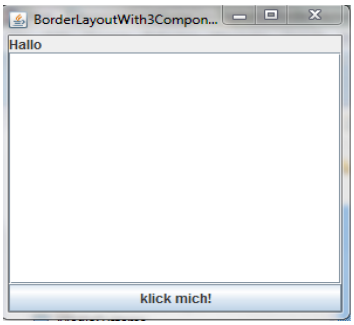
1. Komponenten und Layouts

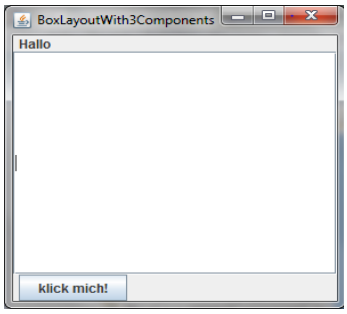
Aufgabe:

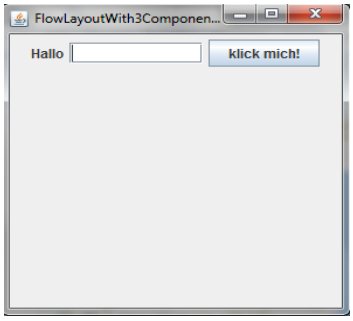
Sie sollen ein Label mit dem Inhalt "Hallo", ein Textfeld mit 10 Zeichen, einen Button mit der Aufschrift "Klick mich" mit verschiedenen Layouts darstellen.

Verwenden Sie das Borderlayout wobei Sie die Komponenten der Reihe nach von oben bis unten angeordnet werden sollen. Tun Sie das gleiche mit dem BoxLayout in Y-Richtung, dem Flowlayout, dem GridLayout mit 1 Spalte und 3 Zeilen und skizzieren Sie das Frame und die Anordnung der Komponenten so, dass erkennbar ist welches Layout verwendet wird.

Um den Unterschied besser sehen zu können, verwenden Sie eine quadratische Framegröße z.B. 300 x 300.







Fragen:

Wie kann man mehrere Layouts in einem Fenster mit einander kombinieren? Welche Komponente oder welchen Container braucht man dazu?

Man braucht dafür JPanels, jedes JPanel bekommt dabei ein eigenes Layout und kann in diesem Layout wieder ein JPanel (Container) enthalten.
(Der Vollständigkeitshalber sei erwähnt, dass natürlich auch ein JFrame ein Layout hat und dem JFrame können JPanels hinzugefügt werden.)

Sie möchten zwischen bestimmten Komponenten Platzhalter hinzufügen, welches Layout brauchen Sie und wie sieht die Anweisung aus, in der Sie einen Platzhalter hinzufügen? (Hilfe: createHorizonalStrut(int width)

Man braucht das BoxLayout. Die Anweisung lautet:
add (Box. CreateHorizonalStrut (5));

Was ist eine JList?

Eine graphische Liste.

Wie kann man zu einer JList Elemente hinzufügen?

```
DefaultListModel<String> model = new DefaultListModel<>();
JList<String> list = new Jlist<>(model);

//hier kommen irgendwelche Daten, z.B. das huhu

String element1 = "Huhu";
model.addElement(element1);
```

2. Listener

Aufgabe:

Prüfen Sie ggf. am Rechner wie folgende Komponenten auf verschiedene Listener reagieren, verwenden Sie dabei ButtonGroups. Wie unterscheiden sich die Listener von einander?

JCheckBox	ActionListener, ItemListener
JRadioButton	ActionListener, ItemListener

Beide Listener reagieren, wenn man eine CheckBox oder ein RadioButton an-oder abwählt. Der Unterschied liegt bei der Verwendung von ButtonGroups. Sind beide Komponenten in einer ButtonGroup, kann nur ein Element zur Zeit anwählbar sein.

Ist die JCheckBox angewählt und der User setzt die Markierung auf die JRadioBox, so würde ein ActionListener nur für die JRadioBox ein Event werfen. Im Gegensatz dazu würde ein ItemListener bei der JCheckBox und der JRadioBox „alarmieren“.

Fragen:

Nennen Sie ein Beispiel wo man ein FocusListener benötigt.

Bei einem Formular, wenn der Focus noch nicht auf einem JTextField ist, soll der User erst mal einen Hinweis bekommen, was er dort eingeben soll (z.B. Name). Sobald der Focus auf dem Textfeld ist, sollte der Hinweis „Name“ jedoch verschinden. Um mitzubekommen, ob der Focus auf einer Komponente ist, fügt man dieser den FocusListener hinzu.

3. File IO

Aufgabe:

Schreibt ein Programm, dass eine Textdatei ausliest, den Text zeilenweise filtert und nur dann wieder in eine Datei schreibt, wenn die Zeile nicht mit einem # beginnt

```
JFileChooser fc = new JFileChooser();

if (JFileChooser.APPROVE_OPTION == fc.showOpenDialog(null)) {

	File file = fc.getSelectedFile();
	File fileOut = new File("Output " + file.getName());

	try {
		FileReader in = new FileReader(file);
		BufferedReader bufread = new BufferedReader(in);

		FileWriter out = new FileWriter(fileOut, false);
		BufferedWriter bufwrite = new BufferedWriter(out);

		String line = bufread.readLine();

		while (line != null) {

			if (line.length() > 0 && line.charAt(0) != '#') {
				bufwrite.write(line);
				bufwrite.newLine();
			}
			else if (line.length() == 0) {
				bufwrite.newLine();
			}

			line = bufread.readLine();
		}

		bufwrite.close();
		bufread.close();
		in.close();
		out.close();

	} catch (FileNotFoundException e) {
		e.printStackTrace();
	} catch (IOException e) {
		e.printStackTrace();
	}

} else {
	System.out.println("da ist was schief gegangen");
}
```

Fragen:

Der FileWriter hat den Konstruktor FileWriter (File file, boolean append). Erklären Sie die beiden Parameter.

Der FileWriter braucht eine Datei in die er schreiben kann, diese Datei ist der erste Parameter. Der zweite bestimmt, ob die Datei überschrieben werden soll (false) oder ob der Inhalt an die bestehende Datei angehängt werden soll.

Wofür ist beim JFileChooser die Konstante APPROVE_OPTION?

Wenn ein Open- oder SaveFileDialog angezeigt wird, kann der User eine Datei auswählen und mit ok/yes den Vorgang abschließen, in diesem Fall ist der Rückgabe Wert APPROVE_OPTION.

Zur Info: Es gibt auch die Rückgabewerte CANCEL_OPTION (Cancel Button wurde gedrückt) und ERROR_OPTION (Ein Fehler ist aufgetreten).

Wofür gibt es beim BufferedWriter die Methode newLine()?

Wenn man einen Zeilenumbruch in einer Datei haben möchte, funktioniert ein „\n“ nicht immer. Das ist abhängig vom laufenden Betriebssystem. Manchmal muss man ein Carriage Return „\r“ mit einem „\n“ verbinden. Um diese Problematik zu umgehen, gibt es die Methode newLine(), die automatisch erkennt welche Steuerzeichen gesetzt werden müssen.

4. Interfaces und Anonyme Klasse

Aufgabe:

Was ist eine innere Klasse? Wie sieht so eine Klasse aus? Schreiben Sie dazu einen ganz kurzen Code, um dies zu verdeutlichen.

Eine Innere Klasse ist eine Klasse in einer Klasse, die von außen nicht zugreifbar ist.

```
public class Bla {

        private class InnereKlasse {
        }
}
```

Fragen:

Was ist der Unterschied zwischen eine inneren und einer anonymen inneren Klasse?

Innere anonyme Klassen werden die Deklaration und Instanziierung in einer Anweisung zusammengefasst, dies passiert innerhalb einer Methode oder eines Konstruktors. Also z.B. beim ActionListener:

```
ActionListener eventHandler = new ActionListener(){ … };
```

Oftmals schreiben wir einfach component.addActionListener (*new ActionListener () { … }*);

Innere Klassen dagegen sind Klassen, die in einer Klasse deklariert werden. Instanziiert werden sie an einer beliebigen Stelle (als Membervariable oder in einer Methode/Konstruktor).

Wozu gibt es das Interface Comparable, wann wird es verwendet? Welche Methode muss implementiert werden?

Comparable ist ein Interface, dass dann implementiert wird, wenn Objekte miteinander vergleichbar (<, >, =) sein sollen. Dazu muss die Methode compareTo (Object o) überschrieben werden. Das aktuelle Objekt wird dabei mit o verglichen.

5. Datentypen

Aufgaben:

Sie möchten die Key-Value Paare der Hashmap ausgeben, schreiben Sie den entsprechenden Code dafür. Dabei könnte die Methode keySet() hilfreich sein.

```
for (Object key : hashmap.keySet()) {
    System.out.println("key:" + key + ", value:" + hashmap.get(i));
}
```

Da kein Datentyp für key und value vorgegeben ist, hab ich einfach Object dafür genommen.

Fragen:

Wann würden Sie eine Hashmap verwenden?

Wenn ich zwei zusammenhängende Datenpaare speichern möchte und das möglichst effizient geschehen soll. Also z.B. ein Buch als Key und die Anzahl des Buchs in einer Biblithek als Value.

Wann würden Sie ein Array und wann eine ArrayListe verwenden?

Ein Array würde ich dann verwenden, wenn die Anzahl der Elemente feststehen. Ein Array ist effizienter als eine ArrayListe. Steht die Anzahl der Elemente einer Liste nicht fest, nimmt man eine ArrayListe.