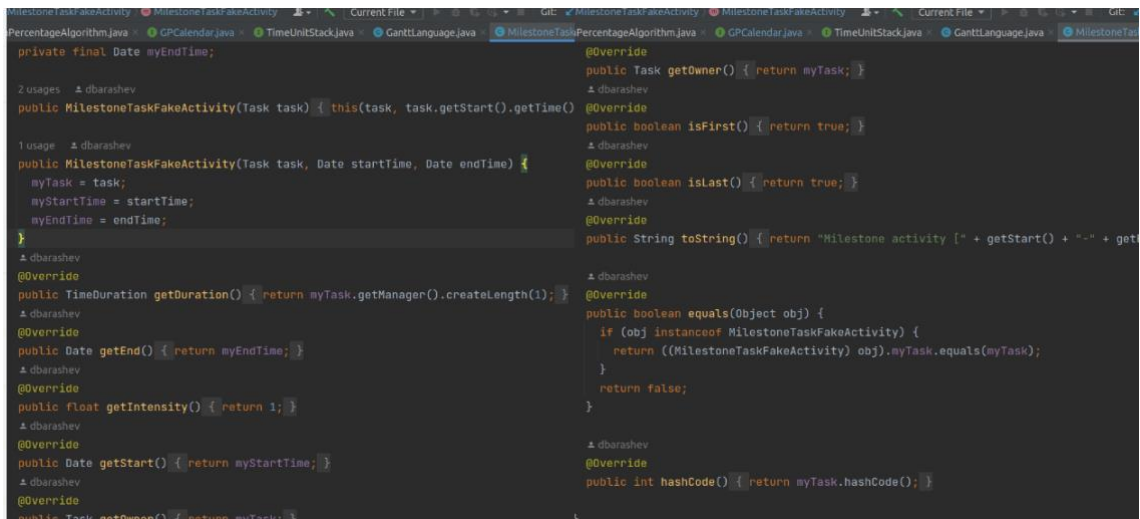


Data Class



```
private final Date myEndTime;

2 usages  ▲ dbarashev
public MilestoneTaskFakeActivity(Task task) { this(task, task.getStart().getTime()

1 usage  ▲ dbarashev
public MilestoneTaskFakeActivity(Task task, Date startTime, Date endTime) {
    myTask = task;
    myStartTime = startTime;
    myEndTime = endTime;
}

▲ dbarashev
@Override
public TimeDuration getDuration() { return myTask.getManager().createLength(1); }
▲ dbarashev
@Override
public Date getEnd() { return myEndTime; }
▲ dbarashev
@Override
public float getIntensity() { return 1; }
▲ dbarashev
@Override
public Date getStart() { return myStartTime; }
▲ dbarashev
@Override
public Task getOwner() { return myTask; }

@Override
public Task getOwner() { return myTask; }
▲ dbarashev
@Override
public boolean isFirst() { return true; }
▲ dbarashev
@Override
public boolean isLast() { return true; }
▲ dbarashev
@Override
public String toString() { return "Milestone activity [" + getStart() + "-" + get

@Override
public boolean equals(Object obj) {
    if (obj instanceof MilestoneTaskFakeActivity) {
        return ((MilestoneTaskFakeActivity) obj).myTask.equals(myTask);
    }
    return false;
}

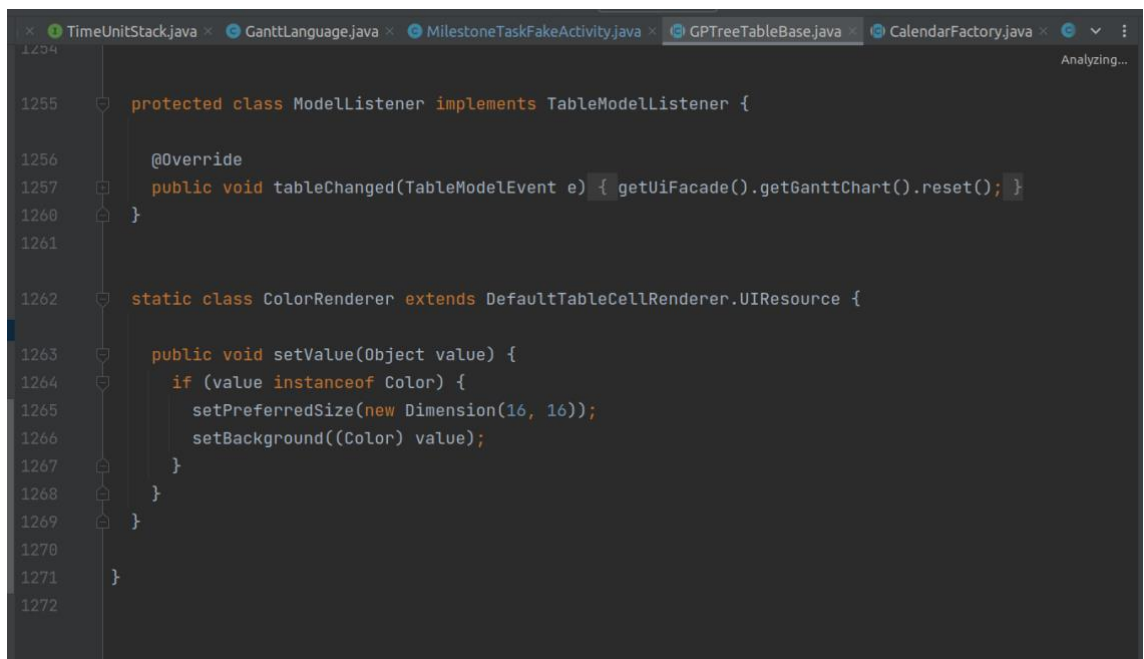
▲ dbarashev
@Override
public int hashCode() { return myTask.hashCode(); }
```

Ficheiro: ganttproject/src/main/java/net/sourceforge/ganttproject/chart/MilestoneTaskFakeActivity.java

Texto: A class MilestoneTaskFakeActivity só tem getters.

Solução: Tentar dar alguma responsabilidade a classe.

Large class



```
1254
1255 protected class ModelListener implements TableModelListener {
1256     @Override
1257     public void tableChanged(TableModelEvent e) { getUiFacade().getGanttChart().reset(); }
1260 }
1261
1262 static class ColorRenderer extends DefaultTableCellRenderer.UIResource {
1263     public void setValue(Object value) {
1264         if (value instanceof Color) {
1265             setPreferredSize(new Dimension(16, 16));
1266             setBackground((Color) value);
1267         }
1268     }
1269 }
1270
1271 }
1272
```

Ficheiro: ganttproject/src/main/java/net/sourceforge/ganttproject/GPTreeTableBase.java

Texto: A class GPTreeTableBase tem mais de 1000 linha de código além de ter muitas outras classes dentro dela mesma e métodos muito longos.

Solução: Dividir a class em outras classes mais pequenas.

Switch statement

```
7 usages
final int mySecondTaskID;

2 usages  dbarashev
public SearchKey(int type, TaskDependencyImpl taskDependency) {
    myType = type;
    Task firstTask, secondTask;
    switch (type) {
        case DEPENDANT: {
            firstTask = taskDependency.getDependant();
            secondTask = taskDependency.getDependee();
            break;
        }
        case DEPENDEE: {
            firstTask = taskDependency.getDependee();
            secondTask = taskDependency.getDependant();
            break;
        }
        default: {
            throw new RuntimeException("Invalid type=" + type);
        }
    }
    myFirstTaskID = firstTask.getTaskID();
    mySecondTaskID = secondTask.getTaskID();
}
```

Ficheiro: ganttproject/src/main/java/net/sourceforge/ganttproject/task/dependency/SearchKey.java

Texto: A classe tem um atributo myType que é um inteiro. As constantes DEPENDANT e DEPENDEE são usadas como tipos. E tem um switch statement no construtor para os dois casos e uma exceção caso contrário.

Solução: Fazer duas subclasses. Além de eliminar o atributo myType vai eliminar a exceção o que já é muito bom.