

PROJETO SE2223

SE2223_59797_60441_60677_60816_60971





Daniel Eugénio 59797
Rafael Costa 60441
Miguel Agostinho 60677
Francisco Silveira 60816
Guilherme Abrantes 60971

Engenharia de Software

Conteúdo

1ª Fase.....	2
<i>Code Smells</i>	3
<i>Padrões</i>	19
<i>Metrics</i>	34
2ª Fase.....	43
<i>Use Cases</i>	44
<i>User Stories</i>	50
<i>Anexos</i>	51



SE 22/23



1ª Fase

Code Smells

Switch statement (VirtualFlow.java) - Miguel Agostinho 60677

```
switch(event.getTextDeltaXUnits()) {  
    case CHARACTERS:  
        // can we get character size here?  
        // for now, fall through to pixel values  
    case NONE:  
        double dx = event.getDeltaX();  
        double dy = event.getDeltaY();  
  
        virtualDelta = (Math.abs(dx) > Math.abs(dy) ? dx : dy);  
    }  
}
```

O switch apresentado abaixo para além de no case “CHARACTERS” não se fazer nada podendo ser um “Dead Code”, poderia ser feito apenas com um if/else evitando assim o uso desnecessário de um switch statement.

Reviews

Daniel Eugénio 59797

Review do code smell nº1 (Switch Statement), é bem identificado, o case Characters não tem qualquer bloco de código, resultando num switch com apenas um caso que, como foi bem evidenciado pelo colega.

Duplicated Code (VirtualFlow.java) - Miguel Agostinho 60677

```
ScrollBar nonVirtualBar = isVertical() ? hbar : vbar;
if (needBreadthBar) {
    double nonVirtualDelta = isVertical() ? event.getDeltaX() : event.getDeltaY();
    if (nonVirtualDelta != 0.0) {
        double newValue = nonVirtualBar.getValue() - nonVirtualDelta;
        if (newValue < nonVirtualBar.getMin()) {
            nonVirtualBar.setValue(nonVirtualBar.getMin());
        } else if (newValue > nonVirtualBar.getMax()) {
            nonVirtualBar.setValue(nonVirtualBar.getMax());
        } else {
            nonVirtualBar.setValue(newValue);
        }
        event.consume();
    }
}
});
```

Neste caso podemos observar que chamamos duas vezes o método `nonVirtualBar.getMin()` e `nonVirtualBar.getMax()` o que deveríamos fazer era criar duas variáveis locais, uma que guardava o valor retornado por `nonVirtualBar.getMin()` e a outra que guarda o valor de `nonVirtualBar.getMax()` e assim evitamos chamar o mesmo método várias vezes.

Reviews

Guilherme Abrantes 60971

Review do code smell Duplicated Code , aqui temos code smell muito bem encontrado, esta duplicação de código poderia passar facilmente despercebida, porém são estas diferenças que tornam tudo mais legível, a solução de criar as variáveis locais acho que seria o ideal.

Comments (ChartUIConfiguration.java) - Miguel Agostinho 60677

```
public class ChartUIConfiguration {

    // dbarashev +1
    public ChartUIConfiguration(UIConfiguration projectConfig) {
        mySpanningRowTextFont = Fonts.TOP_UNIT_FONT;
        mySpanningHeaderBackgroundColor = new Color(0.93f, 0.93f, 0.93f);
        myHeaderBorderColor = new Color(0.482f, 0.482f, 0.482f);
        myWorkingTimeBackgroundColor = Color.WHITE;
        myHolidayTimeBackgroundColor = new Color(0.9f, 0.9f, 0.9f);
        myPublicHolidayTimeBackgroundColor = new Color(240, 220, 240);
        // mySpawningHeaderColor = new Color(0f, 0f, 0f);
        myBottomUnitColor = new Color(0.482f, 0.482f, 0.482f);
        myProjectConfig = projectConfig;
        myChartStylesOption = new ChartPropertiesOption();
    }

    // dbarashev
    ListOption<Map.Entry<String, String>> getChartStylesOption() { return myChartStylesOption; }
    // dbarashev
    Font getSpanningHeaderFont() { return mySpanningRowTextFont; }

    // usage A dbarashev
    public int getHeaderHeight() { return myHeaderHeight; }

    // usage A dbarashev
    public void setHeaderHeight(int headerHeight) { myHeaderHeight = headerHeight; }

    // dbarashev
    public int getSpanningHeaderHeight() { return myHeaderHeight / 2; }

    // dbarashev
    public Color getSpanningHeaderBackgroundColor() { return mySpanningHeaderBackgroundColor; }

    // dbarashev
    public Color getHeaderBorderColor() { return myHeaderBorderColor; }
```

Neste caso a classe ChartUIConfiguration têm uma ausência de comentários muito notória e os comentários nas classes são muito importantes porque se outra pessoa for programar a mesma classe precisa de saber o que os métodos ou até a classe fazem, portanto deveríamos comentar esta.

Reviews

Francisco Silveira 60816

Review code smell nº3 :

Concordo com o que o colega disse. A falta de comentários torna o código pouco legível e torna o trabalho de quem vai trabalhar nele muito mais difícil. Os comentários são uma parte essencial do código.

Repeated Code - Daniel Eugénio 59797

```
234 |getViewModel().createView(myGanttChartTabContent, new ImageIcon(getClass().getResource("/icons/tasks_16.gif")));
235 |getViewModel().toggleVisible(myGanttChartTabContent);
236
237 |myResourceChartTabContent = new ResourceChartTabContentPanel(getProject(), getUIFacade(), getResourcePanel(),
238 |getResourcePanel().area);
239 |getViewModel().createView(myResourceChartTabContent, new ImageIcon(getClass().getResource("/icons/res_16.gif")));
240 |getViewModel().toggleVisible(myResourceChartTabContent);
```

- Neste trecho, é possível reparar que as linhas 234-235 e 239-240 são muito semelhantes, diferindo apenas no último argumento passado no método `toggleVisible()`.
- Uma solução possível seria fazer um método que receberia o argumento que queremos passar no `toggleVisible()`. Esta simples mudança tornaria o código nesta região mais limpo e legível.

-----> (GanttProject.java) <-----

Reviews

Miguel Agostinho 60677

Review do code smell nº1, concordo com o meu colega pois a repetição de código torna-o mais difícil de ler e entender e a solução que forneceu também me parece boa.

Long Parameter List - Daniel Eugénio 59797

```
146  
147 private void constructTopOffsets(TimeUnit timeUnit, List<Offset> topOffsets, List<Offset> bottomOffsets,  
148 int initialEnd, int baseUnitWidth) {
```

- Neste exemplo, os argumentos topOffsets e bottomOffsets podem ser atributos numa classe (Offsets.java, por exemplo) e, nesse cenário, estes dois argumentos podiam ser 1 apenas.

-----> (OffsetBuilderImpl.java) <-----

Reviews

Rafael Costa 60441

Review do long parameter list.

Concordo com o referido, é um bad smell code que devemos evitar fazer, pois torna o código confuso e denso.

Divergent Change - Daniel Eugénio 59797

```
85
86     Box colorLabels = Box.createHorizontalBox();
87     for (final Color c : myRecentColors) {
88         final JLabel label = new JLabel();
89         label.setBackgroundPainter(new Painter<JLabel>() {
90             @Override
91             public void paint(Graphics2D g, JLabel object, int width, int height) {
92                 g.setColor(c);
93                 g.fillRect(4, 4, width-8, height-8);
94             }
95         });
96         label.setCursor(Cursor.getPredefinedCursor(Cursor.HAND_CURSOR));
97         label.setFocusable(true);
98         label.setPreferredSize(new Dimension(20, 20));
99         label.setMaximumSize(new Dimension(20, 20));
100
101         final Border outsideFocusBorder = BorderFactory.createLineBorder(c.darker(), 2);
102         final Border outsideNoFocusBorder = BorderFactory.createEmptyBorder(2,2,2,2);
103         label.setBorder(outsideNoFocusBorder);
104         label.addFocusListener(new FocusAdapter() {
105             @Override
106             public void focusGained(FocusEvent e) {
107                 label.setBorder(outsideFocusBorder);
108                 myChooserImpl.setColor(c);
109                 mySelectedColor = c;
110             }
111             @Override
112             public void focusLost(FocusEvent e) { label.setBorder(outsideNoFocusBorder); }
113         });
114
115         label.addMouseListener(new MouseAdapter() {
116             @Override
117             public void mouseClicked(MouseEvent e) { label.requestFocus(); }
118         });
119
120     };
121     colorLabels.add(label);
122     colorLabels.add(Box.createHorizontalStrut(5));
123 }
124
125 colorLabels.setBorder(BorderFactory.createEmptyBorder(0, 7, 0, 0));
```

- Aqui é possível ver a completa ausência do uso de constantes, o que nos leva a ter de mexer em várias zonas do código na mesma classe. A utilização dos chamados *magic numbers* também dificulta a legibilidade do código por parte de pessoas que não estiveram envolvidas inicialmente no seu desenvolvimento.

- A solução seria utilizar constantes com nomes explicativos para podermos fatorizar várias zonas do código ao mesmo tempo.

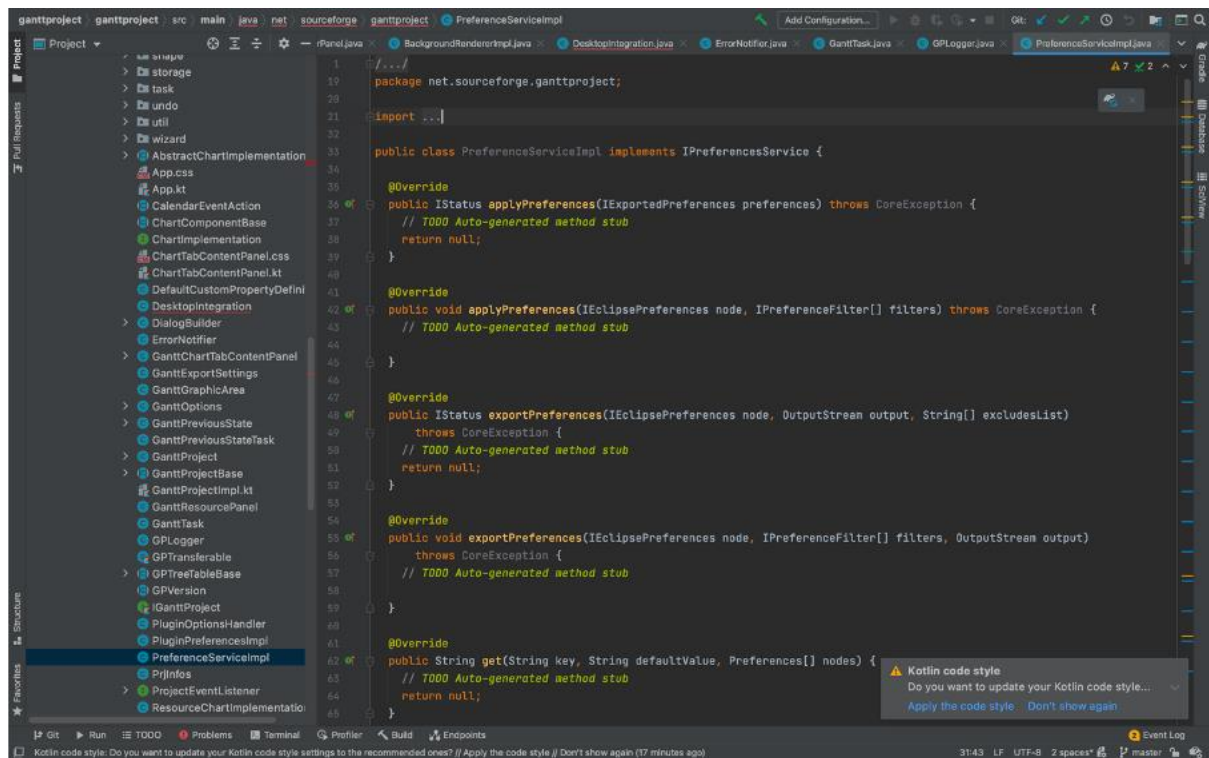
-----> (GPColorChooser.java) <-----

Reviews

Guilherme Abrantes 60971

Review do code smell Divergent Change, acho que foi um code smell bem encontrado sendo difícil a alteração do código pois vamos ter de alterar todos os números e também dificulta a legibilidade, a solução encontrada acho que é perfeita pois resolveria estes dois problemas mesmo que estes números só tenham sido usados neste método variáveis locais com nomes explicativos tornaria o código muito mais claro.

Speculative Generality - Rafael Costa 60441



É constituído por código que é genérico ou abstrato e o mais importante, não é realmente necessário hoje. Esse código está lá para apoiar o comportamento futuro, que pode ou não, ser necessário no futuro.

A solução será apenas criar os métodos e parâmetros à medida que é necessário.

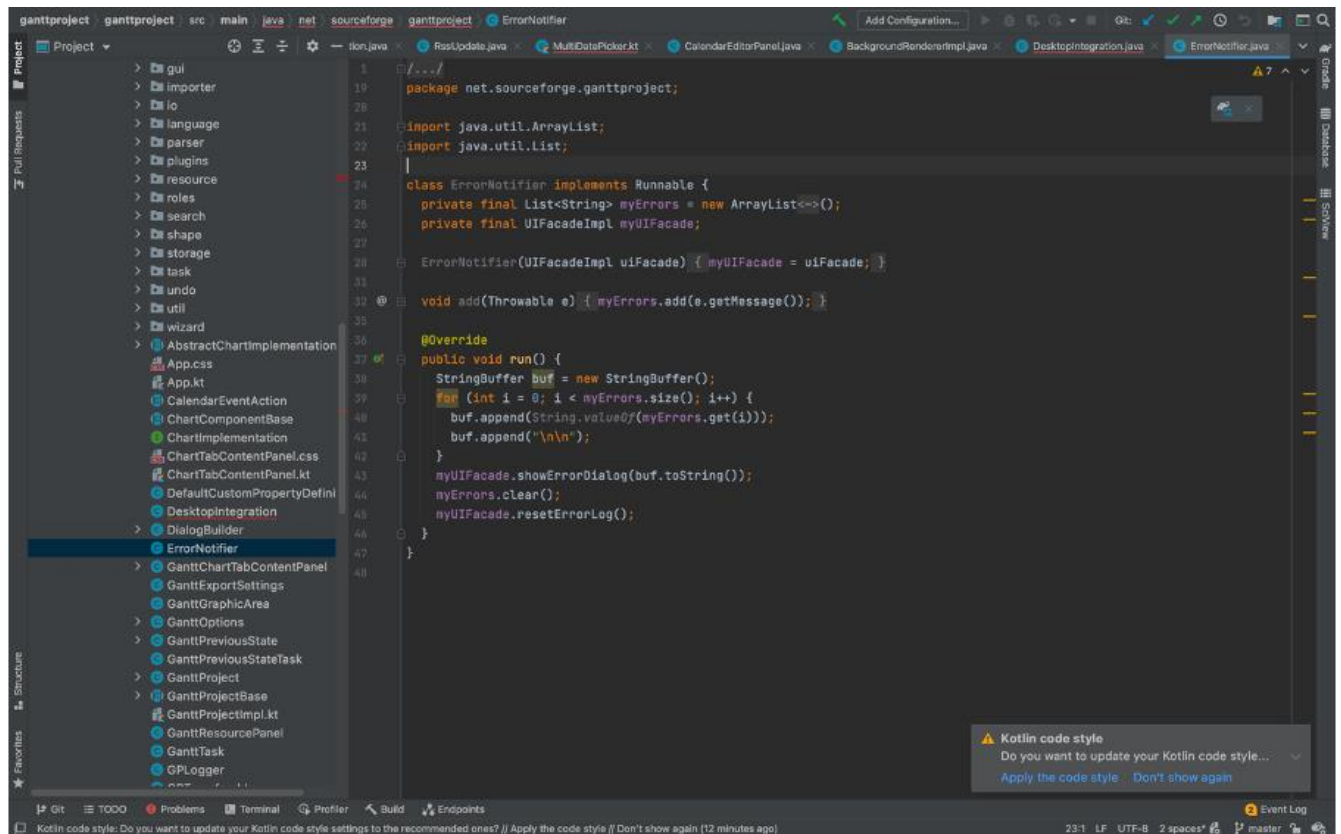
<PreferenceServiceImpl.java>

Reviews

Guilherme Abrantes 60971

Review do code smell Speculative Generality, este code smell foi bem encontrado a explicação está bastante boa tenho apenas a acrescentar que além disto não existe qualquer tipo de comentário a explicar o que aquela classe poderia acrescentar no futuro, dificultando ainda mais o código.

Not Comment - Rafael Costa 60441



A classe apresenta-se sem comentários, o que é algo essencial para a interpretação da mesma.

Os comentários são essenciais uma vez que ao passar o código a outra pessoa, ou a trabalhar em equipa vai facilitar a interpretação do código pelos mesmos

<ErrorNotifier.java>

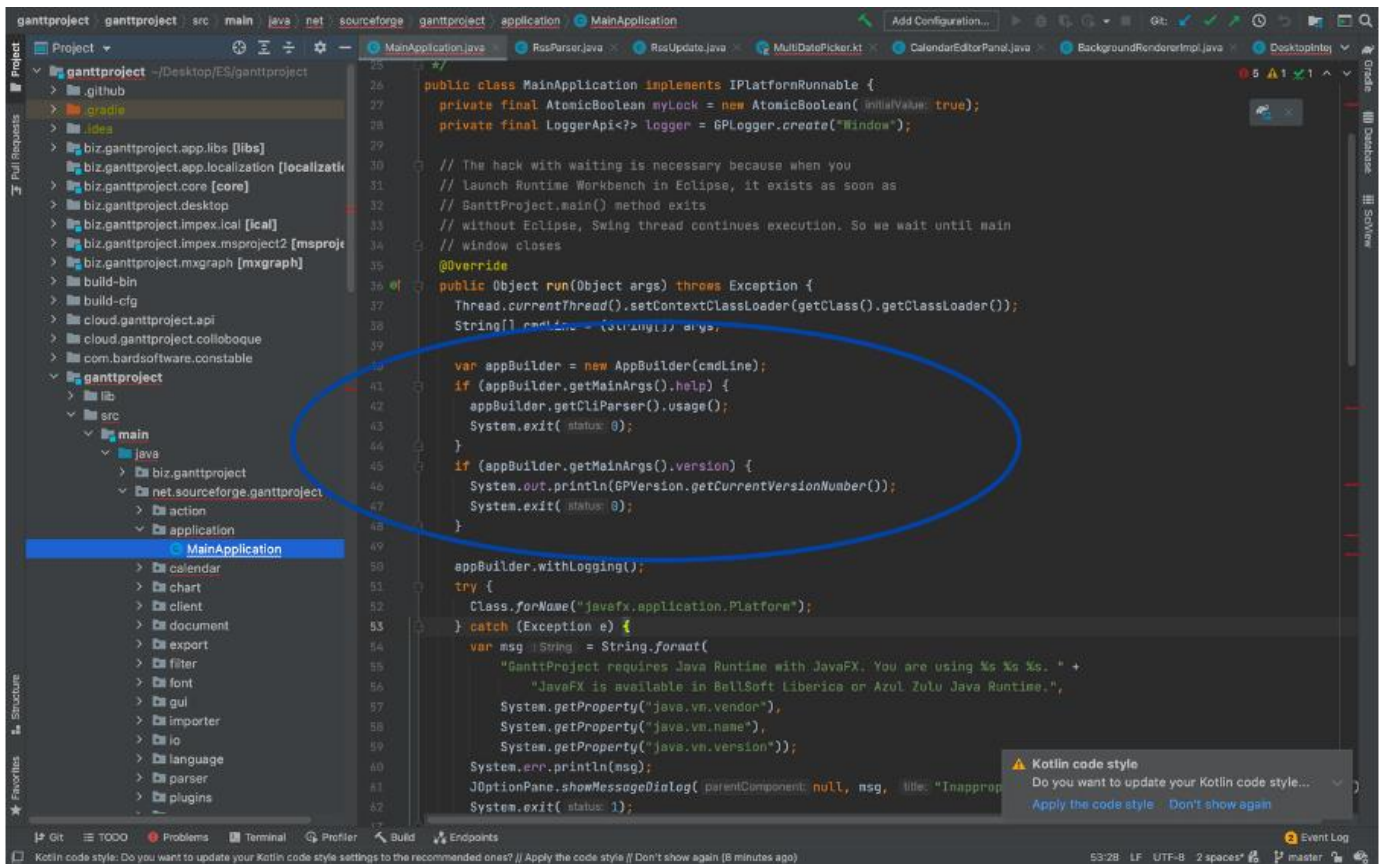
Reviews

Francisco Silveira 60816

Review Not Commented:

Concordo com o que o colega disse. A falta de comentários torna o código pouco legível e torna o trabalho de quem vai trabalhar nele muito mais difícil. Os comentários são uma parte essencial do código.

Duplicated Code - Rafael Costa 60441



Dentro de cada "if" é chamada uma função para chegar a um valor da mesma. Neste caso, para evitarmos estarmos sempre a chamar a mesma função, podemos fazer a chamada antes do "if" e guardar `appBuilder.getMainArgs()` em uma variável, com isso não necessitamos de estar a chamar 2 vezes o `getMainArgs()` da classe `AppBuilder`

<MainApplication.java>

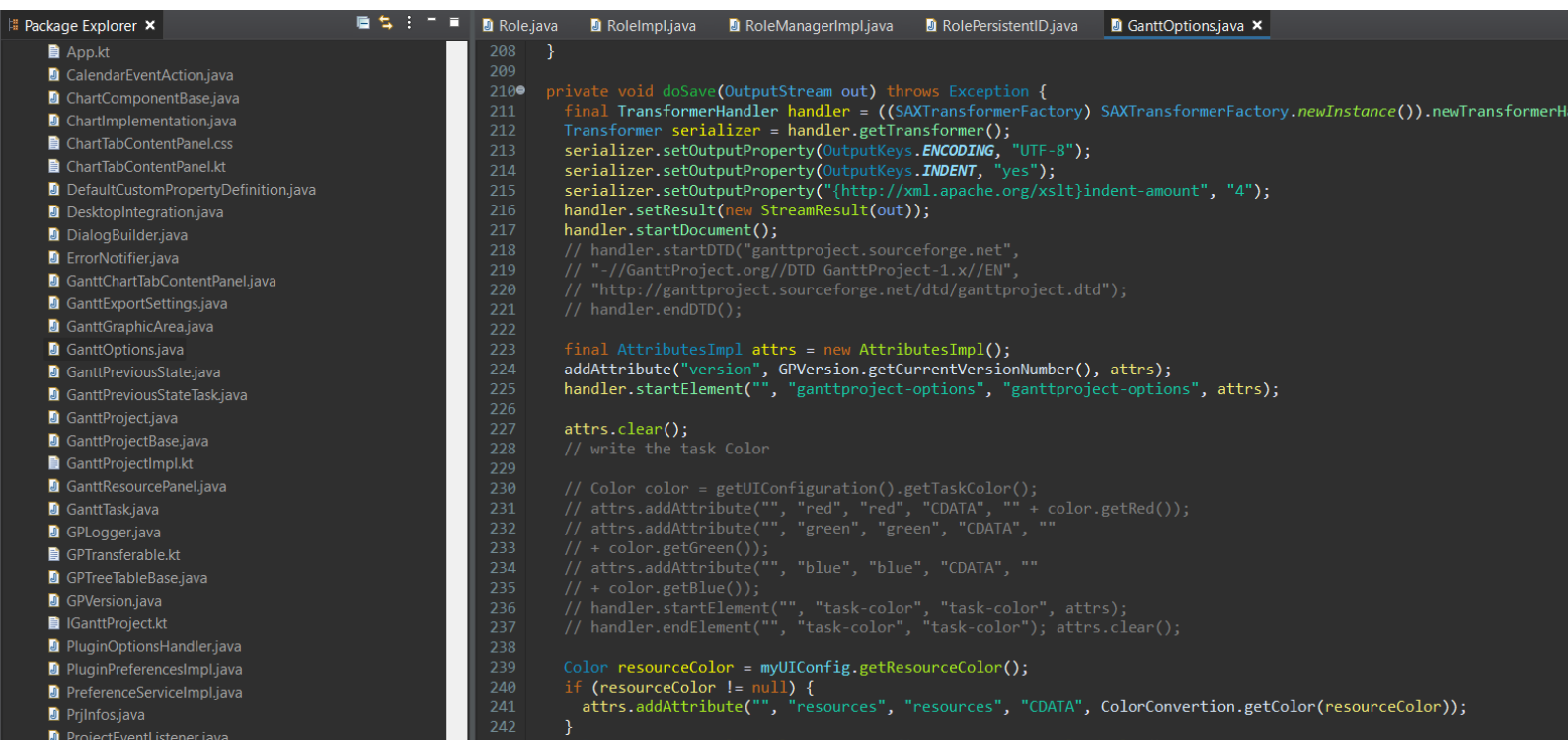
Reviews

Francisco Silveira 60816

Review duplicated code:

Concordo com o que o colega disse. Em alguns casos, guardar as variáveis que utilizamos várias vezes é extremamente importante pois, por exemplo, pode ser necessário correr grandes estruturas de dados para encontrar esses valores para além de retirar o código duplicado.

Comments that take on a “reminder” nature - Guilherme Abrantes 60971



Aqui vemos um comentário que evidencia que algo precisa de ser feito no futuro, o que aplica alterações em outros métodos para resolver devíamos ter implementado logo esta funcionalidade para não resultar em mais problemas.

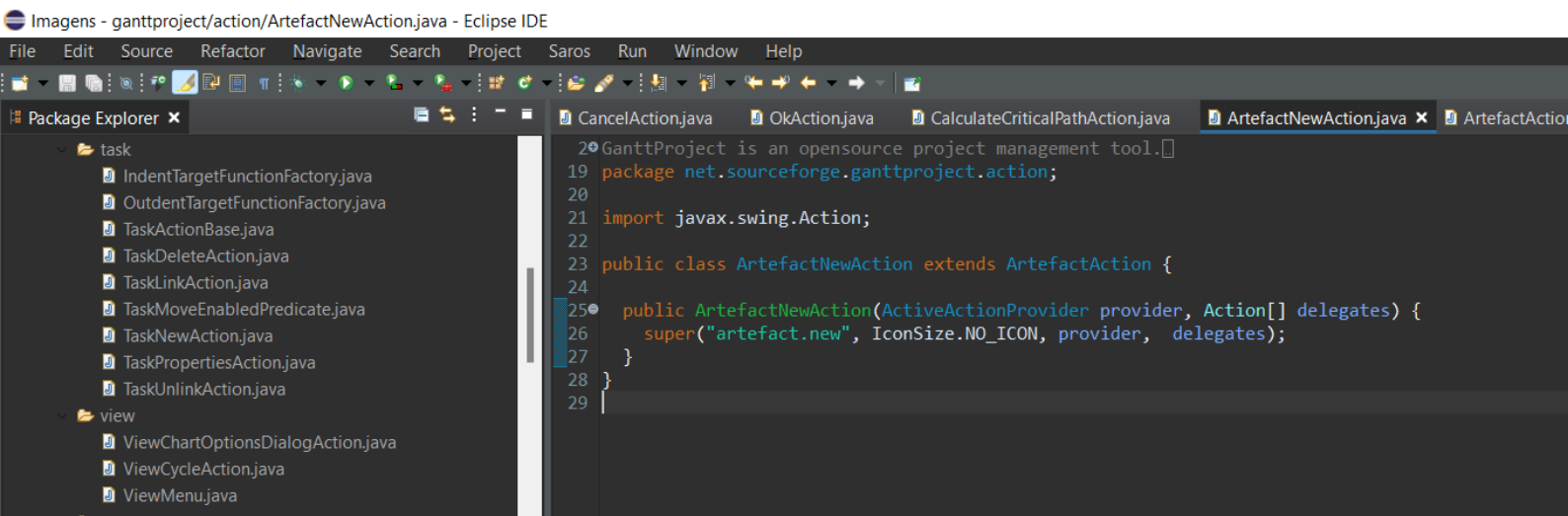
-----> (GanttOptions.java) <-----

Reviews

Rafael Costa 60441

O exposto no code smell nº1, a meu ver pode não evidenciar que pertence algo precisa de ser feito no futuro, poderá também ser algo que simplesmente deixou de ser necessário de utilizar e para "jogar pelo seguro" o troço de código foi mantido em comentário.

Data Class - Guilherme Abrantes 60971



Aqui vemos o Code smell Dataclass sendo esta classe bastante desnecessária pois apenas tem o construtor, esta classe seria facilmente substituída por um simples método na classe ArtefactAction.

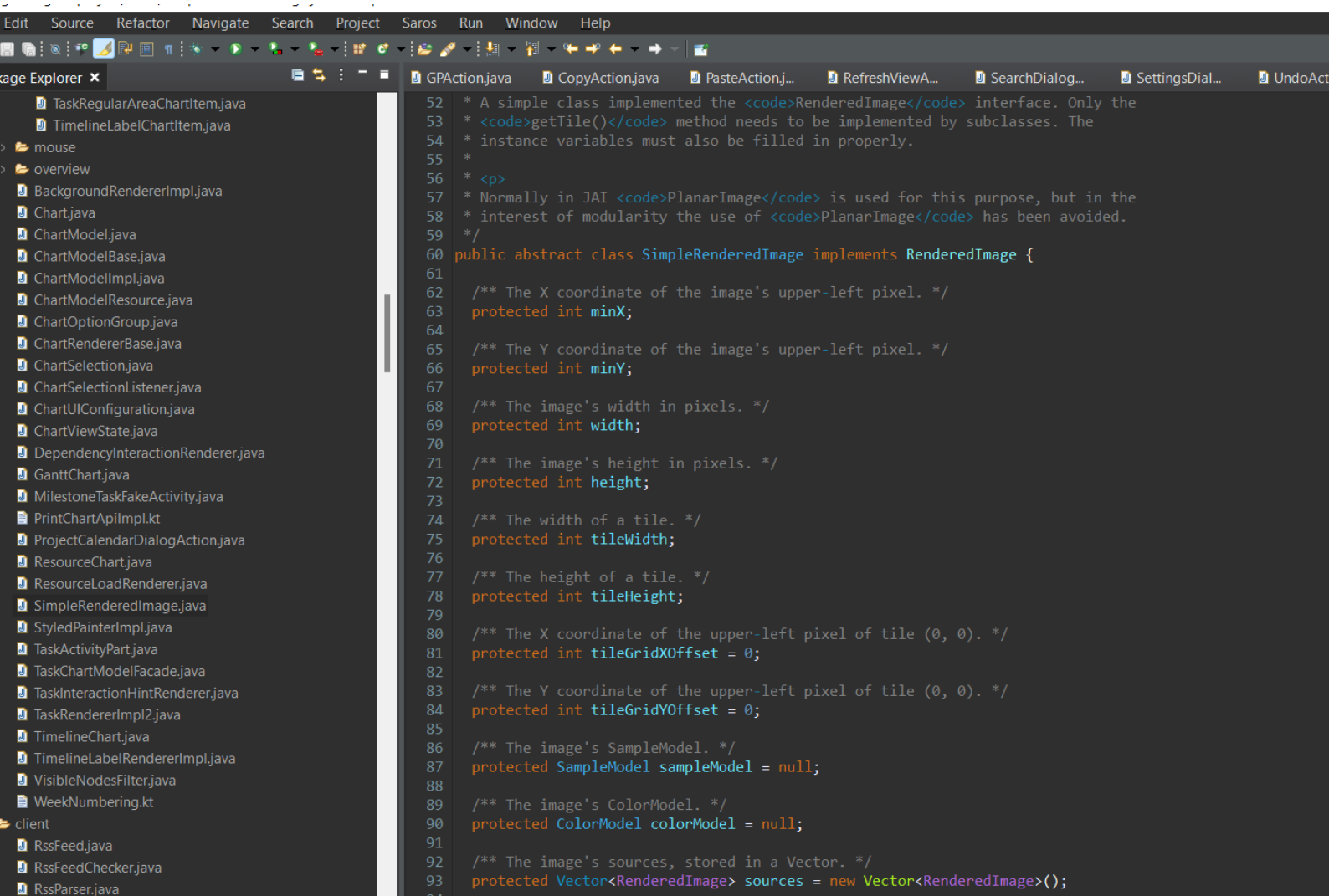
-----> (ArtefactNewAction.java) <-----

Reviews

Miguel Agostinho 60677

Concordo com o code smell nº2 pois, também acho desnecessário a criação desta classe porque apenas tem um construtor sem nenhum método.

Data Clumps - Guilherme Abrantes 60971



Podemos observar nesta classe bastantes inteiros que representam coordenadas e dimensões, é possível então ter uma classe com estas constantes e fazendo nesta classe alguns dos cálculos tornando tudo mais claro e fácil de perceber.

-----> (SimpleRenderedImage.java) <-----

Reviews

Daniel Eugénio 59797

Concordo com o code smell nº3, penso que o code smell mencionado é corretamente identificado. No entanto, penso que também poderia ser considerado Primitive Obsession, pelo uso excessivo de atributos com o tipo inteiro.

Switch statement - Francisco Silveira 60816

(GanttCSVExport.java)

Neste programa podemos ver que existem cases dentro do switch sem qualquer tipo de código, ou seja, não servem para nada. Neste caso, deveríamos remover esses cases ou em alguns casos específicos criar um case "Default" que trataria de todos os cases que não é suposto executar código.


```
case END_DATE:
    writer.print(task.getDisplayEnd());
    break;
case DURATION:
    writer.print(task.getDuration().getLength());
    break;
case COMPLETION:
    writer.print(task.getCompletionPercentage());
    break;
case OUTLINE_NUMBER:
    List<Integer> outlinePath = task.getManager().getTaskHierarchy().getOutlinePath(task);
    writer.print(Joiner.on('.').join(outlinePath));
    break;
case COORDINATOR:
    ResourceAssignment coordinator = Iterables.tryFind(Arrays.asList(task.getAssignments()), COORDINATOR_PREDICATE).orNull();
    writer.print(coordinator == null ? "" : coordinator.getResource().getName());
    break;
case PREDECESSORS:
    writer.print(TaskProperties.formatPredecessors(task, ";", true));
    break;
case RESOURCES:
    writer.print(getAssignments(task));
    writer.print(buildAssignmentSpec(task));
    break;
case COST:
    writer.print(task.getCost().getValue());
    break;
case COLOR:
    if (!Objects.equal(task.getColor(), task.getManager().getTaskDefaultColorOption().getValue())) {
        writer.print(ColorConversion.getColor(task.getColor()));
    } else {
        writer.print("");
    }
    break;
case INFO:
case PRIORITY:
case TYPE:
    break;
}
```

Reviews

Rafael Costa 60441

Concordo com o code smell nº1, pois não nos serve de nada ter condições no switch nas quais não utilizamos, para isso deveria ser inserido default no final do switch ou eliminado completamente aqueles "cases" que não estão a ser utilizados

(TaskDisplayColumnsTagHandler.java) - Francisco Silveira 60816

```
 @Override
protected boolean onStartElement(Attributes attrs) {
    if (!isEnabled) {
        return false;
    }
    loadTaskDisplay(attrs);
    return true;
}
```

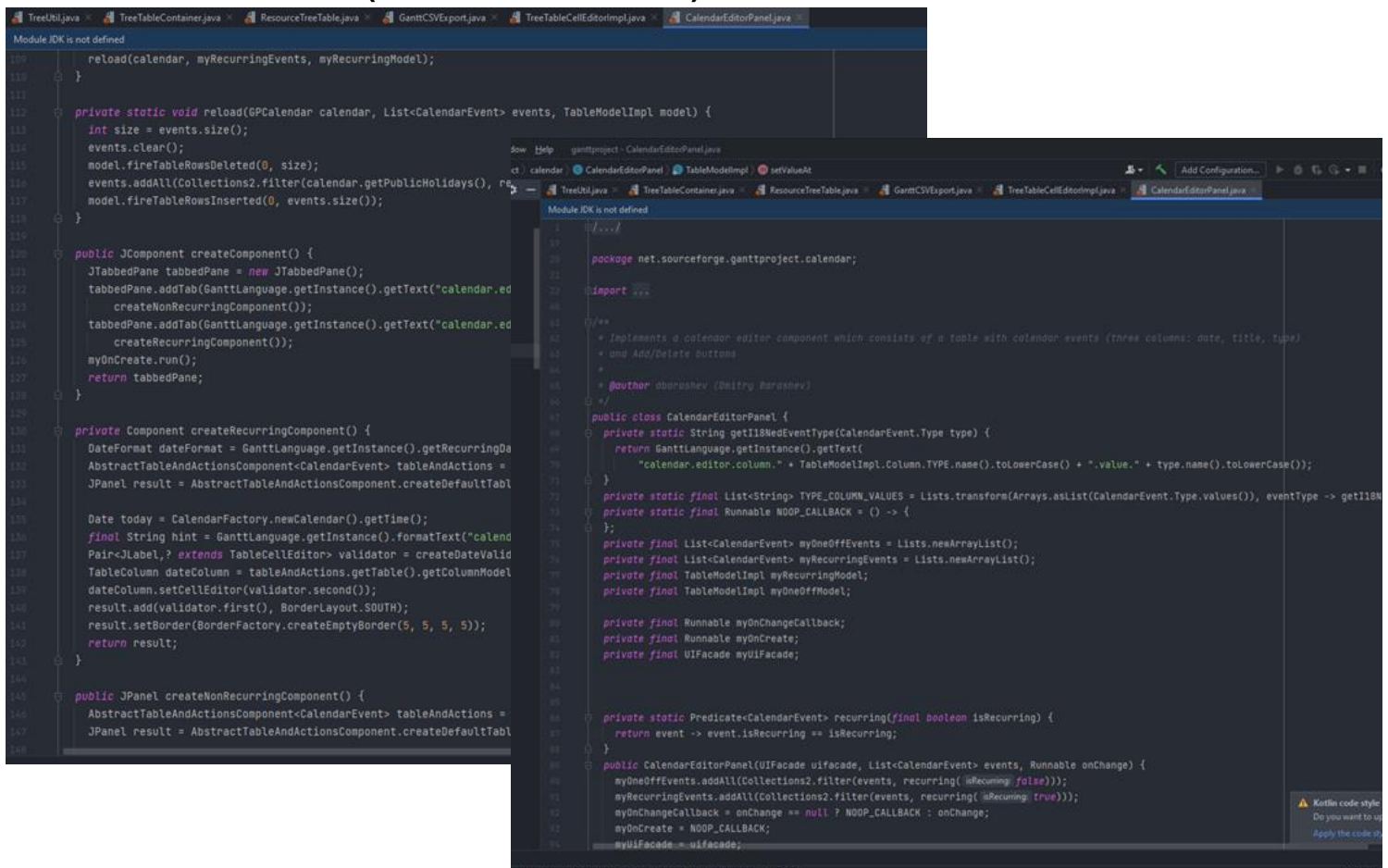
Como podemos ver neste pedaço de código no primeiro “if” tratamos primeiro da negação de um método booleano. Deveríamos testar a condição “isEnabled” dentro do “if” e executar “loadTaskDisplay()” e retornar “true”. De seguida depois do “if” retornamos “false”.

Reviews

Miguel Agostinho 60677

Concordo com o code smell nº2 pois, a negação de uma condição no primeiro if pode causar uma dificuldade de percepção do código.

No Comments (CalendarEditorPanel) - Francisco Silveira 60816



Esta Classe esta sem comentários, o que é essencial para a interpretação da mesma. Sem comentários, alguém ou alguma equipa que fosse trabalhar no nosso código iria ter bastante dificuldade em entender o programa. O que dificultaria o trabalho da equipa e faria com que demorasse muito mais tempo.

Reviews

Daniel Eugénio 59797

Review do code smell Not Commented, concordo com a opinião do colega. Apesar desta classe ser relativamente simples, os comentários são boas adições ao código e podem ser cruciais para o entendimento de pessoas novas, tanto ao código como ao mundo da programação.

Padrões

Prototype Pattern (ProjectRolesOptionPageProvider.java) - Miguel Agostinho 60677

```
@Override
protected Role createValue(Role prototype) {
    RoleSet projectRoles = getRoleManager().getProjectRoleSet();
    return projectRoles.createRole(prototype.getName());
}
```

Neste padrão estamos a criar um objeto copiando informações de um outro, para o podermos alterar sem danificar o original.

Reviews

Guilherme Abrantes 60971

Review do pattern Prototype Pattern, concordo com a identificação deste padrão acho que o meu colega deu uma boa explicação

Iterator Pattern (TableRowSkinBase.java) - Miguel Agostinho 60677

```
2 usages  Dmitry Barashev
private void recreateCells() {
    if (cellsMap != null) {
        Collection<Reference<R>> cells = cellsMap.values();
        Iterator<Reference<R>> cellsIter = cells.iterator();
        while (cellsIter.hasNext()) {
            Reference<R> cellRef = cellsIter.next();
            R cell = cellRef.get();
            if (cell != null) {
                cell.updateIndex(-1);
                cell.getSkin().dispose();
                cell.setSkin(null);
            }
        }
        cellsMap.clear();
    }

    ObservableList<? extends TableColumnBase*<T, ?*>/> columns = getVisibleLeafColumns();

    cellsMap = new WeakHashMap<>(columns.size());
    fullRefreshCounter = DEFAULT_FULL_REFRESH_COUNTER;
    getChildren().clear();

    for (TableColumnBase col : columns) {
        if (cellsMap.containsKey(col)) {
            continue;
        }

        // create a TableCell for this column and store it in the cellsMap
        // for future use
        createCellAndCache(col);
    }
}
```

Padrão identificado pela criação de um iterador para percorrer elementos “Reference”.

Reviews

Daniel Eugénio 59797

Review do pattern Iterator, concordo com o que foi dito pelo colega, é bastante evidente a utilização deste padrão e não tenho nada a acrescentar.

Template Method Pattern (TreeTableCells.kt) - Miguel Agostinho 60677

```
Dmitry Barashev +2
override fun startEdit() {
    if (this.index == -1) {
        return
    }
    if (!isEditable) {
        onEditingCompleted()
        return
    }
    super.startEdit()
    contentDisplay = ContentDisplay.GRAPHIC_ONLY
    disclosureNode?.let {
        it.isVisible = false
    }
}

if (isEditing) {
    treeTableView.requestFocus()
    doStartEdit()
} else {
    onEditingCompleted()
}
}
```

Este padrão consiste em fazer “override” de um método da classe super chamando esse mesmo método e adicionando mais algumas funcionalidades.

Reviews

Francisco Silveira 60816

Review do Template Method Pattern:

Concordo com o que o colega disse. Facilmente identificamos que este método chama o método da classe super e adiciona novo código.

Iterador - Daniel Eugénio 59797

```
112     private DefaultMutableTreeNode buildTree() {  
113  
114         DefaultMutableTreeNode root = new DefaultMutableTreeNode();  
115         List<HumanResource> listResources = myResourceManager.getResources();  
116         Iterator<HumanResource> itRes = listResources.iterator();  
117  
118         while (itRes.hasNext()) {  
119             HumanResource hr = itRes.next();  
120             ResourceNode rnRes = new ResourceNode(hr); // the first for the resource  
121             root.add(rnRes);  
122         }  
123         return root;  
124     }
```

- Este padrão é facilmente identificado pela criação de um objeto do tipo Iterator. É acompanhado com um ciclo while para percorrer os elementos da lista

-----> (ResourceTreeTableModel.java) <----

Reviews

Miguel Agostinho 60677

Concordo com o pattern nº1 pois, conseguimos ver facilmente a criação de um iterador com o intuito de percorrer uma coleção de objetos.

Prototype - Daniel Eugénio 59797

```
141     public TaskBuilder withPrototype(Task prototype) {  
142         myPrototype = prototype;  
143         return this;  
144     }
```

- Este pedaço de código permite criar uma cópia dum objeto Task que nele seja passado. O método em questão é usado na classe ClipboardTaskProcessor, para criar uma cópia de uma task.

-----> (TaskManager.java) <-----

Reviews

Francisco Silveira 60816

Prototype: Neste padrão podemos facilmente encontrar um prototype. É recebido um objeto é retornado outro com os mesmos dados. Concordo com tudo o que o meu colega disse.

Façade - Daniel Eugénio 59797

```
102 class UIFacadeImpl extends ProgressProvider implements UIFacade {
103     private final JFrame myMainFrame;
104     private final ScrollingManager myScrollingManager;
105     private final ZoomManager myZoomManager;
106     private final GanttStatusBar myStatusBar;
107     private final UIFacade myFallbackDelegate;
108     private final TaskSelectionManager myTaskSelectionManager;
109     private final List<GPOptionGroup> myOptionGroups = Lists.newArrayList();
110     private final GPOptionGroup myOptions;
111     private final LafOption myLafOption;
112     private final GPOptionGroup myLogoOptions;
113     private final DefaultFileOption myLogoOption;
114     private final NotificationManagerImpl myNotificationManager;
115     private final TaskView myTaskView = new TaskView();
116     private final DialogBuilder myDialogBuilder;
117     private final Map<String, Font> myOriginalFonts = Maps.newHashMap();
118     private final List<Runnable> myOnUpdateComponentTreeUiCallbacks = Lists.newArrayList();
119     private float myLastScale = 0;
120
121     private static Map<FontSpec.Size, String> getSizeLabels() {
122         Map<FontSpec.Size, String> result = Maps.newHashMap();
123         for (FontSpec.Size size : FontSpec.Size.values()) {
124             result.put(size, GanttLanguage.getInstance().getText("optionValue.ui.appFontSpec." + size.toString() + ".Label"));
125         }
126         return result;
127     }
```

- Aqui está implementado o padrão Façade. Neste caso, o padrão é aplicado para simplificar o API ProgressProvider.

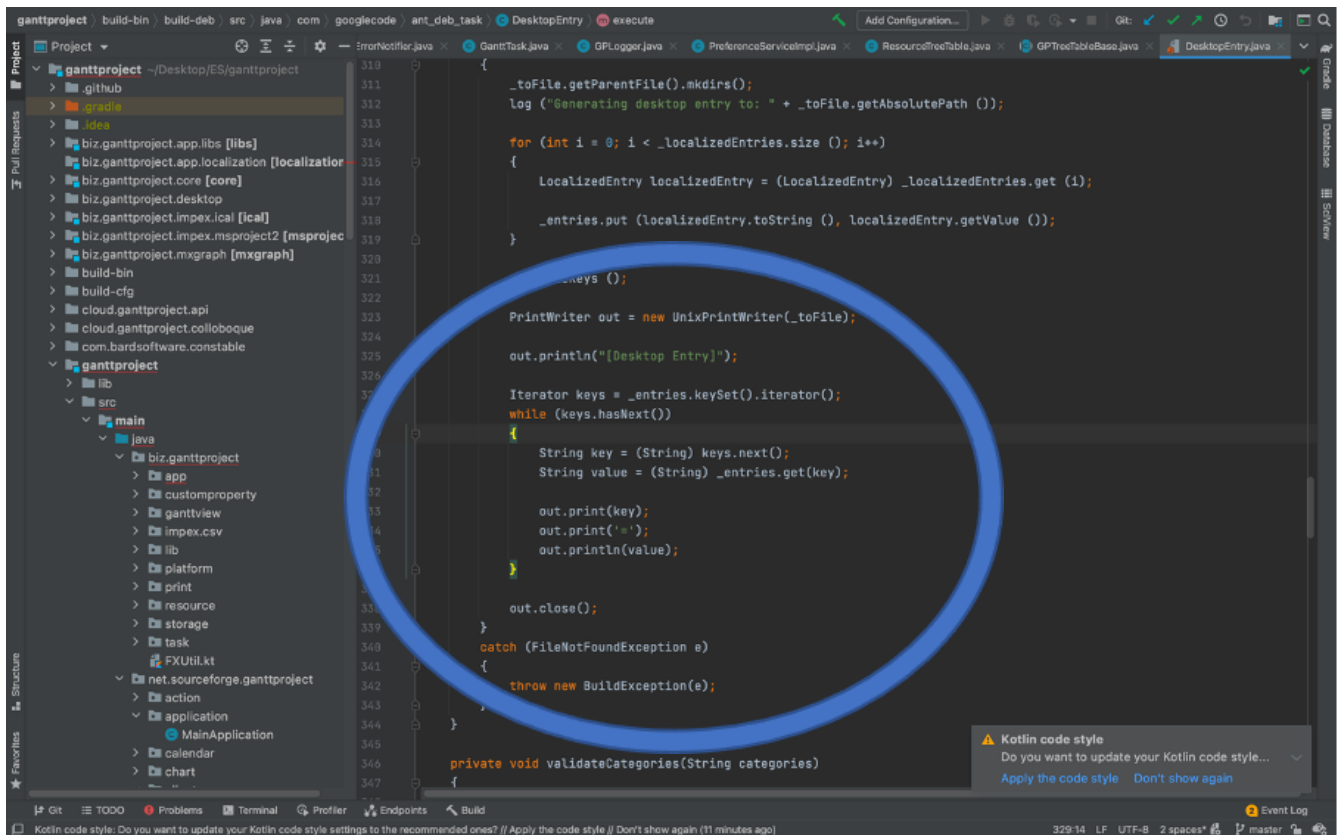
-----> (UIFacadeImpl.java) <-----

Reviews

Guilherme Abrantes 60971

Review do pattern Façade , temos aqui um padrão bastante explícito e muito bem encontrado que não levanta dúvidas, claramente aquela classe é uma façade que vai tornar as subclasses mais fáceis de usar

Iterator pattern - Rafael Costa 60441



Este padrão é facilmente identificado pela criação de um `Iterator`, que vai iterar/percorrer um objeto

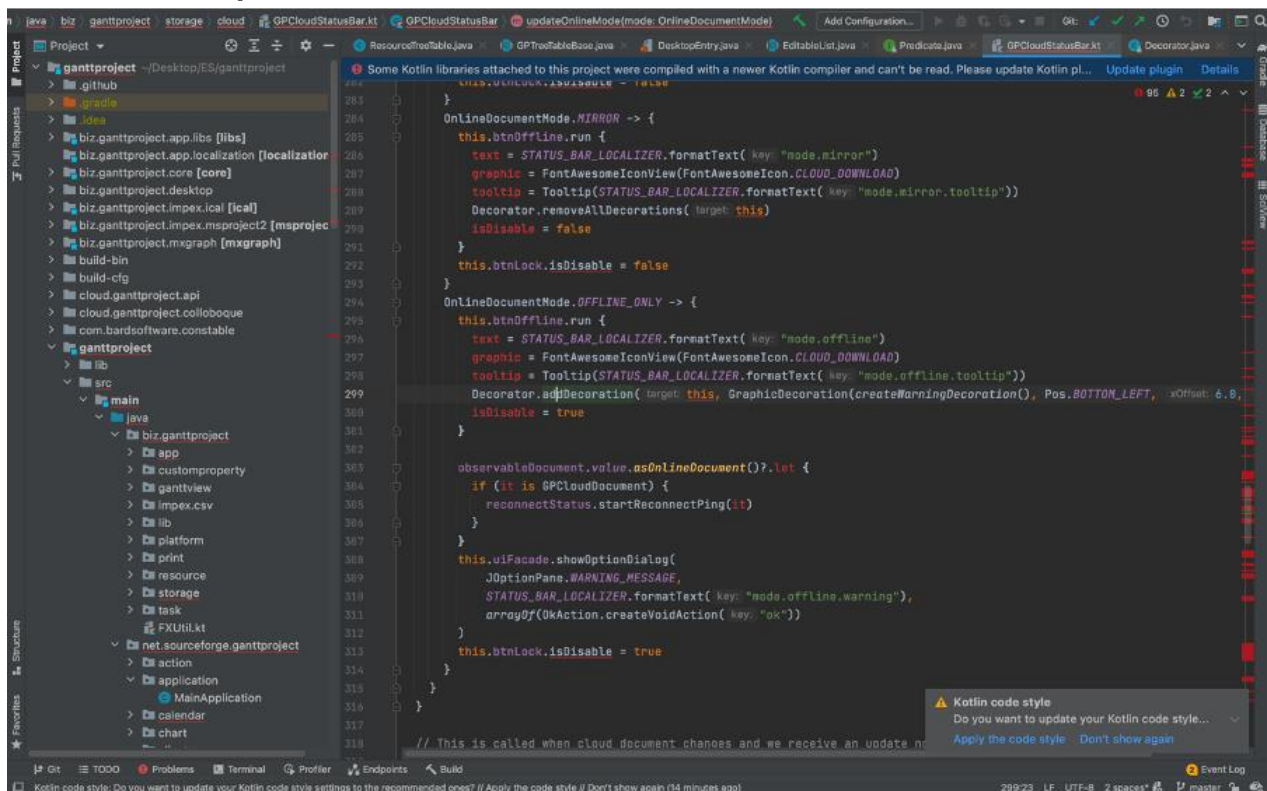
<DesktopEntry.java>

Reviews

Guilherme Abrantes 60971

Review do pattern `Iterator`, vemos aqui um padrão bem identificado que não levanta grandes dúvidas pela criação de um `iterator` que fornece uma maneira de percorrer uma coleção de objetos sem expor sua implementação.

Decorator pattern - Rafael Costa 60441



Este padrão permite anexar novos comportamentos, funcionalidades ou estados extra a um objeto em tempo de execução colocando esses objetos dentro de objetos especiais que contêm os comportamentos.

<GPCloudStatusBar.kt>

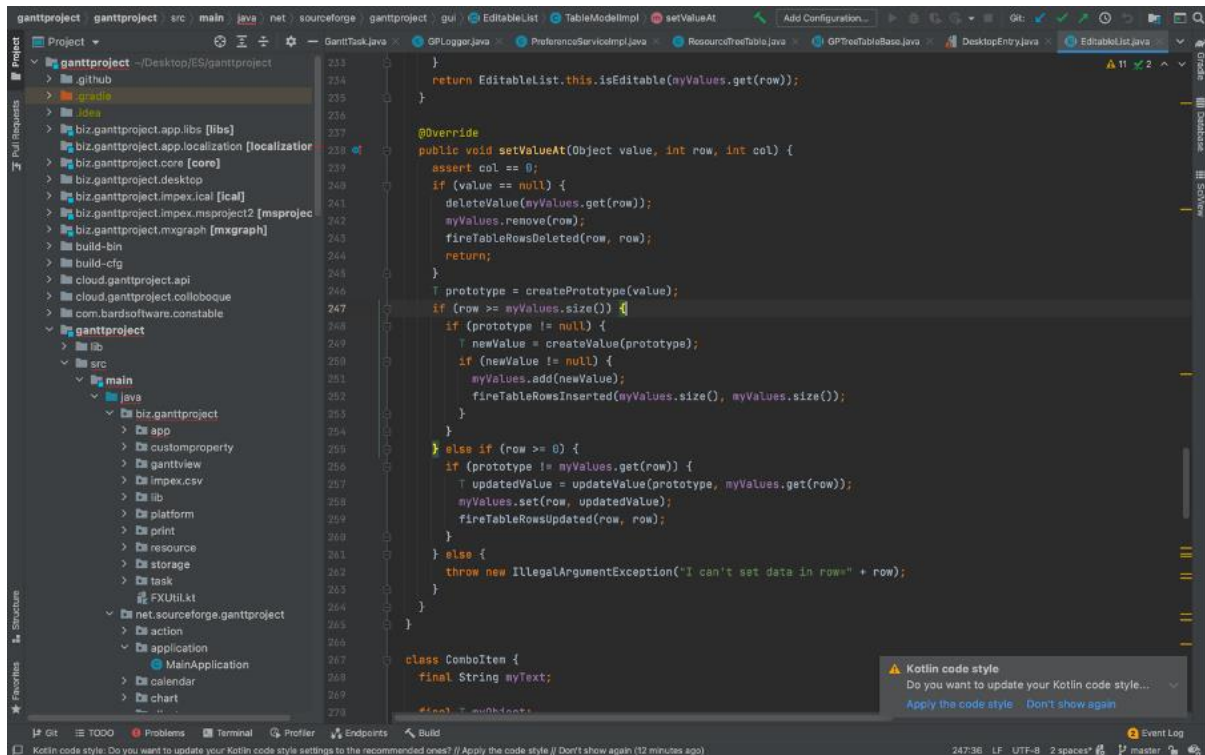
Reviews

Miguel Agostinho 60677

Review do pattern nº2, aqui podemos observar um padrão "Decorator" bem identificado pois, podemos ver claramente o objeto "this"

a ser enviado com o método addDecoration para obter um novo comportamento.

Prototype Pattern - Rafael Costa 60441



Através do prototype podemos criar uma cópia dum objeto que nele seja passado.

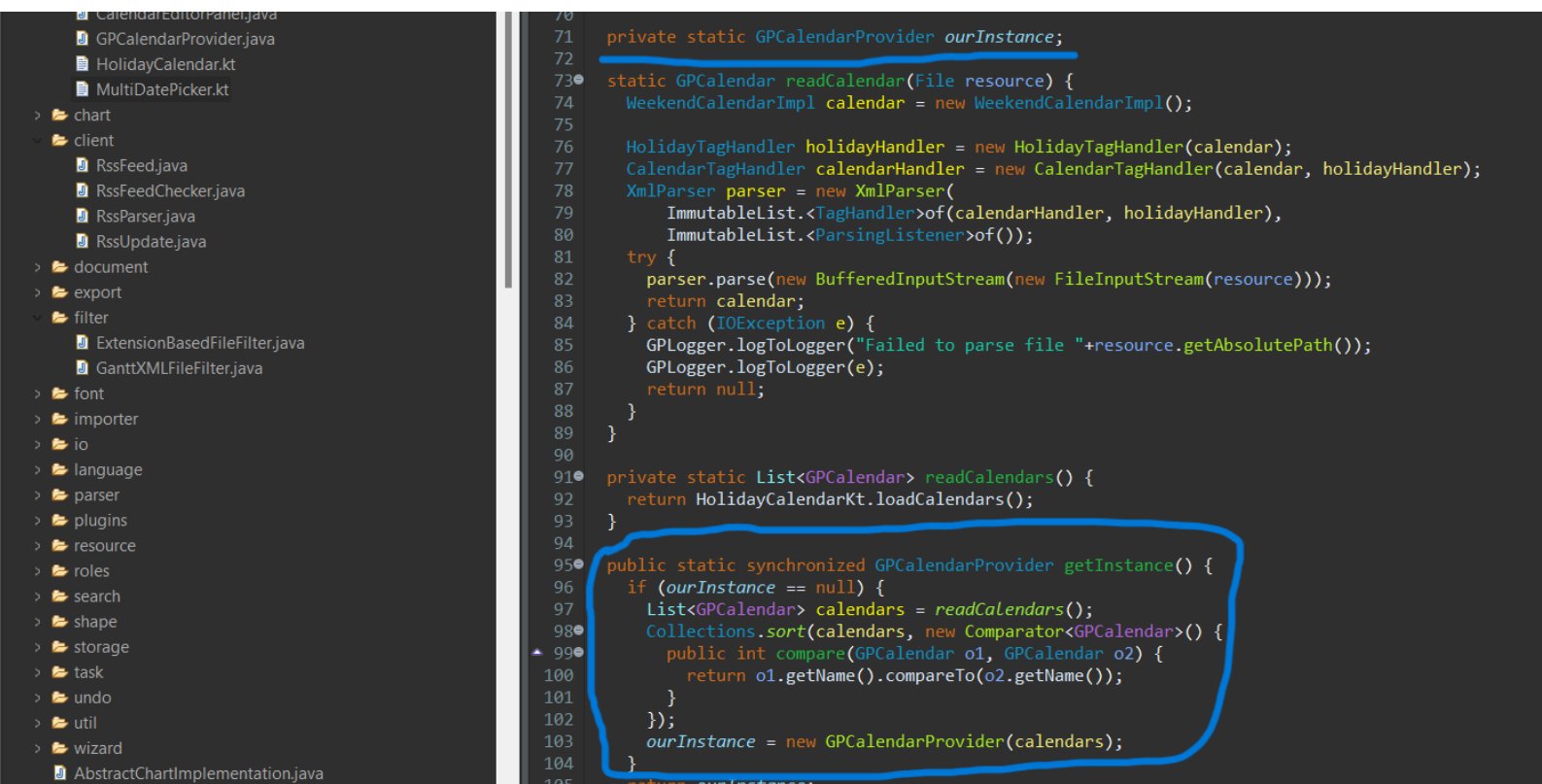
<EditableList.java>

Reviews

Daniel Eugénio 59797

Review do pattern nº3 (Prototype), concordo com o colega, é evidente o uso deste padrão, mais concretamente na linha 246. Podemos ainda ver o protótipo a ser usado mais à frente, na linha 257.

Singleton Pattern - Guilherme Abrantes 60971



Este padrão é identificável pela criação de uma instância única `OurInstance` e pelo método `public static synchronized` que fornece acesso global a esta instância.

-----> (GPCalenderProvider.java) <-----

Reviews

Rafael Costa 60441

Singleton Pattern: Concordo com a identificação do padrão, mas penso que não é explícito a função do padrão

Memento Pattern - Guilherme Abrantes 60971

```
30 * @author bard
31 */
32 public class RedoAction extends GPAction implements GUndoListener {
33     private final GUndoManager myUndoManager;
34
35     public RedoAction(GUndoManager undoManager) {
36         this(undoManager, IconSize.MENU);
37     }
38
39     private RedoAction(GUndoManager undoManager, IconSize size) {
40         super("redo", size);
41         myUndoManager = undoManager;
42         myUndoManager.addUndoableEditListener(this);
43         setEnabled(myUndoManager.canRedo());
44     }
45
46     @Override
47     public void actionPerformed(ActionEvent e) {
48         if (calledFromAppleScreenMenu(e)) {
49             return;
50         }
51
52         myUndoManager.redo();
53     }
54
55     @Override
56     public void undoableEditHappened(UndoableEditEvent e) {
57         setEnabled(myUndoManager.canRedo());
58         updateTooltip();
59     }
60
61     @Override
62     public void undoOrRedoHappened() {
63         setEnabled(myUndoManager.canRedo());
64         updateTooltip();
65     }
66
67     @Override
68     public void undoReset() {
69         undoOrRedoHappened();
70     }
71
72     @Override
```

Aqui podemos observar uma classe que permite que a ação corrente volte atrás para a ação antiga a partir do método undoReset().

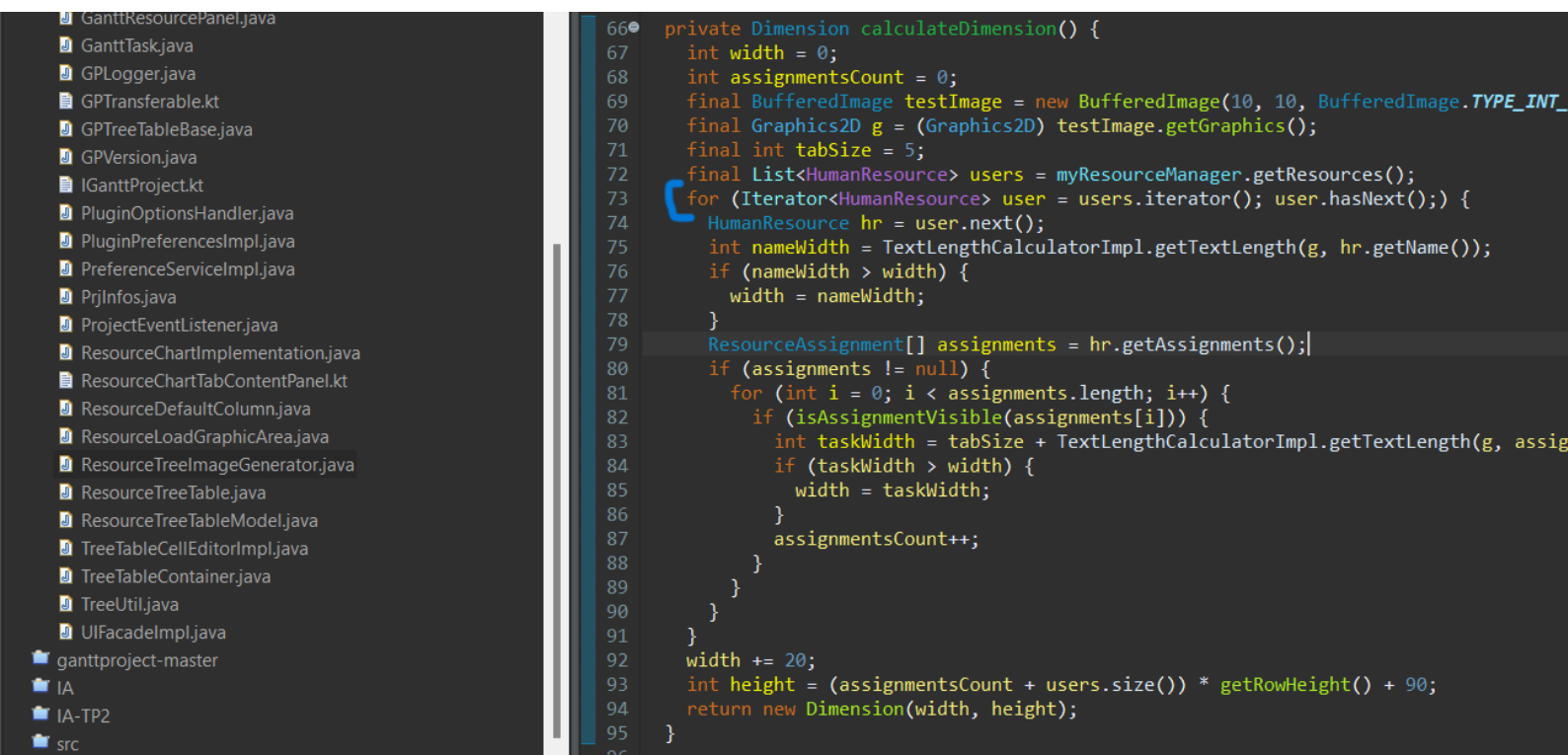
-----> (RedoAction.java) <-----

Reviews

Miguel Agostinho 60677

Review do pattern nº2, este pattern foi bem identificado e o meu colega fez uma boa explicação do mesmo pois é notório o uso de um "undo" para voltar atrás.

Iterator Pattern - Guilherme Abrantes 60971



Este padrão é facilmente identificado pela criação de um objeto do tipo Iterator. É acompanhado com um ciclo for para percorrer os elementos da lista.

-----> (ResourceTreeImageGenerator.java) <-----

Reviews

Francisco Silveira 60816

Concordo com o que o colega disse. Facilmente identificamos um padrão pela criação de um iterador.

Template Method Pattern - Francisco Silveira 60816

(ShortDateFormatOption.java & DefaultStringOption.java)

```
@Override  
public void loadPersistentValue(String value) {  
    super.loadPersistentValue(value);  
    GanttLanguage.getInstance().setShortDateFormat(myDateFormat);  
}
```

```
@Override  
public void loadPersistentValue(String value) { setValue(value); }  
}
```

Neste padrão podemos observar que existe um “override” do método “loadPersistentValue”.

Isto permite modificar ou fazer algumas alterações ao método da superclasse pois as classes que estendem a super classe podem por exemplo ter uma análise e processamento de dados idêntica mas lidar com os vários formatos de dados de maneira diferente.

Reviews

Rafael Costa 60441

Sobre o pattern nº1, padrão bem identificado.

Iterator pattern - Francisco Silveira 60816

(CustumColumnsStorage.java)

```
public void addCustomColumnsListener(CustomPropertyListener listener) { myListeners.add(l  
private void fireCustomColumnsChange(CustomPropertyEvent event) {  
    Iterator<CustomPropertyListener> it = myListeners.iterator();  
    while (it.hasNext()) {  
        CustomPropertyListener listener = it.next();  
        listener.customPropertyChange(event);  
    }  
}
```

Neste pedaço de código conseguimos facilmente identificar a criação de um iterador. Estes são utilizados para percorrer uma lista ou outro tipo de estrutura de dados com maior eficiência.

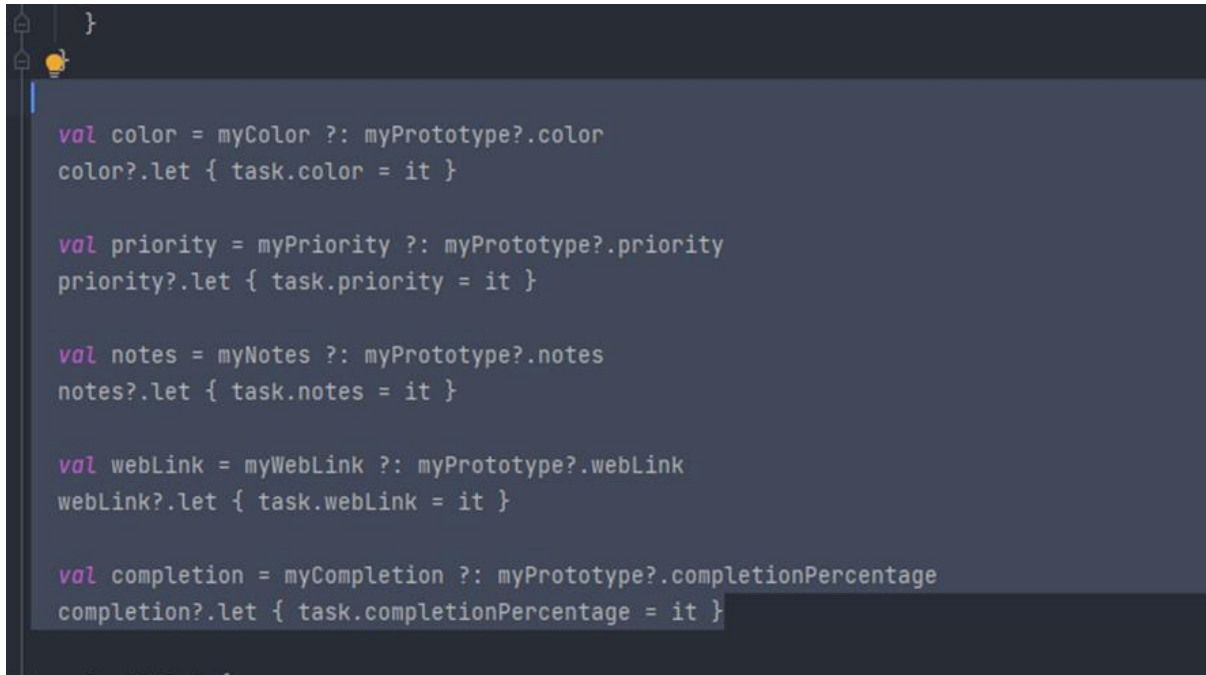
Reviews

Daniel Eugénio 59797

Sobre o pattern nº2 (iterador), concordo com o padrão identificado, no entanto, a meu ver, o iterador serve mais o propósito de percorrer uma estrutura de dados sem mexer na estrutura em si, algo que não acontece com um ciclo for ou while, e não tanto com a eficiência.

Prototype pattern - Francisco Silveira 60816

(taskManagerImpl.kt)

A screenshot of a code editor showing Kotlin code for the Prototype pattern. The code is in a dark-themed editor with a light blue cursor. The code defines several properties (color, priority, notes, webLink, completion) and assigns them to a task object using the 'let' function. The properties are retrieved from either a direct variable or a prototype object (myPrototype) using the Elvis operator (?:).

```
}  
  
val color = myColor ?: myPrototype?.color  
color?.let { task.color = it }  
  
val priority = myPriority ?: myPrototype?.priority  
priority?.let { task.priority = it }  
  
val notes = myNotes ?: myPrototype?.notes  
notes?.let { task.notes = it }  
  
val webLink = myWebLink ?: myPrototype?.webLink  
webLink?.let { task.webLink = it }  
  
val completion = myCompletion ?: myPrototype?.completionPercentage  
completion?.let { task.completionPercentage = it }
```

Aqui podemos ver que é utilizado um protótipo de um objeto já existente para ser comparado com outros valores, ou seja, utilizamos a cópia de um objeto para podermos trabalhar nele à vontade.

Isto permite que não seja possível alterar o objeto inicial criando uma maior segurança no nosso código e dar mais liberdade ao programador pois este pode fazer os testes e alterações que ache necessário para o seu programa no protótipo.

Reviews

Rafael Costa 60441

Sobre o pattern nº3, concordo na plenitude com o exposto acerca do padrão prototype, diria completo e explícito

Metrics

Complexity Metrics Miguel Agostinho (60677)

Complexity metrics	quarta 30 nov. 2022 10:57:30 WET		
Package	v(G)avg	v(G)tot	
net.sourceforge.ganttproject	1,77	1 949	
net.sourceforge.ganttproject.action	1,89	123	
net.sourceforge.ganttproject.action.edit	1,45	61	
net.sourceforge.ganttproject.action.help	1,43	20	
net.sourceforge.ganttproject.action.project	1,48	86	
net.sourceforge.ganttproject.action.resource	1,47	78	
net.sourceforge.ganttproject.action.scroll	1,38	18	
net.sourceforge.ganttproject.action.task	1,59	126	
net.sourceforge.ganttproject.action.view	1,54	20	
net.sourceforge.ganttproject.action.zoom	1,18	13	
net.sourceforge.ganttproject.application	2	2	
net.sourceforge.ganttproject.calendar	2,02	87	
net.sourceforge.ganttproject.chart	1,64	535	
net.sourceforge.ganttproject.chart.export	1,2	18	
net.sourceforge.ganttproject.chart.gantt	2,03	138	
net.sourceforge.ganttproject.chart.item	1	10	
net.sourceforge.ganttproject.chart.mouse	1,72	110	
net.sourceforge.ganttproject.chart.overview	1,7	63	
net.sourceforge.ganttproject.client	2,71	57	
net.sourceforge.ganttproject.document	1,51	214	
net.sourceforge.ganttproject.document.webdav	2,36	337	
net.sourceforge.ganttproject.excel	2	12	
net.sourceforge.ganttproject.export	1,75	210	
net.sourceforge.ganttproject.filter	1,67	10	
net.sourceforge.ganttproject.gui	1,95	805	
net.sourceforge.ganttproject.gui.about	1,33	12	
net.sourceforge.ganttproject.gui.options	1,95	332	
net.sourceforge.ganttproject.gui.options.model	12	12	
net.sourceforge.ganttproject.gui.projectwizard	1,5	93	
net.sourceforge.ganttproject.gui.scrolling	1,5	9	
net.sourceforge.ganttproject.gui.tableView	2	54	
net.sourceforge.ganttproject.gui.taskproperties	2,1	172	
net.sourceforge.ganttproject.gui.view	1,37	26	
net.sourceforge.ganttproject.gui.window	1,14	8	
net.sourceforge.ganttproject.gui.zoom	1,42	27	
net.sourceforge.ganttproject.importer	1,3	86	
net.sourceforge.ganttproject.io	2,34	208	
net.sourceforge.ganttproject.language	1,93	145	
net.sourceforge.ganttproject.parser	2,11	260	
net.sourceforge.ganttproject.plugins	3	9	
net.sourceforge.ganttproject.print	2,32	72	
net.sourceforge.ganttproject.resource	1,6	217	
net.sourceforge.ganttproject.roles	1,57	69	
net.sourceforge.ganttproject.search	1,59	59	
net.sourceforge.ganttproject.shape	1,6	8	
net.sourceforge.ganttproject.task	1,74	794	
net.sourceforge.ganttproject.task.algorithm	2,37	406	
net.sourceforge.ganttproject.task.dependency	1,6	147	
net.sourceforge.ganttproject.task.dependency.constraint	1,68	52	
net.sourceforge.ganttproject.task.event	1	24	
net.sourceforge.ganttproject.task.hierarchy	2,22	20	
net.sourceforge.ganttproject.undo	1,4	35	
net.sourceforge.ganttproject.util	3,36	84	
net.sourceforge.ganttproject.util.collect	1	4	
net.sourceforge.ganttproject.wizard	1,81	76	
project	1,81	8 622	

Obs: Para esta métrica optamos por agrupá-la por packages pois, caso contrário ficaria muito extenso.

Cyclomatic Complexity - A Cyclomatic Complexity consiste no número de decisões mais 1, sendo este número de decisões if, if ... else, switch, for loop, while loop

v(G)avg - A Average Cyclomatic Complexity é a complexidade ciclomática por função de um ficheiro/package. Portanto, esta é a soma das complexidades ciclomáticas de todas as funções a dividir pelo número de funções presentes nesse mesmo ficheiro/package.

v(G)tot - A Total Cyclomatic Complexity é a soma das complexidades ciclomáticas de funções presentes num ficheiro/package.

Pontos problemáticos:

Nesta métrica podemos observar que na package

net.sourceforge.ganttproject.gui.options.model obtemos um valor extremo de Cyclomatic Complexity sendo esse valor 12 para **v(G)avg** ou seja, a média da complexidade de cada função nesta package é 12. Sendo este um valor elevado deveríamos rever o código desta package para tentar reduzir esta mesma complexidade pois, funções com grande cyclomatic complexity são difíceis de entender e também de obter cobertura total do código em testes unitários.

Por outro lado encontramos também outros extremos como por exemplo na package net.sourceforge.ganttproject.gui pois conseguimos observar que o valor de **v(G)tot** é 805 o que significa que a Cyclomatic Complexity tem um valor total de 805 nesta package mas, sendo o valor de **v(G)avg** baixo quando comparado com outras packages.

Relação com code smells:

Como explicado acima, o valor desta métrica depende do número de decisões, portanto, sempre que temos um número elevado isto pode significar que podemos encontrar uma grande sequência de if's o que representará um code smell "Long Method" (Um método que contém muitas linhas de código).

Reviews

Francisco Silveira 60816

Primeiramente, penso que o colega tenha feito um excelente trabalho na explicação de cada parâmetro, pois para além de explicar o significado de cada um ainda deu breves exemplos bastante esclarecedores. Concordo com os code smells do meu colega pois com esta métrica conseguimos facilmente identificar "Long Method"s assim como o excesso de if's.

MARTIN PACKAGING METRICS

RAFAEL COSTA (60441)

Martin packaging metrics					
Package	A	Ca	Ce	D	I
net.sourceforge.ganttproject	0,17	0	0	0,83	1
net.sourceforge.ganttproject.action	0,33	0	0	0,67	1
net.sourceforge.ganttproject.action.edit	0	0	0	1	1
net.sourceforge.ganttproject.action.help	0	0	0	1	1
net.sourceforge.ganttproject.action.project	0,07	0	0	0,93	1
net.sourceforge.ganttproject.action.resource	0,1	0	0	0,9	1
net.sourceforge.ganttproject.action.scroll	0	0	0	1	1
net.sourceforge.ganttproject.action.task	0,2	0	0	0,8	1
net.sourceforge.ganttproject.action.view	0	0	0	1	1
net.sourceforge.ganttproject.action.zoom	0	0	0	1	1
net.sourceforge.ganttproject.application	0	0	0	1	1
net.sourceforge.ganttproject.calendar	0,12	0	0	0,88	1
net.sourceforge.ganttproject.chart	0,34	0	0	0,66	1
net.sourceforge.ganttproject.chart.export	0,25	0	0	0,75	1
net.sourceforge.ganttproject.chart.gantt	0	0	0	1	1
net.sourceforge.ganttproject.chart.item	0	0	0	1	1
net.sourceforge.ganttproject.chart.mouse	0,38	0	0	0,62	1
net.sourceforge.ganttproject.chart.overview	0	0	0	1	1
net.sourceforge.ganttproject.client	0	0	0	1	1
net.sourceforge.ganttproject.document	0,37	0	0	0,63	1
net.sourceforge.ganttproject.document.webdav	0,11	0	0	0,89	1
net.sourceforge.ganttproject.excel	0	0	0	1	1
net.sourceforge.ganttproject.export	0,25	0	0	0,75	1
net.sourceforge.ganttproject.filter	0	0	0	1	1
net.sourceforge.ganttproject.font	0	0	0	1	1
net.sourceforge.ganttproject.gui	0,27	0	0	0,73	1
net.sourceforge.ganttproject.gui.about	0	0	0	1	1
net.sourceforge.ganttproject.gui.options	0,22	0	0	0,78	1
net.sourceforge.ganttproject.gui.options.model	1	0	0	0	1
net.sourceforge.ganttproject.gui.projectwizard	0,15	0	0	0,85	1
net.sourceforge.ganttproject.gui.scrolling	0,67	0	0	0,33	1
net.sourceforge.ganttproject.gui.tableView	0	0	0	1	1
net.sourceforge.ganttproject.gui.taskproperties	0,17	0	0	0,83	1
net.sourceforge.ganttproject.gui.view	0,5	0	0	0,5	1
net.sourceforge.ganttproject.gui.window	0	0	0	1	1
net.sourceforge.ganttproject.gui.zoom	0,25	0	0	0,75	1
net.sourceforge.ganttproject.importer	0,27	0	0	0,73	1
net.sourceforge.ganttproject.io	0,06	0	0	0,94	1
net.sourceforge.ganttproject.language	0,29	0	0	0,71	1
net.sourceforge.ganttproject.parser	0,2	0	0	0,8	1
net.sourceforge.ganttproject.plugins	0	0	0	1	1
net.sourceforge.ganttproject.print	0	0	0	1	1
net.sourceforge.ganttproject.resource	0,35	0	0	0,65	1
net.sourceforge.ganttproject.roles	0,4	0	0	0,6	1
net.sourceforge.ganttproject.search	0,33	0	0	0,67	1
net.sourceforge.ganttproject.shape	0	0	0	1	1
net.sourceforge.ganttproject.task	0,33	0	0	0,67	1
net.sourceforge.ganttproject.task.algorithm	0,31	0	0	0,69	1
net.sourceforge.ganttproject.task.dependency	0,42	0	0	0,58	1
net.sourceforge.ganttproject.task.dependency.constrain	0,17	0	0	0,83	1
net.sourceforge.ganttproject.task.event	0,33	0	0	0,67	1
net.sourceforge.ganttproject.task.hierarchy	0	0	0	1	1
net.sourceforge.ganttproject.undo	0,5	0	0	0,5	1
net.sourceforge.ganttproject.util	0,11	0	0	0,89	1
net.sourceforge.ganttproject.util.collect	0	0	0	1	1
net.sourceforge.ganttproject.wizard	0,5	0	0	0,5	1

Acoplamentos aferentes (Afferent couplings - Ca):

O número de classes em outros pacotes que dependem de classes dentro de um pacote é um indicador da responsabilidade do pacote.

Acoplamentos eferentes (Efferent couplings - Ce):

O número de classes em outros pacotes dos quais as classes em um pacote dependem é um indicador da dependência de pacotes externos.

Abstração (A):

A relação entre o número de classes abstratas (e interfaces) e o número total de classes no pacote analisado.

O intervalo para esta métrica varia entre 0 e 1, sendo $A=0$ a indicação de que o pacote é completamente concreto e $A=1$ a indicação de que o pacote é completamente abstrato.

Instabilidade (I):

A relação entre o acoplamento eferente (C_e) e o acoplamento total ($C_e + C_a$) tal que $I = C_e / (C_e + C_a)$.

Esta métrica é um indicador da resiliência do pacote à mudança.

O intervalo para esta métrica varia entre 0 e 1, sendo $I=0$ a indicação de que o pacote é completamente estável e $I=1$ a indicação de que o pacote é completamente instável.

Distância da sequência principal (D):

A distância perpendicular de um pacote da linha idealizada $A + I = 1$. D é calculado como o módulo de $(A + I - 1)$ ou seja $|A + I - 1|$.

Esta métrica é um indicador do equilíbrio do pacote entre abstração e estabilidade. Um pacote diretamente na sequência principal é perfeitamente equilibrado em relação à sua abstração e estabilidade, ou seja, os **pacotes ideais** são completamente **abstratos e estáveis ($I=0, A=1$)** ou completamente **concretos e instáveis ($I=1, A=0$)**.

O intervalo para essa métrica varia entre 0 e 1, com $D=0$ indicando um pacote que coincide com a sequência principal e $D=1$ indicando um pacote que está o mais longe possível da sequência principal.

Pontos problemáticos:

A **instabilidade dos pacotes é visível** em todos os pacotes presentes na tabela acima representada como se pode ver através da coluna **I** e contudo através da coluna **D** a existência de pacotes ideais é elevada e muitos pacotes apresentam valores de D próximos ao que seria o valor idealizado 1, uma vez que apesar de todos os pacotes serem instáveis muitos são concretos, o que faz com que o indicador D seja quase ideal em muitos casos. Apenas no caso do pacote `net.sourceforge.ganttproject.gui.options.model` o valor do indicador D apresentado é 0, ou seja não é ideal.

Code Smells:

Como explicado acima, os valores de **Afferent couplings (C_a)** e **Efferent couplings (C_e)**, são relativos ao número de classes que dependem ou estão dependentes de outras classes de outros pacotes, e daí, encontramos relação com o Code Smell *Inappropriate Intimacy*, uma vez que este é referente ao uso de campos e métodos de outras classes.

Reviews

Daniel Eugénio 59797

Primeiramente, penso que tenha feito um bom resumo explicativo dos parâmetros da métrica de forma concisa. Quanto à análise dos dados, penso que foi acertada, de facto há muitos pacotes quase ideais, com um valor de D perto de 1, sendo poucos os que têm um valor mais baixo. O *Code Smell* é também bem identificado, visto que a *inappropriate intimacy* está relacionada com acoplamento e dependência entre classes, fatores esses que podem ser analisados pelos parâmetros da métrica em causa.

MOOD Metrics Guilherme Abrantes (60971)

MOOD metrics	quarta	30 nov. 2022 10:56:14 WET				
Project	AHF	AIF	CF	MHF	MIF	PF
project	89,63%	68,73%	0,20%	45,86%	0,00%	100,00%

As **mood metrics** são constituídas por 6 métricas diferentes: **MHF**, **AHF**, **MIF**, **AIF**, **PF** e **CF**.

AHF e MHF são métricas relacionadas com a visibilidade.

AHF = 1 - Attributes Visible;

MHF = 1 - Methods Visible;

MHF tem em conta o número de métodos visíveis no nosso projeto podemos ver um MHF de 45,86% o que indica que existem quase tantos métodos públicos como privados o que pode resultar em vários métodos especializados que não podem ser reutilizados (45,86% é um número relativamente alto quando comparado com outros projetos).

AHF tem em conta o número de atributos visíveis, sendo que o nosso projeto possui um AHF de 89,63% acho um número bastante aceitável onde a maioria dos atributos são privados.

MIF e AIF são métricas relacionadas com a herança de métodos.

MIF = inherited methods / total methods available in classes

AIF = inherited attributes / total attributes available in classes

Podemos então constatar um **MIF** de 0 que pode estar a ser causado por as classes que herdam métodos dos pais estarem a redefinir todos os métodos ou então acrescentar novos, este MIF é considerado bastante mau.

O **AIF** deste projeto é de 68,73% o que sugere que bastantes atributos são herdados pelas classes filhas, o que pode não ser o ideal.

PF Polymorphism Factor:

PF = overrides / sum for each class(new methods * descendants)

A **PF** indica o grau de redefinição de métodos durante a herança de classes, podemos então observar um PF de 100% o que indica que nos estamos a dar override a tudo o que condiz com o facto do MIF ser 0%.

CF Coupling Factor:

CF = Actual couplings / Maximum possible couplings

CF mede as copulações que existem, para existir uma população de A para B, A tem de chamar métodos ou variáveis que existem em B, neste projeto este valor é de 0,20% o que indica que quase não existem classes copuladas.

Pontos problemáticos e code smells:

Sobre os pontos problemáticos temos o PF e o MIF que se completam e indicam que grande parte dos métodos são redefinidos, o que leva às classes filhas a não aproveitarem os métodos que herdam dos pais, o que pode levar ao code Smell Divergent Change, como não existem métodos comuns sempre que queremos adicionar algo temos de reescrever todos os métodos que já existem.

Reviews

Rafael Costa 60441

A descrição de cada parâmetro da métrica, penso que tenha sido simples e intuitiva, em relação ao Code Smell, realmente não é fácil de identificar um Code Smell para a métrica em questão, mas talvez como o colega descreveu o code Smell Divergent Change seja o adequado.

Lines of Code Metrics Daniel Eugénio (59797)

Lines of code metrics														
Package	CLOC	CLOC(rec)	JLOC	JLOC(rec)	LOC	LOC(rec)	LOCp	LOCp(rec)	LOCt	LOCt(rec)	NCLOC	NCLOCp	NCLOCp(rec)	NCLOCt
	n/a	8 720	n/a	3 124	n/a	62 376	n/a	62 376	n/a	0	n/a	n/a	53 372	n/a
biz	n/a	97	n/a	61	n/a	1 169	n/a	1 169	n/a	0	n/a	n/a	1 072	n/a
biz.ganttproject	n/a	97	n/a	61	n/a	1 169	n/a	1 169	n/a	0	n/a	n/a	1 072	n/a
biz.ganttproject.impex	n/a	97	n/a	61	n/a	1 169	n/a	1 169	n/a	0	n/a	n/a	1 072	n/a
biz.ganttproject.impex.csv	0	97	61	61	1 169	1 169	1 169	1 169	0	0	1 072	1 072	1 072	0
net	n/a	8 623	n/a	3 063	n/a	61 204	n/a	61 204	n/a	0	n/a	n/a	52 297	n/a
net.sourceforge	n/a	8 623	n/a	3 063	n/a	61 204	n/a	61 204	n/a	0	n/a	n/a	52 297	n/a
net.sourceforge.ganttproject	0	8 623	458	3 063	12 258	61 204	12 258	61 204	0	0	10 993	10 993	52 297	0
net.sourceforge.ganttproject.action	0	1 142	48	151	894	4 687	894	4 687	0	0	676	676	3 545	0
net.sourceforge.ganttproject.action.edit	0	140	10	10	556	556	556	556	0	0	416	416	416	0
net.sourceforge.ganttproject.action.help	0	16	5	5	186	186	186	186	0	0	170	170	170	0
net.sourceforge.ganttproject.action.project	0	229	21	21	786	786	786	786	0	0	557	557	557	0
net.sourceforge.ganttproject.action.resource	0	152	16	16	652	652	652	652	0	0	500	500	500	0
net.sourceforge.ganttproject.action.scroll	0	76	0	0	214	214	214	214	0	0	138	138	138	0
net.sourceforge.ganttproject.action.task	0	221	36	36	1 087	1 087	1 087	1 087	0	0	866	866	866	0
net.sourceforge.ganttproject.action.view	0	45	6	6	182	182	182	182	0	0	137	137	137	0
net.sourceforge.ganttproject.action.zoom	0	45	9	9	130	130	130	130	0	0	85	85	85	0
net.sourceforge.ganttproject.application	0	8	3	3	44	44	44	44	0	0	36	36	36	0
net.sourceforge.ganttproject.calendar	0	31	11	11	593	593	593	593	0	0	563	563	563	0
net.sourceforge.ganttproject.chart	0	998	356	455	3 846	6 638	3 846	6 638	0	0	3 376	3 376	5 654	0
net.sourceforge.ganttproject.chart.export	0	65	0	0	225	225	225	225	0	0	164	164	164	0
net.sourceforge.ganttproject.chart.gantt	0	102	42	42	925	925	925	925	0	0	823	823	823	0
net.sourceforge.ganttproject.chart.item	0	56	27	27	146	146	146	146	0	0	90	90	90	0
net.sourceforge.ganttproject.chart.mouse	0	211	19	19	914	914	914	914	0	0	703	703	703	0
net.sourceforge.ganttproject.chart.overview	0	84	11	11	582	582	582	582	0	0	498	498	498	0
net.sourceforge.ganttproject.client	0	58	5	5	388	388	388	388	0	0	330	330	330	0
net.sourceforge.ganttproject.document	0	425	181	328	1 666	4 115	1 666	4 115	0	0	1 433	1 433	3 367	0
net.sourceforge.ganttproject.document.webdav	0	192	147	147	2 449	2 449	2 449	2 449	0	0	1 934	1 934	1 934	0
net.sourceforge.ganttproject.excel	0	22	2	2	111	111	111	111	0	0	89	89	89	0
net.sourceforge.ganttproject.export	0	241	24	24	1 462	1 462	1 462	1 462	0	0	1 221	1 221	1 221	0
net.sourceforge.ganttproject.filter	0	30	11	11	83	83	83	83	0	0	53	53	53	0
net.sourceforge.ganttproject.font	0	15	5	5	35	35	35	35	0	0	21	21	21	0
net.sourceforge.ganttproject.gui	0	1 569	324	566	6 637	12 408	6 637	12 408	0	0	5 840	5 840	10 845	0
net.sourceforge.ganttproject.gui.about	0	15	0	0	122	122	122	122	0	0	107	107	107	0
net.sourceforge.ganttproject.gui.options	0	359	107	107	2 493	2 676	2 493	2 676	0	0	2 180	2 180	2 321	0
net.sourceforge.ganttproject.gui.options.model	0	42	0	0	183	183	183	183	0	0	141	141	141	0
net.sourceforge.ganttproject.gui.projectwizard	0	125	7	7	770	770	770	770	0	0	645	645	645	0
net.sourceforge.ganttproject.gui.scrolling	0	9	25	25	88	88	88	88	0	0	79	79	79	0
net.sourceforge.ganttproject.gui.tableView	0	24	20	20	418	418	418	418	0	0	394	394	394	0
net.sourceforge.ganttproject.gui.taskproperties	0	136	36	36	1 200	1 200	1 200	1 200	0	0	1 064	1 064	1 064	0
net.sourceforge.ganttproject.gui.view	0	61	16	16	260	260	260	260	0	0	199	199	199	0
net.sourceforge.ganttproject.gui.window	0	15	6	6	73	73	73	73	0	0	58	58	58	0
net.sourceforge.ganttproject.gui.zoom	0	26	25	25	164	164	164	164	0	0	138	138	138	0
net.sourceforge.ganttproject.importer	0	179	19	19	784	784	784	784	0	0	605	605	605	0
net.sourceforge.ganttproject.io	0	256	33	33	1 425	1 425	1 425	1 425	0	0	1 175	1 175	1 175	0
net.sourceforge.ganttproject.language	0	76	186	186	942	942	942	942	0	0	866	866	866	0
net.sourceforge.ganttproject.parser	0	295	83	83	1 687	1 687	1 687	1 687	0	0	1 393	1 393	1 393	0
net.sourceforge.ganttproject.plugins	0	14	5	5	63	63	63	63	0	0	49	49	49	0
net.sourceforge.ganttproject.print	0	109	2	2	808	808	808	808	0	0	699	699	699	0
net.sourceforge.ganttproject.resource	0	158	57	57	1 200	1 200	1 200	1 200	0	0	1 042	1 042	1 042	0
net.sourceforge.ganttproject.roles	0	95	24	24	509	509	509	509	0	0	414	414	414	0
net.sourceforge.ganttproject.search	0	116	26	26	593	593	593	593	0	0	477	477	477	0
net.sourceforge.ganttproject.shape	0	31	3	3	90	90	90	90	0	0	59	59	59	0
net.sourceforge.ganttproject.task	0	1 194	247	443	4 708	8 725	4 708	8 725	0	0	4 223	4 223	7 532	0
net.sourceforge.ganttproject.task.algorithm	0	256	112	112	2 176	2 176	2 176	2 176	0	0	1 921	1 921	1 921	0
net.sourceforge.ganttproject.task.dependency	0	346	47	72	1 065	1 504	1 065	1 504	0	0	804	804	1 158	0
net.sourceforge.ganttproject.task.dependency.constrain	0	85	25	25	439	439	439	439	0	0	354	354	354	0
net.sourceforge.ganttproject.task.event	0	78	12	12	220	220	220	220	0	0	142	142	142	0
net.sourceforge.ganttproject.task.hierarchy	0	29	0	0	117	117	117	117	0	0	88	88	88	0
net.sourceforge.ganttproject.undo	0	60	14	14	284	284	284	284	0	0	224	224	224	0
net.sourceforge.ganttproject.util	0	179	110	113	631	666	631	666	0	0	466	466	487	0
net.sourceforge.ganttproject.util.collect	0	14	3	3	35	35	35	35	0	0	21	21	21	0
net.sourceforge.ganttproject.wizard	0	48	36	36	606	606	606	606	0	0	558	558	558	0
org	n/a	n/a	n/a	n/a	n/a	3	n/a	3	n/a	0	n/a	n/a	3	n/a
org.ganttproject	0	0	0	0	3	3	3	3	0	0	3	3	3	0

Obs: Para esta métrica optamos por agrupá-la por packages pois, caso contrário ficaria muito extenso.

Alguns constituintes da métrica Lines of Code:

LOC - Lines Of Code

- Todas linhas de código, incluindo comentários

NCLOC - Non-Comment Lines of Code

- Linhas de código excluindo os comentários

LOCp - Lines of product code

- Linhas de código “eficaz” (source code). Exclui comentários e linhas em branco.

Pontos problemáticos:

Esta métrica é difícil de avaliar, visto ser mais eficaz quando avaliada ao nível da classe, no entanto, avaliar um projeto desta dimensão à classe demoraria muito tempo, pelo que achámos melhor ver os valores ao nível da package. A nível da package, a package com mais linhas é a package “mãe”, pelo que se justifica o valor de linhas apresentado, por esta ter várias packages em si.

net.sourceforge.ganttproject.gui.GanttTaskPropertiesBean	27	16	466
net.sourceforge.ganttproject.gui.UIUtil	19	4	416
Média de linhas por classe =	73,8726		

Aqui estão dois exemplos, ao nível da classe, de valores que estão fora do normal para a média da aplicação, o que pode implicar dificuldades na leitura e interpretação destas classes e que podiam ser divididas em classes mais pequenas.

Code Smells

Apesar de difícil de avaliar, esta métrica pode ser extremamente útil para identificar code smells como Large Class ou Duplicate Code, no caso de uma classe se destacar das outras no número de linhas. Outro Code Smell é o No Comments, detetável em classes onde o número de linhas seja igual, independentemente do parâmetro ser LOC ou NCLOC.

Em relação à análise do meu colega, concordo que esta métrica é difícil de ser avaliada no formato de packages, em relação aos code smells concordo que possa ajudar a identificar os code smells referidos mas também o code smell “Comments” que é identificado quando uma classe tem excesso de comentários.

Reviews

Miguel Agostinho 60677

Em relação à análise do meu colega, concordo que esta métrica é difícil de ser avaliada no formato de packages, em relação aos code smells concordo que possa ajudar a identificar os code smells referidos mas também o code smell “Comments” que é identificado quando uma classe tem excesso de comentários.

Dependency Metrics Francisco Silveira (60816)

Package	Cyclic	PDcy	PDpt	PDpt*
net.sourceforge.ganttproject	50	53	52	64
net.sourceforge.ganttproject.action	50	8	21	64
net.sourceforge.ganttproject.action.edit	50	12	2	64
net.sourceforge.ganttproject.action.help	50	8	1	64
net.sourceforge.ganttproject.action.project	50	13	2	64
net.sourceforge.ganttproject.action.resource	50	10	1	64
net.sourceforge.ganttproject.action.scroll	50	7	1	64
net.sourceforge.ganttproject.action.task	50	10	2	64
net.sourceforge.ganttproject.action.view	50	8	1	64
net.sourceforge.ganttproject.action.zoom	50	2	5	64
net.sourceforge.ganttproject.application	0	1	0	1
net.sourceforge.ganttproject.calendar	50	12	2	64
net.sourceforge.ganttproject.chart	50	27	22	64
net.sourceforge.ganttproject.chart.export	50	5	5	64
net.sourceforge.ganttproject.chart.gantt	50	18	1	64
net.sourceforge.ganttproject.chart.item	50	1	4	64
net.sourceforge.ganttproject.chart.mouse	50	18	2	64
net.sourceforge.ganttproject.chart.overview	50	7	1	64
net.sourceforge.ganttproject.client	50	7	1	64
net.sourceforge.ganttproject.document	50	17	12	64
net.sourceforge.ganttproject.document.webdav	50	8	1	64
net.sourceforge.ganttproject.excel	50	1	1	64
net.sourceforge.ganttproject.export	50	17	7	64
net.sourceforge.ganttproject.filter	0	0	3	65
net.sourceforge.ganttproject.font	0	0	3	65
net.sourceforge.ganttproject.gui	50	30	39	64
net.sourceforge.ganttproject.gui.about	50	4	2	64
net.sourceforge.ganttproject.gui.options	50	19	19	64
net.sourceforge.ganttproject.gui.options.model	50	4	8	64
net.sourceforge.ganttproject.gui.projectwizard	50	8	4	64
net.sourceforge.ganttproject.gui.scrolling	0	1	6	65
net.sourceforge.ganttproject.gui.tableView	50	7	1	64
net.sourceforge.ganttproject.gui.taskproperties	50	17	2	64
net.sourceforge.ganttproject.gui.view	50	7	4	64
net.sourceforge.ganttproject.gui.window	50	1	1	64
net.sourceforge.ganttproject.gui.zoom	0	1	9	65
net.sourceforge.ganttproject.importer	50	20	6	64
net.sourceforge.ganttproject.io	50	19	9	64
net.sourceforge.ganttproject.language	50	5	35	64
net.sourceforge.ganttproject.parser	50	16	7	64
net.sourceforge.ganttproject.plugins	50	3	6	64
net.sourceforge.ganttproject.print	50	8	2	64
net.sourceforge.ganttproject.resource	50	10	21	64
net.sourceforge.ganttproject.roles	50	1	16	64
net.sourceforge.ganttproject.search	50	7	2	64
net.sourceforge.ganttproject.shape	0	0	1	65
net.sourceforge.ganttproject.task	50	22	36	64
net.sourceforge.ganttproject.task.algorithm	50	12	14	64
net.sourceforge.ganttproject.task.dependency	50	8	20	64
net.sourceforge.ganttproject.task.dependency.constraint	50	7	10	64
net.sourceforge.ganttproject.task.event	50	3	4	64
net.sourceforge.ganttproject.task.hierarchy	50	1	1	64
net.sourceforge.ganttproject.undo	50	6	13	64
net.sourceforge.ganttproject.util	50	3	15	64
net.sourceforge.ganttproject.util.collect	0	0	12	65
net.sourceforge.ganttproject.wizard	50	5	3	64

Dependency Metrics

Uma métrica dependente é uma métrica em que suas variáveis de influência são explicitadas. Um exemplo é a escalabilidade da taxa de transferência em relação ao número de clientes que acessam uma conexão de dados. Aqui, uma métrica (a taxa de transferência) é fornecida em dependência de outra variável (o número de clientes).

Cyclic:

- número de dependências cyclic por package. Esta ocorre quando um package A depende de outro package B, e o B também depende do A.

PDcy:

- número de dependências por package, ou seja, por quantos packages está dependente o package em questão.

PDpt:

- número de packages dependentes, ou seja, quantos packages estão dependentes do package em questão.

PDpt*:

- número de packages dependentes transitivamente. Considerando três packages: A, B, C. Se C for dependente de B e B for dependente de A então C é uma dependência transitiva para A.

Pontos Problemáticos:

Como podemos ver nas colunas "Cyclic" e "PDpt*" Os valores apresentados são praticamente os mesmos, ou seja, não conseguimos observar algo em concreto apenas que estes packages dependem quase todos uns dos outros.

Code Smells:

Esta métrica pode ajudar-nos a identificar vários tipos de code smells como por exemplo o Shotgun Surgery no caso de querermos alterar algo simples no programa mas temos que alterar várias classes para o fazermos. Outro code smell que podemos identificar é o Feature Envy no caso de existir um método que está mais preocupado em manipular dados de outra classe em vez da sua. Estes code smells podem ser identificados observando as suas dependências. No caso do Feature Envy podemos observá-lo se existir uma classe com poucas dependências e muitos dependentes.

Atenção estes exemplos podem ocorrer (ou ocorrer de outra maneira dependendo do programa em questão) e não existir nenhum code smell apenas são algumas maneiras de podermos identificá-los.

Reviews

Guilherme Abrantes 60971

Em relação à análise do meu colega, acho que o colega conseguiu descrever bem e explicar o que consiste cada ponto desta métrica porém podia ter pegado num exemplo em específico e ter dado a sua análise, em relação aos code smells concordo com o que o meu colega disse é difícil identificar um em específico mas certos valores de dependências podem levar a certos smells.

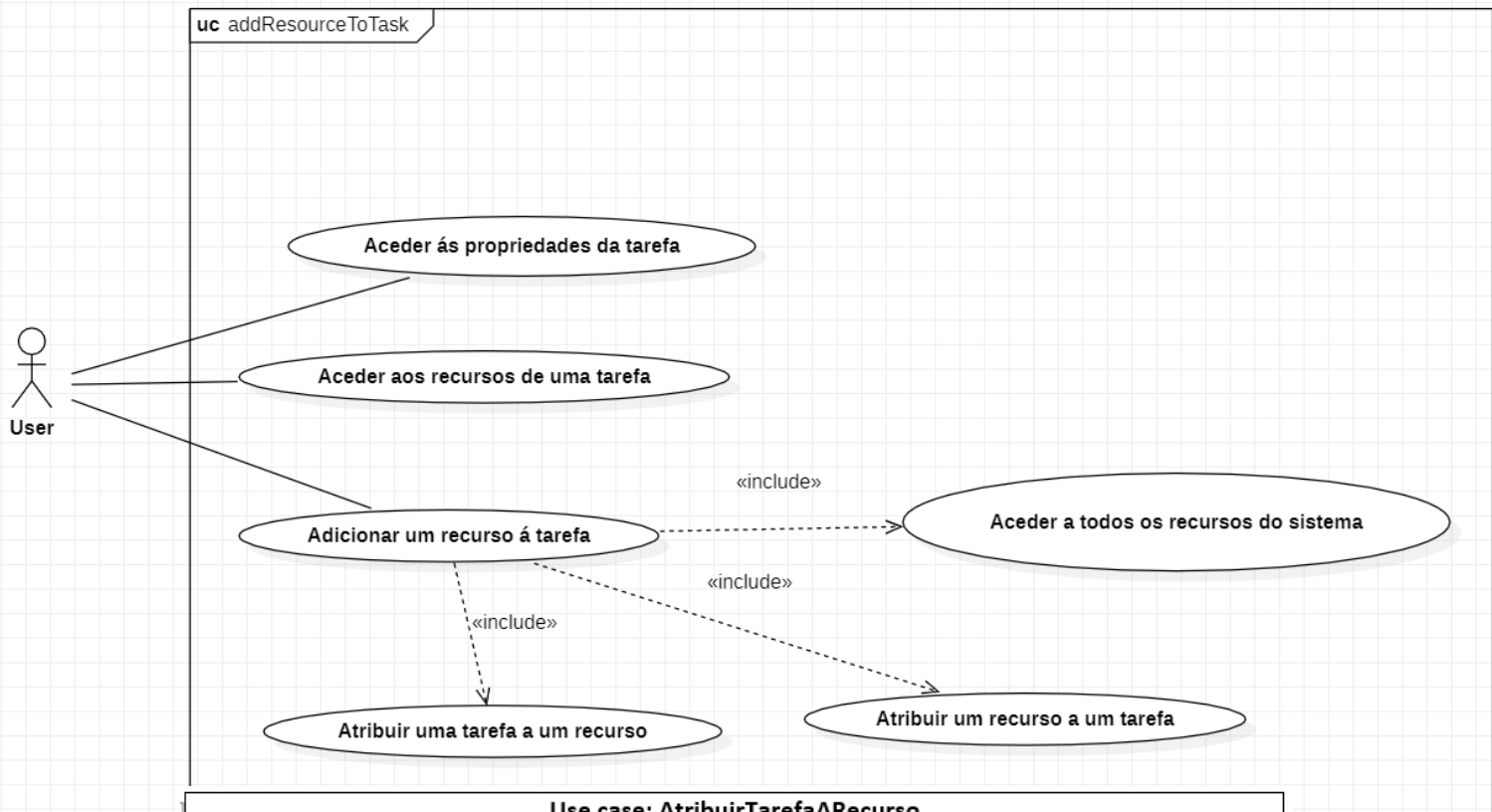
A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the text 'SE 22/23'.

SE 22/23

2ª Fase

Use Cases

Use Case Guilherme Abrantes (60971)



Use case: AtribuirTarefaARecurso

ID: 1

Description: Inserir um recurso a uma tarefa

Primary Actors: Utilizador

Secondary Actors: Não existem

Preconditions: Existir pelo menos uma tarefa no sistema
Existir pelo menos um recurso no sistema

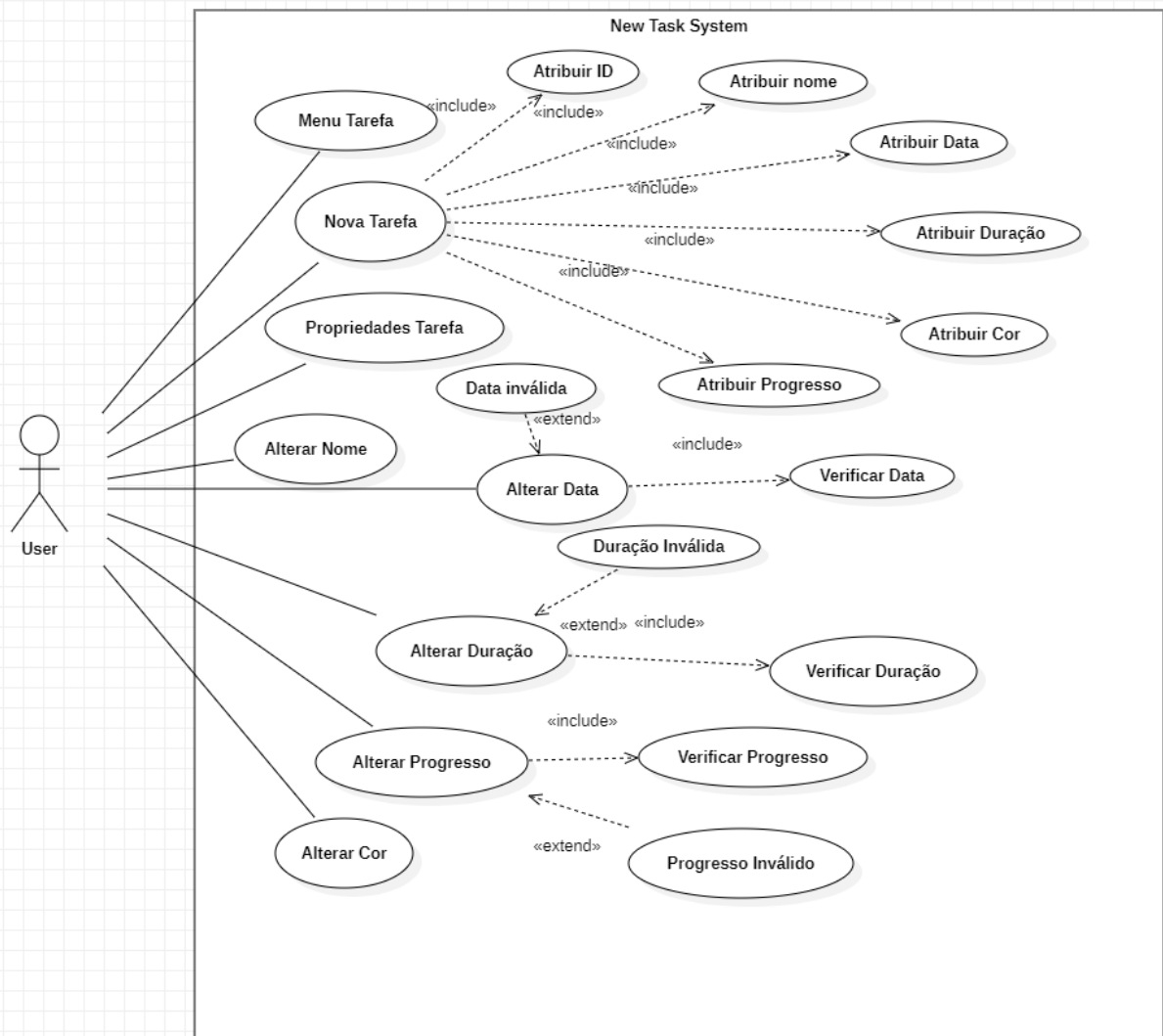
Main Flow:

1. O use case começa quando o utilizador seleciona escolhe uma tarefa e seleciona a opção propriedades da tarefa.
2. Utilizador seleciona a secção dos recursos.
3. O sistema exibe todos os recursos que já estão associadas à tarefa.
4. O utilizador consegue clicar num botão adicionar .
5. O sistema cria uma linha na tabela dos recursos e providência uma caixa de seleção onde estão presentes todos os recursos existentes no sistema identificados pelo nome.
6. O utilizador escolhe o recurso que quer adicionar à tarefa.

Postconditions: O recurso possui a tarefa onde foi adicionado no diagrama de recursos

Alternative flows:

Use Case Miguel Agostinho (60677)



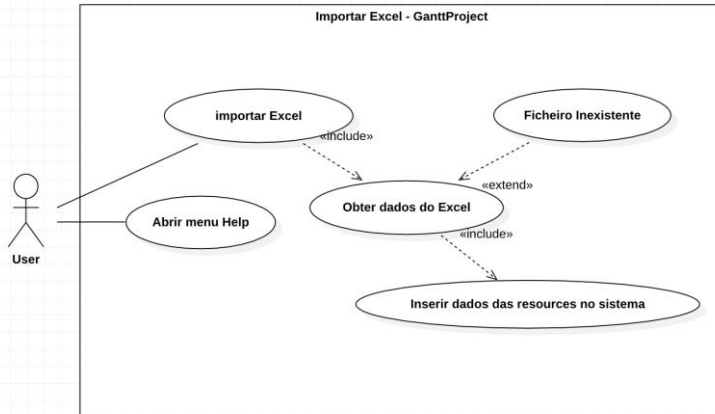
Use case: CriarNovaTarefa
ID: 1
Description: Inserção de uma nova tarefa no sistema
Primary Actors: Utilizador
Secondary Actors: Não existem
Preconditions: Não existem
Main Flow: <ol style="list-style-type: none"> 1. O use case começa quando o utilizador seleciona a opção de inserir uma “nova tarefa” no sistema. 2. Sistema cria uma nova tarefa. 3. Utilizador seleciona a opção “propriedades da tarefa”. 4. Utilizador pode preencher os campos nome, data, duração, progresso e cor. <ol style="list-style-type: none"> a. O utilizador pode optar por preencher apenas alguns ou até nenhum dos campos. 5. O sistema regista a informação dada pelo utilizador e altera a tarefa.
Postconditions: Uma nova tarefa foi criada
Alternative flows: DataInválida DuraçãoInválida ProgressoInválido

Alternative flow: CriarNovaTarefa.DataInválida
ID: 1.1
Description: O sistema corrige a data inserida pelo utilizador se não existir.
Primary Actors: Utilizador
Secondary Actors: Não existem
Preconditions: O utilizador inseriu uma data que não existe
Alternative Flow: <ol style="list-style-type: none"> 1. O alternative flow começa depois da etapa 4 do main flow. 2. O sistema corrige a data inválida inserida pelo utilizador.
Postconditions: Data corrigida

Alternative flow: CriarNovaTarefa.DuraçãoInválida
ID: 1.2
Description: O sistema repõe a duração se inválida
Primary Actors: Utilizador
Secondary Actors: Não existem
Preconditions: O utilizador inseriu uma duração inválida
Alternative Flow: <ol style="list-style-type: none"> 1. O alternative flow começa depois da etapa 4 do main flow. 2. O sistema repõe a duração inválida inserida pelo utilizador.
Postconditions: Duração reposta

Alternative flow: CriarNovaTarefa.ProgressoInválido
ID: 1.3
Description: O sistema ignora um progresso inválido.
Primary Actors: Utilizador
Secondary Actors: Não existem
Preconditions: O utilizador inseriu um progresso inválido
Alternative Flow: <ol style="list-style-type: none"> 1. O alternative flow começa depois da etapa 4 do main flow. 2. O sistema ignora o progresso inválido inserido pelo utilizador.
Postconditions: Progresso corrigido

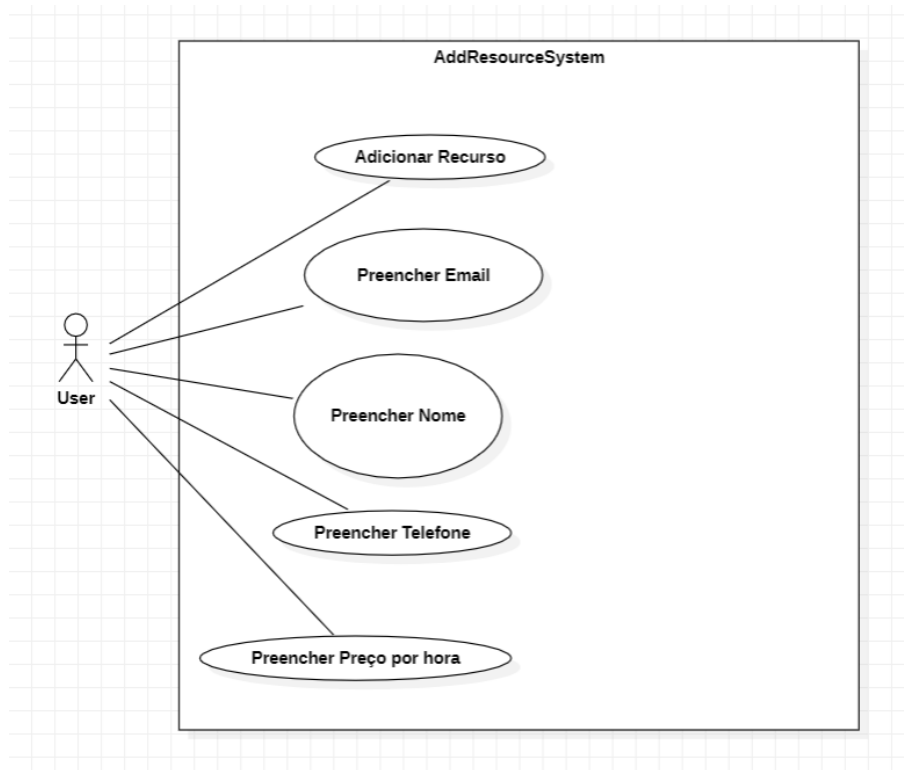
Use Case Rafael Costa (60441)



Use case: ImportarExcel
ID: 1
Description: Importar recursos através de um ficheiro Excel
Primary Actors: Utilizador
Secondary Actors: Não existem
Preconditions: Existir um ficheiro Excel guardado numa determinada pasta do sistema
Main Flow: <ol style="list-style-type: none">1. O use case começa quando o utilizador seleciona a o Menu Help<ol style="list-style-type: none">a. O utilizar escolhe a opção importar Excel2. O Sistema vai ler o ficheiro Excel guardado numa determinada pasta do sistema.3. O Sistema decompõe cada linha da tabela Excel em várias variáveis.<ol style="list-style-type: none">a. A cada linha decompõe o sistema aglomera tudo e regista um novo recurso.4. O sistema regista a informação dada pelo utilizador e altera a tarefa.
Postconditions: recursos foram adicionados ao sistema
Alternative flows: FicheiroInexistente

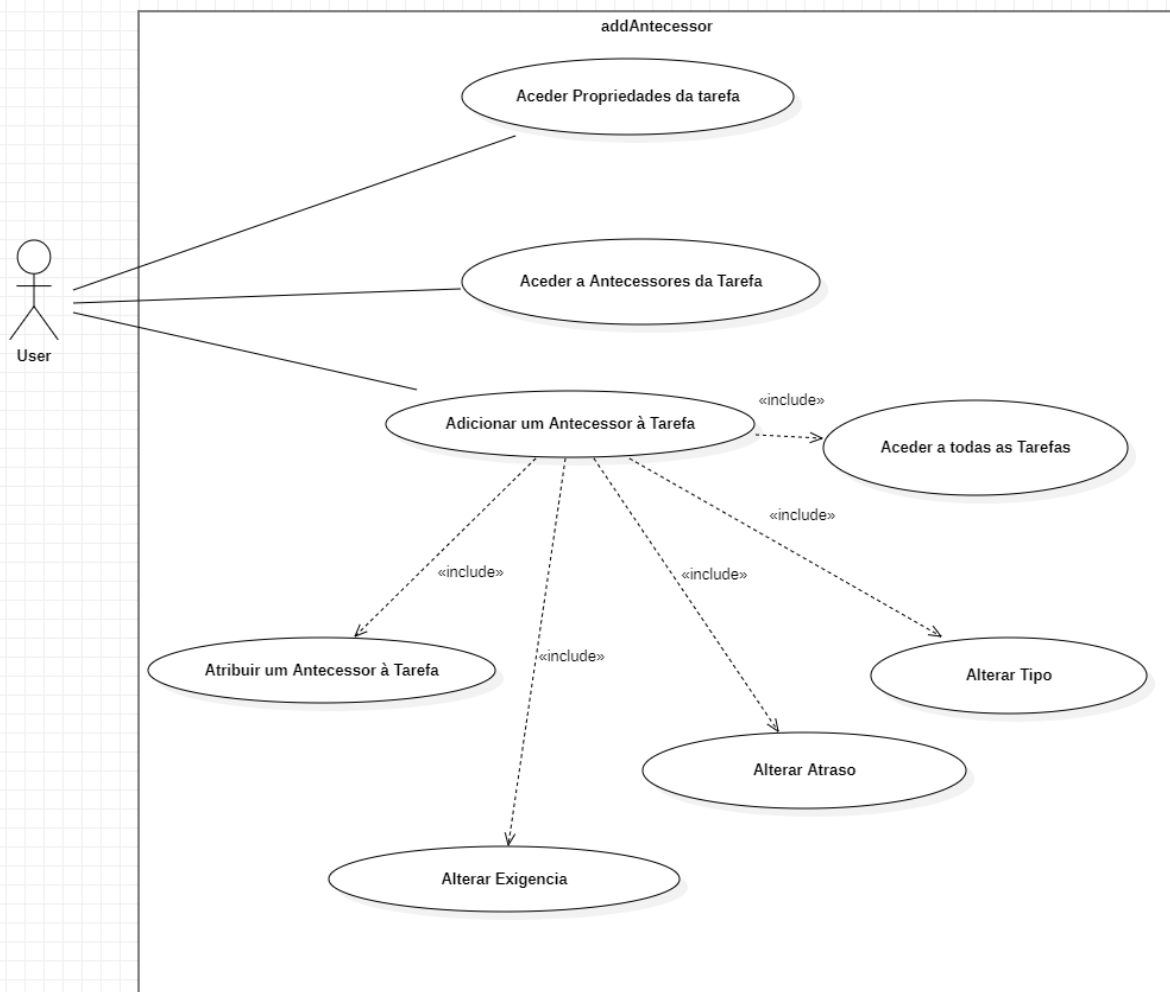
Alternative flow: ImportarExcel.FicheiroInexistente
ID: 1.1
Description: O sistema não encontra o ficheiro Excel
Primary Actors: Não existem
Secondary Actors: Não existem
Preconditions: Ficheiro Excel não se encontra na diretoria.
Alternative Flow: <ol style="list-style-type: none">1. O alternative flow começa depois da etapa 2 do main flow.2. O sistema envia uma mensagem de erro
Postconditions: Recursos não adicionados, mensagem de erro enviada ao utilizador

Use Case Daniel Eugénio (59797)



ID: 2
Description: Inserção de um novo recurso no sistema.
Primary Actors: Utilizador
Secondary Actors: Não existem
Preconditions: Não existem
Main Flow: <ol style="list-style-type: none">1. O use case começa quando o utilizador seleciona a opção de inserir um novo recurso no sistema2. O sistema abre uma janela com o formulário para inserção dos dados3. O utilizador pode preencher o formulário com o nome, email, telefone, função e preço por hora do novo recurso<ol style="list-style-type: none">a. O utilizador pode optar por preencher apenas alguns ou até nenhum dos campos4. O sistema regista o novo recurso que pode ser consultado na aba "Resources"
Postconditions: Um recurso foi criado
Alternative flows: Não existem

Use Case Francisco Silveira (60816)



Use case: AddAntecessor
ID: 1
Description: Inserir um antecessor a uma tarefa
Primary_Actors: Utilizador
Secondary Actors: Não existem
Preconditions: Existir pelo menos duas tarefas no sistema
Main Flow: <ol style="list-style-type: none"> 1. O use case começa quando o utilizador seleciona uma tarefa e seleciona a opção propriedades da tarefa. 2. Utilizador seleciona a secção dos antecessores. 3. O sistema exibe todos os antecessores atribuídos à tarefa. 4. O utilizador clica no botão adicionar. 5. O utilizador escolhe a tarefa que quer adicionar como antecessor. 6. O utilizador pode ainda escolher alterar o tipo, o atraso e a exigência do antecessor. <ol style="list-style-type: none"> a. O utilizador pode optar por preencher algum destes campos. Os que não forem alterados possuem valores default(Fim-Inicio,0,Muita, respetivamente).
Postconditions: Foi adicionado um novo antecessor à tarefa
Alternative flows:

User Stories

User Stories Daniel Eugénio (59797)

- As a user I want to be able to view all the resources in the system so that I don't add the same person twice.
- As a user I want to be able to insert new columns to the resources so that I can complete their information as I see fit.

User Stories Rafael Costa (60441)

- As a user I want to be able to import resources to the system using excel so that It will not be necessary to add a lots of resources by hand
- As a user I want to save the project file so that I can send to my co-workers.

User Stories Guilherme Abrantes (60971)

- As a user I want to see what tasks each resource has so I don't waste time going through every task to check.
- As a user I want to zoom out my calendar so that I don't waste time scrolling right to plan my work.

User Stories Miguel Agostinho (60677)

- As a user I want to add tasks so that I can manage my project.
- As a user I want to highlight completed tasks so that I can see these tasks better.

User Stories Francisco Silveira (60816)

- As a user I want to add predecessors to my tasks so that I can organize which tasks to do first.
- As a user I want to add a resource's days off so you can organize the project without delay.

Anexos

Github: <https://github.com/Miguell5/ganttproject>

Vídeo Youtube: <https://youtu.be/u2j2Hhprzqk>