G. Einsendeaufgabe

Objektorientierte Programmierung mit C++ und Qt

Code:
CPBS 19E-XX1-N01

Name: Rein	Vorname: Wjatscheslaw
Postleitzahl und Ort: 14193 Berlin	Straße: Cunostr. 54A
Studien- bzw. Vertrags-Nr.: 800463563	Lehrgangs-Nr.: 246

Fernlehrer/in:

Datum:

Note:

Unterschrift Fernlehrer/in:

Bitte reichen Sie Ihre Lösungen über die Online-Lernplattform ein oder schicken Sie uns diese per Post. Geben Sie bitte immer den Code zum Studienheft an (siehe oben rechts).

Inhaltsverzeichnis

Inhaltsverzeichnis **2**

1 Aufgabe: **3**
1.1 Aufgabe: 3
1.2 Lösung: 3

2 Aufgabe: **4**
2.1 Aufgabe: 4
2.2 Lösung: 4

3 Aufgabe: **5**
3.1 Aufgabe: 5
3.2 Lösung: 5

Abbildungsverzeichnis **11**

Listings **12**

1 Aufgabe:

Aufgabe:

Sorgen Sie dafür, dass in unserem Taschenrechner aus diesem Studienheft nicht mehr der Punkt für ein Komma eingegeben werden muss. Ersetzen Sie auch in der Ausgabe den Punkt durch ein Komma.

Ein kleiner Tipp für die Lösung:

JavaScript kennt eine Funktion, mit der Sie in einer Zeichenkette ein Zeichen durch ein anderes ersetzen können. Der Name der Funktion lässt sich aus der englischen Übersetzung für „ersetze“ ableiten.

20 Pkt.

Lösung:

Um unsichtbar für den Benutzer der Punkt durch das Komma zu ersetzen und die Berechnung aufrechtzuerhalten, werden die beiden Symbole mehrmals in dem Ablauf ausgetauscht. Bei der Eingabe wird gleich das Komma in die Zeichenkette eingefügt. Danach, unmittelbar vor der Berechnung, wird das Komma stillschweigend durch den Punkt getauscht. Und vor der Ergebnis-Ausgabe werden wieder die Punkte in den entsprechenden Zeichenketten durch Komma ersetzt (siehe Listing 1.1).

```
1       function operatorEingabe(operator) {
2         ...
3
4         //Aufgabe 1
5         //Das Komma vor der Rechnung durch einen Punkt austauschen
6         textField.text = textField.text.replace(",", ".");
7         textField1.text = textField1.text.replace(",", ".");
8
9         ...
10
11        //das Ergebnis im Label ausgeben
12        //Aufgabe 1
13        //Die Punkte nach der Rechnung durch Kommas wieder ersetzen
14        textField.text = textField.text.replace(".", ",");
15        textField1.text = textField1.text.replace(".", ",");
16        ausgabe.text = ergebnis;
17        ausgabe.text = ausgabe.text.replace(".", ",");
18      }
19
20
```

Listing 1.1: Lösung der Aufgabe 1

2 Aufgabe:

Aufgabe:

Sorgen Sie dafür, dass im Taschenrechner aus diesem Studienheft nur maximal ein Komma für jede Zahl eingegeben werden kann.

20 Pkt.

Lösung:

Es wird dafür gesorgt, dass nur maximal ein Komma für jede Zahl eingegeben werden kann. Die Funktion „indexOf“ gibt die Position des Zeichens in der Zeichenkette zurück oder -1 wenn kein Zeichen gefunden wurde. Somit kann die mehrfache Eingabe vom Komma verhindert werden (siehe Listing 2.1).

```
1    button11.onClicked: {
2      //Aufgabe 2
3      //Es wird dafuer gesorgt, dass nur maximal ein Komma fuer
4      //jede Zahl eingegeben werden kann. Die Funktion "indexOf"
5      //gibt die Position des  Zeichens in der Zeichenkette
6      //zurueck oder -1 wenn kein Zeichen gefunden wurde.
7      if (aktiveEingabe == 1)
8      {
9        if (textField.text.indexOf(",") == -1)
10       {
11         textField.text = textField.text + ",";
12       }
13       else
14       {
15         //Das Komma ist bereits vorhanden
16       }
17     }
18     else
19     {
20       if (textField1.text.indexOf(",") == -1)
21       {
22         textField1.text = textField1.text + ",";
23       }
24       else
25       {
26         //Das Komma ist bereits vorhanden
27       }
28     }
29   }
30
31
```

Listing 2.1: Lösung der Aufgabe 2

3 Aufgabe:

Aufgabe:

Erstellen Sie eine Uhr als mobile App. Beim Anklicken oder Antippen der Uhrzeit soll für zwei Sekunden das aktuelle Datum erscheinen.

Einige Hinweise zur Lösung:

Über das Ereignis `Component.completed` können Sie auf das vollständige Laden einer Komponente reagieren. So können Sie zum Beispiel nach dem Laden der Seite direkt die Uhrzeit anzeigen lassen.

Ein Label kann nicht angeklickt oder angetippt werden. Sie können aber zum Beispiel ein Steuerelement vom Typ **MouseArea** verwenden und das Label in diesem Steuerelement ablegen.
Wie Sie die Aufgabe lösen, bleibt Ihnen überlassen. Dokumentieren Sie Ihre Lösung aber bitte.

60 Pkt.

Lösung:

Vorausgesetzt die Makros für die Verbindungen, sowie Methoden und Attribute in der Headerdatei (Listing 3.1) deklariert und in der entsprechenden Quelldatei (Listing 3.2) definiert bzw. implementiert sind, können die Anweisungen, Funktionen und Signale in QML-Projektdateien realisiert werden.

```
1   class qtUhr : public QObject
2   {
3     Q_OBJECT
4     //das Makro fuer die Verbindung
5     Q_PROPERTY(QString zeitAct READ liefereZeit)
6     Q_PROPERTY(QString datumAct READ liefereDatum)
7
8     public:
9       explicit qtUhr(QObject *parent = nullptr);
10      QString liefereZeit();
11      QString liefereDatum();
12
13    private:
14      QString zeitAct;
15      QString datumAct;
16  };
17
18
```

Listing 3.1: Deklaration von einem Klass qtUhr in der Headerdatei

Die Methoden sorgen dafür, dass die aktuelle Zeit bzw. Datum anhand bekannten Funktionen ermittelt und an die aufrufende Funktion zurück geliefert werden (siehe Listing 3.2).

```
1   qtUhr::qtUhr(QObject *parent) : QObject(parent)
2   {
3     zeitAct = "00:00:00:00";
4   }
5
6   /*****************************************************************
7   * QString qtUhr::liefereZeit()
8   *
9   * Aufgabe 3
10  * Funktion: Die aktuelle Zeit wird ermittelt und an die aufrufende
11  * Funktion zurueck geliefert.
12  *****************************************************************/
13  QString qtUhr::liefereZeit() {
14    //die Zeit abfragen und aufbereiten
15    zeitAct = QTime::currentTime().toString().left(8);
16    return zeitAct;
17  }
18
19  /*****************************************************************
20  * QString qtUhr::liefereDatum()
21  *
22  * Aufgabe: Das aktuelle Datum wird ermittelt und an die aufrufende
23  * Funktion zurueck geliefert.
24  *****************************************************************/
25  QString qtUhr::liefereDatum() {
26    datumAct = QDate::currentDate().toString("dd.MM.yyyy");
27    return datumAct;
28  }
29
```

Listing 3.2: Implementierung von den Methoden für aktuellen Datum und Zeit

Nach dem vollständigen Laden einer Komponente wird automatisch das Anzeigen der aktuellen Uhrzeit gestartet (Listing 3.3) .

```
1   ...
2   //Aufgabe 3: Defininition der Eigenschaft "datumV"
3   property bool datumV: false
4
5   //Aufgabe 3: Nach dem vollstaendigen Laden einer Komponente wird das
6   //Anzeigen der aktuellen Uhrzeit gestartet.
7   Component.onCompleted:
8   {
9     meinTimer.start();
10    aktualisiereZeit()
11  }
12  ...
13
```

Listing 3.3: Quellcode für Komponente vollständig geladen

Beim ersten Anklicken wird ein Label mit der Uhrzeit ausgeblendet, ein Timer für 2 Sekunden gestartet und in der Zeit ein zweites Label mit aktuellem Datum versorgt und angezeigt. Zusätzlich, kann beim zweiten Anklicken wieder die Uhrzeit erneut eingeblendet werden (Listing 3.4).

```
1    ...
2    /*****************************************************************
3    * mouseArea.onClicked: {}
4    *
5    * Aufgabe 3
6    * Funktion: Beim ersten Anklicken wird ein Label mit der Uhrzeit
7    * ausgeblendet, ein Timer fuer 2 Sekunden gestartet und in der Zeit
8    * ein zweites Label mit aktuellem Datum versorgt und angezeigt.
9    * Zusaetzlich, kann beim zweiten Anklicken wieder die Uhrzeit erneut
10   * eingeblendet werden.
11   *****************************************************************/
12   mouseArea.onClicked: {
13     if (datumV == false)
14     {
15       datumV = true;
16       label1.visible = true;
17       label.visible = false;
18       label1.text = meineQUhr.datumAct;
19       meinTimerDatum.start();
20     }
21     else
22     {
23       datumV = false;
24       label1.visible = false;
25       label.visible = true;
26       meinTimerDatum.stop();
27     }
28   }
29
30   ...
31
```

Listing 3.4: Quellcode für einen Signal beim Anklicken der MausArea

Die beiden Timer für Datum- und Zeitanzeige werden vorkonfiguriert und ein Zeit-Intervall wird geladen. Die Anzeige für die Zeit wird wiederholend mit einer Schrittgröße von 100 ms upgedatet und die Datumanzeige wird für zwei Sekunden lang eingeblendet (Listing 3.5).

```
1     ...
2     /**********************************************************************
3     * Timer {}
4     *
5     * Aufgabe 3
6     * Funktion: Konfiguration von dem Timer, der in einem Zeitintervall von
7     * 100[ms] wiederholend die Uhrzeit anzeigt.
8     **********************************************************************/
9     Timer {
10      id: meinTimer
11      //er soll alle 100 Millisekunden ausgeloest werden
12      interval: 100
13      //wiederholt werden
14      repeat: true
15      //und die Funktion aktualisiereZeit() aufrufen
16      onTriggered: aktualisiereZeit()
17    }
18
19    /**********************************************************************
20    * Timer {}
21    *
22    * Aufgabe 3
23    * Funktion: Konfiguration von dem Timer, der in einem Zeitintervall von
24    * 2[s] die Uhrzeit ausblendet und die Anzeige vom aktuellen
25    * Datum ermoeglicht.
26    **********************************************************************/
27    Timer {
28      id: meinTimerDatum
29      //er soll 2000 Millisekunden lang das Datum anzeigen
30      interval: 2000
31      //darf nicht wiederholt werden
32      repeat: false
33      //und am Ende die Funktion zuUhr() aufrufen
34      onTriggered: zuUhr()
35    }
36    ...
37
```

Listing 3.5: Timer-Funktionen für das Datum und die Zeit

Die Funktion für den ersten Timer zeigt die aktuelle Zeit und die Funktion für den zweiten Timer führt ein Wiederkehren zu der Uhrzeit durch mit Setzen bzw. Rücksetzen von entsprechenden Eigenschaften (Listing 3.6).

```
    ...
    //die Funktion fuer den Timer
    /*****************************************************************
    * function aktualisiereZeit()
    *
    * Aufgabe 3
    * Funktion: Die aktuelle Zeit wird vom Timer gesteuert angezeigt.
    *****************************************************************/
    function aktualisiereZeit() {
      label.text = meineQUhr.zeitAct;
    }
    //die Funktion fuer den Timer
    /*****************************************************************
    * function zuUhr()
    *
    * Aufgabe 3
    * Funktion: Fuehrt ein Wiederkehren zu der Uhrzeit durch mit Setzen bzw.
    * Ruecksetzen von entsprechenden Eigenschaften.
    *****************************************************************/
    function zuUhr() {
      datumV = false;
      label1.visible = false;
      label.visible = true;
    }
    ...

```

Listing 3.6: Die Funktionen für das Datum und die Zeit

Das Listing (Listing 3.7) zeigt Definition von den Eigenschaften und die Konfiguration eines RowLayouts.

```
    //Aufgabe 3
    //Funktion: Definition von den Eigenschaften
    property alias label: label
    property alias mouseArea: mouseArea
    property alias label1: label1
    property alias rowLayout: rowLayout

    RowLayout {
      id: rowLayout
      anchors.fill: parent

      Label {
        id: label
        text: qsTr("00:00:00:00")
        font.pointSize: 35
        verticalAlignment: Text.AlignVCenter
        horizontalAlignment: Text.AlignHCenter
        Layout.fillHeight: true
        Layout.fillWidth: true
      }
    }

```

Listing 3.7: Definition von den Eigenschaften und RowLayout

In dem folgendem Quellcode-Abschnitt wird ein Steuerelement vom Typ MouseArea definiert und das Label in diesen Element verschachtelt abgelegt (Listing 3.8).

```
1     ...
2     /*****************************************************************
3     * MouseArea {}
4     *
5     * Aufgabe 3
6     * Funktion: Ein Steuerelement vom Typ MouseArea wird definiert und das
7     * Label in diesen Steuerelement verschachtelt abgelegt.
8     *****************************************************************/
9     MouseArea {
10      id: mouseArea
11      anchors.fill: parent
12
13      Label {
14        id: label1
15        visible: false
16        text: qsTr("Datum")
17        anchors.fill: parent
18        horizontalAlignment: Text.AlignHCenter
19        verticalAlignment: Text.AlignVCenter
20        anchors.rightMargin: 0
21        font.pointSize: 35
22      }
23    }
24    ...
25
```

Listing 3.8: Definition von einer MausArea

Die Abbildung (3.1) zeigt die Ergebnisse der dritten Aufgabe.

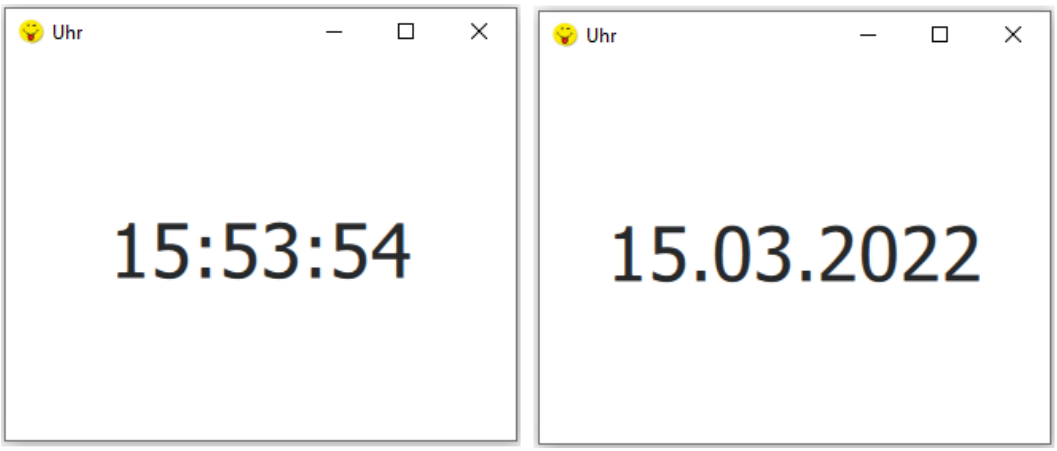

Abbildung 3.1: Ergebnis der Aufgabe 3

Abbildungsverzeichnis

3.1 Ergebnis der Aufgabe 3 10

Listings

1.1 Lösung der Aufgabe 1 3

2.1 Lösung der Aufgabe 2 4

3.1 Deklaration von einem Klass qtUhr in der Headerdatei 5
3.2 Implementierung von den Methoden für aktuellen Datum und Zeit . 6
3.3 Quellcode für Komponente vollständig geladen 6
3.4 Quellcode für einen Signal beim Anklicken der MausArea 7
3.5 Timer-Funktionen für das Datum und die Zeit 8
3.6 Die Funktionen für das Datum und die Zeit 9
3.7 Definition von den Eigenschaften und RowLayout 9
3.8 Definition von einer MausArea 10