Pattern 1:

Code Snippet:

(…)

boolean contains(Task var1);  
  
 public interface Factory {  
 TaskContainmentHierarchyFacade createFacade();  
 }  
}

Location on the code base: ganttproject/bin/main/net/sourceforge/ganttproject/task/TaskContainmentHierarchyFacade.class

Pattern identification: Factory method - This interface hides the creation of the instance TaskContainmentHierarchyFacade. This interface is useful in the class FacadeFactoryImpl nested in the class TaskManagerImpl

Pattern 2:

Code Snippet:

Fig1

public class CommandLineExportApplication {  
 private LoggerApi logger;  
  
 public CommandLineExportApplication() {  
 throw new Error("Unresolved compilation problems: \n\tThe import biz.ganttproject.LoggerApi cannot be resolved\n\tThe import net.sourceforge.ganttproject.IGanttProject cannot be resolved\n\tLoggerApi cannot be resolved to a type\n\tThe method create(String) from the type GPLogger refers to the missing type LoggerApi\n\tIGanttProject cannot be resolved to a type\n\tIGanttProject cannot be resolved to a type\n\tLoggerApi cannot be resolved to a type\n\tConsoleProgressProvider cannot be resolved to a type\n");  
 }  
  
 public boolean export(Args var1, IGanttProject var2, UIFacade var3) {  
 throw new Error("Unresolved compilation problem: \n\tIGanttProject cannot be resolved to a type\n");  
 }  
  
 boolean export(Exporter var1, Args var2, IGanttProject var3, UIFacade var4) {  
 throw new Error("Unresolved compilation problems: \n\tIGanttProject cannot be resolved to a type\n\tLoggerApi cannot be resolved to a type\n\tConsoleProgressProvider cannot be resolved to a type\n");  
 }  
  
 public static class Args {  
 @Parameter(  
 names = {"-export"},  
 description = "Export format"  
 )  
 public String exporter;  
 @Parameter(  
 names = {"-stylesheet"},  
 description = "Stylesheet used for export"  
 )  
 public String stylesheet;  
 @Parameter(  
 names = {"-chart"},  
 description = "Chart to export (resource or gantt)"  
 )  
 public String chart;  
 @Parameter(  
 names = {"-zoom"},  
 description = "Zoom scale to use in the exported charts"  
 )  
 public Integer zooming;  
 @Parameter(  
 names = {"-o", "-out"},  
 description = "Output file name",  
 converter = FileConverter.class  
 )  
 public File outputFile;  
 @Parameter(  
 names = {"-expand-resources"},  
 description = "Expand resource nodes on the resource load chart"  
 )  
 public boolean expandResources;  
 @Parameter(  
 names = {"-expand-tasks"},  
 description = "Expand all tasks nodes on the Gantt chart",  
 arity = 1  
 )  
 public boolean expandTasks;  
  
 public Args() {  
 throw new Error("Unresolved compilation problems: \n\tThe import biz.ganttproject.LoggerApi cannot be resolved\n\tThe import net.sourceforge.ganttproject.IGanttProject cannot be resolved\n\tLoggerApi cannot be resolved to a type\n\tThe method create(String) from the type GPLogger refers to the missing type LoggerApi\n\tIGanttProject cannot be resolved to a type\n\tIGanttProject cannot be resolved to a type\n\tLoggerApi cannot be resolved to a type\n\tConsoleProgressProvider cannot be resolved to a type\n");  
 }  
 }  
}

Fig2

} else {  
 appBuilder.whenDocumentReady(project -> {  
 var executor = Executors.newSingleThreadExecutor();  
 executor.submit(() -> {  
 var cliApp = new CommandLineExportApplication();  
 cliApp.export(appBuilder.getCliArgs(), project, ((GanttProject) project).getUIFacade());  
 GanttProject.doQuitApplication(true);  
 });  
 return Unit.INSTANCE;  
 });  
}

Location on the code base:

Fig1 - code/ganttproject/src/main/java/net/sourceforge/ganttproject/application/MainApplication.java

Fig2 - ganttproject/bin/main/net/sourceforge/ganttproject/export/CommandLineExportApplication.class

Pattern identification: Command Pattern – the class MainApplication – the invoker – invokes an object from CommandLineExportApplication to complete a task. This class CommandLineExportApplication can be considered as a command manager because manipulates and invokes the commands needed to finish this method.

Pattern 3:

Code Snippet:

Fig1

package net.sourceforge.ganttproject.gui;  
  
import biz.ganttproject.core.table.ColumnList;  
import com.google.common.base.Predicate;  
import java.awt.Component;  
import javax.swing.AbstractAction;  
import net.sourceforge.ganttproject.action.GPAction;  
  
public interface TreeUiFacade<T> {  
 Component getTreeComponent();  
  
 ColumnList getVisibleFields();  
  
 boolean isVisible(T var1);  
  
 boolean isExpanded(T var1);  
  
 void setExpanded(T var1, boolean var2);  
  
 void applyPreservingExpansionState(T var1, Predicate<T> var2);  
  
 void setSelected(T var1, boolean var2);  
  
 void clearSelection();  
  
 void makeVisible(T var1);  
  
 GPAction getNewAction();  
  
 GPAction getPropertiesAction();  
  
 GPAction getDeleteAction();  
  
 void startDefaultEditing(T var1);  
  
 void stopEditing();  
  
 AbstractAction[] getTreeActions();  
}

Fig2

public interface ResourceTreeUIFacade extends TreeUiFacade<HumanResource> {  
 AbstractAction getMoveUpAction();  
  
 AbstractAction getMoveDownAction();  
  
 TimelineChart.VScrollController getVScrollController();  
}

Location on the code base:

Fig1 - ganttproject/bin/main/net/sourceforge/ganttproject/gui/TreeUiFacade.class

Fi2 - ganttproject/bin/main/net/sourceforge/ganttproject/gui/ResourceTreeUIFacade.class

Uma imagem com texto

Descrição gerada automaticamentePattern identification:

Facade – The interface ResourceTreeUIFacade, an extension of the interface TreeUiFacade, in an encapsulation of a subsystem with four classes. This interface acts as a point of entry into this subsystem.