

The S Team

1st Part



Carolina Simonet, 59748

Filipe Santo, 64859

Jaime Russo, 60062

Margarida Carvalho, 60437

Rui Capareira, 57046

Code Smells

Carolina Simonet, 59748

1. Message Chains

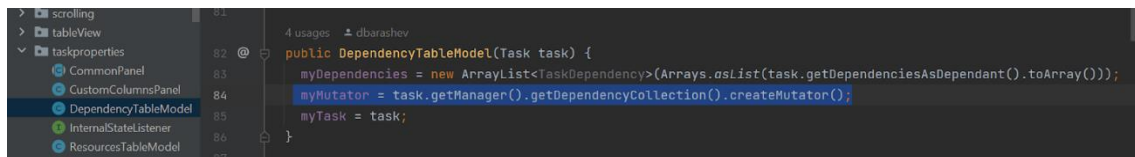
Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/gui/taskproperties/DependencyTableModel.java

Explanation:

We are getting an object, and again getting another object back and again calling another method.

Reviewed by Margarida Carvalho.



```
42 4 usages dbarashev
43 public DependencyTableModel(Task task) {
44     myDependencies = new ArrayList<TaskDependency>(Arrays.asList(task.getDependenciesAsDependant().toArray()));
45     myMutator = task.getManager().getDependencyCollection().createMutator();
46     myTask = task;
47 }
```

2. Dead Code

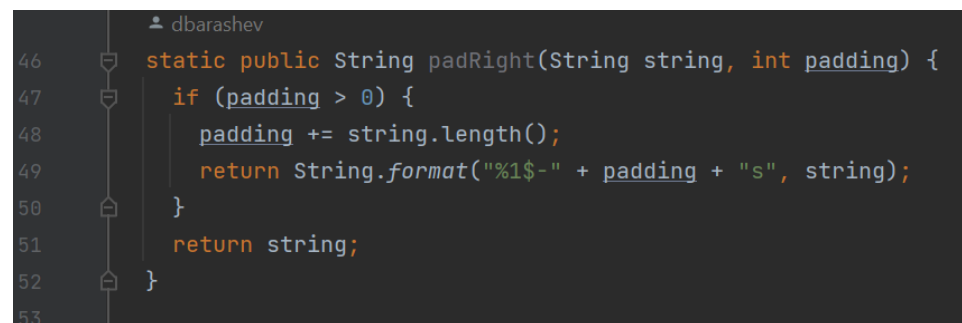
Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/util/StringUtils.java

Explanation:

The code is not used for now, and we already have a similar method padLeft. So we could delete this method.

Reviewed by Filipe Santo.



```
46 dbarashev
47 static public String padRight(String string, int padding) {
48     if (padding > 0) {
49         padding += string.length();
50         return String.format("%1${-}" + padding + "s", string);
51     }
52     return string;
53 }
```

3. Long Method

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/task/TaskProperties.java

Explanation:

This method is very long with 79 lines and it's confusing to understand it because it has a lot of if statements and no comments. The method is responsible for doing more things than it actually should.

Reviewed by Rui Capareira.

```
286 @ public static void parseDependency(String depSpec, final Task successor, Function<Integer, Task> taskIndex,
287                                     Map<Integer, Supplier<TaskDependency>> out) {
288     final TaskManager taskMgr = successor.getManager();
289     int posDash = depSpec.indexOf('-');
290     String maybeId = posDash < 0 ? depSpec : depSpec.substring(0, posDash);
291     final Integer predecessorId;
292     try {
293         predecessorId = Integer.parseInt(maybeId);
294     } catch (NumberFormatException e) {
295         throw new IllegalArgumentException(String.format("%s is not a number", maybeId));
296     }
297     if (posDash < 0) {
298         final Task predecessor = taskIndex.apply(predecessorId);
299         if (predecessor == null) {
300             throw new IllegalArgumentException(String.format("Can't find task with ID=%s", depSpec));
301         }
302         out.put(predecessorId, () -> {
303             if (taskMgr.getDependencyCollection().canCreateDependency(successor, predecessor)) {
304                 return taskMgr.getDependencyCollection().createDependency(successor, predecessor);
305             }
306         });
307     }
308     throw new TaskDependencyException(String.format(
```

Filipe Santo, 64859

1. Dead Code

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/wizard/AbstractFileChooser.java

Explanation:

This function doesn't do nothing, is a dead code, it should be deleted.

Reviewed by Jaime Russo.

```
private void reportMalformedUrl(Exception e) {
}
```

2. Comments

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/client/RssFeedChecker.java

Explanation:

Instead of having the comments explaining the following complex expression, the expression should go to another function and make it name self explanatory. The comments should be deleted.

Reviewed by Carolina Simonet.

```
public void run() {
    Runnable command = null;
    CheckOption checkOption = CheckOption.valueOf(myCheckRssOption.getValue());
    if (CheckOption.NO == checkOption) {
        if (myOptionsVersion == null) {
            // We used opt-in before GP 2.7; now we use opt-out, and we suggest to
            // subscribe once again to those who previously chosen not to.
            checkOption = CheckOption.UNDEFINED;
            myCheckRssOption.setSelectedValue(checkOption);
            markLastCheck();
        } else {
            NotificationChannel.RSS.setDefaultNotification(myRssProposalNotification);
        }
        return;
    }
    Date lastCheck = myLastCheckOption.getValue();
    if (lastCheck == null) {
        // It is the first time we run, just mark it. We want to suggest
        // subscribing to updates only to
        // those who runs GP at least twice.
        markLastCheck();
    } else if (wasToday(lastCheck)) {
        // It is not the first run of GP but it was last run today and RSS
        // proposal has not been shown yet.
        // Add it to RSS button but don't promote it, wait until tomorrow.
        if (CheckOption.UNDEFINED == checkOption) {
            NotificationChannel.RSS.setDefaultNotification(myRssProposalNotification);
        }
    } else {
        // So it is not the first time and even not the first day we start GP.
        // If no decision about subscribing, let's proactively suggest it,
        // otherwise
        // run check RSS.
        if (CheckOption.UNDEFINED == checkOption) {
```

3. Long Method

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/GanttOptions.java

Explanation:

This method is very long with 204 lines. It should be divided into multiple methods and if possible follow the SOLID principles because it accumulates a lot of responsibility making it very hard to understand.

Reviewed by Margarida Carvalho.

```

@Override
public void startElement(String namespaceURI, String sName, // simple name
                        String qName, // qualified name
                        Attributes attrs) throws SAXException {

    if ("ganttproject-options".equals(qName)) {
        myVersion = attrs.getValue("version");
        return;
    }
    if ("configuration".equals(qName) || "instance".equals(qName)) {
        myPluginOptionsHandler = new PluginOptionsHandler(myPluginPreferencesRootNode);
    }
    if (myPluginOptionsHandler != null) {
        myPluginOptionsHandler.startElement(namespaceURI, sName, qName, attrs);
        return;
    }
    int r = 0, g = 0, b = 0;

    if ("option".equals(qName)) {
        String id = attrs.getValue("id");
        GPOption option;
        if (id.equals(csvOptions.getBomOption().getID())) {
            option = csvOptions.getBomOption();
        }
    }
}

```

Jaime Russo, 60062

1. Long Method

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/GanttProject

Explanation: This method has 60 lines, which we can identify as a long method code smell. To prevent this smell we could create additional auxiliar methods to help.

Reviewed by Rui Capareira.

```

470  /**
471   * Create the button on toolbar
472   */
473  @ 1 usage 2 dbarashev +4
474  private FXToolBar createToolBar() {
475      FXToolBarBuilder builder = new FXToolBarBuilder();
476      builder.addButton(myProjectMenu.getOpenProjectAction().asToolBarAction())
477          .addButton(myProjectMenu.getSaveProjectAction().asToolBarAction())
478          .addWhitespace();
479
480      final ArtefactAction newAction;
481      {
482          final GPAction taskNewAction = myTaskActions.getCreateAction().asToolBarAction();
483          final GPAction resourceNewAction = getResourceTree().getNewAction().asToolBarAction();
484          newAction = new ArtefactNewAction() -> getTabs().getSelectedIndex() == UIFacade.GANTT_INDEX ? taskNewAction : resourceNewAction, new A
485          builder.addButton(taskNewAction).addButton(resourceNewAction);
486      }
487
488      final ArtefactAction deleteAction;
489      {
490          final GPAction taskDeleteAction = myTaskActions.getDeleteAction();
491          final GPAction resourceDeleteAction = getResourceTree().getDeleteAction().asToolBarAction();
492          deleteAction = new ArtefactDeleteAction() -> getTabs().getSelectedIndex() == UIFacade.GANTT_INDEX ? taskDeleteAction : resourceDeleteA
493      }
494      builder.setArtefactActions(newAction, deleteAction);
495
496      final ArtefactAction propertiesAction;
497      {
498          final GPAction taskPropertiesAction = myTaskActions.getPropertiesAction().asToolBarAction();
499          final GPAction resourcePropertiesAction = getResourceTree().getPropertiesAction().asToolBarAction();
500          propertiesAction = new TaskResourcePropertiesAction(
501              taskPropertiesAction, resourcePropertiesAction,
502              () -> getTabs().getSelectedIndex(),
503              () -> getTaskSelectionManager().getSelectedTasks());
504      }
505
506      UIUtil.registerActions(getRootPane(), recursive: false, newAction, propertiesAction, deleteAction);
507      UIUtil.registerActions(myGanttChartTabContent.getComponent(), recursive: true, newAction, propertiesAction, deleteAction);
508      UIUtil.registerActions(myResourceChartTabContent.getComponent(), recursive: true, newAction, propertiesAction, deleteAction);
509      getTabs().getModel().addChangeListener(e -> {

```

2. No Comments

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/GanttProject

Explanation: This method don't have any comments at all, only pure code, it may be confusing even for the coder.

Reviewed by Margarida Carvalho.

```

470  /**
471   * Create the button on toolbar
472   */
473  @usage 1 dbarashev +4
474  private FXToolBar createToolBar() {
475      FXToolBarBuilder builder = new FXToolBarBuilder();
476      builder.addButton(myProjectMenu.getOpenProjectAction().asToolBarAction())
477          .addButton(myProjectMenu.getSaveProjectAction().asToolBarAction())
478          .addWhitespace();
479
480      final ArtefactAction newAction;
481      {
482          final GPAction taskNewAction = myTaskActions.getCreateAction().asToolBarAction();
483          final GPAction resourceNewAction = getResourceTree().getNewAction().asToolBarAction();
484          newAction = new ArtefactNewAction() -> getTabs().getSelectedIndex() == UIFacade.GANTT_INDEX ? taskNewAction : resourceNewAction, new A
485      }
486
487      final ArtefactAction deleteAction;
488      {
489          final GPAction taskDeleteAction = myTaskActions.getDeleteAction();
490          final GPAction resourceDeleteAction = getResourceTree().getDeleteAction().asToolBarAction();
491          deleteAction = new ArtefactDeleteAction() -> getTabs().getSelectedIndex() == UIFacade.GANTT_INDEX ? taskDeleteAction : resourceDeleteA
492      }
493      builder.setArtefactActions(newAction, deleteAction);
494
495      final ArtefactAction propertiesAction;
496      {
497          final GPAction taskPropertiesAction = myTaskActions.getPropertiesAction().asToolBarAction();
498          final GPAction resourcePropertiesAction = getResourceTree().getPropertiesAction().asToolBarAction();
499          propertiesAction = new TaskResourcePropertiesAction(
500              taskPropertiesAction, resourcePropertiesAction,
501              () -> getTabs().getSelectedIndex(),
502              () -> getTaskSelectionManager().getSelectedTasks());
503      }
504
505      UIUtil.registerActions(getRootPane(), recursive: false, newAction, propertiesAction, deleteAction);
506      UIUtil.registerActions(myGanttChartTabContent.getComponent(), recursive: true, newAction, propertiesAction, deleteAction);
507      UIUtil.registerActions(myResourceChartTabContent.getComponent(), recursive: true, newAction, propertiesAction, deleteAction);
508      getTabs().getModel().addChangeListener(e -> {

```

3. Repeated Code

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/undo/UndoableEditImpl

Explanation: These methods have only 1 different line from one to the other, we could only have 1 method with a condition to implement 1 line or the other.

Reviewed by Filipe Santo.

```

76      @Override
77      public void redo() throws CannotRedoException {
78          try {
79              restoreDocument(myDocumentAfter);
80              if (projectDatabaseTxn != null) {
81                  try {
82                      projectDatabaseTxn.redo();
83                  } catch (ProjectDatabaseException e) {
84                      GPLLogger.log(e);
85                  }
86              }
87          } catch (DocumentException | IOException e) {
88              undoRedoExceptionHandler(e);
89          }
90      }
91
92      @Override
93      public void undo() throws CannotUndoException {
94          try {
95              restoreDocument(myDocumentBefore);
96              if (projectDatabaseTxn != null) {
97                  try {
98                      projectDatabaseTxn.undo();
99                  } catch (ProjectDatabaseException e) {
100                      GPLLogger.log(e);
101                  }
102              }
103          } catch (DocumentException | IOException e) {
104              undoRedoExceptionHandler(e);
105          }
106      }
107

```

Margarida Carvalho, 60437

1. Dead Code

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/util/StringUtils.java

Explanation:

The method is not used. To fix this code smell, the method should be deleted.

Reviewed by Carolina Simonet.


```

54      /** @return a comma separated list showing the names of the given objects */
55      @ public static String getDisplayNames(Object[] objects) {
56          if (objects.length == 1) {
57              return objects[0].toString();
58          }
59          StringBuffer result = new StringBuffer();
60          for (int i = 0; i < objects.length; i++) {
61              result.append(objects[i].toString());
62              if (i < objects.length - 1) {
63                  result.append(", ");
64              }
65          }
66          return result.toString();
67      }

```

2. Data Class

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/util/CustomColumnn.java

Explanation:

This class doesn't contain real functionality. Besides the equals and hashCode methods, it only has getter and setter methods.

Reviewed by Rui Capareira.

```

53      public void setId(String newId) { id = newId; }
54
55      @Override
56      public Object getDefaultValue() { return defaultValue; }
57
58      @Override
59      public void setDefaultValue(Object defaultValue) { this.defaultValue = defaultValue; }
60
61      @Override
62      public void setDefaultValueAsString(String value) {
63          CustomPropertyDefinition stub = PropertyTypeEncoder.INSTANCE.decodeTypeAndDefaultValue(
64              getTypeAsString(), value);
65          defaultValue = stub.getDefaultValue();
66      }
67
68      @Override
69      public Map<String, String> getAttributes() { return myAttributes; }
70
71      @Override
72      public String getName() { return name; }
73
74      @Override
75      public void setName(String name) {
76          String oldName = this.name;

```

3. Feature Envy

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/task/TaskActivitiesAlgorithm.java

Explanation:

This method manipulates the data of another class. It could have been made in the TaskActivity class.

Reviewed by Jaime Russo.

```
21 import ...
26
27
28 public class TaskActivitiesAlgorithm {
29     2 usages
30     private final GPCalendarCalc myCalendar;
31
32     1 usage
33     public TaskActivitiesAlgorithm(GPCalendarCalc calendar) { myCalendar = calendar; }
34
35     1 usage
36     @ public void recalculateActivities(Task task, List<TaskActivity> output, Date startDate, Date endDate) {
37         output.clear();
38         List<GPCalendarActivity> activities = myCalendar.getActivities(startDate, endDate);
39         for (int i = 0; i < activities.size(); i++) {
40             GPCalendarActivity activity = activities.get(i);
41             TaskActivity nextTaskActivity;
42             if (activity.isWorkingTime()) {
43                 nextTaskActivity = new TaskActivityImpl(task, activity.getStart(), activity.getEnd());
44             } else if (i > 0 && i + 1 < activities.size()) {
45                 nextTaskActivity = new TaskActivityImpl(task, activity.getStart(), activity.getEnd(), intensity: 0);
46             } else {
47                 continue;
48             }
49             output.add(nextTaskActivity);
50         }
51     }
52 }
```

Rui Capareira, 57046

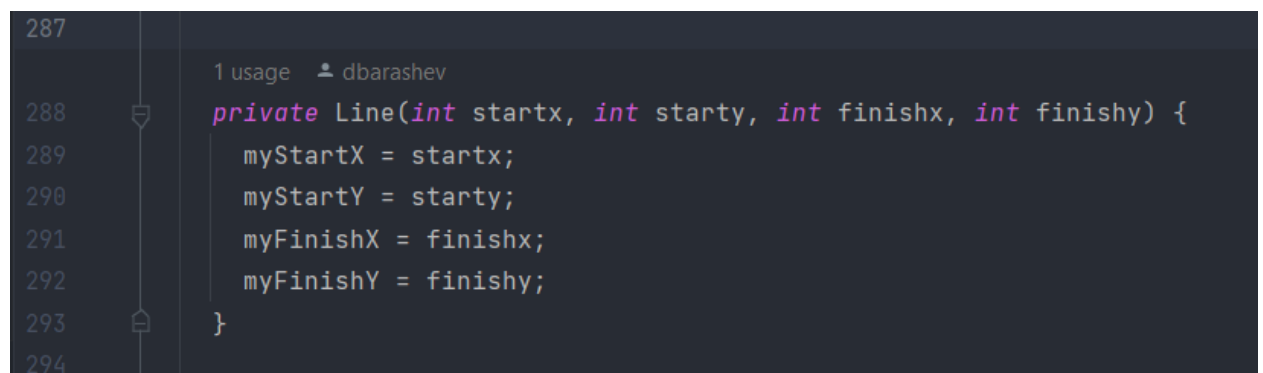
1. Data Clump

Location:

ganttproject\biz.ganttproject.core\src\main\java\biz\ganttproject\core\chart\canvas\Canvas.java

Explanation: This method could be rewritten by passing two 2DPoint objects as arguments, instead it uses a group of variables passed as a clump.

Reviewed by Filipe Santo.



```
287  
1 usage  dbarashev  
288  private Line(int startx, int starty, int finishx, int finishy) {  
289      myStartX = startx;  
290      myStartY = starty;  
291      myFinishX = finishx;  
292      myFinishY = finishy;  
293  }  
294
```

2. Switch Statements

Location:

ganttproject\biz.ganttproject.core\src\main\java\biz\ganttproject\core\calendar\GPCalendarBase.java

Explanation:

Switch statement with conditionals checking on type instead of reducing conditionals down to a design that uses polymorphism. Two switch cases evaluated without any code being executed.

Reviewed by Jaime Russo.

```

2 usages  dbarashev
100  protected Date doFindClosest(Date time, DateFrameable framer, MoveDirection direction, DayType dayType, Date limit) {
101      Date nextUnitStart = direction == GPCalendarCalc.MoveDirection.FORWARD ? framer.adjustRight(time)
102          : framer.jumpLeft(time);
103      int nextUnitMask = getDayMask(nextUnitStart);
104      switch (dayType) {
105          case WORKING:
106              if ((nextUnitMask & DayMask.WORKING) == DayMask.WORKING) {
107                  return nextUnitStart;
108              }
109              break;
110          case WEEKEND:
111          case HOLIDAY:
112          case NON_WORKING:
113              if ((nextUnitMask & DayMask.WORKING) == 0) {
114                  return nextUnitStart;
115              }
116              break;
117          default:
118              assert false : "Should not be here";
119      }

```

3. Comments

Location:

ganttproject\ganttproject\src\main\java\net\sourceforge\ganttproject\chart\ChartModellImpl.java

Explanation:

This section of code shows several lines of comments that take on a reminder nature, showing that something needs to be done or this code needs to be updated later.

Reviewed by Carolina Simonet.

```
153 // java.awt.Rectangle nextAwtRectangle = new java.awt.Rectangle(
154 // nextRectangle.myLeftX, nextRectangle.myTopY,
155 // nextRectangle.myWidth, nextRectangle.myHeight);
156 // if (result == null) {
157 // result = nextAwtRectangle;
158 // } else {
159 // result = result.union(nextAwtRectangle);
160 // }
161 // }
162 // }
163 // return result;
164 // }
165
166 // GraphicPrimitiveContainer.Rectangle[] getTaskActivityRectangles(Task task)
167 // {
168 // List<Rectangle> result = new ArrayList<Rectangle>();
169 // TaskActivity[] activities = task.getActivities();
170 // for (int i = 0; i < activities.length; i++) {
171 // GraphicPrimitiveContainer.Rectangle nextRectangle = myTaskRenderImpl
172 // .getPrimitive(activities[i]);
173 // if (nextRectangle!=null) {
174 // result.add(nextRectangle);
175 // }
176 // }
177 // return result.toArray(new GraphicPrimitiveContainer.Rectangle[0]);
178 // }
179
180 // dbarashev +1
180 public List<Task> getVisibleTasks() {
181     return myVisibleTasks == null ? Collections.<Task> emptyList() : myVisibleTasks;
182 }
```

Design Patterns

Carolina Simonet, 59748

1. Facade Pattern

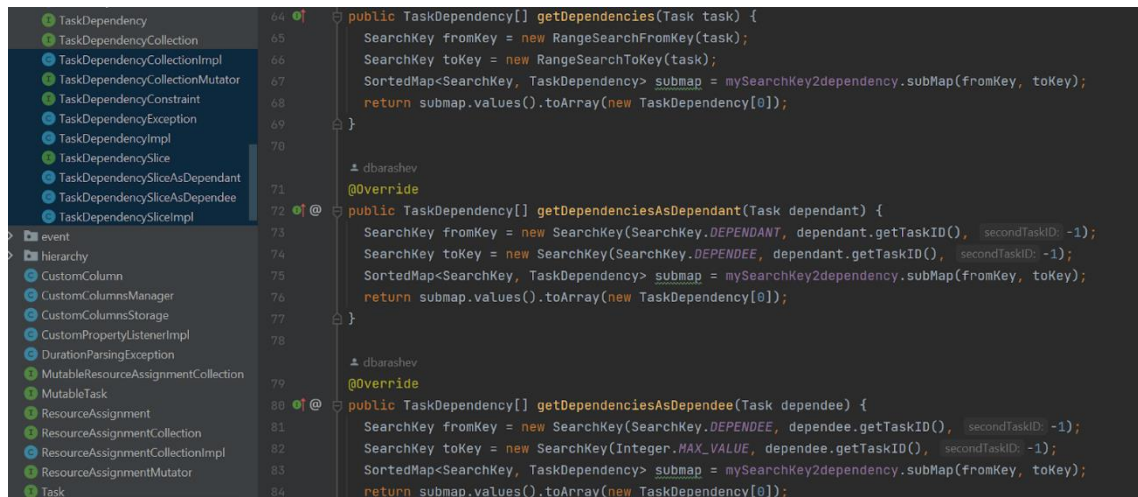
Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/task/dependency/TaskDependencyCollectionImpl.java

Explanation:

This class works with a subsystem of other classes to be easier to access all of them through this TaskDependencyCollectionImpl class.

Reviewed by Filipe Santo.



```
64 public TaskDependency[] getDependencies(Task task) {
65     SearchKey fromKey = new RangeSearchFromKey(task);
66     SearchKey toKey = new RangeSearchToKey(task);
67     SortedMap<SearchKey, TaskDependency> submap = mySearchKey2dependency.subMap(fromKey, toKey);
68     return submap.values().toArray(new TaskDependency[0]);
69 }
70
71
72 @Override
73 public TaskDependency[] getDependenciesAsDependant(Task dependant) {
74     SearchKey fromKey = new SearchKey(SearchKey.DEPENDANT, dependant.getTaskID(), secondTaskID: -1);
75     SearchKey toKey = new SearchKey(SearchKey.DEPENDEE, dependant.getTaskID(), secondTaskID: -1);
76     SortedMap<SearchKey, TaskDependency> submap = mySearchKey2dependency.subMap(fromKey, toKey);
77     return submap.values().toArray(new TaskDependency[0]);
78 }
79
80 @Override
81 public TaskDependency[] getDependenciesAsDependee(Task dependee) {
82     SearchKey fromKey = new SearchKey(SearchKey.DEPENDEE, dependee.getTaskID(), secondTaskID: -1);
83     SearchKey toKey = new SearchKey(SearchKey.DEPENDANT, dependee.getTaskID(), secondTaskID: -1);
84     SortedMap<SearchKey, TaskDependency> submap = mySearchKey2dependency.subMap(fromKey, toKey);
85     return submap.values().toArray(new TaskDependency[0]);
86 }
```

2. Command Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/undo/UndoManagerImpl.java

Explanation:

Commands are being manipulated as objects, allows to do and undo operations.

Reviewed by Rui Capareira.

```

119 4 usages dbarashev
120 @Override
121 public boolean canUndo() {
122     return mySwingUndoManager.canUndo();
123 }
124
125 4 usages dbarashev
126 @Override
127 public boolean canRedo() {
128     return mySwingUndoManager.canRedo();
129 }
130
131 dbarashev
132 @Override
133 public void undo() throws CannotUndoException {
134     mySwingUndoManager.undo();
135     fireUndoOrRedoHappened();
136 }
137
138 dbarashev
139 @Override
140 public void redo() throws CannotRedoException {
141     mySwingUndoManager.redo();
142     fireUndoOrRedoHappened();
143 }

```

3. Singleton Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/gui/ListAndFieldsPanel.java

Explanation:

Creates an object, the class has only one instance that generates a new box and returns it.

Reviewed by Jaime Russo.

```

30 3 usages dbarashev
31 public class ListAndFieldsPanel<T> {
32     4 usages
33     private EditableList<T> myList;
34     3 usages
35     private JComponent myFields;
36     7 usages
37     private Box myPanel;
38
39     1 usage dbarashev
40     public ListAndFieldsPanel(EditableList<T> list, JComponent fields) {
41         myList = list;
42         myFields = fields;
43     }
44
45     dbarashev
46     public JComponent getComponent() {
47         if (myPanel == null) {
48             SpringLayout topPanelLayout = new SpringLayout();
49             JPanel topPanel = new JPanel(topPanelLayout);
50
51             JComponent depsComponent = myList.getTableComponent();
52             JComponent titleComponent = new JLabel(myList.getTitle());
53             JComponent actionsComponent = myList.getActionsComponent();
54             topPanel.add(titleComponent);

```

Filipe Santo, 64859

1. Prototype Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/ChartComponentBase.java

Explanation:

This class lets us create copies of objects without depending on the concrete class.

Reviewed by Jaime Russo.

```
public abstract class ChartComponentBase extends JPanel implements TimelineChart {
    public static final Cursor HAND_CURSOR = Cursor.getPredefinedCursor(Cursor.HAND_CURSOR);
    public static final Cursor DEFAULT_CURSOR;
    public static final Cursor CURSOR_DRAG;

    static {
        Cursor drag = Cursor.getPredefinedCursor(Cursor.HAND_CURSOR);
        Cursor hand = Cursor.getPredefinedCursor(Cursor.HAND_CURSOR);
        try {
            drag = Toolkit.getDefaultToolkit().createCustomCursor(
                ImageIO.read(ChartComponentBase.class.getResource("/icons/16x16/chart-drag.png")),
                new Point(16, 16), ChartComponentBase.class.getSimpleName() + "-drag");
            hand = Toolkit.getDefaultToolkit().createCustomCursor(
                ImageIO.read(ChartComponentBase.class.getResource("/icons/16x16/chart-hand.png"))
```

2. Singleton Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/gui/GanttLookAndFeels.java

Explanation:

This singleton class provides us global access to the info about the installed LookAndFeels.

Reviewed by Carolina Simonet.


```

*/
public class GanttLookAndFeels {

    protected Map<String, GanttLookAndFeelInfo> infoByClass;

    protected Map<String, GanttLookAndFeelInfo> infoByName;

    protected static GanttLookAndFeels singleton;

    static {
        UIManager.put("ClassLoader", LookUtils.class.getClassLoader());
        UIManager.installLookAndFeel("Plastic", "com.jgoodies.looks.plastic.PlasticLookAndFeel");
    }

    protected GanttLookAndFeels() {

```

3. Proxy Pattern

Location:

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

Explanation:

This class has the same interface as the original service object, and when it's updated it passes from this class to the original documents object. Delegating it all the work to it.

Reviewed by Margarida Carvalho.

```

*/
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();
    }
}

```

Jaime Russo, 60062

1. Memento Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/undo/UndoManagerImpl

Explanation: In this algorithm, there is a class that saves the last state of the object, so we can undo and redo whenever we need.

Reviewed by Rui Capareira.

```
129      @Override
130      public void undo() throws CannotUndoException {
131          mySwingUndoManager.undo();
132          fireUndoOrRedoHappened();
133      }
```

2. Singleton Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/task/event/TaskDependencyEvent

Explanation: This class only have one instance and that instance can be accessed through a method.

Reviewed by Margarida Carvalho.

```
20  *
21  28 usages dbarashev
22  public class TaskDependencyEvent extends EventObject {
23      2 usages
24      private final TaskDependency myDependency;
25
26      3 usages dbarashev
27      public TaskDependencyEvent(TaskDependencyCollection source, TaskDependency dependency) {
28          super(source);
29          myDependency = dependency;
30      }
31
32      dbarashev
33      public TaskDependency getDependency() { return myDependency; }
```

3. Command Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/export/CommandLineExportApplication

Explanation: This class will export the command line text, other class will call this one to do that job.

Reviewed by Carolina Simonet.

```
77 @ private boolean export(Exporter exporter, Args args, IGanttProject project, UIFacade uiFacade) {
78     logger.debug(msg: "Using exporter {}", new Object[]{exporter}, new HashMap<>());
79     ConsoleUIFacade consoleUI = new ConsoleUIFacade(uiFacade);
80     GPLogger.setUIFacade(consoleUI);
81     // TODO: bring back task expanding
82     // if (myArgs.expandTasks) {
83     //     for (Task t : project.getTaskManager().getTasks()) {
84     //         project.getUIFacade().getTaskTree().setExpanded(t, true);
85     //     }
86     // }
87
88     Job.getJobManager().setProgressProvider(new ConsoleProgressProvider());
89     File outputFile = args.outputFile == null ? FileChooserPage.proposeOutputFile(project, exporter)
90         : args.outputFile;
91
92     Preferences prefs = new PluginPreferencesImpl(parent: null, name: "");
93     prefs.putInt(s: "zoom", args.zooming);
94     prefs.put(
95         s: "exportRange",
96         s1: DateParser.getIsoDate(project.getTaskManager().getProjectStart()) + " "
97         + DateParser.getIsoDate(project.getTaskManager().getProjectEnd());
98     prefs.putBoolean(s: "commandLine", b: true);
99
100     // If chart to export is defined, then add a string to prefs
101     if (args.chart != null) {
102         prefs.put(s: "chart", args.chart);
103     }
104 }
```

Margarida Carvalho, 60437

1. Memento Pattern

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/undo/UndoableEditImpl.java

Explanation:

With this method, we are able to access a previous state of an object and return it.

Reviewed by Carolina Simonet.

```

92      @Override
93      public void undo() throws CannotUndoException {
94          try {
95              restoreDocument(myDocumentBefore);
96              if (projectDatabaseTxn != null) {
97                  try {
98                      projectDatabaseTxn.undo();
99                  } catch (ProjectDatabaseException e) {
100                      GPLogger.log(e);
101                  }
102              }
103          } catch (DocumentException | IOException e) {
104              undoRedoExceptionHandler(e);
105          }
106      }

```

2. Factory Method

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/chart/ChartModelBase.java

Explanation:

It allows the creation of objects in a superclass.

Reviewed by Filipe Santo.

```

3 usages  dbarashev
454      public OffsetBuilder.Factory createOffsetBuilderFactory() {
455          OffsetBuilder.Factory factory = new OffsetBuilderImpl.FactoryImpl()
456              .withAtomicUnitWidth(getBottomUnitWidth())
457              .withBottomUnit(getBottomUnit())
458              .withCalendar(myTaskManager.getCalendar())
459              .withRightMargin(myScrollingSession == null ? 0 : 1)
460              .withStartDate(getOffsetAnchorDate())
461              .withViewportStartDate(getStartDate())
462              .withTopUnit(myTopUnit)
463              .withWeekendDecreaseFactor(
464                  getTopUnit().isConstructedFrom(getBottomUnit()) ? OffsetBuilderImpl.WEEKEND_UNIT_WIDTH_DECREASE_FACTOR : 1f);
465          if (getBounds() != null) {
466              factory.withEndOffset((int) getBounds().getWidth());
467          }
468          return factory;
469      }

```

3. Observer Pattern

Location:

org/apache/commons/io/input/ObservableInputStream.java

Explanation:

It is a mechanism that let us notify multiple objects about anything that happens to the observed object.

Reviewed by Rui Capareira.

```
243     protected void noteFinished() throws IOException {
244         for (final Observer observer : getObservers()) {
245             observer.finished();
246         }
247     }
248
249     2 usages
250     @ private void notify(final byte[] buffer, final int offset, final int result, final IOException ioe) throws IOException {
251         if (ioe != null) {
252             noteError(ioe);
253             throw ioe;
254         }
255         if (result == EOF) {
256             noteFinished();
257         } else if (result > 0) {
258             noteDataBytes(buffer, offset, result);
259         }
260     }
```

Rui Capareira, 57046

1. Template Method Pattern

Location:

ganttproject\ganttproject\src\main\java\net\sourceforge\ganttproject\importer\ImporterBase.java

Explanation:

Defines a general implementation for import related features, deferring the implementation of more specific steps to subclasses.

Reviewed by Margarida Carvalho.

```
36     3 usages dbarashev +1
    public abstract class ImporterBase implements Importer {
        5 usages
```

```
1 usage  👤 dbarashev +4
public class ImporterFromCsvFile extends ImporterBase {
    2 usages
```

```
60 1 usage  👤 dbarashev +1
public class IcsFileImporter extends ImporterBase {
    4 usages
```

```
44 2 usages  👤 dbarashev +2
public class ImporterFromMsProjectFile extends ImporterBase implements Importer {
    3 usages
```

```
41 4 usages  👤 dbarashev +3
public class ImporterFromGanttFile extends ImporterBase {
    9 usages
```

```
25 2 usages  👤 dbarashev
public class ImporterFromTxtFile extends ImporterBase {
26
```

2. Singleton Pattern

Location:

ganttproject\biz.ganttproject.core\src\main\java\biz\ganttproject\core\chart\canvas
\Canvas.java

Explanation:

The Singleton myStyles can only be accessed through its instance operation. The constructor is private and the public methods instantiate the singleton if it still doesn't exist.

Reviewed by Jaime Russo.

3 usages

```
private LinkedHashSet<String> myStyles;
```

2 usages  dbarashev

```
private LinkedHashSet<String> getStyles() {  
    if (myStyles == null) {  
        myStyles = new LinkedHashSet<String>();  
    }  
    return myStyles;  
}
```

```
87     public void addStyle(String style) {  
88         getStyles().add(style);  
89     }  
90  
91     public boolean hasStyle(String style) {  
92         return getStyles().contains(style);  
93     }  
94
```

3. Iterator Pattern

Location:

ganttproject\biz.ganttproject.core\src\main\java\biz\ganttproject\core\time\TimeUnitStack.java

Explanation:

The iterator is used to traverse a collection of elements and access them without exposing the underlying representation of the data structure.

Reviewed by Filipe Santo.

```
49 // Now compare lists to find a common unit
50 current = unit2;
51 while (current != null) {
52     Iterator<TimeUnit> u1Iterator = units1.iterator();
53     while (u1Iterator.hasNext()) {
54         TimeUnit nextU1 = u1Iterator.next();
55         if (current.equals(nextU1)) {
56             return current;
57         }
58     }
59     current = current.getDirectAtomUnit();
60 }
61 return null;
62 }
63 }
```