**Joana Tomás 60152**

Code Smells

1. Duplicated code

*public* ResourceNewAction(HumanResourceManager hrManager, ProjectDatabase projectDatabase, RoleManager roleManager, TaskManager taskManager, UIFacade uiFacade) {  
 *super*("resource.new", hrManager);  
 myUIFacade = uiFacade;  
 myRoleManager = roleManager;  
 myTaskManager = taskManager;  
 myProjectDatabase = projectDatabase;  
}  
  
*private* ResourceNewAction(HumanResourceManager hrManager, ProjectDatabase projectDatabase, RoleManager roleManager, TaskManager taskManager, UIFacade uiFacade, IconSize size) {  
 *super*("resource.new", hrManager, *null*, size);  
 myUIFacade = uiFacade;  
 myRoleManager = roleManager;  
 myTaskManager = taskManager;  
 myProjectDatabase = projectDatabase;

Both code fragments look identical, making the code more difficult to change and update.

Solution: Make a method with the repeated code that can adapt to many tasks.

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/action/resource/ResourceNewAction.java

1. Large class

*public class* ConsoleUIFacade *implements* UIFacade

This class contains many lines of code. This occurs because this class has more responsibilities than it should.

Solution: Some of the responsibilities in this class can be moved to another class, making it more simple.

Location:

ganttproject/src/main/java/net/sourceforge/ganttproject/export/ConsoleUIFacade.java

1. Long Method

*final* TaskContainmentHierarchyFacade taskHierarchy = getTaskManager().getTaskHierarchy();  
 Function<Task, Task> getParent = task -> taskHierarchy.getContainer(task);  
  
 *// If there are tasks in selection which are in ancestor-descendant relationship,  
 // we'll retain only topmost ones.* List<Task> indentRoots = Lists.newArrayList();  
 myRetainRootsAlgorithm.run(selection, getParent, indentRoots);  
  
 *// We use dependency graph transaction to test if we get a loop after move.  
 // THROWING\_LOGGER will throw TaskDependencyException if graph finds a loop.* DependencyGraph dependencyGraph = getTaskManager().getDependencyGraph();  
 DependencyGraph.Logger oldLogger = dependencyGraph.getLogger();  
 dependencyGraph.setLogger(DependencyGraph.THROWING\_LOGGER);  
 dependencyGraph.startTransaction();  
 *try* {  
 Function<Task, Task> getParentFxn = myGetMoveTargetFxnFactory.apply(indentRoots);  
 *for* (Task task : indentRoots) {  
 Task moveTarget = getParentFxn.apply(task);  
 *if* (moveTarget == *null*) {  
 *return false*;  
 }  
 *if* (moveTarget.isMilestone()) {  
 *return false*;  
 }  
 dependencyGraph.move(task, moveTarget);  
 }  
 } *catch* (TaskDependencyException e) {  
 *return false*;  
 } *finally* {  
 dependencyGraph.rollbackTransaction();  
 dependencyGraph.setLogger(oldLogger);  
 }  
 *return true*;  
}

This method contains too many lines of code. This probably means that is more complex than it needs to be.

Solution: Make auxiliary methods to simplify the code.

Location: ganttproject/src/main/java/net/sourceforge/ganttproject/action/task/TaskMoveEnabledPredicate.java