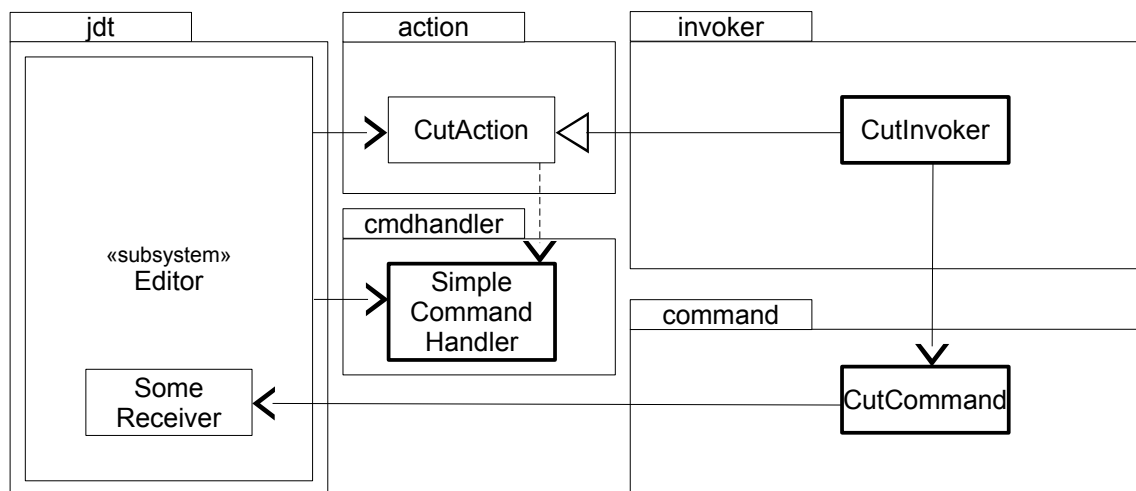# Undoing and Redoing Operations

Introduction

Interactive applications usually allow users to undo an operation just performed. In a graphical editor, for example, unintentionally deleting a recently added shape can be undone by pressing the *undo* operation of the tool.

The *Command pattern* provides the class structure allowing the programmers of the editor to introduce this user-friendly behavior in a systematic manner. In this exercise, you'll complete the editor's behavior to support multiple *undo* and *redo* operations.

Invokers, Commands and Receivers

The following sketch of classes illustrates the structure of this pattern:

Assume a user uses the editor. After having selected one or more shapes, the user invokes the *cut*-operation. Thus, the selected shapes are eliminated from the editor's current *sheet* and moved onto the *clipboard*. That is, the editor invokes the `actionPerformed(...)` method of a `CutAction`, the superclass of class `CutInvoker`. Method `actionPerformed(...)` calls the *Factory Method* createCommand(...) which in turn creates and returns the appropriate command. Method `actionPerformed(...)` then registers the returned command with the `SimpleCommandHandler` instance and executes the command.

Similar things can be said for other operations of the editor.

Your Tasks

- Complete the given classes in the packages `*.invoker`, `*.command`, and `*.cmdhandler`.
- Test your code by running the provided JUnit test classes. You may add your own test classes, too.
- Run the application by executing `ch.bfh.due1.jdt.simple.JdtEditor`.

Remarks

- Modify and/or extend existing classes in sub-packages of package `ch.bfh.sed.commandpattern`.

References

- Software frameworks: http://en.wikipedia.org/wiki/Software_framework
- Wikipedia: http://en.wikipedia.org/wiki/Command_pattern
- SourceMaking: https://sourcemaking.com/design_patterns/command
- WikiBooks: https://en.wikibooks.org/wiki/Computer_Science_Design_Patterns/Command