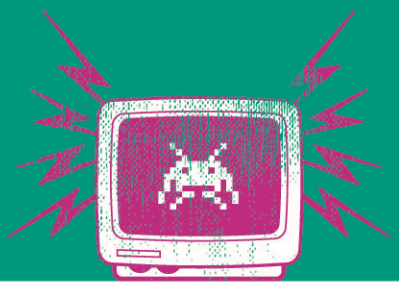
Informatik



BIF/SWE - Übungsbeispiel

Arthur Zaczek

Feb 2015

1 Allgemein

1.1 Ziele

Ziele dieses Übungsbeispieles ist es:

- **GUI**: Implementierung einer grafischen Oberfläche mit JavaFX oder WPF
- **UI-Komponente**: Implementierung einer eigenen grafischen Komponente
- **Presentation-/Viewmodelle**: Kapseln eines Models/Entität in eine ViewModel zur optimalen Aufbereitung für die UI. Anwenden von objektorientierten Grundsätzen.
- **Layer**: Kapselung von Aufgaben in Layer (BL, DAL, VM)

1.2 Ziele II

- **Design Patterns**: Anwenden mind. eines Design Patterns.
- **Logging**: Anwenden einer Logging-Komponente (log4j, log4net)
- **Reporting**: Erstellen von Berichten mit Hilfe einer geeigneten Bibliothek. Dazu müssen Sie eine geeignete Bibliothek auswählen und anwenden.

- **Codemetriken**: Einhaltung von Codemetriken mit Hilfe von Continuous Integration.

1.3 Aufgabenstellung

Implementieren Sie eine einfache Bilddatenbank “PicDB”.

- Verwalten von Bilder aus **einem** Verzeichnis
- Auslesen & Ändern der IPTC & EXIF Informationen des Bildes (**Simulieren!**)
- Speichern der IPTC & EXIF Informationen in einer Datenbank
- Verwaltung von Fotografen_innen in einer Datenbank
- Zuordnung der Fotografen_innen zu Bildern
- Suche nach Bilder anhand der IPTC, EXIF und Fotografen_innen Daten

2 Konkretes

2.1 GUI

- Die grafische Oberfläche muss nach den Vorgaben gestaltet sein
- Die Liste alle Bilder im unteren Bereich muss als wiederverwendbare Komponente implementiert werden

2.2 GUI - Gestaltung

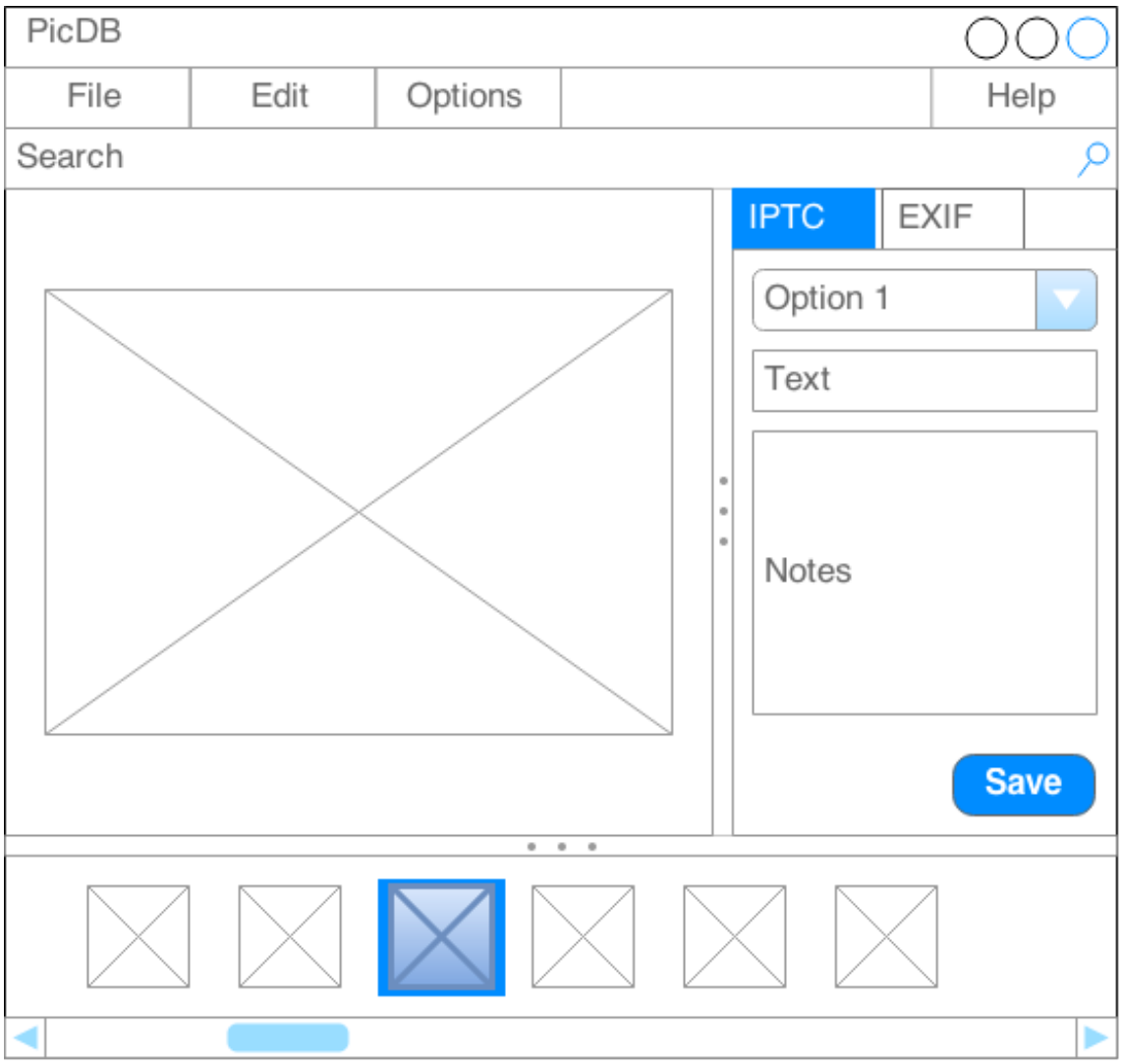


Figure 1: PicDB Mockup

2.3 UML

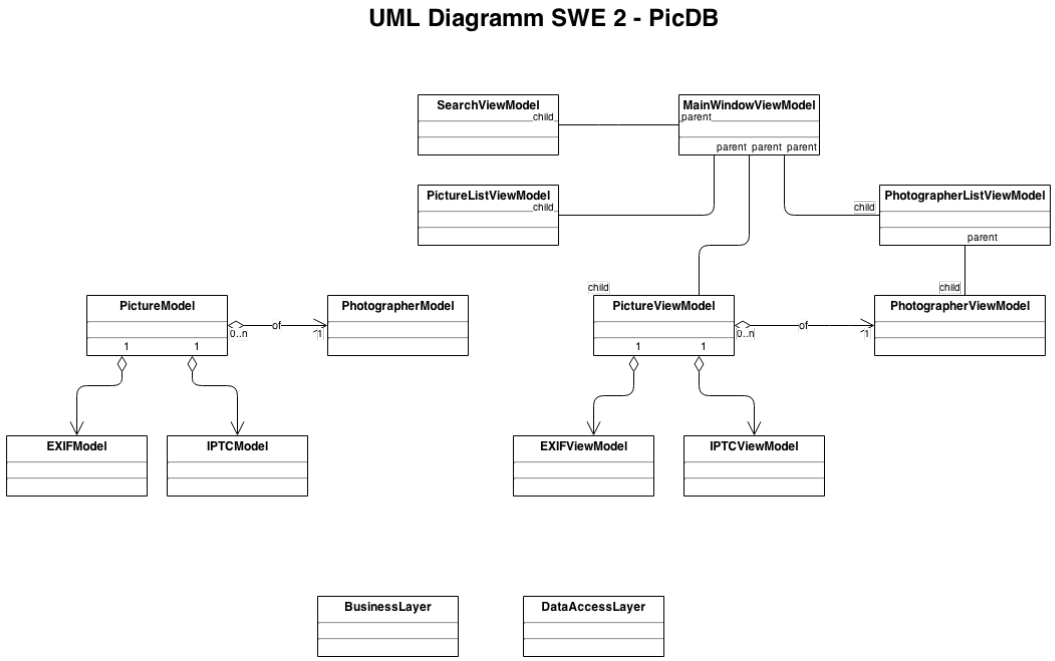


Figure 2: UML PicDB

2.4 Fotografen_innen

Fotografen_innen können über ein Menü aufgelistet und bearbeitet/neu angelegt werden.

Name	Typ	Validierung	Anmerkungen
Vorname	String(100)	-	Enthält auch zweiten Vornamen
Nachname	String(50)	Pflichtfeld	
Geburtstag	Datum	< Today()	
Notizen	Text	nein	Kann beliebig lang werden

2.5 Berichte

Bericht	Inhalt
Einzelbild	Ein einzelnes Bild wird gedruckt. Es werden die IPTC sowie EXIF Informationen gedruckt. Der oder die Fotograf_inn wird enfalls gedruckt.
Tags	Ein Bericht der eine Liste aller Tags mit der Anzahl der Fotos beinhaltet.

2.6 IPTC & EXIF Informationen

Diese Funktionalität ist als Mockup zu implementieren. Wenn Sie eine geeignete Bibliothek finden, steht es Ihnen frei, diese auch zu benutzen.

Unabhängig davon **müssen** diese Informationen in der Datenbank gespeichert werden (Cache, für die Suche). Diese Bibliothek würde also eine Synchronisierung durchführen.

2.7 Nicht Dokumentierte Anforderungen

Anforderungen, welche nicht explizit dokumentiert oder durch Unit-Test festgelegt sind, dürfen Sie selbst bestimmen, wie sie diese umsetzen.

Beispiele wären:

1. Das Aussehen der Berichte
2. Der Ort im Menü, von wo aus Fotograf_innen gelistet/bearbeitet/hinzugefügt werden
3. Das aussehen der Maske zum Bearbeiten von Fotograf_innen

Wenn Sie uns fragen, werden Sie Antworten erhalten.

2.8 20 eigene Unit Tests

Implementieren Sie 20 eigene Unit Tests. Testen Sie ihren Code genauer.

2.9 JavaDoc/Doxygen/XML Documentation Comments

(0 Punkte!)

Versehen Sie alle *public* Klassen/Methoden/Eigenschaften mit Kommentaren, aus denen mittels Tools eine API Dokumentation erstellt werden kann. Führen Sie das Tool Ihrer Wahl aus und zeigen Sie uns das Ergebnis.

Wenn Sie unsicher sind bzgl. des Tools dann fragen Sie uns.

Mögliche Tools:

- Doxygen
- Sandcastle

Falls Sie ein anderes Tool nutzen möchten, können Sie dies sehr gerne tun. Evtl. sprechen Sie sich mit uns ab.

Sie erhalten keine Punkte für diese Aufgabe. Es gehört aber zum guten Ton, seinen Code ausreichend zu Dokumentieren - zumindest die öffentliche Schnittstelle. Ihre Kolleginnen und Kollegen in großen Projekten werden es Ihnen danken.

2.10 Dokumentation

Ausgedruckt 1 ca. A4 Seite lang.

Inhalt

1. Benutzerhandbuch – Wie wird die Applikation verwendet
2. Lösungsbeschreibung – Wie wurde die Aufgabe gelöst
3. Worauf bin ich stolz
4. Was würde ich das nächste mal anders machen

2.11 Technologien

Bereich	Java	C#
UI-Technologie	JavaFX	WPF
Datenbank	JDBC	ADO.NET
Logging	log4j	log4net
Reporting	...	...

3 Bewertung

3.1 Automatisierte Tests

Die Übung wird automatisiert getestet und bewertet. Näheres finden Sie im Moodle-Kurs.

Die Unit-Tests geben Ihnen eine Struktur vor, *wie* sie die Anforderungen zu implementieren haben.

> Da die Aufgabe diesmal "komplexer" ist, müssen Sie alle Interfaces und Unit-Tests als instabil betrachten.

3.2 Codereview

In der letzten Übung findet ein Code-Review statt welches gesondert bewertet wird. Benötigt wird:

- Ein lauffähiges Programm (.exe, .jar)
- SourceCode

3.3 Notenrechner

Bewertung	Punkte	Max. Punkte
GUI		
Aussehen entspricht der Angabe		1
Bilder werden angezeigt		1
IPTC & EXIF Informationen werden bei Bild angezeigt		1
IPTC Informationen können bearbeitet werden		2
Liste alle Bilder ist eine UI-Komponente		2
Suche nach Bildern		1
Fotografen_innen		
Fotografen_innen auflisten		1
Fotografen_innen bearbeiten		2
Bilder zu Fotografen_innen zuordnen		1
Felder alle vorhanden		1
Validierung funktioniert		1
Berichte		
Einzelbild		1
Tags		1
Nichtfunktionale Anforderungen		
20 eigene Unittests		2
Konfiguration wird benutzt		1
Dokumentation vorhanden		1
Abzüge		
Sonstiges 1		
Sonstiges 2		
Sonstiges 3		
Summen	**0**	**20**

Figure 3: Notenrechner Codereview