ArrayList in method clear().

# Slide.java

public SlideItem getSlideItem(int number) {  
 return (SlideItem)items.elementAt(number);  
}

Made into this (IntelliJ signalized that it was redundant)

\/

public SlideItem getSlideItem(int number) {  
 return items.elementAt(number);  
}

Removed Vector object type from a constructor

public Slide() {  
 items = new Vector<SlideItem>();  
}

\/

public Slide() {  
 items = new Vector<>();  
}

getSideItem method can be removed since it’s not being used.

public SlideItem getSlideItem(int number) {  
 return items.elementAt(number);  
}

Doubled method name. Changed one to addSlideItem(). Changed few depending on it methods.

public void addSlideItem(SlideItem anItem) {  
 items.addElement(anItem);  
}

# SlideItem.java

Removed second constructor since it was not being used.

public SlideItem() {  
 this(0);  
}

# SlideViewerComponent.java

Removed, because of not actual use in any of the classes.

private static final long serialVersionUID = 227L;

# Style.java

Variables didn’t have the access modifiers. Added them just in case (yes color is very important component and needs to be protected) for limited access.

# TextItem.java

Removed second constructor because it was not being used.

public TextItem() {  
 this(0, EMPTYTEXT);  
}

Removed variable. It wasn’t being used

private static final String EMPTYTEXT = "No Text Given";

# Additions.java

Created *MenuControlStatic* and *SlideViewerStatic* classes for data clumps with static variables and separated them making the code cleaner.

# 

# RESIT CODE CHANGES

Code placed below is an extension to the changes above and should be treated as one complete body.

Some changes were marked in this color to make it more visible

# XML Accessor.java

Has been split to XMLAccessorSave.java and XMLAccessorLoad.java. They contain only the methods that match their name. Implemented because: “Factories good, joint classes bad”

public class XMLAccessorLoad implements AccessorLoad {

public class XMLAccessorSave implements AccessorSave {

Those changes also required to change few naming inconsistencies/dependencies in other classes like (MenuController, SlideViewerComponent, Presentation):

public void actionPerformed(ActionEvent e) {  
 AccessorSave xmlAccessorSave = new XMLAccessorSave();  
 try {  
 xmlAccessorSave.saveFile(slideViewerComponent.getPresentation(), MenuControlStatic.SAVEFILE);

public void actionPerformed(ActionEvent actionEvent) {  
 presentation.clear();  
 AccessorLoad xmlAccessorLoad = new XMLAccessorLoad();  
 try {  
 xmlAccessorLoad.loadFile(slideViewerComponent.getPresentation(, MenuControlStatic.TESTFILE);

Vector collection remains unchanged since the speed of loading it won’t be that impacted.

Changing this class also lead to splitting interface:

\/

# Accessor.java

This interface has been split to AccessorSave and AccessorLoad to suit the needs of the implementation of XMLAccesorSave and XMLAccesorLoad

public interface AccessorLoad {  
 static AccessorLoad getDemoAccessor() {  
 return new DemoPresentation();  
 }  
 void loadFile(Presentation p, String fn) throws IOException;  
}

public interface AccessorSave {  
 void saveFile(Presentation p, String fn) throws IOException;  
}

# DemoPresentation.java

Only implemented interface (name change)

class DemoPresentation implements AccessorLoad {

# JabberPoint.java

Creating new SlideViewerFrame in main to make use of new referencing.

SlideViewerFrame slideViewerFrame = new SlideViewerFrame(JABVERSION, presentation);

slideViewerFrame.getSlideViewerComponent().setSlideNumber(0);

Slight naming for the Accessors changes like this one:

AccessorLoad.*getDemoAccessor*().loadFile(presentation, "");

new XMLAccessorLoad().loadFile(presentation, argv[0]);

# KeyController.java

Added SlideViewerComponent variable.

private SlideViewerComponent slideViewerComponent;

keyPressed() method needed to have changed reference in green marked places

public void keyPressed(KeyEvent keyEvent) {  
 switch (keyEvent.getKeyCode()) {  
 case KeyEvent.VK\_PAGE\_DOWN, KeyEvent.VK\_DOWN, KeyEvent.VK\_ENTER, '+' -> slideViewerComponent.nextSlide();  
 case KeyEvent.VK\_PAGE\_UP, KeyEvent.VK\_UP, '-' -> slideViewerComponent.prevSlide();  
 case 'q', 'Q' -> System.*exit*(0);  
 *//Should not be reached* default -> {  
 }  
 }  
}

# MenuController.java

Constructor now requires a new parameter of SlideViewerComponent instead of Presentation

public MenuController(Frame frame, SlideViewerComponent slideViewerComponent) {

Classes like loadPresentation, savePresentation and moveToSlide needed reference changes due to the SlideViewerComponent partially replacing Presentation functions. Example below. There were more changes – savePresentation() encountered similar changes, as well as moveToSlide(), nextSlide() and prevSlide().

public void loadPresentation() {  
 fileMenu.add(menuItem = mkMenuItem(MenuControlStatic.OPEN));  
 menuItem.addActionListener(actionEvent -> {  
 slideViewerComponent.clear();  
 AccessorLoad xmlAccessorLoad = new XMLAccessorLoad();  
 try {  
 xmlAccessorLoad.loadFile(slideViewerComponent.getPresentation(), MenuControlStatic.TESTFILE);  
 slideViewerComponent.setSlideNumber(0);  
 } catch (IOException exc) {  
 JOptionPane.*showMessageDialog*(parent, MenuControlStatic.IOEX + exc,  
 MenuControlStatic.LOADERR, JOptionPane.ERROR\_MESSAGE);  
 }  
 parent.repaint();  
 });

# Presentation.java

Presentation uses update() method (in SetSlideNumber()) that is placed in SlideViewerComponent and requires parameters Presentation and Slide. This could be changed by moving the class or connections between the classes so that they don’t use each other and have only one way connection

Text

Description automatically generated

Text

Description automatically generated

One constructor got removed and other minimalized to this:

public Presentation() {}

Presentation has lost a lot of functions that were moved to SlideViewerComponent. One of the method that were added is that is being used later in SlideViewerComponent:

public void setCurrentSlideNumber(int currentSlideNumber){  
 this.currentSlideNumber = currentSlideNumber;  
}

Show list also got changed from null value to creating a new ArrayList.

private ArrayList<Slide> showList = new ArrayList<>();

Now in Presentation class there is no referencing towards SlideViewerComponent

# SlideViewerComponent.java

A lot of methods were transferred from Presentation class like:

oid clear() {  
 presentation.getShowList().clear();  
 setSlideNumber(-1);  
}

public void setSlideNumber(int number) {  
 this.presentation.setCurrentSlideNumber(number);  
 this.update(presentation, this.presentation.getCurrentSlide());  
}

public void nextSlide() {  
 if (presentation.getSlideNumber() < (presentation.getSize() - 1)) {  
 setSlideNumber(presentation.getSlideNumber() + 1);  
 }  
}

public void prevSlide() {  
 if (presentation.getSlideNumber() > 0) {  
 this.setSlideNumber(presentation.getSlideNumber() - 1);  
 }  
}

Those methods had their internal reference changed, to relate back to Presentation class. \

New method was also added:

public Presentation getPresentation() {  
 return this.presentation;  
}

# SlideViewerFrame.java

Added new methods in relation to big changes in SlideViewerComponent class.

public SlideViewerComponent getSlideViewerComponent() {  
 return this.slideViewerComponent;  
}

Added variable

private SlideViewerComponent slideViewerComponent;

And changed in-constructor reference

this.slideViewerComponent = new SlideViewerComponent(presentation, this);

SlideViewerFrame was also used in JabberPoint Class.

# Additional Changes

1. Classes ending with …Static contain now variables separated from their original class for decluttering.
2. Listeners/actionEvents/Events were changes to the lambda expressions
3. Vectors are unchanged intentionally
4. Some small obsolete items or parameters while using built-in Java libraries were removed