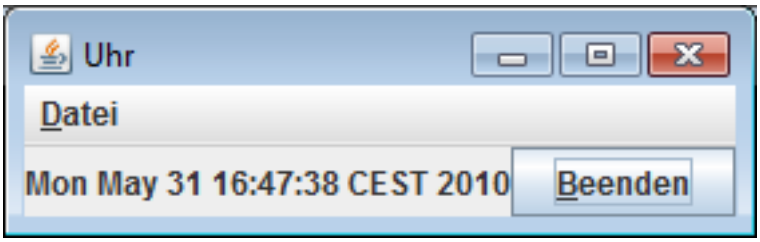
Praktikum Programmieren II (Java)

Sommersemester 2010

1. Juni 2010

Übung 5.1 Oberfläche mit Menü

Entwerfen Sie die folgende Oberfläche, die einmal das aktuelle Datum und Uhrzeit anzeigt:



Die im Screenshot dargestellt Stringdarstellung erhält man mit `new Date().toString()`, wobei `Date` die Datumsklasse aus `java.util` ist. Hinweis: Fügen Sie Ihrer Anwendung eine Methode `private void setzeUhr()` hinzu, die für das `JLabel` den Text auf diese Weise setzt, und die beim Aufbau der GUI aufgerufen wird.

Das Menü „Datei“ besitzt dabei nur den Menüpunkt „Beenden“ mit dem Mnemonic-Zeichen „B“ und dem Tastaturkürzel „Ctrl-Q“.

Beim Klick auf die Schaltfläche „Beenden“ oder Auswahl des Menüpunktes „Datei“ → “Beenden“ soll die Anwendung beendet werden. Implementieren Sie hierzu eine Methode `private void beenden()`. Hinweis: Zum Schließen und Beenden des Threads, in dem ein `JFrame` läuft, machen Sie das Fenster zunächst unsichtbar (`setVisible(false)`) und rufen dann die Methode `dispose()` auf.

Zusatzaufgabe 1

Beim Aufruf der Methode `beenden()` soll zunächst ein Bestätigungsdialog erscheinen, ob die Anwendung wirklich geschlossen werden soll.

Hierfür gibt es in der Klasse `javax.swing.JOptionPane` die Methode

```
public static int showConfirmDialog(JComponent parent, String message, String title, int option_mask)
```

wobei `parent` das Elternfenster ist, das solange blockiert wird, bis der Dialog beendet wird, `message` die angezeigte Frage, `title` der im Dialograhmen angezeigte Titel und `option_mask` die angezeigten Auswahloptionen (benutzen Sie hier `JOptionPane.YES_NO_OPTION`).

Der Rückgabewert dieser statischen Methode ist eine der Integer-Konstanten `JOptionPane.YES_OPTION`, `JOptionPane.NO_OPTION` oder `JOptionPane.CLOSED_OPTION`, die darüber informiert, ob der Dialog mit „Ja“, „Nein“ oder durch Schließen ohne Auswahl beendet wurde.

Zusatzaufgabe 2

Auch beim Schließen der Anwendung über das Fensterelement „X“ oben rechts soll zunächst der Dialog erscheinen. Rufen Sie hierzu beim Erzeugen der Oberfläche

`setDefaultCloseOperation(JFrame.DO_NOTHING_ON_CLOSE)`

auf, melden Sie bei Ihrer Anwendung als `WindowListener` einen `WindowAdapter` an, dessen Methode `windowClosing` so überladen ist, dass beim Schließen des Fensters analog zum Menü und zur Schaltfläche vor dem Beenden der Bestätigungsdialog erscheint.

Übung 5.2 Aktualisierte Uhr

Sorgen Sie dafür, dass das Label mit Datum/Uhrzeit alle 0,5 Sekunden aktualisiert wird (hierfür sind die Funktionalität der Menüs und die Zusatzaufgaben optional).

Hierfür gibt es drei prinzipielle Möglichkeiten:

1. Eigener Thread, der periodisch nach dem Setzen von Datum/Uhrzeit 500 ms pausiert, bis er unterbrochen wird.
2. Ein `Timer` sowie Implementierung eines `TimerTask` aus `java.util`. Dies sieht folgendermaßen aus:

   ```
   timer = new java.util.Timer(true);
   timer.schedule(new TimerTask() {
       @Override
       public void run() {
           // TODO: Code, der jeweils aufgeführt wird
       }
   }, 500, 500);
   ```

   wobei das `schedule` hier den TimerTask erstmalig nach 500 ms ausführt und alle 500 ms wiederholt. Mit `cancel()` kann der Timer beendet werden. Das `true` kennzeichnet den Timer als Daemon.
3. Mit einem `Timer` aus `javax.swing`. Dieser wird folgendermaßen benutzt:

   ```
   timer = new javax.swing.Timer(500,new ActionListener() {
       @Override
       public void actionPerformed(ActionEvent e) {
           // TODO: Code, der jeweils aufgeführt wird
       }
   });
   ```

   d. h. dem Konstruktor wird eine Zeitdifferenz mitgegeben, nach der Timer erstmals ausgeführt wird und dann periodisch wiederholt wird, indem für den übergebenen `ActionListener` jeweils die Methode `actionPerformed` aufgerufen wird. Dieser Timer kann mit `start` bzw. `stop` gestartet und beendet werden.

Entscheiden Sie sich für einen der Wege zum Realisieren des Timers. Beachten Sie dabei, dass evtl. beim Schließen der Anwendung über Menü/Schaltfläche der Timer explizit gestoppt werden muss, damit die Anwendung tatsächlich beendet wird.

Tipp: Wenn Sie in einer inneren/anonymen Klasse eine Referenz auf das `this` der äußeren Klasse `MyApp` brauchen, geht dies mit `MyApp.this`.