Interaktive Testklasse für Konto

Forderungen:

- In einer Schleife soll die Schnittstelle der Klasse interaktiv aufgerufen werden können
- Nach jedem Schleifendurchlauf sollen die Inhalte des Konto-Objektes ausgegeben werden

KontoDialog: Grobstruktur

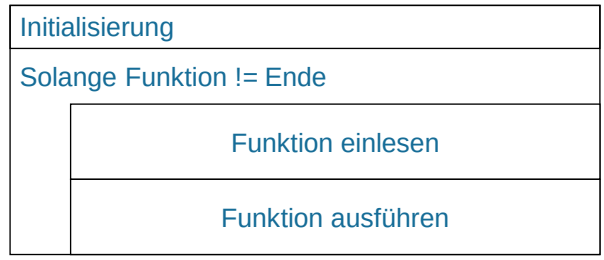
```
public class KontoDialog {

    public KontoDialog() { }
    /**
     * Hauptschleife des Testprogramms
     */
    public void start() { ... }

    /**
     * Main-Methode zum Erzeugen des KontoDialog-
     * Objektes und zum Anstarten der Testschleife
     */
    public static void main (String[] args) {
        new KontoDialog().start();
    }
}
```

KontoDialog: Hauptschleife

- Struktogramm für die Hauptsteuerung



KontoDialog: Hauptschleife

```
public class KontoDialog {
    private Konto konto1;
    private Scanner input = new Scanner(System.in);

    /**
     * Hauptschleife des Testprogramms
     */
    public void start() {
        konto1 = null;
        int funktion = -1;

        while (funktion != ENDE) {
            funktion = einlesenFunktion();
            ausfuehrenFunktion(funktion);
        }
    }
```

Input als Attribut anlegen und initialisieren

Methode einlesenFunktion()

```
// Klassenkonstanten
private static final int ANLEGEN     = 1;
private static final int EINZAHLEN   = 2;
private static final int ABHEBEN     = 3;
private static final int UEBERWEISEN = 4;
private static final int SET_INHABER = 5;
private static final int ENDE        = 0;

private int einlesenFunktion() {
    System.out.print(ANLEGEN      + ": anlegen; " +
                     EINZAHLEN    + ": einzahlen; " +
                     ABHEBEN      + ": abheben; " +
                     UEBERWEISEN  + ": überweisen; " +
                     SET_INHABER  + ": setInhaber; " +
                     ENDE         + ": beenden -> ");
    return input.nextInt();
}
```


Methode ausfuehrenFunktion()

```
// grobe Struktur:
private void ausfuehrenFunktion(int funktion) {
    ...
    if (funktion == ANLEGEN) {
        ...
    } else if (funktion == EINZAHLEN) {
        ...
    } else if (funktion == ABHEBEN) {
        ...
    } else if (funktion == UEBERWEISEN) {
        ...
    } else if (funktion == SET_INHABER) {
        ...
    } else if (funktion == ENDE) {
        System.out.println("Programmende");
    } else {
        System.out.println("Falsche Funktion!");
    }
}
```

Methode ausfuehrenFunktion (1)

```
private void ausfuehrenFunktion(int funktion) {
    int kontonr;
    String inhaber;
    double kontostand;
    double betrag;

    if (funktion == ANLEGEN) {
        System.out.print("Kontonummer: ");
        kontonr = input.nextInt();
        System.out.print("Inhaber    : ");
        inhaber = input.next();
        System.out.print("Kontostand : ");
        kontostand = input.nextDouble();
        konto1 = new Konto(kontonr, inhaber, kontostand);

    } else if (funktion == EINZAHLEN) {
```


Methode ausfuehrenFunktion (2)

```
    } else if (funktion == EINZAHLEN) {
        System.out.print("Betrag: ");
        betrag = input.nextDouble();
        konto1.einzahlen(betrag);

    } else if (funktion == ABHEBEN) {
        System.out.print("Betrag: ");
        betrag = input.nextDouble();
        konto1.abheben(betrag);

    } else if (funktion == UEBERWEISEN) {

    } else if (funktion == SET_INHABER) {
        System.out.println("Neuer Inhaber: ");
        inhaber = input.next();
        konto1.setInhaber(inhaber);

    } else if (funktion == ENDE) {
        System.out.println("Programmende");
```

Probleme mit diesem Programm?

- Wie vermeidet man Programmabbrüche bei throw?
- Warum geht das Eingeben einen Namens schief, wenn der Name aus mehr als einem Wort besteht?
- Wie implementiert man den Test für das Überweisen?

Ausnahmebehandlung einführen

```
public void start() {
        konto1 = null;
        int funktion = -1;

        while (funktion != ENDE) {
            try {
                funktion = einlesenFunktion();
                ausfuehrenFunktion(funktion);
            } catch (IllegalArgumentException e) {
                System.out.println(e);
            } catch (Exception e) {
                System.out.println(e);
                e.printStackTrace(System.out);
            }
        }
    }
```

Einlesen des Namens

```
private void ausfuehrenFunktion(int funktion) {
    int kontonr;
    String inhaber;
    double kontostand;
    double betrag;

    if (funktion == ANLEGEN) {
        System.out.print("Kontonummer: ");
        kontonr = input.nextInt();
        System.out.print("Inhaber    : ");
        inhaber = input.next();
        System.out.print("Kontostand : ");
        kontostand = input.nextDouble();
        konto1 = new Konto(kontonr, inhaber, kontostand);
```

Liest nur ein Wort ein

`input.nextLine()` liest eine ganze Zeile ein, klappt aber zunächst auch nicht, warum?

Einlesen des Namens: besser

```
private void ausfuehrenFunktion(int funktion) {
    int kontonr;
    String inhaber;
    double kontostand;
    double betrag;

    if (funktion == ANLEGEN) {
        System.out.print("Kontonummer: ");
        kontonr = input.nextInt();
        input.nextLine();
        System.out.print("Inhaber    : ");
        inhaber = input.nextLine();
        System.out.print("Kontostand : ");
        kontostand = input.nextDouble();
        konto1 = new Konto(kontonr, inhaber, kontostand);
```

Linefeed lesen

Zeile lesen

Zusätzlich sollte noch eine `InputMismatchException` gefangen werden, damit das Dialogprogramm nicht in eine Schleife gerät

InputMismatchException fangen

```
public void start() {
    konto1 = null;
    zielkonto = null;
    int funktion = -1;

    while (funktion != ENDE) {
        try {
            funktion = einlesenFunktion();
            ausfuehrenFunktion(funktion);
        } catch (IllegalArgumentException e) {
            System.out.println(e);
        } catch (InputMismatchException e) {
            System.out.println(e);
            input.nextLine();

        } catch (Exception e) {
            System.out.println("Ausnahme gefangen: " + e);
            e.printStackTrace(System.out);
        }
```

Falsche Eingabe weglesen

Überweisen implementieren

```
/**
 * Überweise einen Betrag vom aktuellen Konto auf das
 * übergebene Zielkonto
 *
 * @param zielkonto Referenz auf ein Konto (!= null)
 * @param betrag zu überweisender Betrag
 */
public void ueberweisen(Konto zielkonto, double betrag) {
    this.abheben(betrag);
    zielkonto.einzahlen(betrag);
}
```

Und wie testet man das jetzt?

Redesign des Tests (1)

```
private Konto kontoAnlegen() {
    int kontonr;
    String inhaber;
    double kontostand;
    System.out.print("Kontonummer: ");
    kontonr = input.nextInt();
    input.nextLine();
    System.out.print("Inhaber    : ");
    inhaber = input.nextLine();
    System.out.print("Kontostand : ");
    kontostand = input.nextDouble();
    return new Konto(kontonr, inhaber, kontostand);
}

private double einleseBetrag() {
    System.out.print("Betrag: ");
    return input.nextDouble();
}
```


Redesign des Tests (2)

```
if (funktion == ANLEGEN) {
    konto1 = kontoAnlegen();

} else if (funktion == EINZAHLEN) {
   konto1.einzahlen(einleseBetrag());

} else if (funktion == ABHEBEN) {
    konto1.abheben(einleseBetrag());

} else if (funktion == UEBERWEISEN) {
    if (zielkonto == null)
        zielkonto = kontoAnlegen();
    konto1.ueberweisen(zielkonto,einleseBetrag());
```