NOVA
NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

Software Engineering

1ª phase – 2022/2023

Written by:

Guilherme Poças 60236

Samuel Pires 60718

Leticia Silva 60221

João Agostinho 57538

Joana Tomás 60152

Index

Introduction

In this report, our team is reviewing the code of the GanttProject, with the intent of finding some of the design patterns and code smells discussed in class. Each member was responsible for finding three patterns and three smells, and for double checking the findings of another member.

Code analysis

Code Patterns

Guilherme Poças 60236

1. Factory method pattern

```
public interface ActiveActionProvider {
  /** Provides the active action, which might depend on external
influences */
  public abstract AbstractAction getActiveAction();
}
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/action/ActiveActionProvider.java

This interface is used to create an Action object, but hides the creation code on the actual class, making the creation process independent of the concrete class.

Review:
The pattern and it's explanation seem correct. -Leticia Silva 60221

2. Facade pattern

```
class UIFacadeImpl extends ProgressProvider implements UIFacade
```

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/UIFacadeImpl.java

This class allows other parts of the program to access functionalities and methods of other classes responsible for the UI, wrapping them in a subsystem, and making it easier for the user to use them.

Review:
Pattern well identified and explained code pattern - Samuel Pires 60718

3. Template method pattern

```
public abstract class ImporterBase implements Importer
public class ImporterFromGanttFile extends ImporterBase
public class ImporterFromTxtFile extends ImporterBase
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/importer/ImporterBase.java

The two importer classes import from different types of files, so have different responsibilities, but a similar function. Because of this, they defer some similar methods to the ImporterBase abstract class, making the code cleaner and avoiding repeated code.

Review:
The Template method pattern is well identified. - Joana Tomás 60152

Samuel Pires 60718

1. Command Pattern

```
public class ZoomActionSet {
 private final ZoomInAction myZoomIn;

 private final ZoomOutAction myZoomOut;

public ZoomActionSet(ZoomManager zoomManager) {
    myZoomIn = new ZoomInAction(zoomManager);
    myZoomIn.putValue(Action.SHORT_DESCRIPTION, null);
    myZoomOut = new ZoomOutAction(zoomManager);
    myZoomOut.putValue(Action.SHORT_DESCRIPTION, null);
  }

 public GPAction getZoomInAction() {
    return myZoomIn;
  }

public GPAction getZoomOutAction() {
    return myZoomOut;
  }
}
```

Location:
ganttproject\src\main\java\net\sourceforge\ganttproject\action\zoom\ZoomActionSet.java

The ZoomActionSet class is used as a Command Pattern in this program. This class manages the concrete commands zoom in (represented by the ZoomInAction class) and zoom out (represented by the ZoomOutAction class).

Review:
Both the identification and justification of the pattern make sense - Guilherme Poças 60236

2. Singleton Pattern

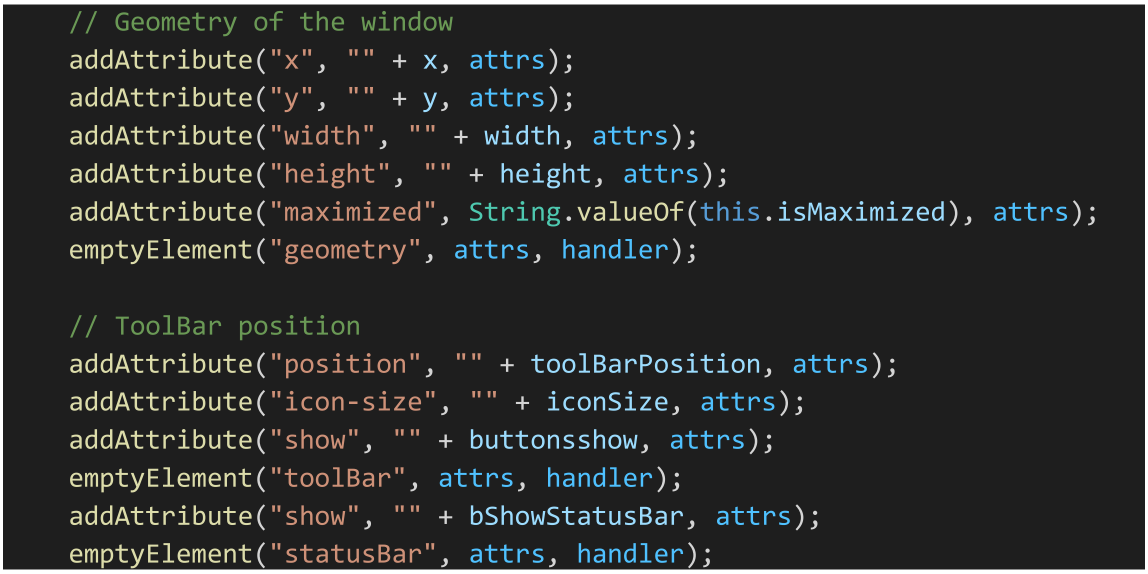
```
private static final GanttLanguage ganttLanguage = new GanttLanguage();
private GanttLanguage() {…}
public static GanttLanguage getInstance() {return ganttLanguage;}
```

Location:
ganttproject\src\main\java\net\sourceforge\ganttproject\language\GanttLanguage.java

The GanttLanguage class only has one instance created at all times, providing a way to access it through the getInstance() method. It creates the instance at initialization and has a private contructor, making it impossible to create another instance.

Review:
Correctly recognized pattern with a good explanation - João Agostinho 57538

3. Facade Pattern



```
    // Geometry of the window
    addAttribute("x", "" + x, attrs);
    addAttribute("y", "" + y, attrs);
    addAttribute("width", "" + width, attrs);
    addAttribute("height", "" + height, attrs);
    addAttribute("maximized", String.valueOf(this.isMaximized), attrs);
    emptyElement("geometry", attrs, handler);

    // ToolBar position
    addAttribute("position", "" + toolBarPosition, attrs);
    addAttribute("icon-size", "" + iconSize, attrs);
    addAttribute("show", "" + buttonsshow, attrs);
    emptyElement("toolBar", attrs, handler);
    addAttribute("show", "" + bShowStatusBar, attrs);
    emptyElement("statusBar", attrs, handler);
```

[...]

Location:
ganttproject\src\main\java\net\sourceforge\ganttproject\GanttOptions.java

The GanttOptions class is used as a Facade Pattern in this program. This class acts as a middleman between the program and the options' classes, hiding these and their complexity.

Review:
This pattern is well identified and the justification is correct - Joana Tomás 60152

Leticia Silva 60221

1. Proxy Pattern

```
public class ReadOnlyProxyDocument implements Document {

  private final Document myDelegate;

  public ReadOnlyProxyDocument(Document delegate) {
    myDelegate = delegate;
  }

  @Override
  public String getFileName() {
    return myDelegate.getFileName();
  }

  @Override
  public boolean canRead() {
    return myDelegate.canRead();
  }
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/document/ReadOnlyProxyDocument.java

This proxy class is something that acts as a simplified version of the original object, providing controlled access of a functionality.

Review:
The identified pattern is correct - Guilherme Poças 60236

2. Template Pattern

```
abstract class SearchServiceBase<SR extends SearchResult<SO>, SO>
implements SearchService<SR, SO>
```

```
public class ResourceSearchService extends
SearchServiceBase<ResourceSearchService.MySearchResult, HumanResource>
```

```
public class TaskSearchService extends
SearchServiceBase<TaskSearchService.MySearchResult, Task>
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/search/ResourceSearchService.java

The two classes differ on the responsibilities but have very similar functionality and order of operations, so the abstract class SearchServiceBase makes the code understandable.

Review:
Well identified and explained code pattern - Samuel Pires 60718

3. Factory Pattern

```
public interface ParserFactory {
  GPParser newParser();

  GPSaver newSaver();
}
```

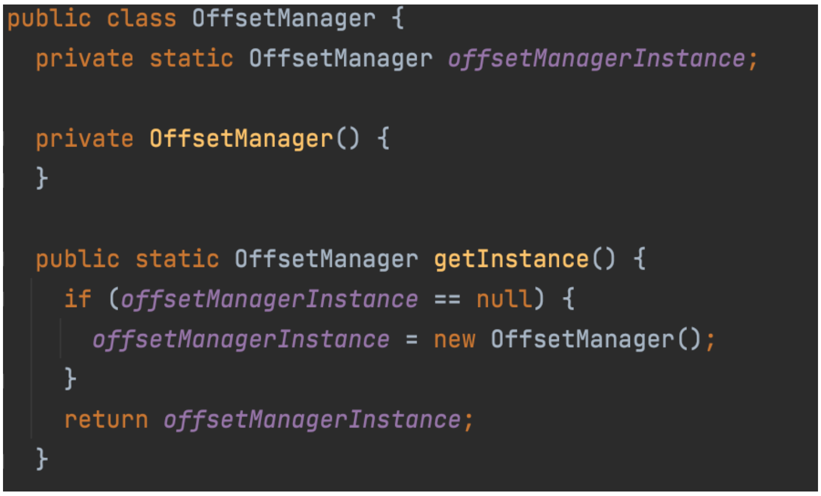
Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/parser/ParserFactory.java

The interface ParserFactory hides the creation code of the objects GPParser and GPSaver, so the creation process does not depend on concrete classes.

Review:
This pattern is well identified - Joana Tomás 60152

João Agostinho 57538

1. Singleton Pattern



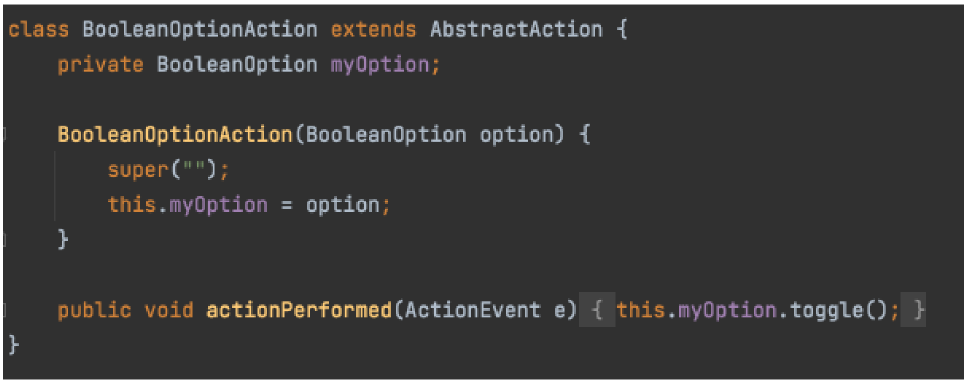
```java
public class OffsetManager {
  private static OffsetManager offsetManagerInstance;

  private OffsetManager() {
  }

  public static OffsetManager getInstance() {
    if (offsetManagerInstance == null) {
      offsetManagerInstance = new OffsetManager();
    }
    return offsetManagerInstance;
  }
```

Location:
biz.ganttproject.core/src/main/java/biz/ganttproject/core/chart/grid/OffsetManager.java

This implementation makes the default constructor private (OffsetManager()) and has a public static method (getInstance()) that acts as the constructor by calling the private constructor to instantiate the class (OffsetManager).

Review:
Pattern well identified and explained - Samuel Pires 60718

2. Command Pattern



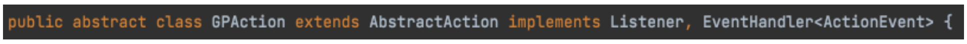
```java
class BooleanOptionAction extends AbstractAction {
    private BooleanOption myOption;

    BooleanOptionAction(BooleanOption option) {
        super("");
        this.myOption = option;
    }

    public void actionPerformed(ActionEvent e) { this.myOption.toggle(); }
}
```

Location:
ganttproject/build/classes/java/main/net/sourceforge/ganttproject/gui/options/BooleanOptionAction.class

This class is a command pattern in this program as it is used to toggle a BooleanOption when an action is performed.

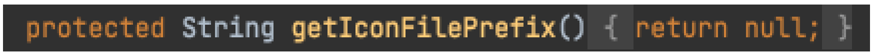
Review:
Pattern well identified, but could specify which object is the invoker. -Leticia Silva 60221

3. Template method pattern



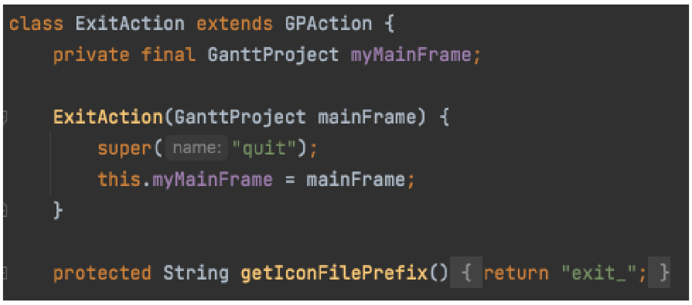
```java
public abstract class GPAction extends AbstractAction implements Listener, EventHandler<ActionEvent> {
```

Location:
ganttproject/build/classes/java/main/net/sourceforge/ganttproject/action/GPAction.class



```java
protected String getIconFilePrefix() { return null; }
```

This method is used as a template for other classses such as ExitAction.



```java
class ExitAction extends GPAction {
    private final GanttProject myMainFrame;

    ExitAction(GanttProject mainFrame) {
        super( name: "quit");
        this.myMainFrame = mainFrame;
    }

    protected String getIconFilePrefix() { return "exit_"; }
```

Location:
ganttproject/build/classes/java/main/net/sourceforge/ganttproject/action/project/ExitAction.class

Review:
The pattern is well identified, but the template for the other action classes is the GPAction class itself, not the getIconFilePrefix method - Guilherme Poças 60236

Joana Tomás 60152

1. Visitor Pattern

```
public abstract class TaskVisitor {
  public String visit(TaskManager taskManager) throws Exception
```

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/export/TaskVisitor.java

This class is an example of a visitor pattern because the original object is now passed to one of the visitor's methods as an argument, providing the method access to all necessary data contained within the object.

Review:
This pattern is well identified. - Leticia Silva 60221

2. Chain of Responsibility Pattern

```
public abstract class AbstractTagHandler implements TagHandler
public class TaskDisplayColumnsTagHandler extends AbstractTagHandler
public class RoleTagHandler  extends AbstractTagHandler
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/parser/AbstractTagHandler.java

This pattern lets us to pass requests along a chain of handlers. Upon receiving a request, each handler decides either to process the request or to pass it to the next handler in the chain.

Review:
Pattern is rightly pointed and the rationalization is logical - João Agostinho 57538

3. Facade Pattern

```
public class TimelineFacadeImpl implements MouseInteraction.TimelineFacade
```

Location:
ganttproject\src\main\java\net\sourceforge\ganttproject\chart\mouse\ChangeTaskProgressInteraction.java

This class is responsible for the timeline of the program and encapsulates multiple classes in order to allow outside classes to use their methods, simplifying the code.

Review:
Pattern's justification is weirdly worded - Guilherme Poças 60236

Code Smells

Guilherme Poças 60236

1. Dead Code

```
private final Action[] myDelegates;
```

```
myDelegates = delegates;
```

Location: ganttproject\src\main\java\net\sourceforge\ganttproject\action\ArtefactAction.java

The variable myDelegates stores an array of Action type objects, but it is never read from, only assigned values, so it serves no purpose and only clutters the code.

Solution: delete the variable.

Review:
Code Smell and it's explanation are correct and the solution is proper. -Leticia Silva 60221

2. Speculative generality

```
public static List<MutableTreeTableNode>
breadthFirstSearch(MutableTreeTableNode rootNode) {
  final List<MutableTreeTableNode> result = Lists.newArrayList();
  breadthFirstSearch(rootNode, new
Predicate<Pair<MutableTreeTableNode,MutableTreeTableNode>>() {
    public boolean apply(Pair<MutableTreeTableNode, MutableTreeTableNode>
parent_child) {
      result.add(parent_child.second());
      return true;
    }
  });
  return result;
}
```

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/TreeUtil.java

This method for Tree breadth first searching is never used, so it’s not currently needed, and it doesn’t help the code.

Solution: if there is no use for the method, delete it.

Review:
Code Smell well identified code smell with correct and understanding explanation and solution.
- Samuel Pires 60718

3. Data Class

```
public class Pair<F, S>
private final F myFirst;
private final S mySecond;
```

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/util/collect/Pair.java

This class serves only to save two values of different types, and only has two getter methods, and one creation method, so it has no real functionality or necessity.

Solution: Instead of using this class, utilize two different variables, or if they're the same type, an array.

Review:
Code Smell is correct and the solution is appropriate for solving the problem. - Joana Tomás 60152

Samuel Pires 60718

1. Dead Code

```
public interface TaskInfo {

}
```

Location: ganttproject\src\main\java\net\sourceforge\ganttproject\task\TaskInfo.java

The TaskIngo class is completely empty, with no variables, methods or any purpose. It's not used anywhere in the program.

Solution: Delete the class

Review:
Code Smell, Dead Code, is correct and the solution is apropriated. -Leticia Silva 60221

2. Refused Request

```
  @Override
  public void setVisibleTasks(List<Task> visibleTasks) {
    // TODO Auto-generated method stub
  }
```

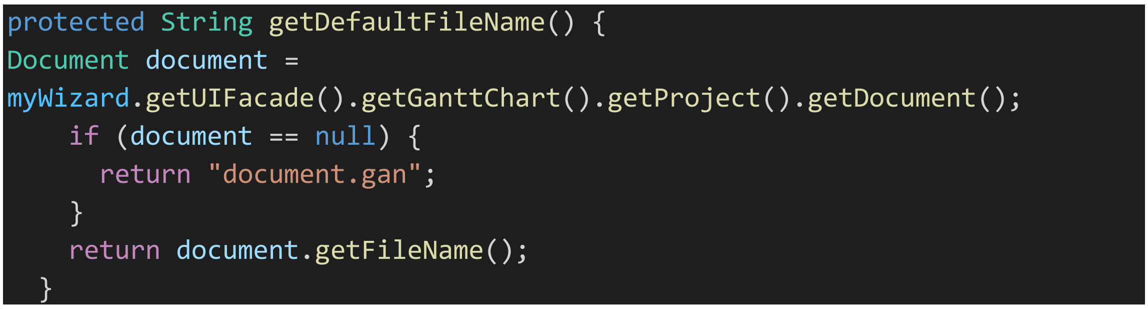
Location:
ganttproject\src\main\java\net\sourceforge\ganttproject\chart\ChartModelResource.java

The ChartModelResource class inherits the method setVisibleTasks from the ChartModel interface which it implements but does not implement it.

Solution: Define this method, not in the interface, but in the other only subclass that implements it (GantModelImpl).

Review:
The smell is adequately identified and justified - Guilherme Poças 60236

3. Message Chain



```
protected String getDefaultFileName() {
Document document =
myWizard.getUIFacade().getGanttChart().getProject().getDocument();
    if (document == null) {
      return "document.gan";
    }
    return document.getFileName();
  }
```

Location:
ganttproject\src\main\java\net\sourceforge\ganttproject\gui\FileChooserPageBase.java

This method contains a considerably long message chain, calling methods that return an object and repeating this process for that object multiple times in the same line.

Solution: Create a method in this class the returns the wanted resource. In case that is not possible, make the function calls inside of each object in order not to have a message chain in the same line.

Review:
Well identified code smell, with a good explanation and solution - João Agostinho 57538

Leticia Silva 60221

1. Long method

```
private Document createDocument(String path, File relativePathRoot, String
user, String pass) {
  assert path != null;
  path = path.trim();
  String lowerPath = path.toLowerCase();
  if (lowerPath.startsWith("http://") || lowerPath.startsWith("https://")) {
    try {
      if (user == null && pass == null) {
        WebDavServerDescriptor server = myWebDavStorage.findServer(path);
        if (server != null) {
          user = server.getUsername();
          pass = server.getPassword();
        }
      }
      return new HttpDocument(path, user, pass,
myWebDavStorage.getProxyOption());
    } catch (IOException e) {
      GPLogger.log(e);
      return null;
    } catch (WebDavException e) {
      GPLogger.log(e);
      return null;
    }
  } else if (lowerPath.startsWith("cloud://")) {
    var patchedUrl = DocumentKt.asDocumentUrl(lowerPath);
    if (patchedUrl.component2().equals("cloud")) {
      return new GPCloudDocument(
        null,

DocumentUri.LocalDocument.createPath(patchedUrl.component1().getPath()).getPa
rent().getFileName(),
        patchedUrl.component1().getHost(),

DocumentUri.LocalDocument.createPath(patchedUrl.component1().getPath()).getFi
leName(),
        null
      );
    }
  } else if (lowerPath.startsWith("ftp:")) {
    return new FtpDocument(path, myFtpUserOption, myFtpPasswordOption);
  } else if (!lowerPath.startsWith("file://") && path.contains("://")) {
    // Generate error for unknown protocol
    throw new RuntimeException("Unknown protocol: " + path.substring(0,
path.indexOf("://")));
  }
  File file = new File(path);
  if (file.toPath().isAbsolute()) {
    return new FileDocument(file);
  }
  File relativeFile = new File(relativePathRoot, path);
  return new FileDocument(relativeFile);
}
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/document/DocumentCreator.java

This method is too long which makes it more difficult to comprehend, is doing more than one thing and produces more code smells.

Solution: Create new methods with fragments of the code.

Review:
The identified smell and it's solution are correct - Guilherme Poças 60236

2. Dead Code

```
public class PreferenceServiceImpl implements IPreferencesService {

  @Override
  public IStatus applyPreferences(IExportedPreferences preferences)
throws CoreException {
    // TODO Auto-generated method stub
    return null;
  }

  @Override
  public void applyPreferences(IEclipsePreferences node,
IPreferenceFilter[] filters) throws CoreException {
    // TODO Auto-generated method stub

  }
```

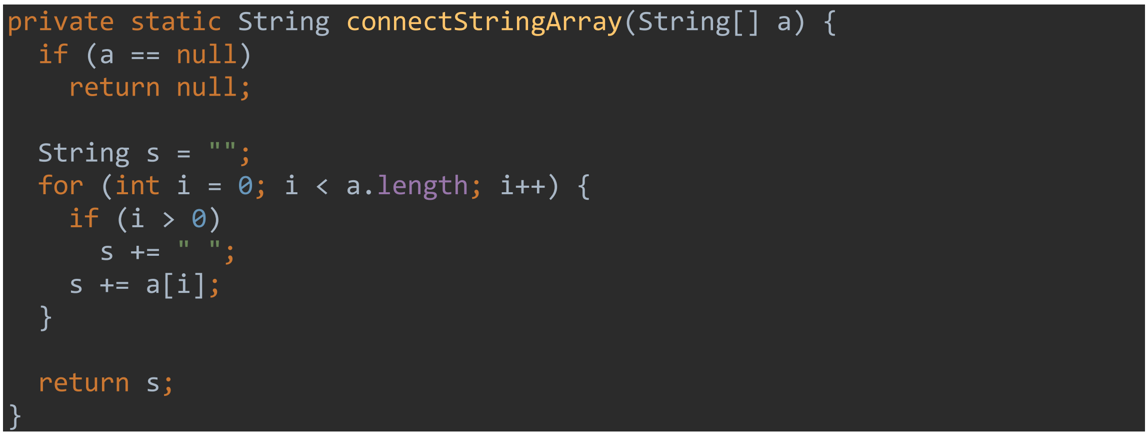
Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/PreferenceServiceImpl.java

This class, PreferenceServiceUmpl , isn’t used, has only the methods inherited from the interfaceIPreferencesService but they aren’t implemented and doesn’t have variables.

Solution: Delete this class.

Review:
Well identified code smell with satisfactory explanation and solution - Samuel Pires 60718

3. Comments



```
private static String connectStringArray(String[] a) {
  if (a == null)
    return null;

  String s = "";
  for (int i = 0; i < a.length; i++) {
    if (i > 0)
      s += " ";
    s += a[i];
  }

  return s;
}
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/util/BrowserControl.java

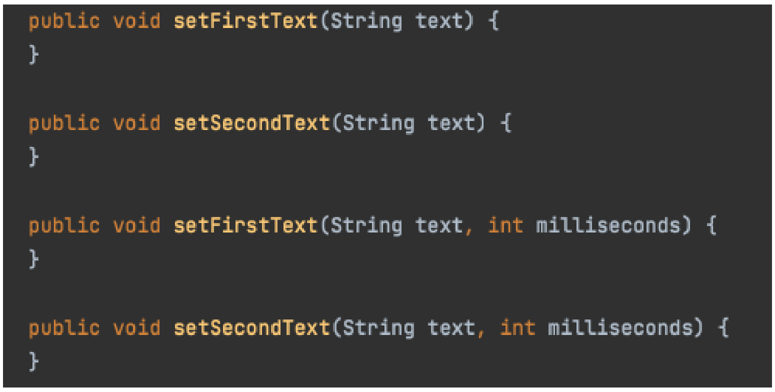
This method has no comments, and so it is hard to understand what the code is doing.

Solution: Add comments to the method to clarify what the code should do.

Review:
Code smell, and solution are correct - Joana Tomás 60152

João Agostinho 57538

1. Dead Code



```
public void setFirstText(String text) {
}

public void setSecondText(String text) {
}

public void setFirstText(String text, int milliseconds) {
}

public void setSecondText(String text, int milliseconds) {
}
```

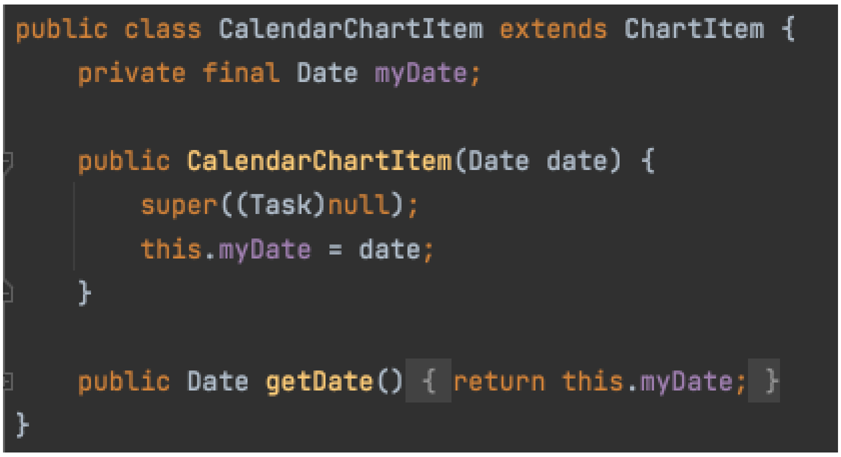
Location:
ganttproject/build/classes/java/main/net/sourceforge/ganttproject/gui/GanttStatusBar.java

These methods are completely empty and have no use for the program.

Solution: Removing these methods.

Review:
Code Smell well spotted code with good explanation and solution - Samuel Pires 60718

2. Data Class



```
public class CalendarChartItem extends ChartItem {
    private final Date myDate;

    public CalendarChartItem(Date date) {
        super((Task)null);
        this.myDate = date;
    }

    public Date getDate() { return this.myDate; }
}
```

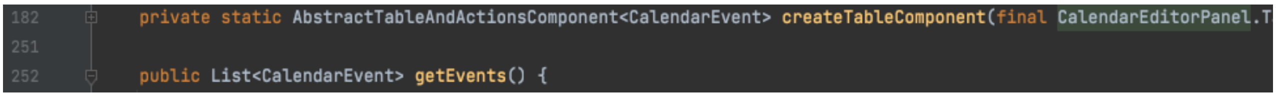
Location:
ganttproject/build/classes/java/main/net/sourceforge/ganttproject/chart/item/CalendarChartItem.class

This class is only used to store a date and only has one method to return it.

Solution: Instead of a class, use an array to store this data.

Review:
Code Smell is well identified, and the solution seems correct. -Leticia Silva 60221

3. Long Method



```
private static AbstractTableAndActionsComponent<CalendarEvent> createTableComponent(final CalendarEditorPanel.T

public List<CalendarEvent> getEvents() {
```

Location:
ganttproject/build/classes/java/main/net/sourceforge/ganttproject/calendar/CalendarEditorPanel.class

This method is very extensive (more than 60 lines), making it hard to read and comprehend.

Solution: Creating private auxiliary methods to distribute its function.

Review:
Code smell is well identified and the solution is also correct. - Joana Tomás 60152

Joana Tomás 60152

1. Duplicated code

```
public ResourceNewAction(HumanResourceManager hrManager, ProjectDatabase
projectDatabase, RoleManager roleManager, TaskManager taskManager, UIFacade
uiFacade) {
  super("resource.new", hrManager);
  myUIFacade = uiFacade;
  myRoleManager = roleManager;
  myTaskManager = taskManager;
  myProjectDatabase = projectDatabase;
}

private ResourceNewAction(HumanResourceManager hrManager, ProjectDatabase
projectDatabase, RoleManager roleManager, TaskManager taskManager, UIFacade
uiFacade, IconSize size) {
  super("resource.new", hrManager, null, size);
  myUIFacade = uiFacade;
  myRoleManager = roleManager;
  myTaskManager = taskManager;
  myProjectDatabase = projectDatabase;
```

Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/action/resource/ResourceNewAction.java

Both code fragments look identical, making the code more difficult to change and update.

Solution: The constructor that has more arguments calls the other and sends the parameters they have in common, avoiding repeated code.

Review:
Smell solution doesn't make much sense because they're both constructors - Guilherme Poças 60236

2. Large class

```
public class OptionsPageBuilder
```

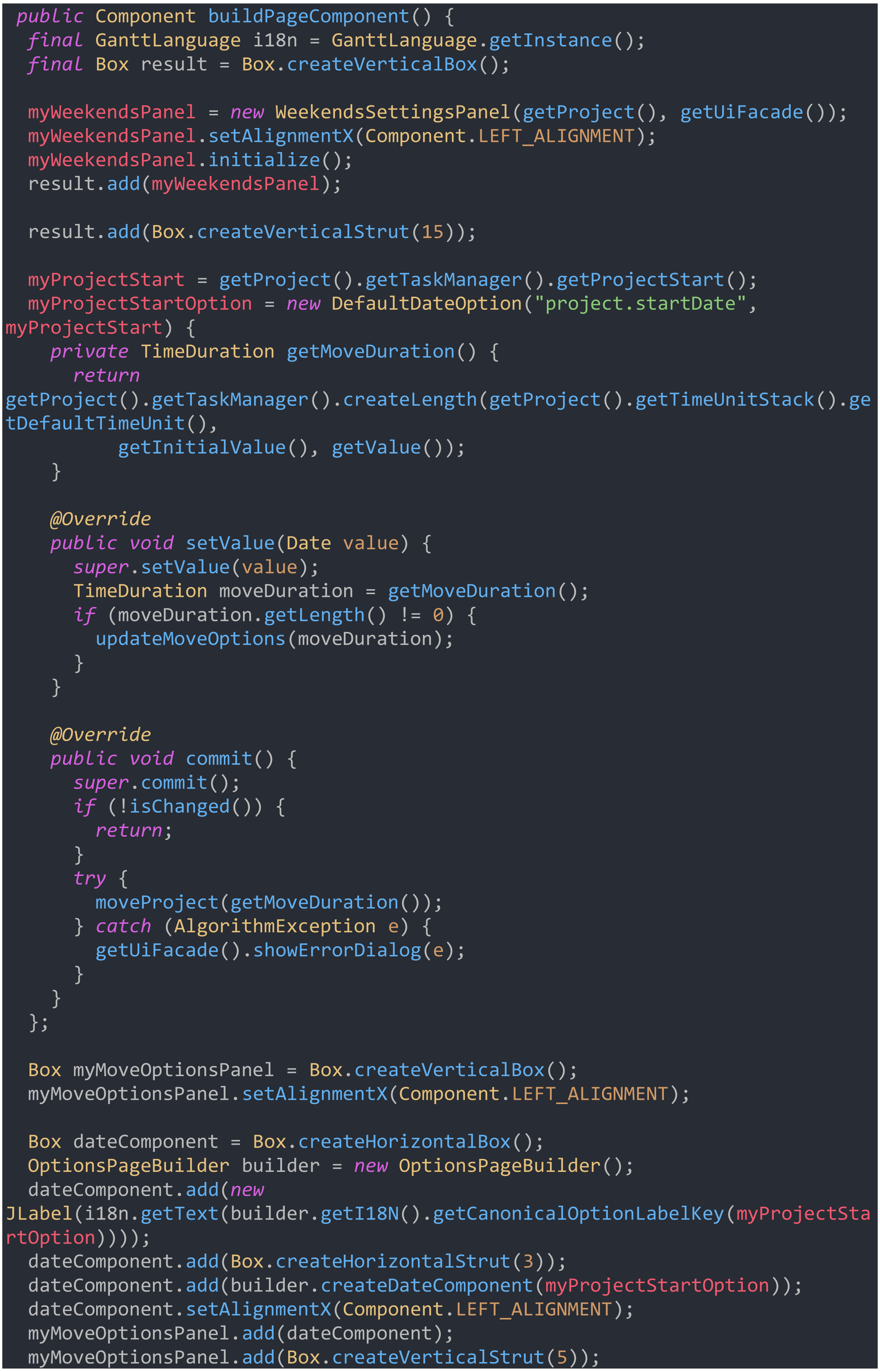
Location:
ganttproject/src/main/java/net/sourceforge/ganttproject/gui/options/OptionsPageBuilder.java

This class contains many lines of code. This occurs because this class has more responsibilities than it should.

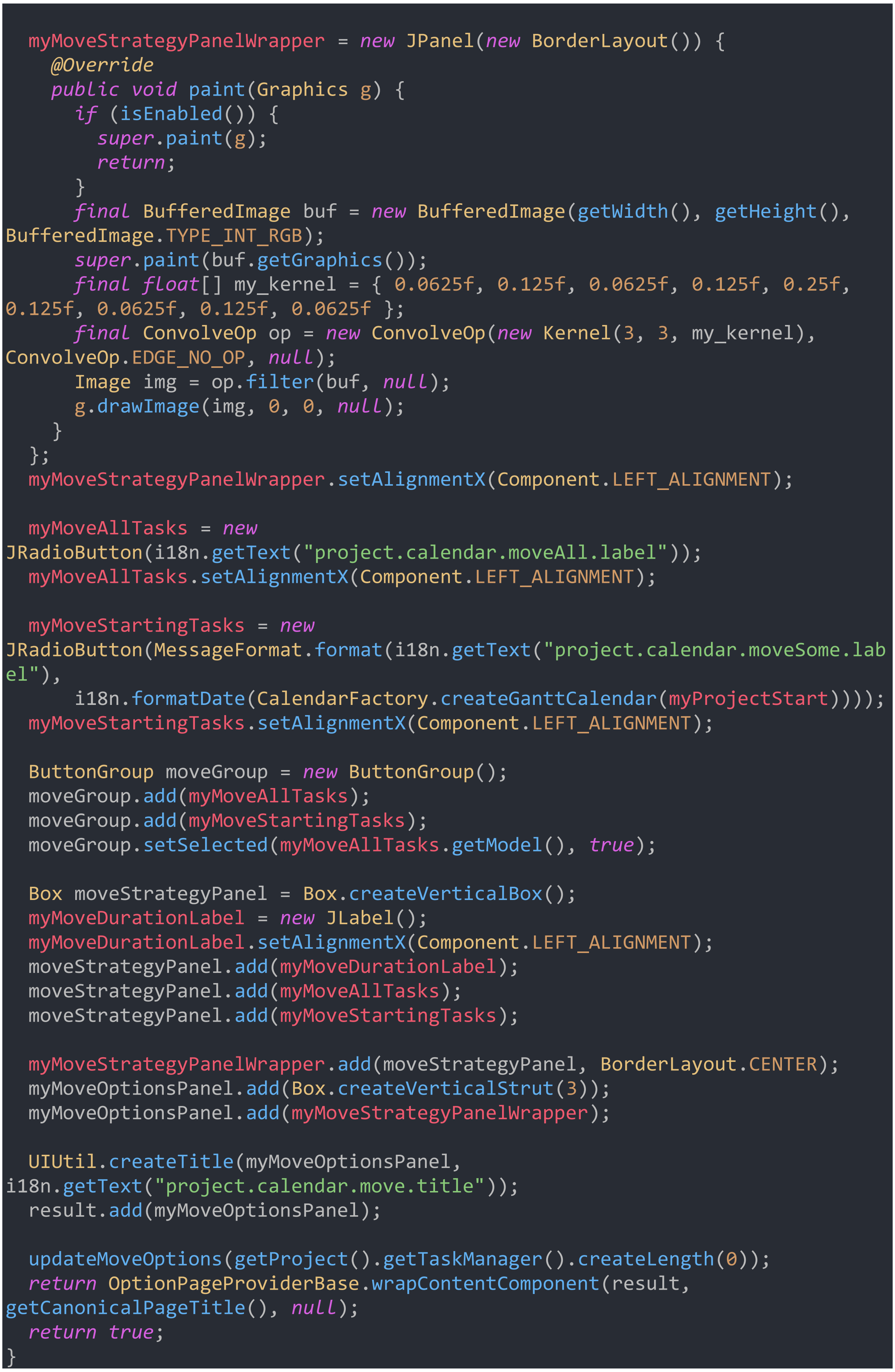
Solution: Some of the responsibilities in this class can be moved to another class, making it simpler.

Review:
This smell isn't suitable because the class isn't that large and mostly made of unimplemented methods - João Agostinho 57538

3. Long Method



```
 public Component buildPageComponent() {
  final GanttLanguage i18n = GanttLanguage.getInstance();
  final Box result = Box.createVerticalBox();

  myWeekendsPanel = new WeekendsSettingsPanel(getProject(), getUiFacade());
  myWeekendsPanel.setAlignmentX(Component.LEFT_ALIGNMENT);
  myWeekendsPanel.initialize();
  result.add(myWeekendsPanel);

  result.add(Box.createVerticalStrut(15));

  myProjectStart = getProject().getTaskManager().getProjectStart();
  myProjectStartOption = new DefaultDateOption("project.startDate",
myProjectStart) {
    private TimeDuration getMoveDuration() {
      return
getProject().getTaskManager().createLength(getProject().getTimeUnitStack().ge
tDefaultTimeUnit(),
          getInitialValue(), getValue());
    }

    @Override
    public void setValue(Date value) {
      super.setValue(value);
      TimeDuration moveDuration = getMoveDuration();
      if (moveDuration.getLength() != 0) {
        updateMoveOptions(moveDuration);
      }
    }

    @Override
    public void commit() {
      super.commit();
      if (!isChanged()) {
        return;
      }
      try {
        moveProject(getMoveDuration());
      } catch (AlgorithmException e) {
        getUiFacade().showErrorDialog(e);
      }
    }
  };

  Box myMoveOptionsPanel = Box.createVerticalBox();
  myMoveOptionsPanel.setAlignmentX(Component.LEFT_ALIGNMENT);

  Box dateComponent = Box.createHorizontalBox();
  OptionsPageBuilder builder = new OptionsPageBuilder();
  dateComponent.add(new
JLabel(i18n.getText(builder.getI18N().getCanonicalOptionLabelKey(myProjectSta
rtOption))));
  dateComponent.add(Box.createHorizontalStrut(3));
  dateComponent.add(builder.createDateComponent(myProjectStartOption));
  dateComponent.setAlignmentX(Component.LEFT_ALIGNMENT);
  myMoveOptionsPanel.add(dateComponent);
  myMoveOptionsPanel.add(Box.createVerticalStrut(5));
```



```
  myMoveStrategyPanelWrapper = new JPanel(new BorderLayout()) {
    @Override
    public void paint(Graphics g) {
      if (isEnabled()) {
        super.paint(g);
        return;
      }
      final BufferedImage buf = new BufferedImage(getWidth(), getHeight(),
BufferedImage.TYPE_INT_RGB);
      super.paint(buf.getGraphics());
      final float[] my_kernel = { 0.0625f, 0.125f, 0.0625f, 0.125f, 0.25f,
0.125f, 0.0625f, 0.125f, 0.0625f };
      final ConvolveOp op = new ConvolveOp(new Kernel(3, 3, my_kernel),
ConvolveOp.EDGE_NO_OP, null);
      Image img = op.filter(buf, null);
      g.drawImage(img, 0, 0, null);
    }
  };
  myMoveStrategyPanelWrapper.setAlignmentX(Component.LEFT_ALIGNMENT);

  myMoveAllTasks = new
JRadioButton(i18n.getText("project.calendar.moveAll.label"));
  myMoveAllTasks.setAlignmentX(Component.LEFT_ALIGNMENT);

  myMoveStartingTasks = new
JRadioButton(MessageFormat.format(i18n.getText("project.calendar.moveSome.lab
el"),
      i18n.formatDate(CalendarFactory.createGanttCalendar(myProjectStart))));
  myMoveStartingTasks.setAlignmentX(Component.LEFT_ALIGNMENT);

  ButtonGroup moveGroup = new ButtonGroup();
  moveGroup.add(myMoveAllTasks);
  moveGroup.add(myMoveStartingTasks);
  moveGroup.setSelected(myMoveAllTasks.getModel(), true);

  Box moveStrategyPanel = Box.createVerticalBox();
  myMoveDurationLabel = new JLabel();
  myMoveDurationLabel.setAlignmentX(Component.LEFT_ALIGNMENT);
  moveStrategyPanel.add(myMoveDurationLabel);
  moveStrategyPanel.add(myMoveAllTasks);
  moveStrategyPanel.add(myMoveStartingTasks);

  myMoveStrategyPanelWrapper.add(moveStrategyPanel, BorderLayout.CENTER);
  myMoveOptionsPanel.add(Box.createVerticalStrut(3));
  myMoveOptionsPanel.add(myMoveStrategyPanelWrapper);

  UIUtil.createTitle(myMoveOptionsPanel,
i18n.getText("project.calendar.move.title"));
  result.add(myMoveOptionsPanel);

  updateMoveOptions(getProject().getTaskManager().createLength(0));
  return OptionPageProviderBase.wrapContentComponent(result,
getCanonicalPageTitle(), null);
  return true;
}
```

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/gui/options/ProjectCalendarOptionPageProvider.java

This method contains too many lines of code. This probably means that is more complex than it needs to be.

Solution: Make auxiliary methods to simplify the code.

Review:
This smell doesn't seem right as the method is not long enough to be a code smell - João Agostinho 57538