Vorlesung "Softwaretechnik" SoSe 2016

Freie Universität Berlin, Institut für Informatik, Arbeitsgruppe Software Engineering
Barry Linnert, Franz Zieris, Michaela Borzechowski, Tanja Linkermann, Marco Ziener

Übungsblatt 8 **Entwurfsmuster** zum 12.06.2016, 16:00 Uhr

Aufgabe 8-1: Begriffe

a) Erklären Sie die offensichtlichsten Unterschiede zwischen *Komponenten*, *Entwurfsmustern* und *Architekturstilen*.

b) Recherchieren Sie das Entwurfsmuster *Einzelstück* (engl. *singleton*) und entwerfen Sie drei Beispiele für seine Verwendung. Zu welcher Klasse der in der Vorlesung genannten Entwurfsmustertaxonomie (*pattern taxonomy*) gehört dieses Muster und warum?

c) Charakterisieren und unterscheiden Sie die folgenden Entwurfsmuster: Stellvertreter (*Proxy*), Adapter (*Adapter*), Fassade (*Facade*), Brücke (*Bridge*).

Aufgabe 8-2: Entwurfsmuster für eigene Software-Idee

a) Überlegen Sie sich, welches der in der Vorlesung vorgestellten Entwurfsmuster Sie in Ihrer zu entwickelnden Software sinnvoll einsetzen können (außer dem Singleton und dem Adapter).

Stellen Sie die Charakteristika des gewählten Entwurfsmusters heraus und begründen Sie, warum und wofür das Muster in Ihrem Kontext geeignet ist.

b) Stellen Sie das Entwurfsmuster in seiner allgemeinen Form in UML-Notation dar.

c) Übertragen Sie nun die UML-Notation des Entwurfsmusters in Ihren Kontext und stellen dieses ebenfalls als UML-Diagramm dar.

d) Implementieren Sie (in Java oder Pseudocode) in Ansätzen die Verwendung des Entwurfsmusters in Ihrem Kontext. Erstellen Sie die notwendigen Schnittstellen und Klassen, mit den jeweils relevanten Codestellen (leere Methodenrümpfe reichen in aller Regel *nicht* aus).

e) Denken Sie wie immer daran, Ihre Arbeitsergebnisse der Teilaufgaben **a)** bis **d)** zusätzlich zur elektronischen Abgabe über das KVV Ihrer Wiki-Seite hinzuzufügen.

Aufgabe 8-3★: Entwurfsmuster anwenden

Stellen Sie sich vor, Sie sind dabei einen einfachen Texteditor mit dem Namen “Notepad−−” in Java zu entwickeln. Der Funktionsumfang ist sehr reduziert, und Sie haben alle Hände voll zu tun, diesen zu erweitern. Daher wollen Sie, so gut es geht, existierende Lösungen wiederverwenden.

a) Für die Dateiverwaltung Ihres Editors wollen Sie eine Klasse `Filesystem` wiederverwenden, die Sie früher einmal entwickelt haben. Diese funktioniert tadellos, und deswegen wollen Sie die auch nicht mehr ändern. Sie haben sich für das Adapter-Entwurfsmuster entschieden. Hier finden Sie einen Auszug der Schnittstellendokumentation:

```
public class Filesystem {
    // Returns the zero-based position on disk where the given file's
    // content starts. Returns -1 if no such file exists.
    public int position(String filename) { /* ... */ }

    // Write the content on the disk, starting at the given position
    public void write(int position, byte[] content) { /* ... */ }

    // Prepares the creation of a new file. Provide the content
    // through the write method! Fails if the name is already taken.
    public void touch(String filename) { /* ... */ }

    // Erases a file, and compacts the content of the disk, so that
    // there are no gaps in between.
    public void remove(String filename) { /* ... */ }
}
```

Der Gebrauch der `Filesystem`-Klasse ist etwas ungelenkt, und in Ihrer Hauptklasse `NotepadMinusMinus` wollen Sie lediglich einen einzigen Aufruf zum Speichern des aktuellen Editor-Inhalts vornehmen. Folgendes Gerüst haben Sie bereits implementiert:

```
public class NotepadMinusMinus {
    /* ... */

    public void save() {
        String content = DocBuffer.getContent();
        String name = DocManager.getCurrentFileName();

        // TODO Save to disk (replace existing file, if any)
    }
}
```

Aufgabe: Implementieren Sie eine neue Klasse (etwa “`Storage`”) mit einer einzigen Methode (etwa “`store`”), die Dateinamen und Inhalt erhält und Ihre alte `Filesytem`-Klasse wiederverwendet – ohne diese zu verändern!

Ergänzen Sie außerdem die `save()`-Implementierung aus obigen Fragment. Führen Sie keine weiten Klassen oder Schnittstellen ein.

b) Das klassische Adapter-Muster kennt die folgenden vier Rollen: Client, Target, Adapter und Adaptee (siehe z.B. http://www.blackwasp.co.uk/Adapter.aspx). Durch welche der Klassen Ihrer Implementierung wird jede dieser Rollen ausgefüllt?

c) Nun haben wollen Sie Ihrem Editor auch eine Dropbox-Integration spendieren. Für solche Zwecke bietet Dropbox selbst ein SDK (eine Bibliothek) an. Gehen Sie für diese Aufgabe von folgender vereinfachter API aus:

```
public class DbxClient {
    public enum WriteMode {
        ADD, REPLACE
    }
```

```
    // Checks whether a file already exists on the server
    public boolean fileExists(String name) { /* ... */ }

    // Uploads a new version of a file; either as a new file or as a
    // replacement for an existing one. New files must be transmitted
    // using the ADD mode, replacements must be done in REPLACE mode.
    // Incorrect modes indicate inconsistencies and the upload will
    // fail.
    public void uploadFile(String name, WriteMode mode, byte[] data) {
        /* ... */ }
}
```

Aufgabe: Nutzen Sie das Dropbox-SDK, um Ihren Editor um eine Speicher-Option zu erweitern. Wie schon bei der `FileSystem`-Wiederverwendung sollen (bzw. *können*) Sie den Dropbox-Quellcode nicht verändern. Schreiben Sie also einen weiteren Adapter hierfür.

Um in der `NotepadMinusMinus`-Klasse nicht direkt von einem der beiden Adapter (einen für das lokale Dateisystem, einen für Dropbox) abzuhängen, führen Sie bitte ein Interface ein (etwa "`IStorage`"), das die beiden Adapter jeweils implementieren. Um in der `NotepadMinusMinus`-Klasse davon Gebrauch machen zu können, implementieren Sie bitte eine Setter-Methode (siehe unten) und verwenden den übergebenen Wert in der `save()`-Methode:

```
public class NotepadMinusMinus {
    /* ... */

    public void setStorage(IStorage s) {
        /* ... */
    }
}
```

d) Wie sieht die Rollenverteilung (also Client, Target, Adapter, Adaptee) auf die Klassen/Schnittstellen der zweiten Implementierung aus?

e) In der zweiten Implementierung "weiß" die zentrale Editor-Klasse nicht mehr, welchen "Storage" sie konkret benutzen wird. Finden Sie heraus, wie man dieses Entwurfsprinzip nennt. Was sind die Vorteile davon?