Algorithmen und Datenstrukturen

Der Experimentierkasten

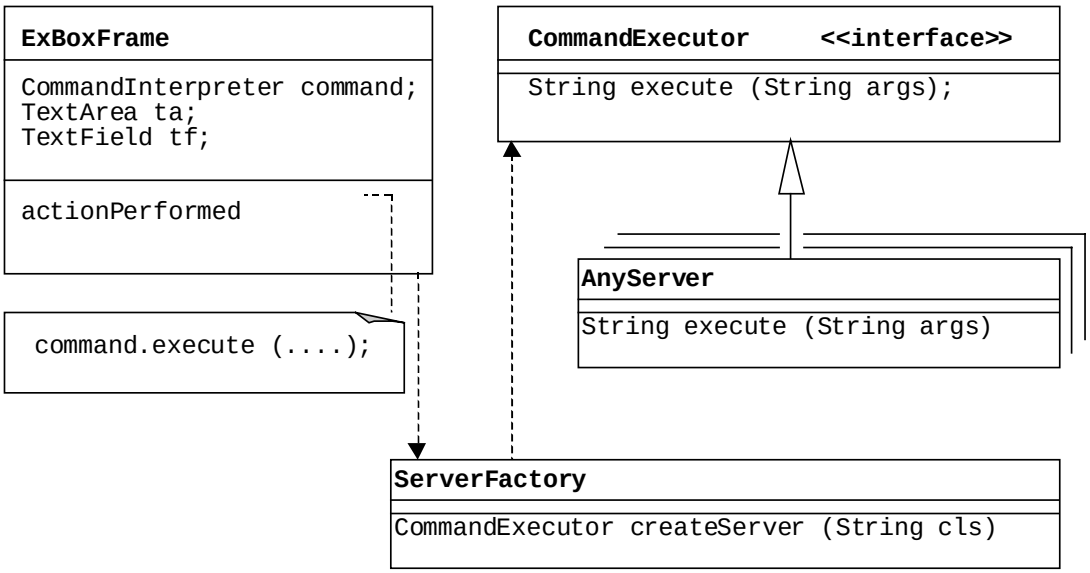
In diesem Semester werden wir eine Reihe von Algorithmen kennenlernen und mit diesen Experimente durchführen.

Für die Praktika empfiehlt Ihnen der Dozent NetBeans 9 oder IntelliJ. Sie können auch jede andere Ihnen gewohnte IDE/Editor verwenden (Notepad++, vi, emacs, Eclipse aber *NICHT* BlueJ[1]).

Sie können die Projekte und Tests in Ihrer IDE ausführen, aber auch ein Build Tool Ihrer Wahl verwenden: z.B. ANT, Maven, Gradle ….

Wir benötigen weiter einen geeigneten "Experimentierkasten". Dieser soll in einem sogenannten Client-Server-Arrangement realisiert werden. Der Client stellt dabei die Benützerschnittstelle zur Verfügung, der Server enthält die Experimente, die Implementierung der Lösung der Aufgaben. Der Client soll so entworfen werden, dass beliebige Server auch nachträglich hinzugefügt werden können (wir werden in den folgenden Aufgaben immer wieder neue Server schreiben). Dabei muss der Client lediglich wissen, wie der Server aufgerufen werden kann, er muss also die Schnittstelle kennen. Die Schnittstelle wird in Java als *Interface* (Bsp. *CommandExecutor*) realisiert, welches vom Server (Bsp. *AnyServer*) implementiert wird. Zusätzlich müssen wir Server nachladen, bzw. Klassen instanzieren können, von denen wir lediglich den Namen (als String) kennen. Für diesen Zweck implementieren wir eine Klasse ServerFactory, mit der Objekte beliebiger Klassen die den *CommandExecutor* implementieren instanziert werden können.

Klassendiagram

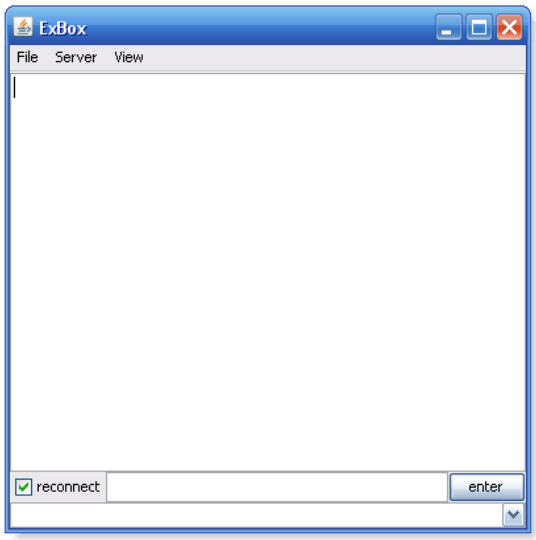


[1] Falls Sie den *unbedingten Wunsch* verspüren, BlueJ verwenden *zu müssen*, dann empfiehlt Ihnen der Dozent den nachfolgenden Link: https://www.zhaw.ch/de/studium/waehrend-des-studiums/beratung/coaching-und-psychologische-beratung/

Beschreibung der Funktionalität

Folgende Funktionen werden implementiert.

AnyServer stellt einen Beispiel-Server dar, welcher das Interface *CommandExecutor* implementiert. Letzteres ist das Interface, das von allen Servern implementiert werden muss. In jedem Praktikum werden Sie einen oder mehrere dieser Server implementieren.




```
public interface CommandExecutor {
  public String
    executor(String command);

}

public class AnyServer implements
   CommandExecutor {
   public String
    executor(String command) {
     // Experiement
   }
```

Bedienung/Befehle

Das Hauptfenster des GUI-Client enthält zwei Bereiche: 1) eine Textzeile für die Eingabe der Befehle und Argumente und 2) ein Textfenster, das die Resultate der ausgeführten Befehle anzeigt. Ferner enthält der Menubalken mehreren Menugruppen:

1. In der Menugruppe "File" ermöglicht "**Exit**" das Fenster zu schliessen (wird automatisch generiert)

2. File→ Open… öffnet ein File und übergibt den Inhalt der execute Methode

3. In der Menugruppe "Server" ermöglicht "**Test**" einen Unit Test durchzuführen.

4. In der Menugruppe "Server" ermöglicht "**Connect**" die Verbindungsaufnahme zu einem Server, der das Interface CommandExecutor implementiert.

   – In der Klasse ServerFactory wird ein Objekt der Klasse deren Namen imArgument der Methode createServer übergeben wird instanziert:

5. Sobald der Enter-Knopf gedrückt wird oder "Enter" eingegeben wird, ruft der Client die Methode *String execute (String command)* des Servers auf. Die von dieser Methode zurückgegebenen Daten übernimmt der Client und stellt sie im Textfenster dar.