Bereich: Grafische Benutzeroberflächen (GUI/Swing) (2)

Datei auswählen und zeilenweise darstellen	**Musterlösung**
Package: `de.dhbwka.java.exercise.ui`	**Klasse:** `TextfileViewer`

Variante 1 (Labels)

```
package de.dhbwka.java.exercise.ui;

import java.awt.GridLayout;
import java.io.BufferedReader;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;

import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.filechooser.FileFilter;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.1
 */
public class TextfileViewer {

   private static final int MAX_LINES = 10;

   public TextfileViewer( File file ) {
      JFrame frame = new JFrame( file.getName() );
      String[] content = TextfileViewer.readTextfile( file );

      final int displayLines =
            Math.min( content.length, TextfileViewer.MAX_LINES );

      frame.setLayout( new GridLayout( displayLines, 1 ) );
      for ( int i = 0; i < displayLines; i++ ) {
         frame.add( new JLabel( content[i] ) );
      }

      frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      frame.setSize( 640, 480 );
      frame.setVisible( true );
   }

   // Continued on next page
```

```java
   /**
    * Read text file and return its content as an string array
    *
    * @param f
    *           text file
    * @return content as string array
    */
   public static String[] readTextfile( File f ) {
      // Count lines first (remember: arrays => fix length!)
      int lines = 0;
      try ( BufferedReader br = new BufferedReader( new FileReader( f ) ) ) {
         while ( br.ready() ) {
            br.readLine();
            lines++;
         }
      } catch ( IOException e ) {
         e.printStackTrace(); // TODO: nice error handling
      }

      // Create array with count of lines as length
      String[] result = new String[lines];
      lines = 0;
      try ( BufferedReader br = new BufferedReader( new FileReader( f ) ) ) {
         while ( br.ready() ) {
            result[lines++] = br.readLine();
         }
      } catch ( IOException e ) {
         e.printStackTrace(); // TODO: nice error handling
      }

      return result;
   }

   public static void main( String[] args ) {
      // FileChooser as presented in slides "Swing"
      JFileChooser fc = new JFileChooser();
      fc.setFileFilter( new FileFilter() {
         @Override
         public boolean accept( File f ) {
            return f.isDirectory()
                  || f.getName().toLowerCase().endsWith( ".txt" );
         }

         @Override
         public String getDescription() {
            return "Text files (*.txt)";
         }
      } );

      int state = fc.showOpenDialog( null );
      if ( state == JFileChooser.APPROVE_OPTION ) {
         new TextfileViewer( fc.getSelectedFile() );
      } else {
         JOptionPane.showMessageDialog( null, "Choice cancelled" );
      }
   }
}
```

Variante 2 (TextArea)

```
package de.dhbwka.java.exercise.ui;

import java.awt.BorderLayout;
import java.io.File;

import javax.swing.JFileChooser;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.filechooser.FileFilter;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.1
 */
public class TextfileViewer2 {

   public TextfileViewer2( File file ) {
      JFrame frame = new JFrame( file.getName() );

      StringBuilder joinedContent = new StringBuilder();
      for ( String s : TextfileViewer.readTextfile( file ) ) {
         joinedContent.append( s ).append( System.lineSeparator() );
      }

      JScrollPane scrollpane =
            new JScrollPane( new JTextArea( joinedContent.toString() ) );

      // remember: default layout for frame content border layout
      frame.add( scrollpane, BorderLayout.CENTER );
      frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      frame.setSize( 640, 480 );
      frame.setVisible( true );
   }


   // Continued on next page
```

```
   public static void main( String[] args ) {
      // FileChooser as presented in slides "Swing"
      JFileChooser fc = new JFileChooser();
      fc.setFileFilter( new FileFilter() {
         @Override
         public boolean accept( File f ) {
            return f.isDirectory()
                  || f.getName().toLowerCase().endsWith( ".txt" );
         }

         @Override
         public String getDescription() {
            return "Text files (*.txt)";
         }
      } );

      int state = fc.showOpenDialog( null );
      if ( state == JFileChooser.APPROVE_OPTION ) {
         new TextfileViewer2( fc.getSelectedFile() );
      } else {
         JOptionPane.showMessageDialog( null, "Choice cancelled" );
      }
   }

}
```

Bereich: Grafische Benutzeroberflächen (GUI/Swing) (2)

Währungsumrechner	**Musterlösung**
Package: `de.dhbwka.java.exercise.ui`	**Klasse:** `CurrencyCalculator`

```java
package de.dhbwka.java.exercise.ui;

import java.awt.BorderLayout;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JTextField;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.1
 */
public class CurrencyCalculator {

   public final static float CHANGE_RATE = 1.14f;

   private JFrame frame = new JFrame( "Currency converter" );

   private JTextField input =
         new JTextField( "Please enter amount to convert!" );

   private JButton btnEur2usd = new JButton( "EUR -> USD" );
   private JButton btnUsd2eur = new JButton( "USD -> EUR" );
   private JButton btnCancel = new JButton( "Cancel" );

   public CurrencyCalculator() {
      // default BorderLayout has no margin!
      this.frame.setLayout( new BorderLayout( 10, 10 ) );
      this.frame.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      this.frame.add( this.input, BorderLayout.NORTH );
      this.frame.add( this.btnEur2usd, BorderLayout.WEST );
      this.frame.add( this.btnUsd2eur, BorderLayout.CENTER );
      this.frame.add( this.btnCancel, BorderLayout.EAST );
      this.frame.setSize( 350, 100 );
      this.frame.setVisible( true );
   }

   public static void main( String[] args ) {
      new CurrencyCalculator();
   }

}
```

Bereich: Grafische Benutzeroberflächen (GUI/Swing) (2)

Editor	**Musterlösung**
Package: `de.dhbwka.java.exercise.ui.editor`	**Klasse:** `EditorSimple`

```
package de.dhbwka.java.exercise.ui.editor;

import java.awt.BorderLayout;
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.KeyEvent;

import javax.swing.JEditorPane;
import javax.swing.JFrame;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
import javax.swing.JScrollPane;
import javax.swing.JSeparator;

/**
 * Part of lectures on 'Programming in Java'. Baden-Wuerttemberg
 * Cooperative State University.
 *
 * (C) 2016-2018 by W. Geiger, T. Schlachter, C. Schmitt, W. Suess
 *
 * @author DHBW lecturer
 * @version 1.2
 */
@SuppressWarnings("serial")
public class EditorSimple extends JFrame {

   private final static String SEPARATOR = "--";
   private final static String SEND_MENU = "-SENDMENU-";
   private final static int INITIAL_WIDTH = 600;
   private final static int INITIAL_HEIGHT = 480;

   private JMenu fileMenu;
   private JMenu editMenu;
   private JEditorPane editPane;

   private final String[] fileMenuItems =
         { "Neu", "\u00D6ffnen", EditorSimple.SEPARATOR,
               "Schlie\u00DFen", EditorSimple.SEPARATOR, "Speichern",
               "Speichern unter...",
               "Als Webseite speichern", "Suchen", EditorSimple.SEPARATOR,
               "Versionen",
               EditorSimple.SEPARATOR, "Webseitenvorschau",
               EditorSimple.SEPARATOR, "Seite einrichten...",
               "Seitenansicht", "Drucken", EditorSimple.SEPARATOR,
               EditorSimple.SEND_MENU,
               "Eigenschaften", EditorSimple.SEPARATOR, "bilanz_2017.doc",
               "bericht_2018_01.doc", "ziele.doc",
               EditorSimple.SEPARATOR, "Beenden" };
```

```
   private final String[] editMenuItems =
         { "R\u00FCckg\u00E4ngig", "Wiederholen",
               EditorSimple.SEPARATOR, "Ausschneiden", "Kopieren",
               "Office-Zwischenablage",
               "Einf\u00FCgen", "Inhalte einf\u00FCgen",
               "Als Hyperlink einf\u00FCgen",
               EditorSimple.SEPARATOR, "L\u00F6schen", "Alles markieren",
               EditorSimple.SEPARATOR,
               "Suchen...", "Ersetzen...", "Gehe zu...", EditorSimple.SEPARATOR,
               "Verkn\u00FCpfungen...", "Objekt" };

   private final String[] sendMenuItems = { "E-Mail-Empf\u00E4nger",
         "E-Mail-Empf\u00E4nger (zur \u00DCberarbeitung)",
         "E-Mail-Empf\u00E4nger (als Anlage)", EditorSimple.SEPARATOR,
         "Verteileremp\u00E4nger...", "Onlinebesprechungsteilnehmer",
         "Exchange-Ordner...", "Fax-Empf\u00E4nger...",
         EditorSimple.SEPARATOR, "Microsoft PowerPoint" };

   /**
    * Constructor for the editor
    */
   public EditorSimple() {
      super( "Editor" );
      this.setLayout( new BorderLayout() );

      // Create scrollable editor
      this.editPane = new JEditorPane();
      this.editPane.setFont( new Font( Font.MONOSPACED, Font.PLAIN, 12 ) );
      this.add( new JScrollPane( this.editPane ), BorderLayout.CENTER );

      // Create menu
      JMenuBar menu = new JMenuBar();
      this.fileMenu =
            this.getMenu( "Datei", KeyEvent.VK_D, this.fileMenuItems );
      this.editMenu =
            this.getMenu( "Bearbeiten", KeyEvent.VK_B, this.editMenuItems );
      menu.add( this.fileMenu );
      menu.add( this.editMenu );

      this.addListener();

      this.setJMenuBar( menu );
      this.setSize( EditorSimple.INITIAL_WIDTH, EditorSimple.INITIAL_HEIGHT );
      this.setDefaultCloseOperation( JFrame.EXIT_ON_CLOSE );
      this.setVisible( true );
   }

   // Continued on next page
```

```
    /**
     * Add the listener to exit button
     */
    private void addListener() {
        // 25 = index of 'Beenden' in this.fileMenuItems
        JMenuItem itemExit = (JMenuItem) this.fileMenu.getMenuComponent( 25 );
        itemExit.addActionListener( new ActionListener() {
            @Override
            public void actionPerformed( ActionEvent e ) {
                System.exit( 0 );
            }
        } );
    }

    /**
     * Produces a JMenu with JMenuItems according to items array
     *
     * @param menuName
     *            name of parent menu
     * @param mnemonic
     *            for the parent menu
     * @param items
     *            string array with sub items labels
     * @return create parent menu
     */
    private JMenu getMenu( String menuName, int mnemonic, String[] items ) {
        JMenu menu = new JMenu( menuName );
        menu.setMnemonic( mnemonic );
        for ( String menuItemName : items ) {
            switch (menuItemName) {
                case SEPARATOR:
                    menu.add( new JSeparator() );
                    break;
                case SEND_MENU:
                    menu.add( this.getMenu( "Senden an", KeyEvent.VK_S,
                            this.sendMenuItems ) );
                    break;
                default:
                    menu.add( new JMenuItem( menuItemName ) );
            }
        }
        return menu;
    }

    /**
     * Run an instance of the editor
     */
    public static void main( String[] args ) {
        new EditorSimple();
    }

}
```