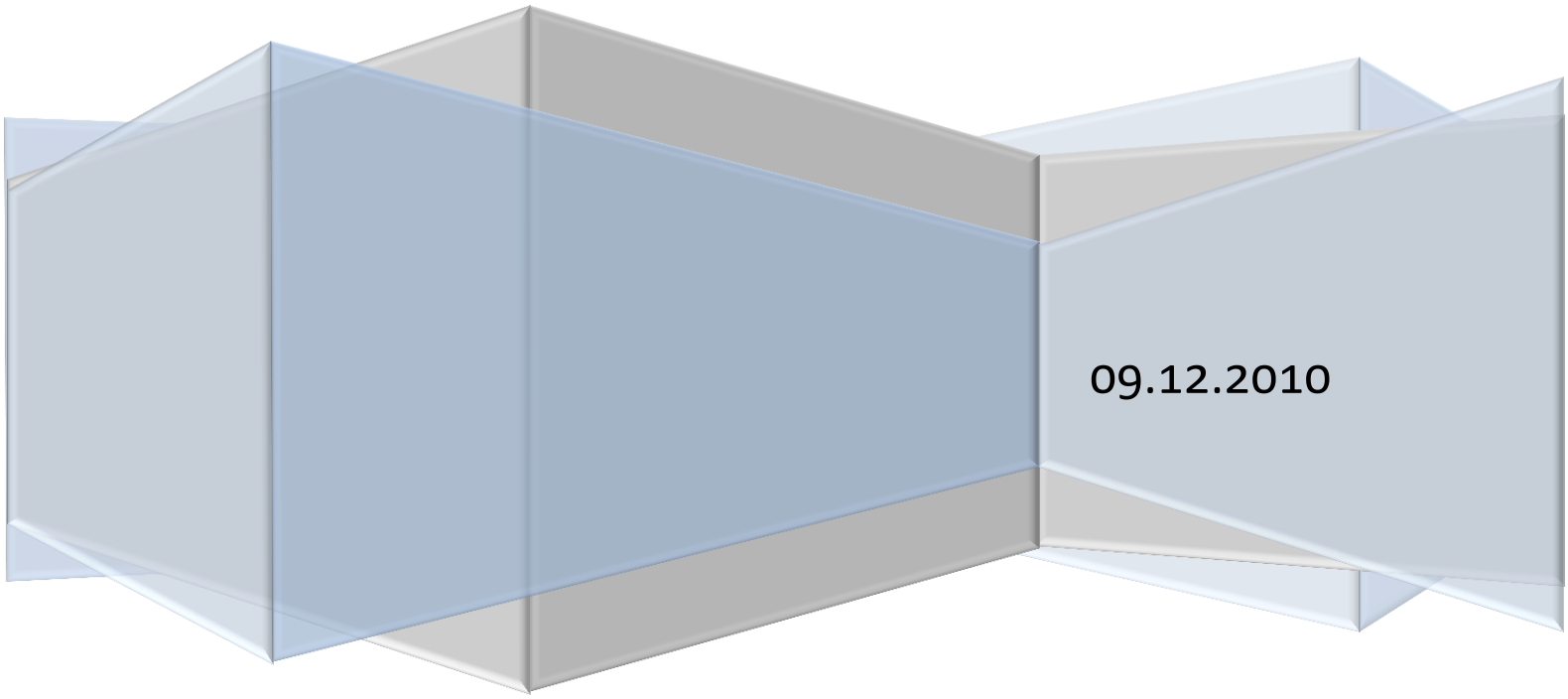
User Interfaces 1, HS 2010

BookMasterPro

Entwicklungsdokumentation

Christina Heidt & Lukas Elmer



09.12.2010

A Inhalt

B Allgemeines

Team

Name	Email
Christina Heidt	cheidt@hsr.ch
Lukas Elmer	lelmer@hsr.ch

Einleitung

Das Ziel des Projektes ist es, konkrete Erfahrungen mit Java Swing zu machen. Dafür gilt es ein User Interface für eine neue Bibliotheksoftware zu realisieren. Die Problem Domain und Business-Logik sind weitgehend vorgegeben, sind aber noch zu ergänzen und anzupassen.

Vorgaben

Vorgegebenes PDF: http://skripte.hsr.ch/Informatik/Fachbereich/User_Interfaces_1/UInt1/2-Uebungen/Vorgaben%20Miniprojekte%202010/UI1%20-%20Miniprojekt.pdf

Bewertungskriterien

- Stabil läuft
- Erwartungskonform verhält
- Code sauber geschrieben
- Geforderte Szenarien sauber umzusetzen
- Bewertungskriterien einzuhalten

Zeitplan

Woche 4

- Kickoff
- Implementierung der “Buch-Master (mit JList)”- und “Buch-Detail”-Wireframes (W1 & W2). Das Ziel ist es Erfahrungen mit der List-Komponente zu sammeln.
- Anwenden des Observer Pattern für die Kommunikation zwischen Master- und Detail-Views.

Woche 5

- Validierung: Die Buch-Detail-View um Validierung der Benutzereingaben erweitern.
- Korrektes En- und Disablen von Buttons abhängig vom Validierungskontext.
- Optionale Features: Commands & Undo, Help

Woche 6

- JTable: Die JTable Komponente der Detail-View durch eine JTable ersetzen. Umsetzung des Wireframes “Buch-Master (mit JTable)” (W3).
- Suche, Filtering & Sortierung: Implementation einer Suchfunktionalität mit Anzeige der Resultate in der JTable.
- Optionale Features II : CellRenderer & Editor

Woche 7

- Ausleihe: Die Wireframes “Ausleihe-Master” (W4) und “Ausleihe-Detail” (W5) implementieren. Um eine Ausleihe abzuschliessen muss der Kunde per Drop Down ausgewählt werden.
- Optionale Features III: Eine Master/Detail View für die Kundenverwaltung umsetzen.

Woche 8

- Rückgabe: Freies Umsetzen der Buchrückgabe-Funktionalität gemäss Rückgabe-Szenario 3.1 und 3.2.
- Optionale Features IV: Finden einer effizienten Lösung für das Rückgabe-Szenario 3.3 und 3.4.

Woche 9

- Vorbereitung für Review durch Betreuer
- Gruppeninterner Systemtest und Bugfixing oder Erarbeitung der optionalen Features zur Vorbereitung für das Review von nächster Woche. Dabei unbedingt die Szenarien beachten.

Woche 10

- Review der Applikation durch den Übungsbetreuer
- Zeigen des aktuellen Standes der Applikation am Übungsraum-PC oder eigenen Laptops (max. 10 Minuten). Jede Gruppe erhält ein kurzes Feedback.

Woche 11

- Feedback Umsetzung: Umsetzung der Verbesserungsvorschläge und Bugfixing.
- Umsetzung Optionale Features: Zeit für Umsetzung der Optionalen Features gemäss der Bewertungs-Kriterienliste oder dieser Tabelle.

Woche 12

- Abgabe Miniprojekt: bis spätestens Freitag 10. Dezember um 12:00 Uhr im IFS. Bitte Abgabekriterien beachten.

Journal

Struktur

Probleme

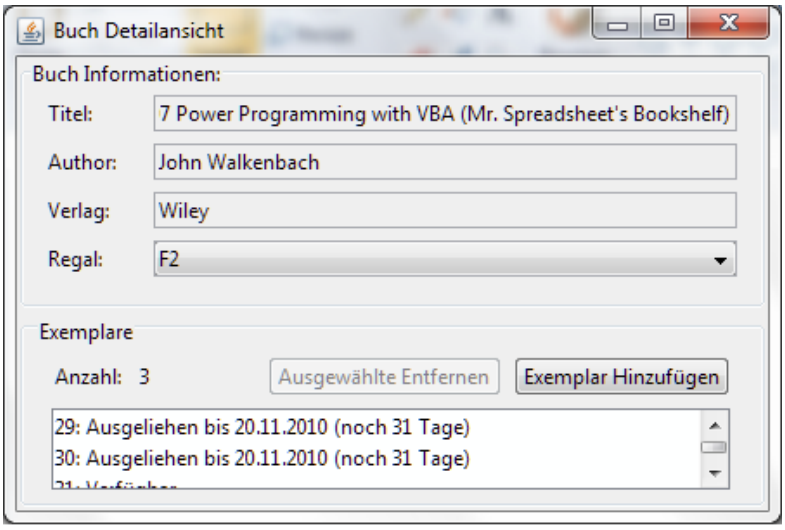
- Validierung von JGoodies verstehen
- Animations API von JGoodies verstehen und anwenden
- Durch GroupLayout wird viel Code generiert, der schwierig anzupassen ist -> durch FormLayout / FormBuilder von JGoodies ersetzen
- Forms von JGoodies verstehen: http://www.jgoodies.com/articles/forms.pdf

Woche 4

- Kickoff
- Einrichten des SVN / Eclipse etc.
- Christina: Kurztutorial SVN
- Analyse der vorgegebenen Library
- Struktur des Projektes besprochen und eingerichtet
- Lukas: Implementierung des “Buch-Master (mit JTable)”, Tab Bücher
- Christina: Implementierung des “Buch-Detail”-Wireframes
- Anwenden des Observer Patterns für die Kommunikation zwischen Master- und Detail-Views.

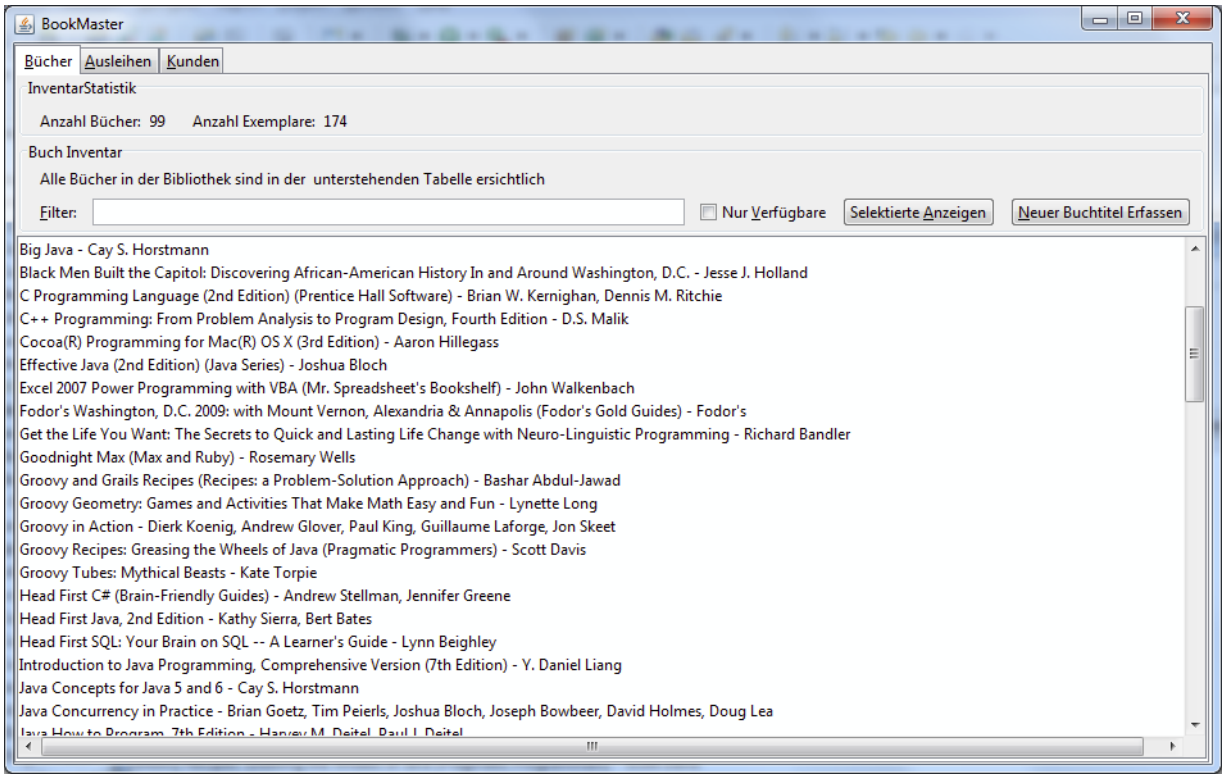
Woche 5

- Implementierung des “Buch-Master (mit JTable)”, Tab Ausleihen und Tab Kunden
- Implementierung Validierung, Buttons deaktivieren je nach Kontext
- Implementierung sortieren und filtern des “Buch-Master (mit JTable)”, alle Tabs
- Kunde erfassen, Kunde editieren
- Anwenden des Observer Patterns für die Kommunikation zwischen Master- und Detail-Views.
- Erste Implementierung des “Buch-Detail”-Wireframes mit JList:

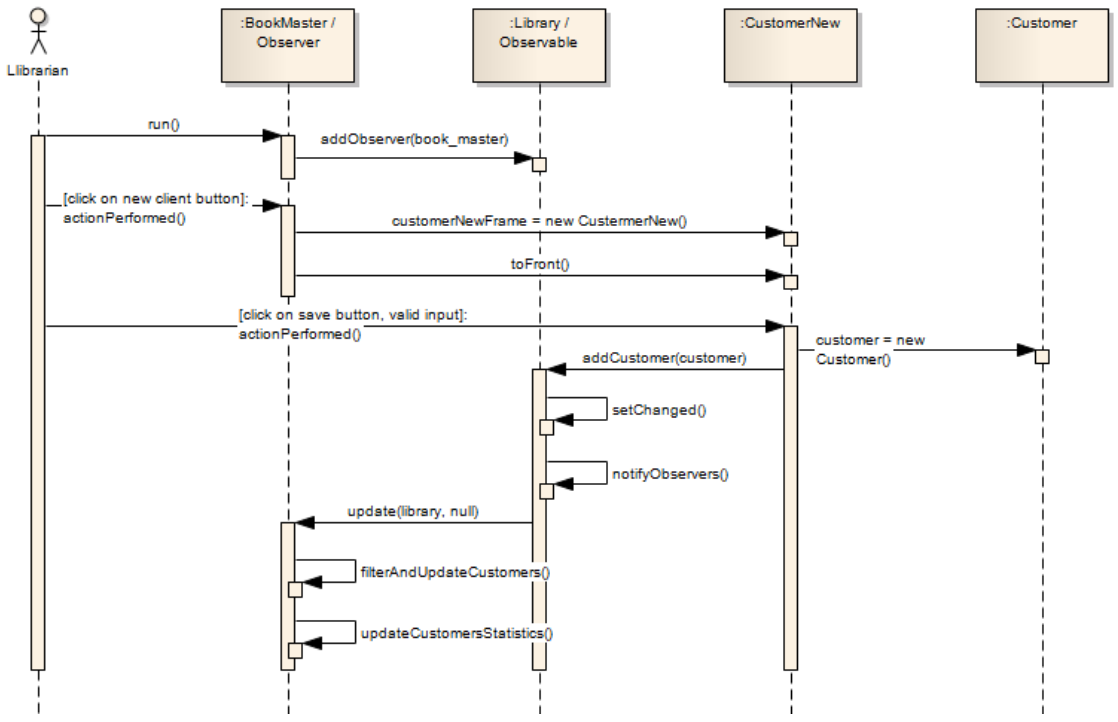


Woche 6

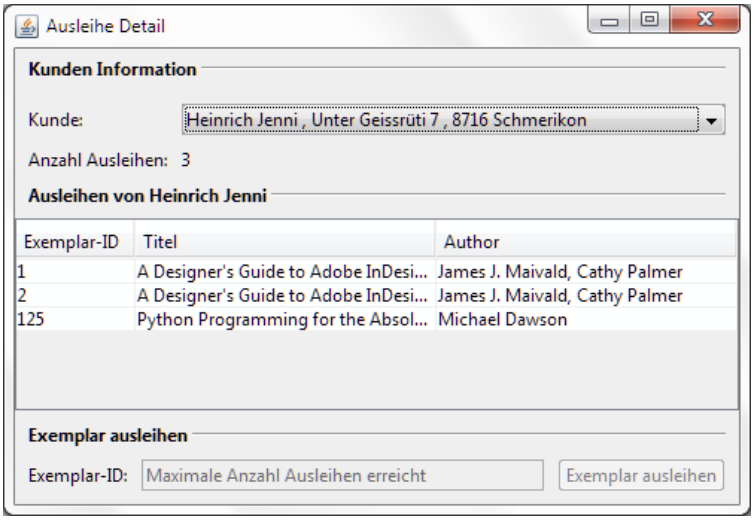
- Observer Pattern im *BookMasterPro* dokumentieren (mit Enterprise Architect)
- Rendering der JTable mit Farben ergänzen
- Filterung für Bücher, Ausleihen und Kunden verbessern
- JList bei Detail-View für Buch-Detail mit JTable ersetzen
- Implementierung des “Buch-Master (mit JList)”, Tab Bücher:



Oberver Pattern Sequenzdiagramm im BookMasterPro (Neuer Kunde erfassen, leicht vereinfacht)



Implementierung des "Ausleihe Detail"-Wireframes (mit JTable)

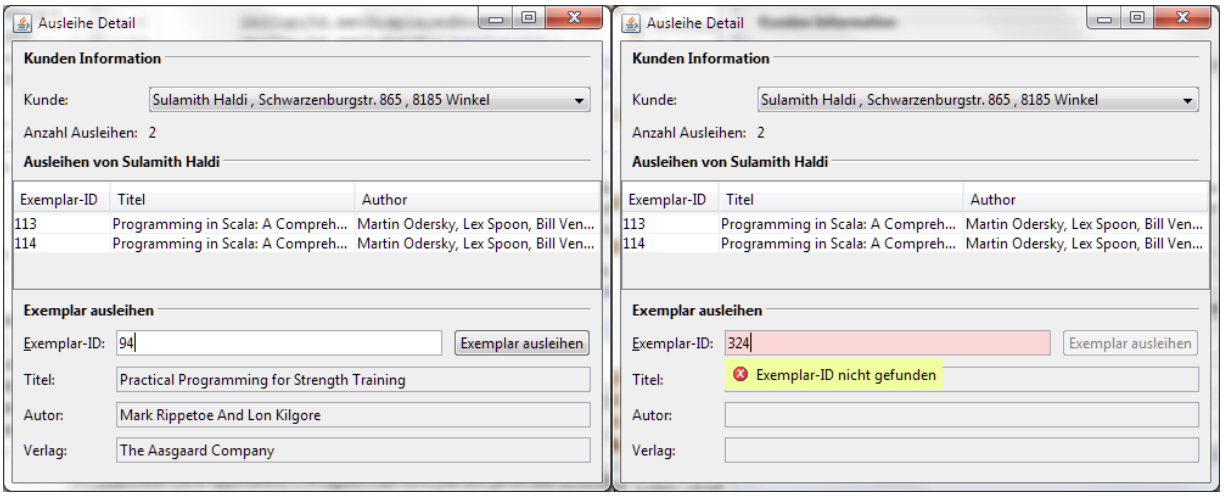


Woche 7

- Refactoring
- Extrahierung TableModel für JTable
- Update Libraries
- Default Column Sorting für JTables
- SplashScreen mit Animations
- Loan Detail ausgebaut (Kundenselektion + Observer Pattern) -> Die Wireframes “Ausleihe-Master” (W4) und “Ausleihe-Detail” (W5) implementiert. Um eine Ausleihe abzuschliessen muss der Kunde per Drop Down Menü ausgewählt werden.
- Book Detail neu gezeichnet
- Einsatz von FormLayout (JGoodies), da weniger / übersichtlicherer Code -> Anordnung / Verhalten des GUIs (Bsp. grow window) über 1 Zeile Code definierbar (http://www.jgoodies.com/articles/forms.pdf)

Woche 8

- Buch erstellen / bearbeiten mit Validierung
- SplashScreen aktualisiert (mit GlyphLabel)
- Evaluation: Optionale Features II : CellRenderer & Editor
    - CellEditor wollen wir nicht bei unseren Tabellen
    - CellRenderer braucht es nicht, bessere Lösung
- Buch Bearbeitung über das "Buch Detail" Wireframe
- Validierung einer Kopie im "Ausleihe Detail" Wireframe:



Woche 9

- Vorbereitung für Review durch Betreuer
- Gruppeninterner Systemtest und Bugfixing oder Erarbeitung der optionalen Features zur Vorbereitung für das Review von nächster Woche. Dabei unbedingt die Szenarien beachten.
- Refactoring
- Key Listener durch Document Listener ersetzen
- Icons für Tabs hinzugefügt
- CellEditor Demo implementiert (obwohl wir aus Usability-Gründen davon abraten würden)
- UI-Manager implementiert, der für die Frames verantwortlich ist
- Multiplatform Tests (Ubuntu)

Woche 10

- Rückgabe im BookMaster
- Accelerators & Mnemonics
- Hinzufügen von Icons & *BookMasterPro* Logo
- Exemplar kann als verloren markiert werden
- Mehrere Ausleihen können im Hauptfenster zurückgegeben werden
- Im Kunden Tab kann auf Ausleihen zugegriffen werden

Woche 11

- Feedback Umsetzung: Umsetzung der Verbesserungsvorschläge und Bugfixing.
- Umsetzung Optionale Features: Zeit für Umsetzung der Optionalen Features gemäss der Bewertungs-Kriterienliste oder dieser Tabelle.
- Refactoring
- Dokumentation schreiben (Word)

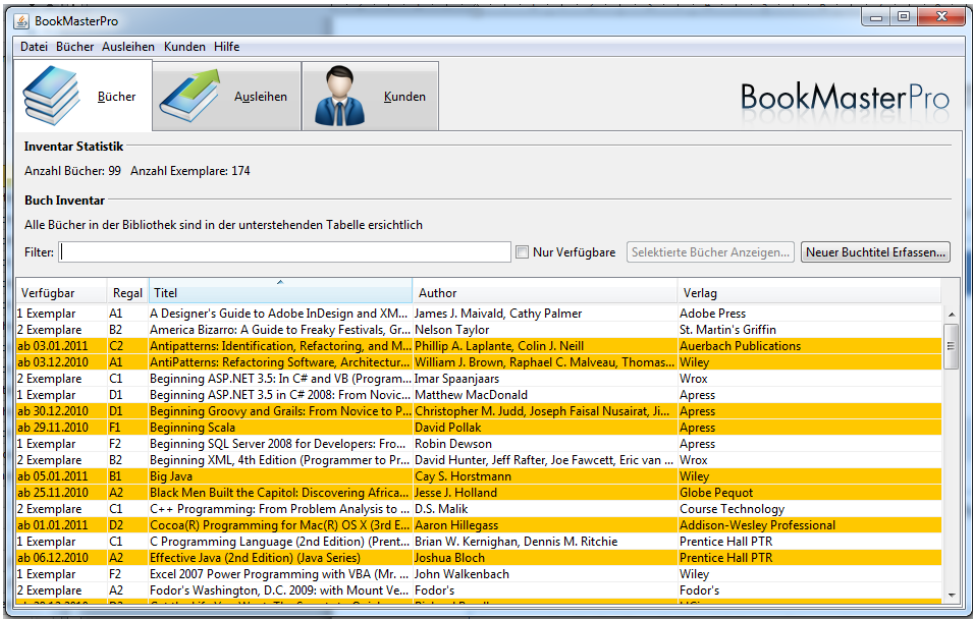
Woche 12

- Dokumentation schreiben (Word)
- Refactoring
- Druck Funktion hinzugefügt

C Szenarien

Da die Szenarios das Zentrale am UI-Projekt sind, haben wir den Ablauf der einzelnen Szenarios kurz beschrieben und mit Bildern illustriert.

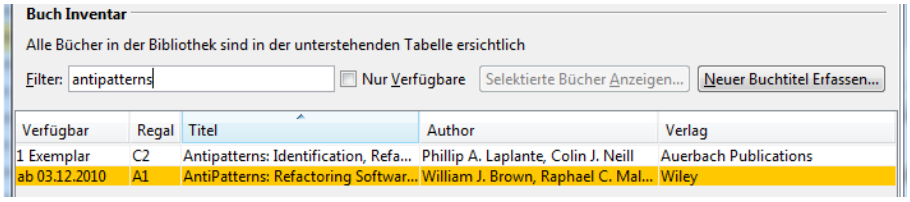
Der Ausgangspunkt für alle Szenarien ist der „BookMaster“ mit dem ausgewählten Tab „Bücher“. Der Cursor befindet sich im „Filter:“ Feld:



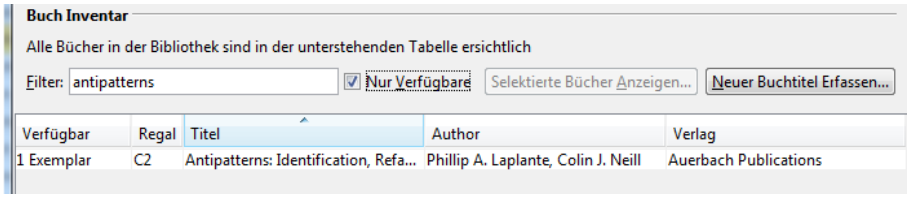
«Muss» Funktionalität

Szenario 1.1 (Buchverfügbarkeit prüfen & Buchstandort finden)

1. Eingabe des Buchtitels im Feld „Filter:“ (Bsp. „antipatterns“)
2. Buchtitel werden in der Tabelle angezeigt. Im Feld „Regal“ kann der Buchstandort herausgelesen werden. Ob sich zurzeit Exemplare in der Bibliothek befinden oder wann wieder Exemplare in der Bibliothek sein sollten, kann dem Feld „Verfügbar“ entnommen werden.



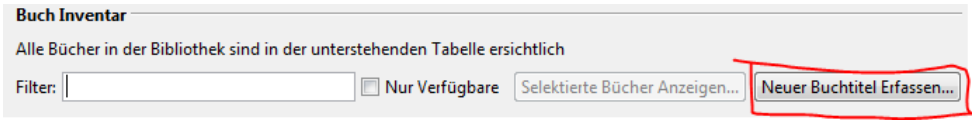
3. Sollen nur die verfügbaren Bücher angezeigt werden, so können diese durch die Check Box „Nur Verfügbare“ gefiltert werden:



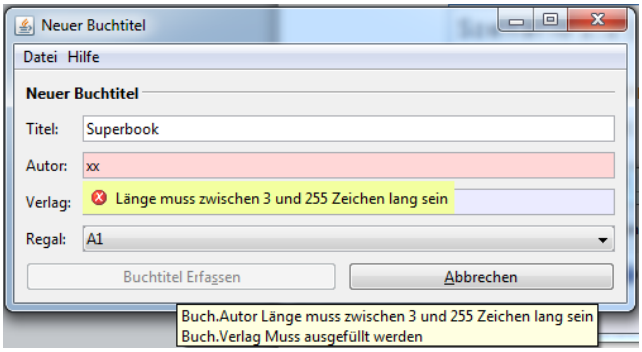
Szenario 1.2 (Buchtitel hinzufügen)

Buchtitel erfassen

1. Um einen neuen Buchtitel zu erfassen muss auf den Button „Neuer Buchtitel Erfassen…“ geklickt werden:



2. Ein neues Fenster „Neuer Buchtitel“ erscheint
3. Formular muss ausgefüllt werden (automatische Validierung, falls Eingabe fehlerhaft)
    a. Anzeige der Fehlermeldung direkt unter dem Textfeld
    b. Anzeige der gesamten Fehlermeldung als ToolTip beim Button „Buchtitel Erfassen“

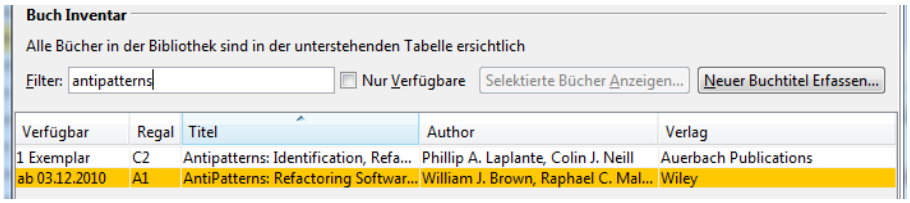


4. Sobald alle Felder richtig ausgefüllt wurden, wird der Button „Buchtitel Erfassen“ aktiviert und der neue Buchtitel kann gespeichert werden.

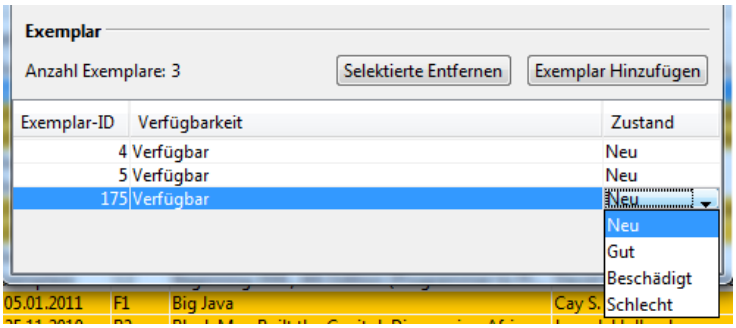
Exemplar zu bestehendem Buchtitel hinzufügen

Wenn der gewünschte Buchtitel bereits besteht und lediglich ein neues Exemplar hinzugefügt werden soll, so soll man folgendermassen vorgehen:

1. Eingabe des Buchtitels im Feld „Filter:“ (Bsp. „antipatterns“)
2. Buchtitel werden in der Tabelle angezeigt.



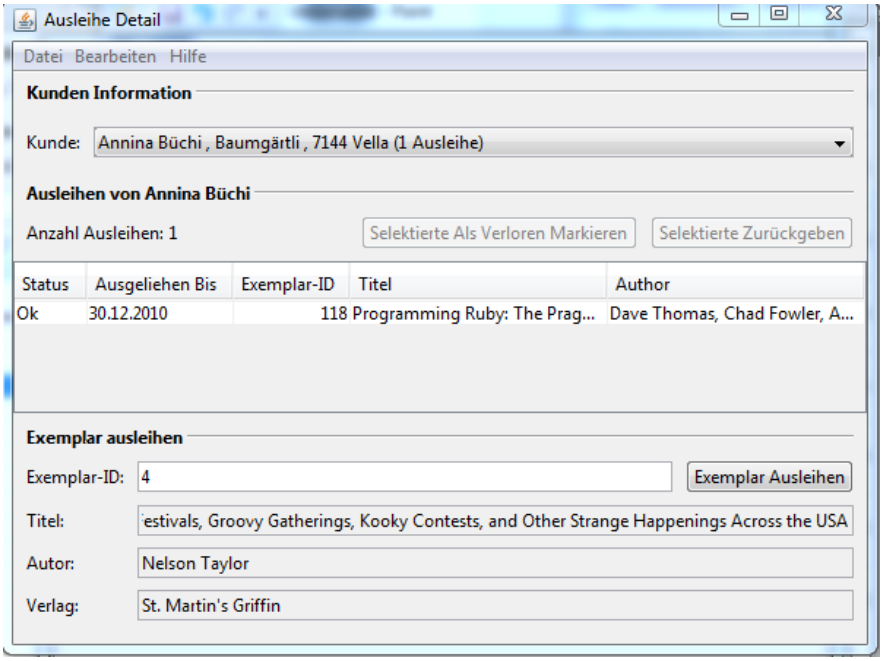
3. Den zu bearbeitenden Buchtitel markieren
4. Auf Button „Selektierte Bücher Anzeigen…“ klicken
5. Neues Fenster „Buch Detail“ wird geöffnet
6. Auf Button „Exemplar Hinzufügen“ klicken, um neues Exemplar hinzuzufügen
7. Durch einen CellEditor kann der Status des neu erfassten Exemplar bearbeitet werden:



Szenario 2.1 (Buch ausleihen)

Da der Kunde in der Bibliothek ein Exemplar auswählt, das mit einer Exemplar-ID angeschrieben ist, ist die Exemplar-ID bekannt. Wenn der Kunde noch nicht vorhanden ist, so muss er zuerst erstellt werden (siehe „Master-Detail View für Kundenverwaltung“). Dieses Szenario geht davon aus, dass der Kunde bereits erstellt wurde.

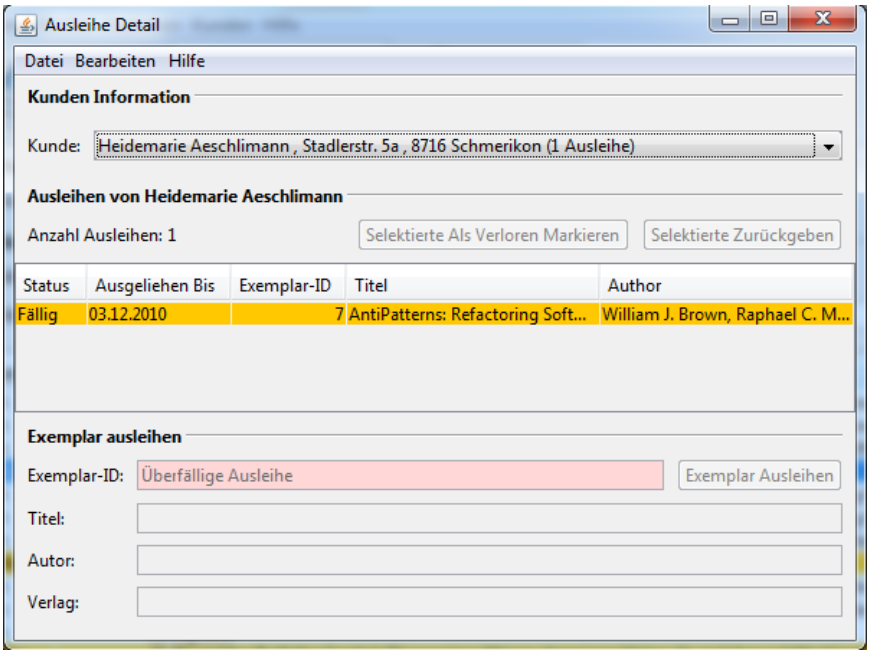
1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Pendente Ausleihen kontrollieren, falls vorhanden → im Fehlerfall am Kunden Feedback geben (Benutzer wird durch GUI unterstützt)
    a. ob Kunde weniger als 3 Ausleihen hat
    b. ob Kunde keine überfälligen Ausleihen hat
7. Exemplar-ID in Text Feld eingeben
8. Unter der Exemplar-ID werden weiter Buchinformationen angezeigt
    a. Kontrollieren, ob richtige Exemplar-ID eingegeben wurde
9. Auf Button „Exemplar Ausleihen“ klicken, um das Buch auszuleihen



Szenario 2.2 (Bücher ausleihen mit überfälligem Buch)

Ähnliches Szenario wie 2.1:

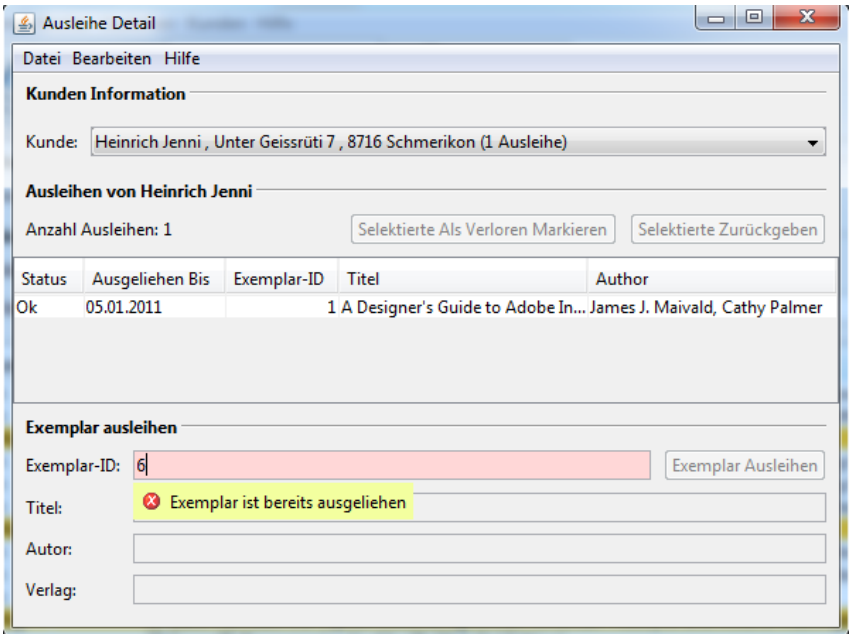
1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Pendente Ausleihen kontrollieren, falls vorhanden → im Fehlerfall am Kunden Feedback geben (Benutzer wird durch GUI unterstützt)
7. Kunde hat überfällige Ausleihe



Szenario 2.3 (Buch ausleihen welches schon ausgeliehen ist)

Ähnliches Szenario wie 2.1:

1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Pendente Ausleihen kontrollieren, falls vorhanden → im Fehlerfall am Kunden Feedback geben (Benutzer wird durch GUI unterstützt)
    a. ob Kunde weniger als 3 Ausleihen hat
    b. ob Kunde keine überfälligen Ausleihen hat
7. Exemplar-ID in Text Feld eingeben
8. Meldung „Exemplar ist bereits ausgeliehen“ wird angezeigt
9. Button „Exemplar Ausleihen“ ist deaktiviert



Szenario 2.4 (Mehrere Bücher ausleihen)

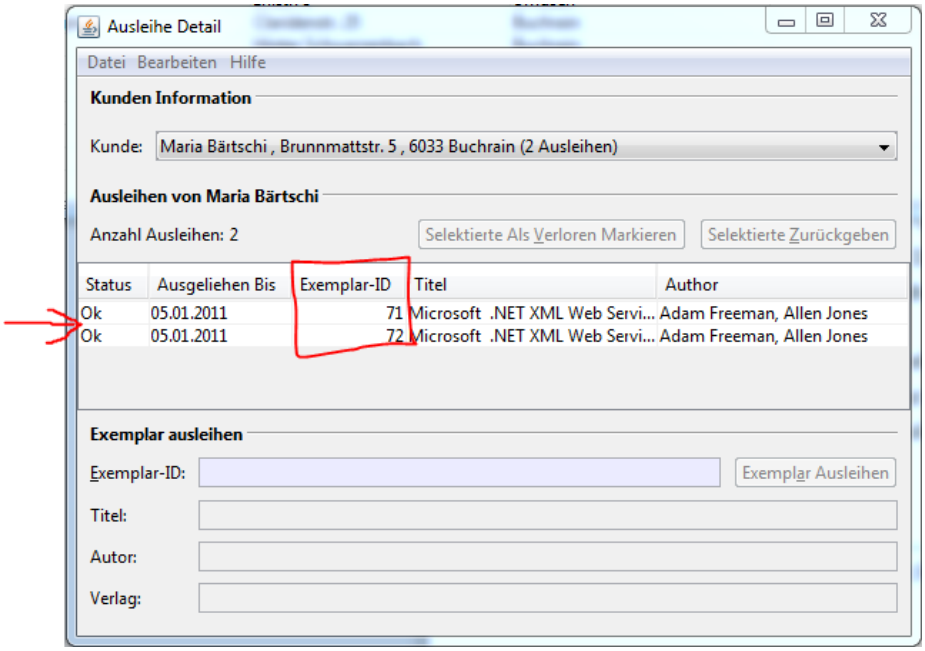
Ähnliches Szenario wie 2.1

1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Pendente Ausleihen kontrollieren, falls vorhanden → im Fehlerfall am Kunden Feedback geben (Benutzer wird durch GUI unterstützt)
    a. ob Kunde weniger als 3 Ausleihen hat
    b. ob Kunde keine überfälligen Ausleihen hat
7. Exemplar-ID in Text Feld eingeben
8. Unter der Exemplar-ID werden weiter Buchinformationen angezeigt
    a. Kontrollieren, ob richtige Exemplar-ID eingegeben wurde
9. Auf Button „Exemplar Ausleihen“ klicken, um das Buch auszuleihen
10. Falls weitere Exemplare an den Kunden ausgeliehen werden sollen, weiter bei 6

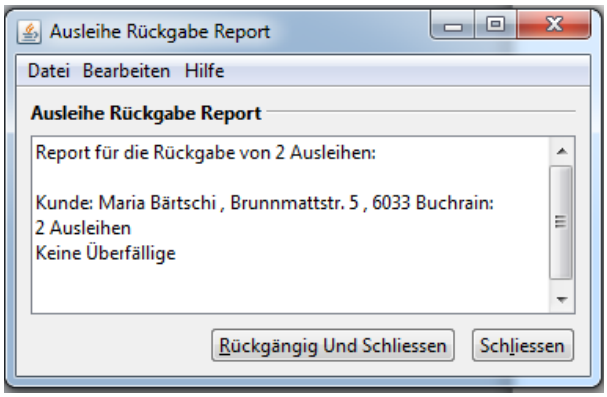
Szenario 3.1 (Buch zurückgeben)

Wie auch in den anderen Szenarien gibt es auch in diesem Szenario mehrere Wege, um das gewünschte Ziel zu erreichen. Nachfolgend ein möglicher Ablauf, um das Szenario zu erfüllen:

1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Gesuchte Exemplare sind in der Tabelle sichtbar:



7. Zurückzugebende(s) Exemplar(e) auswählen
8. Auf Button „Selektierte Zurückgeben“ klicken
9. Rückgabe bestätigen
10. Report für Ausleihe Rückgabe wird angezeigt:

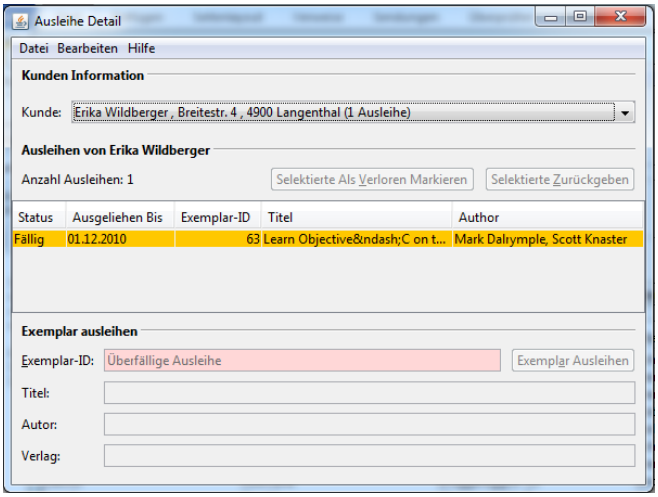


Szenario 3.2 (Buch zurückgeben welches überfällig ist)

Damit überfällige Ausleihen schnell erkannt werden können, sind diese mit oranger Farbe hinterlegt.

Ähnliches Szenario wie 3.1:

1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Gesuchte Exemplare sind in der Tabelle sichtbar, überfällige Ausleihen sind orange hinterlegt:



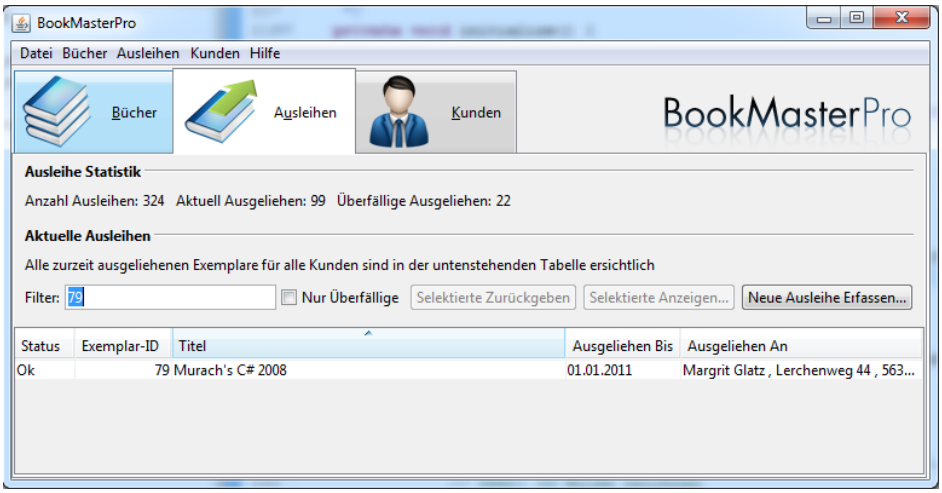
7. Zurückzugebende(s) Exemplar(e) auswählen
8. Auf Button „Selektierte Zurückgeben“ klicken
9. Rückgabe bestätigen
10. Report für Ausleihe Rückgabe wird angezeigt, überfällige Ausleihen werden gezählt:

«Kann» Funktionalität

Szenario 3.3 (Mehrere Bücher zurückgeben)

Wenn mehrere Exemplare zurückgegeben werden sollen, so wird das am Einfachsten durch die Ausleihe Übersicht im „BookMaster“ gemacht.

1. Wechseln zum Tab „Ausleihen“
2. Nach Exemplar-ID filtern (z.B. nach „79“):

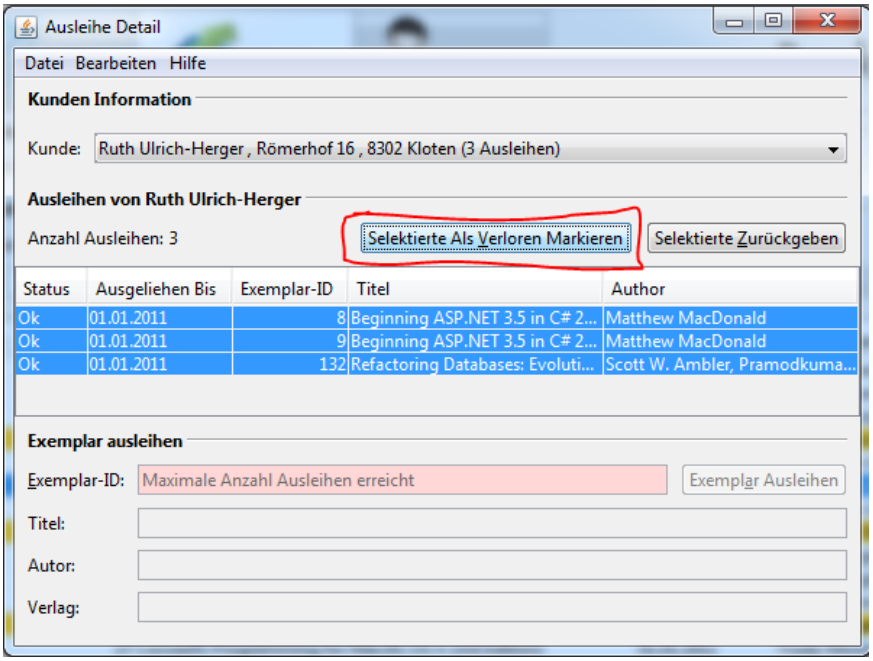


3. Zurückzugebendes Exemplar auswählen
4. Auf Button „Selektierte Zurückgeben“ klicken
5. Dialog bestätigen
6. Report für Ausleihe Rückgabe wird angezeigt:

Falls mehrere Exemplare markiert werden, können natürlich auch gleich mehrere Exemplare zurückgegeben werden.

Szenario 3.4 (Buch zurückgeben welches nicht der Ausleihe entspricht)

In diesem Szenario geht es darum, dass der Kunde eine Ausleihe verloren hat. Da das Szenario sehr ähnlich ist wie „3.1 Ausleihe zurückgeben“ haben wir einen Button „Ausleihe Verloren“ im Kontext zur Rückgabe platziert:



Ansonsten ist das Szenario sehr ähnlich wie 3.1:

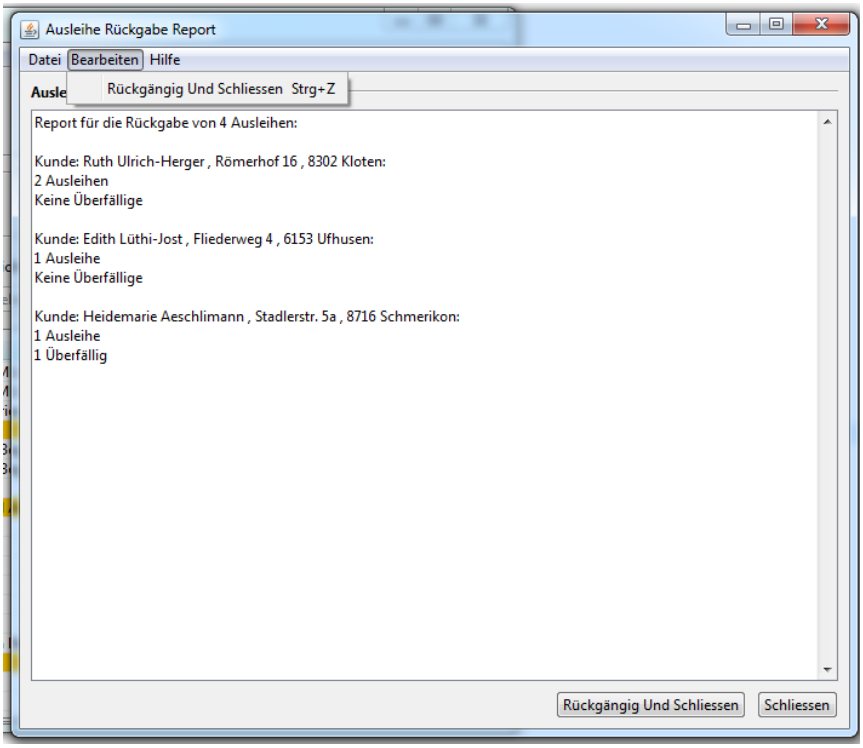
1. Wechseln zum Tab „Kunden“
2. Kunde Filtern (z.B. nach „büchi“)
3. Kunde „Büchi Annina“ auswählen
4. Auf Button „Ausleihe Detail Für Kunden Anzeigen…“ klicken
5. Neues Fenster „Ausleihe Detail“ wird geöffnet
6. Gesuchte Exemplare sind in der Tabelle sichtbar, überfällige Ausleihen sind orange hinterlegt:
7. Zurückzugebende(s) Exemplar(e) auswählen
8. Auf Button „Selektierte Als Verloren Markieren“ klicken
9. Eingabe bestätigen
10. Die Ausleihe für den Kunden ist nun abgeschlossen und das Exemplar wurde gelöscht. Um ein neues Exemplar hinzuzufügen, siehe Szenario 1.1

«Optionale» Funktionalität

Commands & Undo/Redo

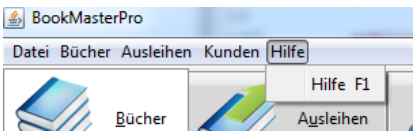
Für gewisse Szenarien ist es sehr schwierig, ein gutes Undo/Redo zu implementieren, das später auch wirklich gebraucht wird. Da der Aufwand dafür sehr gross sein kann, haben wir nur an einer Stelle im Miniprojekt ein Undo implementiert.

Da die Wahrscheinlichkeit für einen Missgriff beim Ausleihe zurückgeben eher gross ist, haben wir dort ein Undo implementiert:



Help

Um zu demonstrieren, wie die Hilfe aufgerufen werden kann, haben wir eine Action implementiert (F1) und ein Menu Eintrag hinzugefügt. Es wurde aber keine Benutzerhilfe geschrieben, deshalb wird, statt dass die Hilfe angezeigt wird, ein Popup angezeigt.



CellRenderer

Für den "Book Master" hatten wir zuerst gedacht, dass wir einen CellRenderer brauchen, um bestimmte Zeilen einzufärben. Allerdings haben wir dafür einen anderen Weg gefunden, ohne einen CellRenderer zu implementieren:

```
tblBooks = new JTable() {
    private static final long serialVersionUID = -6660470510160948438L;

    @Override
    public Component prepareRenderer(TableCellRenderer renderer, int row, int column) {
        Component c = super.prepareRenderer(renderer, row, column);
        if (!isCellSelected(row, column)) {
            Color col = colorForRow(row);
            c.setBackground(col != null ? col : UIManager.getColor("Table.background"));
            c.setForeground(UIManager.getColor("Table.foreground"));
        } else {
            c.setBackground(UIManager.getColor("Table.selectionBackground"));
            c.setForeground(UIManager.getColor("Table.selectionForeground"));
        }
        return c;
    }

    private Color colorForRow(int row) {
        Book b = (Book) getModel().getValueAt(convertRowIndexToModel(row), -1);
        return library.getAvailibleCopiesOfBook(b).size() == 0 ? Color.ORANGE : null;
    };
};
```

Bei dieser Lösung hat man den Vorteil, dass nicht für jede einzelne Spalte ein CellRenderer hinzugefügt werden muss. Da sich in unseren Tabellen eine ganze Zeile immer auf ein bestimmtes Objekt bezieht (z.B. auf ein Buch), haben wir diese Lösung bevorzugt.

Um zu zeigen, dass wir auch den CellRenderer beherrschen, haben wir eine Demoversion dafür erstellt, die in der Klasse „LoanDetailWithCellRenderer" gefunden werden kann:

```
tblLoans = new JTable();
tblLoans.setModel(loanTableModel);

for (int i = 0; i < tblLoans.getColumnModel().getColumnCount(); ++i) {
    tblLoans.getColumnModel().getColumn(i).setCellRenderer(new DefaultTableCellRenderer() {
        private static final long serialVersionUID = 439302311044429919L;

        @Override
        public Component getTableCellRendererComponent(JTable table, Object value, boolean isSelected, boolean hasFocus, int row,
                int column) {
            Component c = super.getTableCellRendererComponent(table, value, isSelected, hasFocus, row, column);
            if (!isSelected) {
                Color col = colorForRow((Loan) loanTableModel.getValueAt(row, -1));
                c.setBackground(col != null ? col : UIManager.getColor("Table.background"));
                c.setForeground(UIManager.getColor("Table.foreground"));
            } else {
                c.setBackground(UIManager.getColor("Table.selectionBackground"));
                c.setForeground(UIManager.getColor("Table.selectionForeground"));
            }
            return c;
        }

        private Color colorForRow(Loan l) {
            return l.isOverdue() ? Color.ORANGE : null;
        };

    });
}
```

In der Klasse „BookMasterUiManager" kann dieses Fenster aktiviert werden, indem man die folgenden Zeilen ändert:

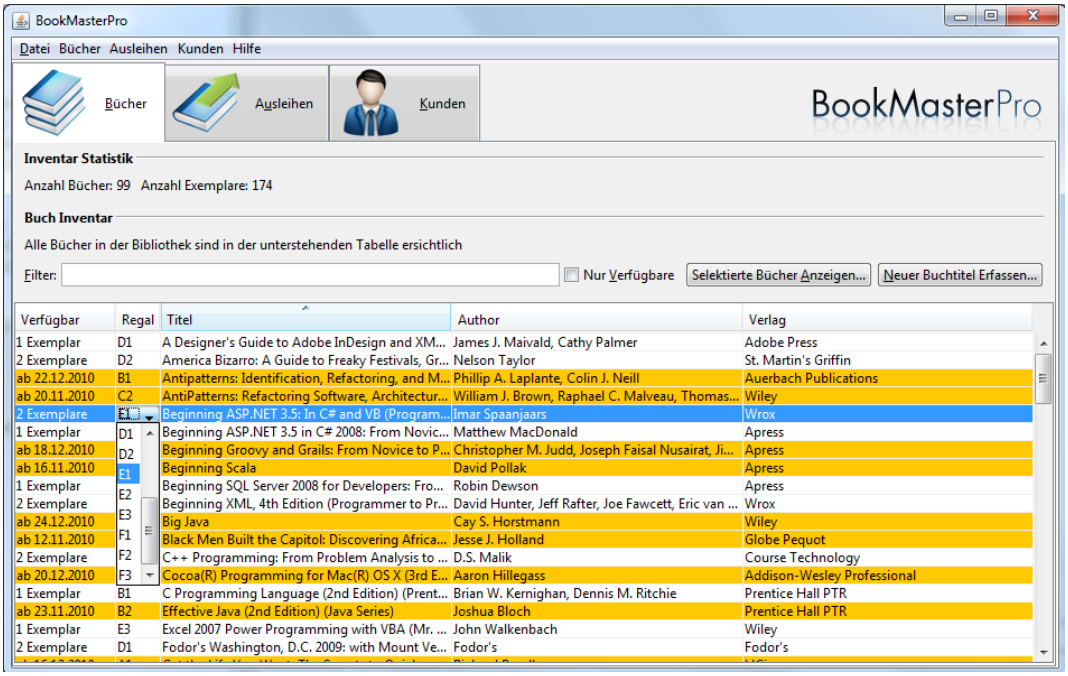
```
protected LoanDetail loanDetailFrame;
// Demo: CellRenderer
// protected LoanDetailWithCellRenderer loanDetailFrame;


loanDetailFrame = new LoanDetail(this, c);
// Demo: CellRenderer
// loanDetailFrame = new LoanDetailWithCellRenderer(this, c);
```

CellEditor

CellEditor im „BookMaster“

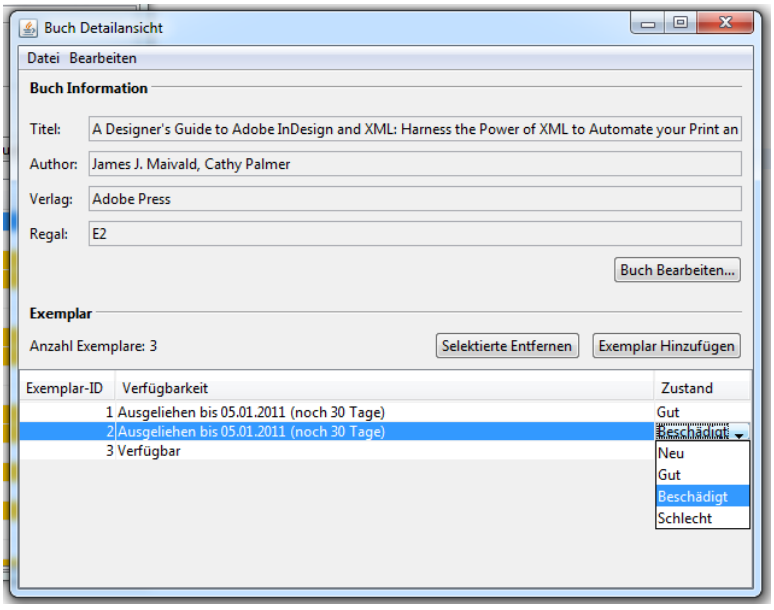
Um zu zeigen, dass wir den CellRenderer und den CellEditor beherrschen, haben wir als Demo einen CellEditor im BookMaster implementiert:



Wir empfehlen aber explizit, dass dieser CellEditor wieder entfernt wird. Denn es soll vermieden werden, dass ein Benutzer einen Fehler macht und ein Buch unabsichtlich bearbeitet (Missgriff).

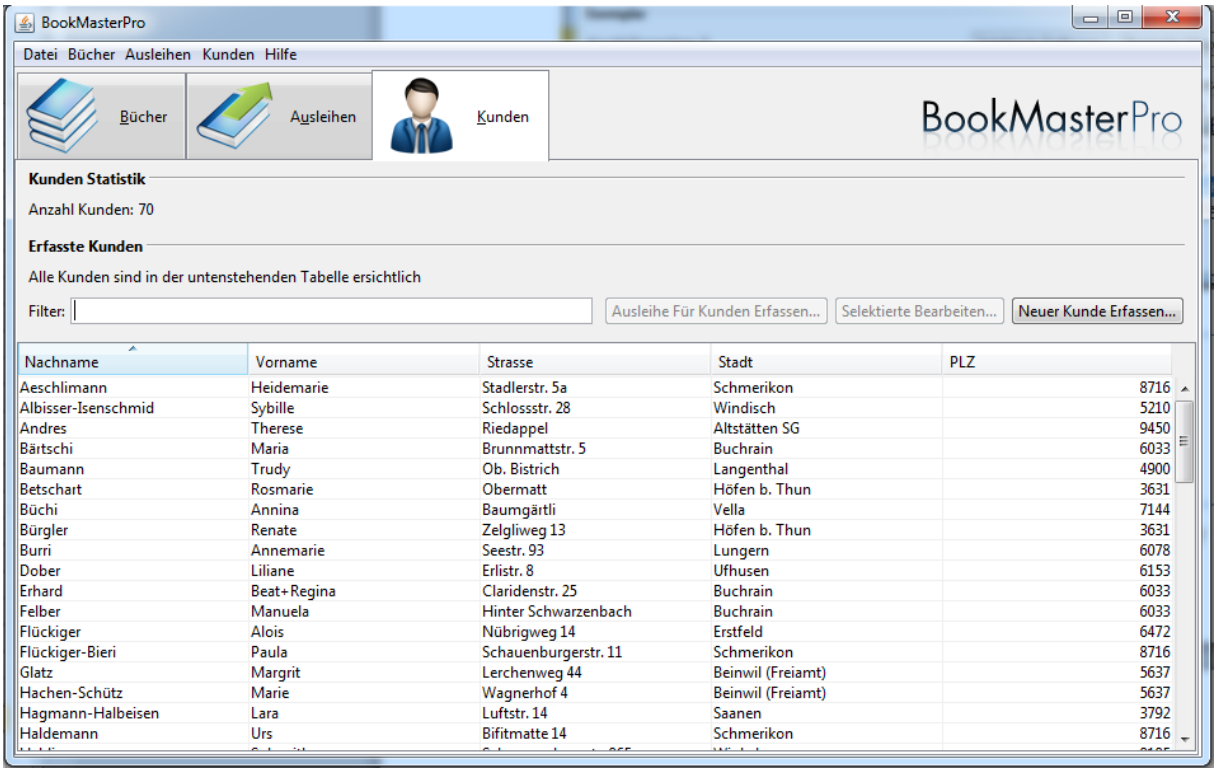
CellEditor im „Buch Detail“, um Exemplar zu bearbeiten

Da das Exemplar nur ein Feld hat, das bearbeitet werden kann, haben wir uns entschieden, dafür einen CellEditor zu verwenden, damit das Exemplar direkt in der Tabelle bearbeitet werden kann:

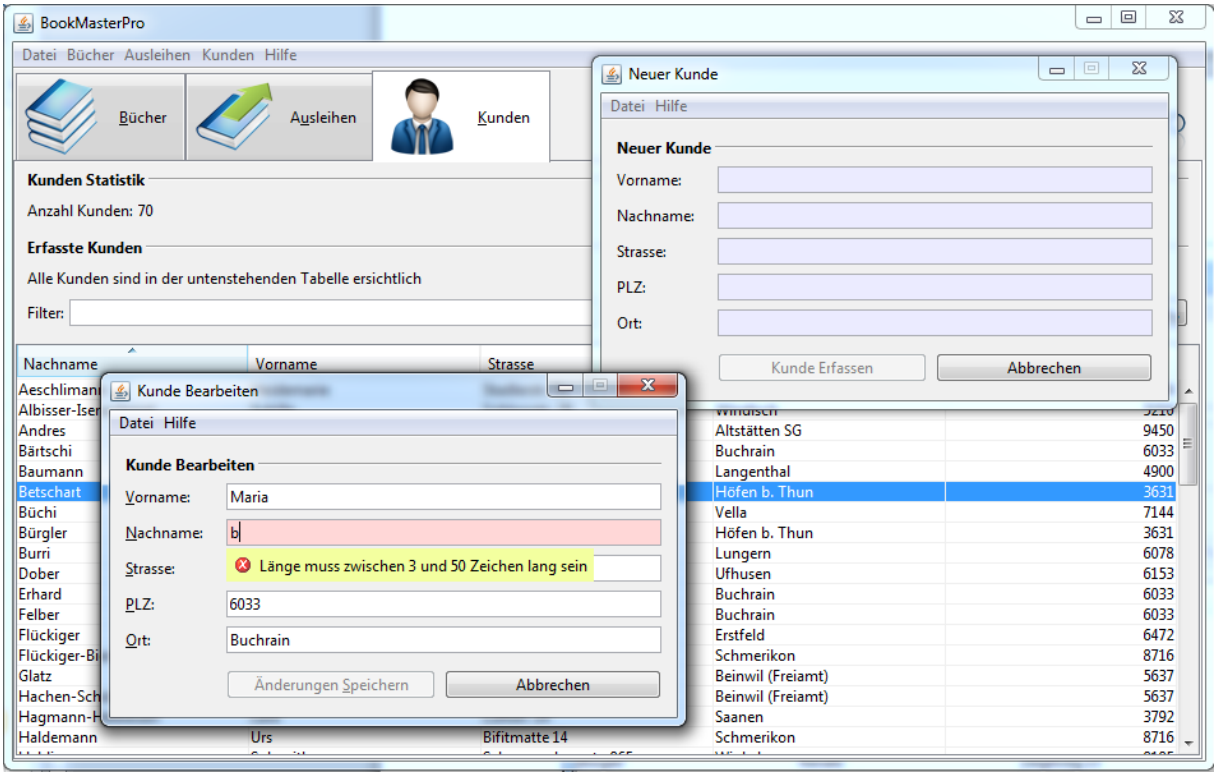


Master-Detail View für Kundenverwaltung

Die Master View für die Kundenverwaltung sieht wie folgt aus:

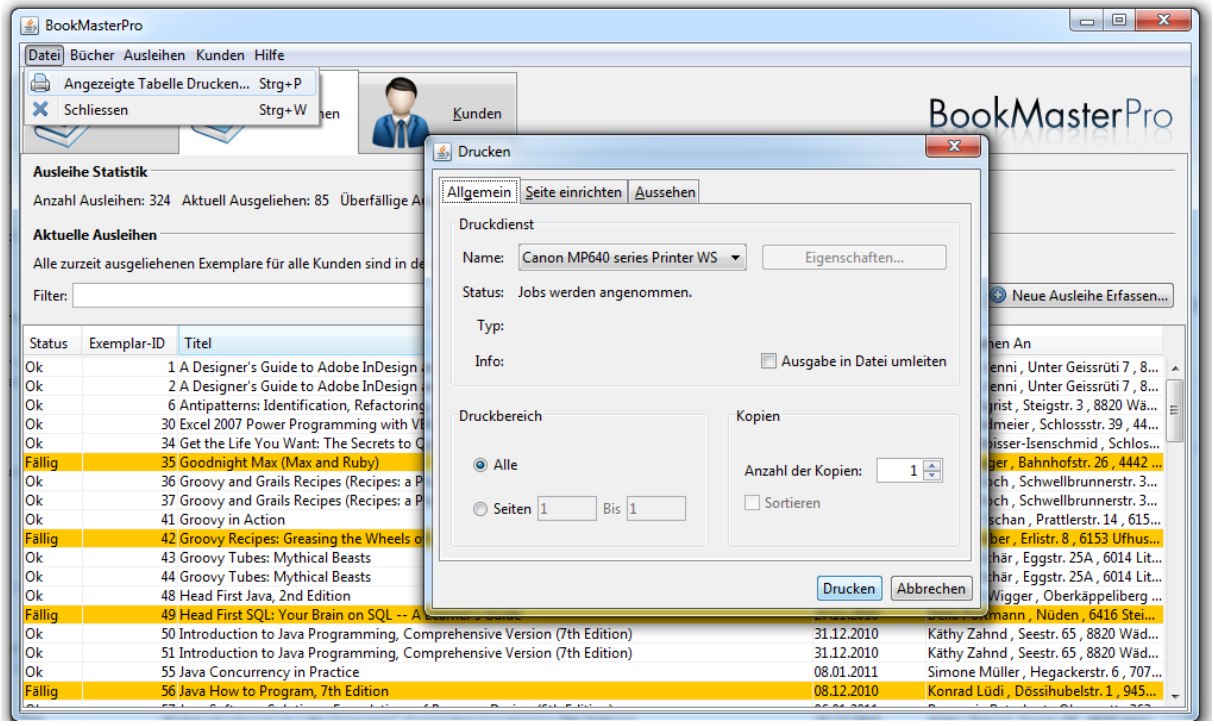


Die Detail View für die Kunden sieht so aus (oben rechts: „Neuer Kunde“, unten links: „Kunde Bearbeiten“):



Printing

Die verschiedenen Tabellen zu den Büchern, Ausleihen und Kunden können ausgedruckt werden, sowie der Ausleihe Rückgabe Report.



Internationalisierung / Mehrsprachigkeit

Wegen zu viel Aufwand nicht implementiert.

Breadcrumbs

Nicht implementiert, da Breadcrumbs nicht zu unserem UI passen.

Trashcan für gelöschte Objekte

Wegen zu viel Aufwand nicht implementiert. Die Library (Model) würde gewisse weitere Funktionalität bereitstellen, wie z.B. abgeschlossene Ausleihen ausgeben.

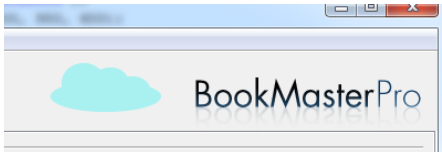
Java 2D eingesetzt

Splash Screen

Für den SplashScreen haben wir zwar nicht direkt Java 2D eingesetzt, aber indirekt durch die JGoodies.

Logo BookMasterPro

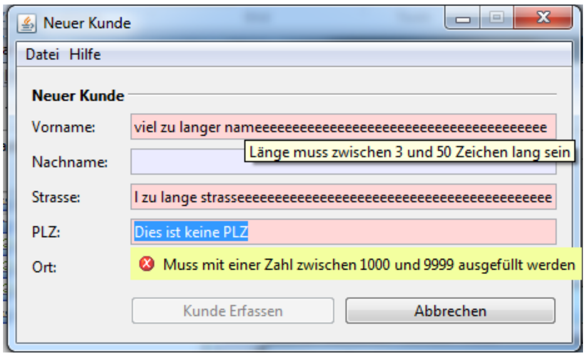
Um das Logo zu zeichnen haben wir Java 2D eingesetzt. Als Demo haben wir auch eine Wolke gezeichnet. Die Demo kann einkommentiert werden:



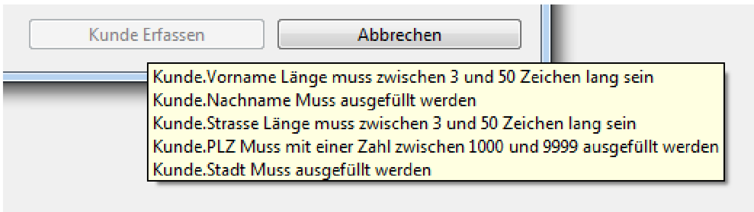
```java
protected void paintComponent(Graphics g) {
    super.paintComponent(g);
    Graphics2D g2 = (Graphics2D) g;
    g2.drawImage(new ImageIcon("data/icons/logo2.png").getImage(), this.getWidth() - 227, 3, this);
    // Demo: 2D Wolke zeichnen
    //g2.setColor(new Color(100, 240, 240, 130));
    //g2.setRenderingHint(java.awt.RenderingHints.KEY_ANTIALIASING, java.awt.RenderingHints.VALUE_ANTIALIAS_ON);
    //java.awt.geom.Ellipse2D cloudPuff1 = new java.awt.geom.Ellipse2D.Double(0, 40, 250, 70);
    //java.awt.geom.Ellipse2D cloudPuff2 = new java.awt.geom.Ellipse2D.Double(25, 0, 100, 90);
    //java.awt.geom.Ellipse2D cloudPuff3 = new java.awt.geom.Ellipse2D.Double(95, 10, 80, 60);
    //java.awt.geom.Ellipse2D cloudPuff4 = new java.awt.geom.Ellipse2D.Double(145, 30, 80, 60);
    //java.awt.geom.Area cloud = new java.awt.geom.Area(cloudPuff1);
    //cloud.add(new java.awt.geom.Area(cloudPuff2));
    //cloud.add(new java.awt.geom.Area(cloudPuff3));
    //cloud.add(new java.awt.geom.Area(cloudPuff4));
    //java.awt.geom.AffineTransform t = new java.awt.geom.AffineTransform();
    //t.translate(this.getWidth() - 350, 10);
    //t.scale(0.4, 0.4);
    //cloud.transform(t);
    //g2.fill(cloud);
};
```

Validierung mit JGoodies

Für die Validierung haben wir die JGoodies verwendet. Durch kontextsensitive Meldungen ist dem Benutzer sofort klar, was er falsch eingegeben hat und wie er die Eingabe verbessern kann. Durch Farben (blau = Pflichtfeld, rot = fehlerhafte Eingabe) ist dem Benutzer auf einen Blick klar, welche Felder ausgefüllt oder korrigiert werden müssen.



Auf dem Button sind ausserdem die nicht oder fehlerhaft ausgefüllten Felder nochmals aufgelistet durch einen ToolTip.



D Accelerators und Mnemonics

BookMaster Hauptfenster

Anwendung	Mnemonic	Accelerator
Schliessen	L	W
Tab Bücher	**B**	
Filter	F	
Nur Verfügbare	V	
Selektierte Bücher Anzeigen...	A	E
Neuer Buchtitel Erfassen...	N	B
Tab Ausleihen	**U**	
Filter	F	
Nur Überfällige	U	
Selektierte Zurückgeben...	R	R
Selektierte Anzeigen...	A	D
Neue Ausleihe Erfassen...	N	U
Tab Kunden	**K**	
Filter	F	
Ausleihe Detail für Kunden anzeigen...	E	L
Selektierte Bearbeiten...	A	H
Neuer Kunde Erfassen...	N	K

Buch Detailansicht

Anwendung	Mnemonic	Accelerator
Schliessen	L	W
Buch Bearbeiten...	B	I
Selektierte Entfernen	E	R
Exemplar Hinzufügen	H	H

Ausleihe Detailansicht

Anwendung	Mnemonic	Accelerator
Schliessen	L	W
Selektierte als Verloren Markieren	V	M
Selektierte Zurückgeben	Z	R
Exemplar Ausleihen	A	N

Neuer Buchtitel

Anwendung	Mnemonic	Accelerator
Buchtitel Erfassen	S	S
Schliessen	L	W

Buchtitel Bearbeiten

Anwendung	Mnemonic	Accelerator
Änderungen Speichern	S	S
Schliessen	L	W

Neuer Kunde

Anwendung	Mnemonic	Accelerator
Kunde Erfassen	S	S
Schliessen	L	W

Kunde Bearbeiten

Anwendung	Mnemonic	Accelerator
Änderungen Speichern	S	S
Schliessen	L	W

Ausleihe Rückgabe Report

Anwendung	Mnemonic	Accelerator
Schliessen	L	W
Rückgängig Und Schliessen	R	Z

E Weitere mögliche Features, die implementiert werden könnten

Wenn diese Bibliothek tatsächlich verwendet werden sollte, dann sollten noch einige weitere Features implementiert werden wie:

- Business Logik auch in der Library prüfen und bei Gui-Fehlverhalten Exceptions werfen
- Ausleihe / Buchtitel / Kunde Löschen
- Trash Scan
- Log
- Datenexport
- Internationalisierung
- Undo / Redo
- Klick rechte Maustaste mit kontextsensitiven Menu
- Usw.

F Zusammenfassung

Es wäre möglich, noch sehr viel mehr zu entwickeln, Prozesse zu optimieren, und weitere Features zu implementieren. Bei der Entwicklung von *BookMasterPro* haben wir viel gelernt. Obwohl die Arbeit sehr zeitaufwändig war, hat uns das Projekt viel Spass bereitet, und wir hoffen natürlich, dass sich dies auch in der Benutzung des Programmes wiederspiegelt.