Technische Hochschule Ingolstadt
Fakultät für Elektrotechnik und Informatik

Praktikum „WaWi“
Vorführaufgabe 4

Behandelte Themen

Vertiefung Java FX: Dialog-Verkettung

Vorbereitung

Kopieren Sie bitte Ihr bisheriges Eclipse-Projekt und benennen Sie die Kopie „`WaWi04`“ (nutzen Sie hierzu einfaches „copy and paste“ im „Package Explorer“ von Eclipse) und erweitern Sie die Inhalte des kopierten Projekts entsprechend nachfolgender Aufgaben.

Aufgabe 1 – Entwicklung eines Login-Dialogs

In diesem Aufgabenblatt geht es darum, den Zugriff auf den POS-Dialog aus Aufgabenblatt 3 zu schützen, so dass nur autorisierte Benutzer den Dialog bedienen können. Beim Start der Anwendung soll daher nicht wie bisher sofort der POS-Dialog angezeigt werden, sondern es soll ein Login-Dialog erscheinen (s. Abbildung 1). Nach erfolgreichem Login gelangt der Benutzer dann zum POS-Dialog.

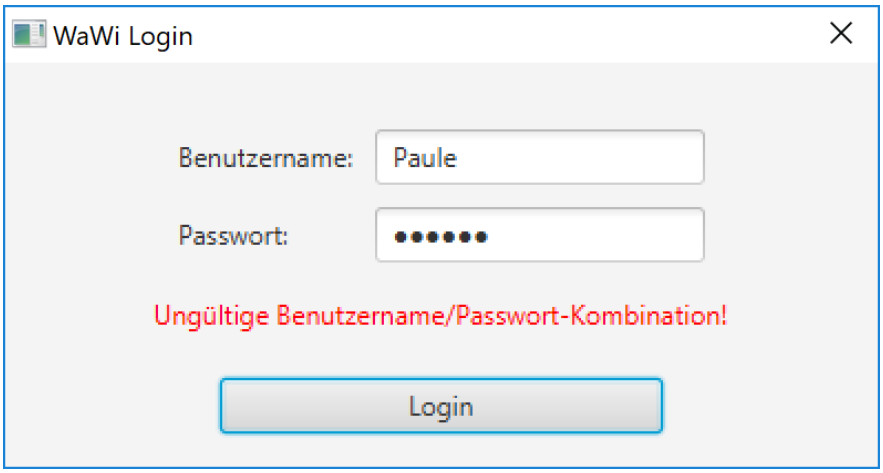

Abbildung 1: Login-Dialog mit Fehlermeldung

Funktionsweise des Dialogs

Der Login-Dialog ermöglicht die Eingabe von Benutzername und Passwort. Beim Klick auf den Login-Button werden die Eingaben geprüft:

- Wurde „Hugo“ und „123“ eingegeben, so wird der Login-Dialog ausgeblendet und der POS-Dialog wird aufgerufen.
- Wurde eine ungültige Benutzername/Passwort-Kombination aufgerufen, so wird die in Abbildung 1 zu sehende Fehlermeldung angezeigt.

- Wurde der Benutzername nicht ausgefüllt (oder nur mit Whitespace-Zeichen), so wird die Fehlermeldung „Bitte geben Sie einen Benutzernamen ein!“ angezeigt. Analog für die Passwort-Eingabe.

Durch Klick auf den Kreuz-Button rechts oben wird die Anwendung beendet. Nach dem Schließen des POS-Dialogs (Kreuz-Button des Dialogs) wird die Anwendung nicht beendet, stattdessen soll der Login-Dialog wieder erscheinen.

Empfohlene Vorgehensweise

1. Definieren Sie eine Klasse `gui.loginDialog.LoginDialog`. Der Konstruktor initialisiert den Dialog und setzt den eigentlichen Login-Handler für den Login-Button. Weitere Attribute und Methoden ergeben sich aus der Notwendigkeit, von außen (durch den Handler) auf das Label für die Fehlermeldung zuzugreifen bzw. den Dialog anzuzeigen bzw. zu verbergen.
2. Definieren Sie analog zu den Event-Handlern aus Aufgabenblatt 3 ein funktionales Interface `gui.loginDialog.LoginHandler` mit einer Methode
   ```
   void checkLogin(String benutzername, String passwort);
   ```
3. Definieren Sie in der Klasse `WaWiApp` eine lokale Methode showLoginDialog (analog zu `showPOSDialog`), die den Login-Dialog aufruft und zuvor den Handler definiert und dem Dialog bekannt macht (per Setter).
4. Verändern Sie die Start-Methode der Klasse `WaWiApp` so, dass `showLoginDialog` aufgerufen wird.
5. Prüfen Sie, ob alle Fehlerfälle korrekt angezeigt werden und ob nach dem Schließen des POS-Dialogs der Login-Dialog erscheint.

Hinweise:

1. Eine rote Fehlermeldung kann folgendermaßen angezeigt werden:
   ```
   <ihre Label-Instanz>.setTextFill(Color.web("red"));
   ```
2. Eine angezeigte Fehlermeldung nach einer fehlgeschlagenen Prüfung soll mit dem ersten Tastendruck wieder gelöscht werden. Einen Handler für den Tastendruck (genauer: das Loslassen einer Taste) können Sie folgendermaßen setzen (hier für den Benutzernamen, analog für das Passwort-Feld):
   ```
   benutzernameTextFeld.setOnKeyReleased(event -> { … });
   ```
3. Nutzen Sie für den Login-Dialog einen modalen[1] Dialog mit einer festen Breite:
   ```
   stage.initModality(Modality.APPLICATION_MODAL);
   stage.setWidth(400.);
   ```
   Als Nebeneffekt bewirkt der Kreuz-Button automatisch das Schließen des Dialogs und damit das Terminieren der Anwendung.

Testat

Dieses Aufgabenblatt sollte zusammen mit dem 5. Aufgabenblatt vorgeführt werden. Im Rahmen des Testats sollten sie die Funktionsweise des Dialogs vorführen und die Implementierung auf Quellcode-Ebene erklären können.

[1] Ein modaler Dialog lässt nur Eingaben im aktuellen Fenster zu, alle anderen Fenster bzw. der Desktop sind gesperrt, bis das Fenster geschlossen wird.