Tagesprogramm

Design-by-Contract

Vertragspartner

Anbieter (Server) bietet Leistungen (Services) an

Kunde (Client) nimmt von Anbietern angebotene Leistungen in Anspruch

Details der Inanspruchnahme in einem **Vertrag** geregelt

man kann gleichzeitig Anbieter und Kunde unterschiedlicher Leistungen sein

Vertragspartner in der Software

Clients und Server sind **Objekte**

Server bietet Services in Form von Methodenausführungen an

Client nimmt Services durch Methodenaufrufe in Anspruch

Details einer Leistung in einem **Software-Vertrag** geregelt

Software-Vertrag wird jedoch nicht durch Objekte selbst vereinbart,
sondern durch Programmierer(innen) entsprechender Programmteile

Vertragsbestandteile = Zusicherungen

Vorbedingung:

Bedingung, die vor Methodenausführung gelten muss;
Client ist dafür verantwortlich

Nachbedingung:

Bedingung, die nach Methodenausführung gelten muss;
Server ist dafür verantwortlich

Invariante:

Bedingung hinsichtlich des Zustands des Servers,
die sowohl vor als auch nach **jeder** Methodenausführung gelten muss;
Server ist dafür verantwortlich

Typischer Software-Vertrag

Rechte und Pflichten des Servers:

Server darf davon ausgehen, dass zu Beginn einer Methodenausführung
- die Vorbedingungen der Methode eingehalten sind,
- die Invarianten des Objekts erfüllt sind

Server muss dafür sorgen, dass am Ende der Methodenausführung
- die Nachbedingungen der Methode eingehalten sind,
- die Invarianten des Objekts erfüllt sind

Rechte und Pflichten des Clients:

Client muss dafür sorgen, dass bei Aufruf einer Methode
- die Vorbedingungen der Methode eingehalten sind

Client darf davon ausgehen, dass nach Rückkehr aus der Methode
- die Nachbedingungen der Methode erfüllt sind

Einfacher Software-Vertrag

```
// berechne 1 + ··· + n
// n > 0
public static int sum(int n) {
    int s = n;
    int i = n - 1;
    while (i != 0) {
        s = s + i;
        i = i - 1;
    }
    return s;
}
```

Nachbedingung

Vorbedingung

Programmverstehen:

annehmen, dass Vorbedingung erfüllt ist,
nachweisen, dass dann auch die Nachbedingung erfüllt wird

Aufgabe: DbC und Programmverstehen

Such Sie in Gruppen zu zwei bis drei Personen Antworten auf folgende Frage:

Welche Zusammenhänge bestehen zwischen

Zusicherungen zwecks Programmverstehen und
Zusicherungen im Zusammenhang mit Design-by-Contract?

Zeit: 2 Minuten

Viel Verantwortung bei Client

```
public class Sparbuch {

    private long guthaben;  // guthaben >= 0          Invariante

    // betrag > 0                                     Vorbedingung
    // guthaben wird um betrag erhöht                 Nachbedingung
    public void einzahlen(long betrag) {
        guthaben += betrag;
    }

    // betrag > 0; betrag <= guthaben                 Vorbedingungen
    // guthaben wird um betrag verringert             Nachbedingung
    public void abheben(long betrag) {
        guthaben -= betrag;
    }

    public long guthaben() { return guthaben; }
}
```

Viel Verantwortung bei Server

```
public class Sparbuch {

    private long guthaben;  // guthaben >= 0          Invariante

    // wenn true: guthaben um betrag erhöht          Nachbed.
    public boolean einzahlen(long betrag) {
        if (betrag <= 0) return false;
        guthaben += betrag;
        return true;
    }

    // wenn true: guthaben um betrag verringert      Nachbed.
    public boolean abheben(long betrag) {
        if (betrag <= 0 || betrag > guthaben) return false;
        guthaben -= betrag;
        return true;
    }
}
```

Verantwortungslos

```
public class Sparbuch {

    private long guthaben;  // guthaben >= 0          Invariante

    // wenn true: guthaben um betrag erhöht          Nachbed.
    public boolean einzahlen(long betrag) {
        return false;
    }

    // wenn true: guthaben um betrag verringert      Nachbed.
    public boolean abheben(long betrag) {
        return false;
    }
}
```

Böswillig

```
public class Sparbuch {

    private long guthaben;  // guthaben >= 0          Invariante

    // wenn true: guthaben um betrag erhöht          Nachbed.
    public boolean einzahlen(long betrag) {
        guthaben = 0;
        return false;
    }

    // wenn true: guthaben um betrag verringert      Nachbed.
    public boolean abheben(long betrag) {
        guthaben = 0;
        return false;
    }
}
```

Usancen

übliches Verhalten von Programmierer(innen) beachten

inhaltliche Bedeutung von Namen berücksichtigen

keine unerwarteten Zustandsänderungen

Richtlinien zur Vertragsgestaltung

möglichst einfach

möglichst eindeutig

Usancen einhalten

„Kleingedrucktes“ vermeiden
(keine unsinnigen Formulierungen aus Angst vor Böswilligkeit)

Verantwortung dort ansiedeln, wo Information vorhanden ist
(eher beim Server als beim Client)

zusammenspielen, nicht gegeneinander ausspielen

Aufgabe: Zusicherungen unterscheiden

Wählen Sie einige Standard-Java-Klassen.

Lesen Sie die API-Beschreibungen dieser Klassen sorgfältig durch und betrachten Sie sie als Software-Verträge.

Wer ist für die Einhaltung welcher Teile der API-Beschreibungen verantwortlich?