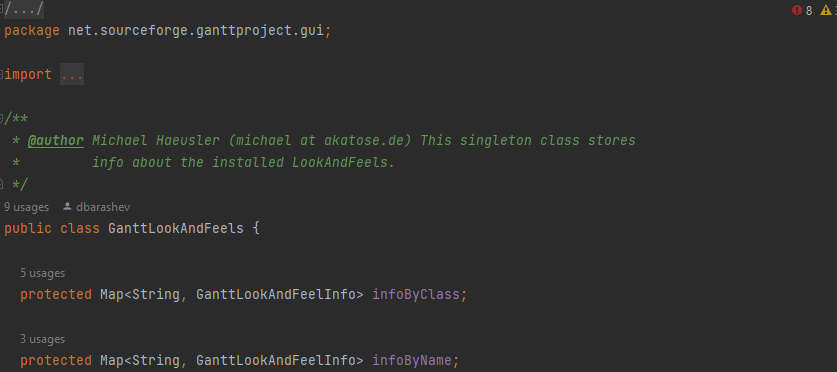
Gof Design Patterns

## Singleton Pattern (Creational Pattern)

Um padrão do estilo singleton tem como função principal central de forma mais restrita as variáveis globais: garante que existe apenas uma instância de classe, ou seja, existe apenas a classe singleton (classe GanttLookAndFeels). Tem também como característica ter um objeto partilhado por diferentes partes do programa.

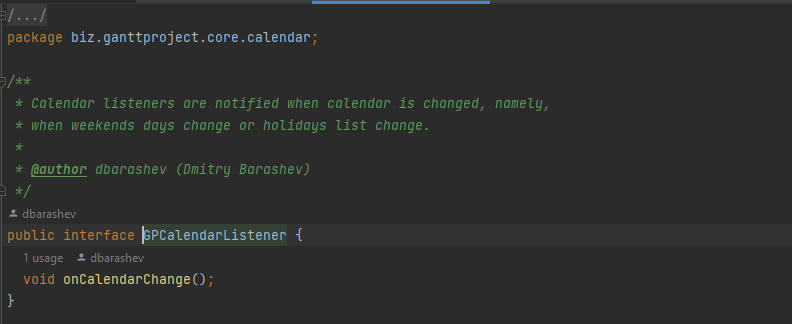


**Localização:** ganttproject/src/main/java/net/sourceforge/ganttproject/gui/GanttLookAndFeels.java

## Observer Pattern(Behavioral Pattern)

Este padrão serve essencialmente para em vez de fazermos a chamada de um método diversas vezes para várias classes, por exemplo update(), fazemos apenas uma vez no observer/listener e este comunica com os restantes caso seja necessário.

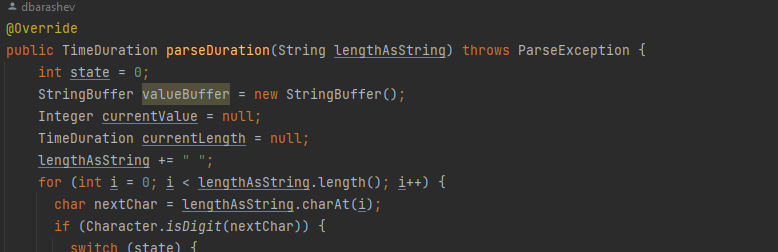
Neste caso temos a classe GPCalendarListener.java que é exatamente um listener de algum update que tenha de haver no calendário e, se houver, envia isso para as classes correspondentes.



**Localização:** ganttproject/biz.ganttproject.core/src/main/java/biz/ganttproject/core/calendar/GPCalendarListener.java

## State Pattern( Behavioural Pattern )

O padrão State tem como objetivo alterar o comportamento de um objeto quando o seu estado se altera. É usado de maneira a simplificar o código, limpando o número de excessivo de condições. No método parseDuration, pertencente à classe GPTimeUnitStack, verifica-se a obediência a este padrão, dado que a condição Character.isDigit(nextChar) muda de comportamento, dependendo do estado (state) em que se encontra.



**Localização:** ganttproject/biz.ganttproject.core/src/main/java/biz/ganttproject/core/time/impl/GPTimeUnitStack.java

**Linhas:** 161-254