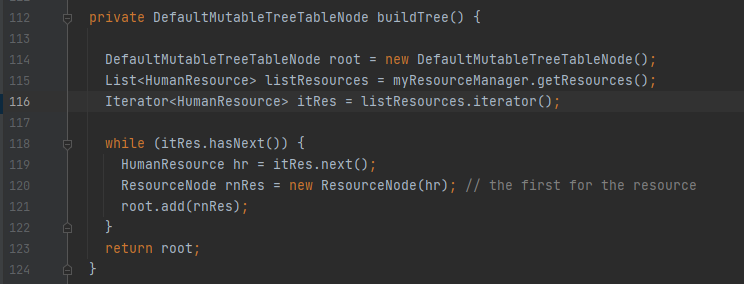
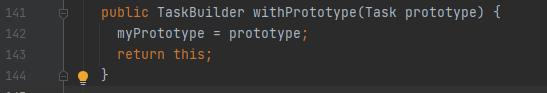
Relatório Fase 1

# Padrões

* Iterador
  + 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) <-----

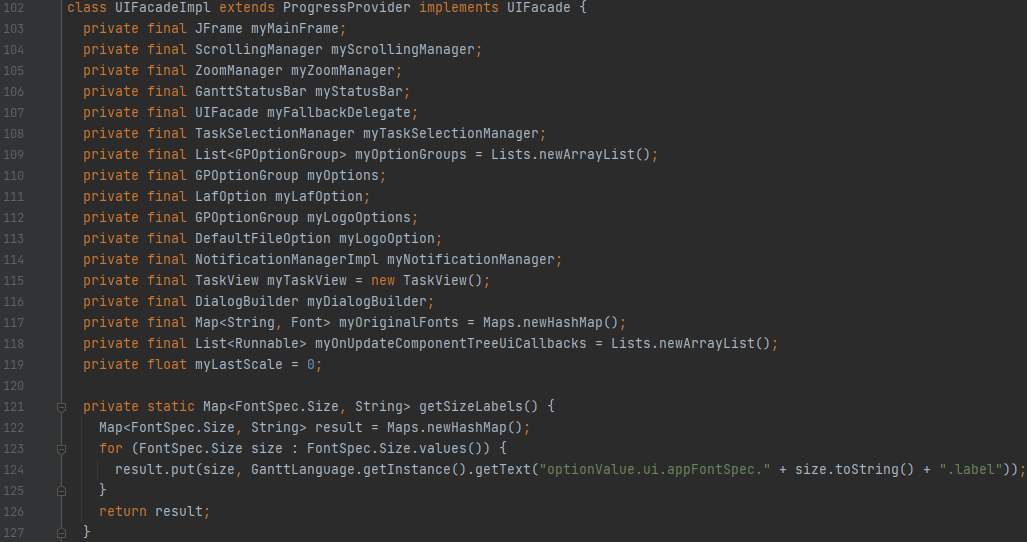
* Prototype



* + 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) <-----

* Façade

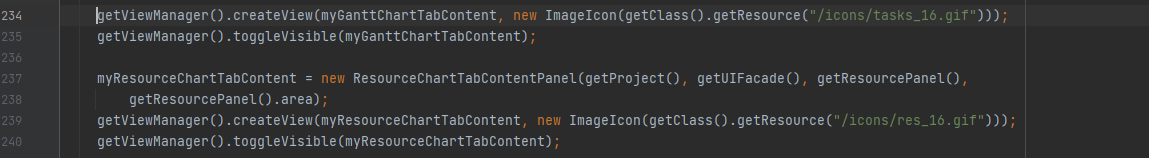


* + Aqui está implementado o padrão Façade. Neste caso, o padrão é aplicado para simplificar o API ProgressProvider.

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

# Code Smells

* Repeated Code



* + Neste troço, 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) <-----

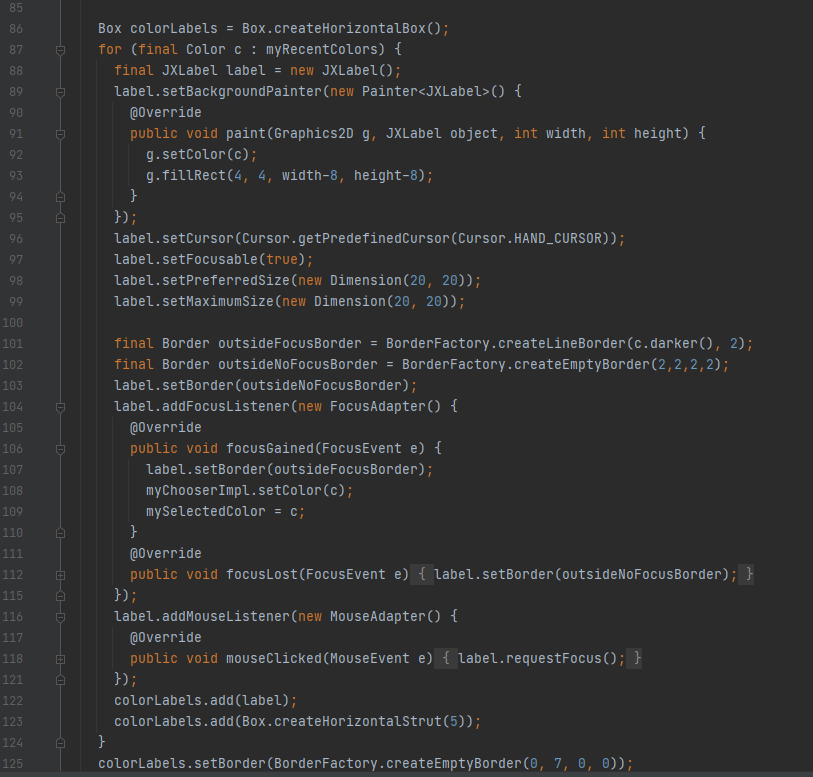
* Long Parameter List



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

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

* Divergent Change



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