**Student**

Francisco Parrinha – 58360

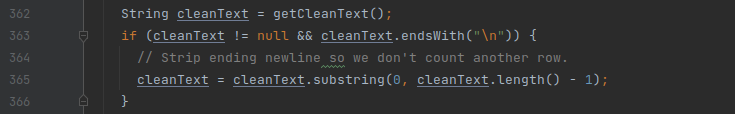
**Code Smells**

1. Long Method:

* Method *computePrefHeight*” in: *ganttproject/src/main/java/biz.gantproject/lib/fx/treetable/LabeledSkinBase.java*

The method starts at line number 350 and ends at number 397, containing 47 lines. There are several refactoring possibilities. The code within lines number 362 and 366 serves as an auxiliary operation to the method. This operation is to strip strings.

The following block of code could be removed from this function to a new *getStrippedString* method, and later be called on the original *computePrefHeight* function. The code to be refactored:

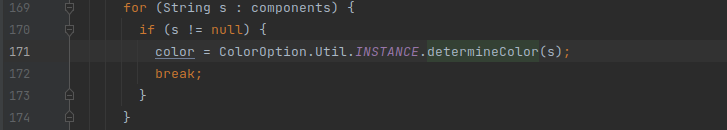


* Método *parse* em: *biz.ganttproject.core/src/main/java/biz/ganttproject/core/chart/render/RectangleRenderer.java*

Although the parse function is not too long, it contains an auxiliary operation and could be refactored.

The block of code shown below (linhas 169 - 174) could be moved to a new method, *gerColor*, for example. This would improve the project 's modularity. The new method would then be called on the original function.

The code to be refactored:

**

2. Comments:

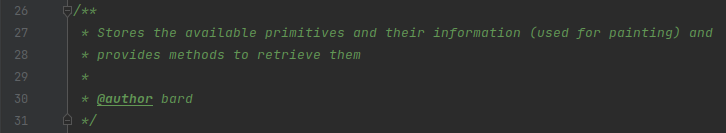
* Classe *Canvas* em: *biz.ganttproject.core/src/main/java/biz/ganttproject/core/chart/canvas/*

This class contains several nested classes The entire *Canvas* class only contains one comment (line 26 -31) and it does not have any *Javadoc* written excluding this one.

All methods and nested classes are to be commented out, making the code understanding much harder.

Creating the Javadoc is enough to refactor this code-smell. Here are some examples of this anti-pattern

1. The only comment:



1. Some examples of the lack of code: