Kapitel 5 Zustand eines Objekts

Lernziel:
Bedingte Anweisung in Java
Objektzustand

5.1 Unverwundbar soll sichtbar sein

Die Methode *VerwundbarSetzen* sorgt dafür, dass der Attributwert von verwundbar entsprechend der Eingabe gesetzt wird. Aber bei der graphischen Darstellung von mampfi ist nicht sichtbar, ob er verwundbar oder unverwundbar ist. Dies soll sich ändern. Wenn mampfi unverwundbar wird, soll die Farbe seiner Form wechseln, z. B. auf rot.

Aufgabe 5.1
Versuche in Worten zu formulieren, wie sich die Anweisungsfolge im Methodenrumpf von *VerwundbarSetzen* ändern muss!

Abhängig vom Eingabewert der Methode *VerwundbarSetzen* muss der Wert der Farbe der form entweder auf gelb oder auf rot gesetzt werden. In Worten formuliert

wenn der Eingabewert verwundbarNeu true ist
dann
    setze den Wert des Attributs der Farbe der form auf „gelb“
sonst
    setze den Wert des Attributs der Farbe der form auf „rot“

Die Struktur dieser Anweisung ist dir von Robot Karol bekannt, es handelt sich um eine **bedingte Anweisung**. Ist die Bedingung erfüllt, so werden die Anweisungen im „dann-Teil“ ausgeführt. Ist die Bedingung nicht erfüllt, so werden die Anweisungen im „sonst-Teil“ ausgeführt. Im Folgenden ist einerseits die Syntax[1] der bedingten Anweisung in der Sprache von Robot Karol und andererseits das zugehörige Struktogramm abgebildet.[2]

```
wenn <Bedingung>
     <Sequenz1>
sonst
     <Sequenz2>
*wenn
```

Abbildung 1: Bedingte Anweisung in der Sprache von Robot Karol

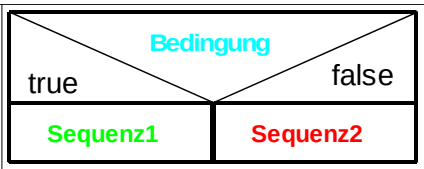


Abbildung 2: Struktogramm einer bedingten Anweisung

1 griechisch: Satzbau; Muster und Regeln nach denen Wörtern zu „Sätzen“ zusammengestellt werden

2 Solltest du dich nicht so genau daran erinnern, ist es hilfreich nochmals ein Karol-Programm mit einer bedingten Anweisung anzusehen. In der Menüzeile kann man mit Struktogramm--> Ansehen sich zu jedem korrekten Programm das zugehörige Struktogramm anzeigen lassen.

© brichzin@tcs.ifi.lmu.de, sebastian.zinkhahn@web.de

In Java sieht die Umsetzung wie folgt aus:

```
if (Bedingung)
{
    //dann-Teil
}
else
{
    //sonst-Teil
}
```

```
if (verwundbar == true)
{
    form.FarbeSetzen("gelb");
}
else
{
    form.FarbeSetzen("rot");
}
```

Abbildung 3: bedingte Anweisung in Java allgemein und am Beispiel innerhalb der Methode VerwundbarSetzen

Hinweise:

- In der bedingten Anweisung kann der sonst-Teil auch weggelassen werden. Bedingte Anweisungen können auch geschachtelt werden.
- Eine **Bedingungen** ist in Java ein beliebiger Ausdruck, der bei seiner Auswertung nur den Wert `true` oder `false` ergeben kann. Zur Formulierung werden häufig die Vergleichsoperatoren verwendet, die in Abbildung 4 aufgelistet sind.
- Vorsicht Verwechslungsgefahr: Ein einzelnes Gleichzeichen "=" steht für eine Wertzuweisung (Kapitel 3). Der Vergleichsoperator besteht aus zwei Gleichzeichen "=="!

Bedeutung	Vergleichsoperator
ungleich	!=
gleich	==
kleiner	<
größer	>
kleiner gleich	<=
größer gleich	>=

Abbildung 4: Vergleichsoperatoren in Java

Abbildung 5 zeigt als Zusammenfassung das Struktogramm zur Methode VerwundbarSetzen

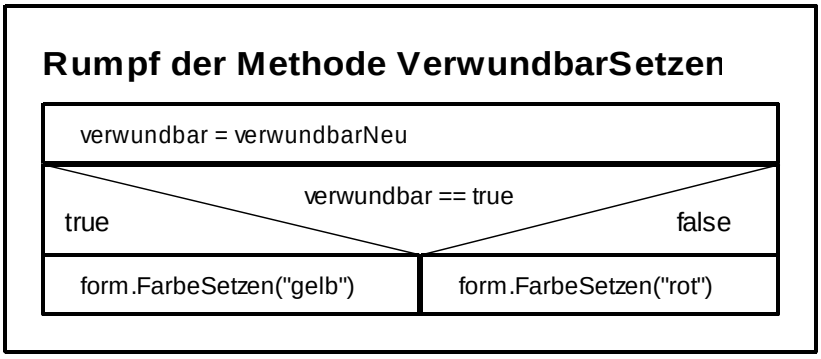


Abbildung 5: Struktogramm zur Planung der Methode VerwundbarSetzen

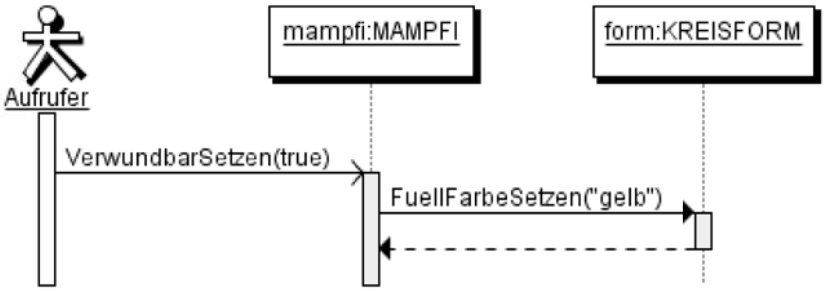
Aufgabe 5.2
Ergänze im Methodenrumpf der Methode *verwundbarSetzen* die bedingte Anweisung aus Abbildung 3 und teste die Methode!

© brichzin@tcs.ifi.lmu.de, sebastian.zinkhahn@web.de

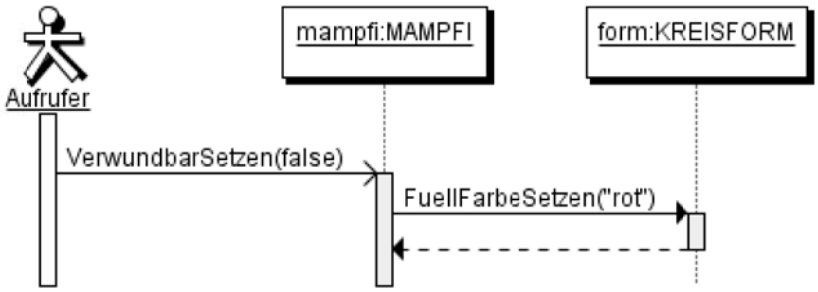
5.2 Immer wieder Objektkommunikation

Ohne Objektkommunikation funktioniert die Zusammenarbeit von Objekten nicht. Aus diesem Grund sind für die Methode verwundbarSetzen in den Abbildungen 5 drei Varianten eines Sequenzdiagramms zu sehen.

a)



b)



c)

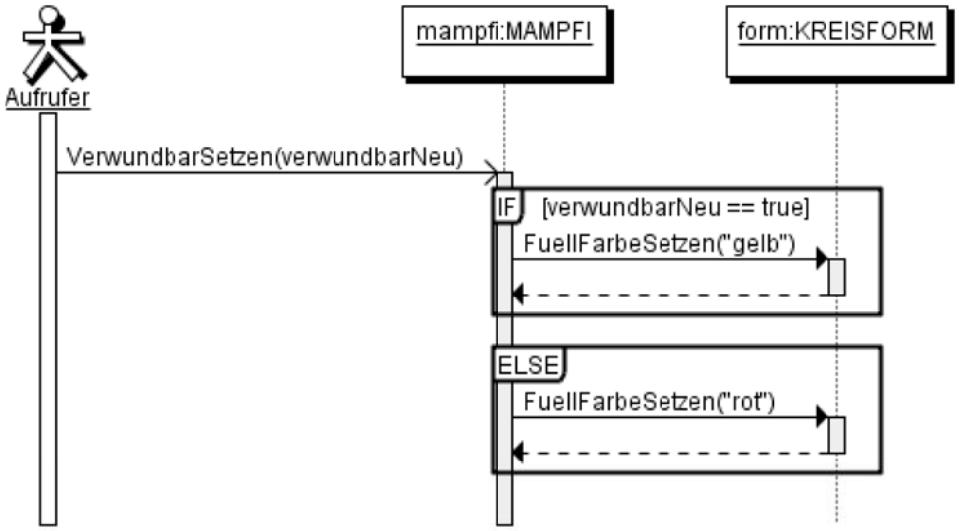


Abbildung 6: Drei Varianten eines Sequenzdiagramm für den Aufruf der Methode VerwundbarSetzen

Hinweis:
Beachte, dass die Sequenzdiagramme in Abbildung 5 den Methodenrumpf der Methode *VerwundbarSetzen* nicht vollständig wiedergeben. Es fehlt die Zuweisung

```
verwundbar = verwundbarNeu
```

Dies hat aber seine Richtigkeit, denn die Zuweisung dient zur Veranschaulichung der Objektkommunikation. Die Zuweisung ist eine Attributwertänderung „innerhalb“ eines Objekts. Es tritt keine Kommunikation dabei auf („Das Objekt führt keine Selbstgespräche“).

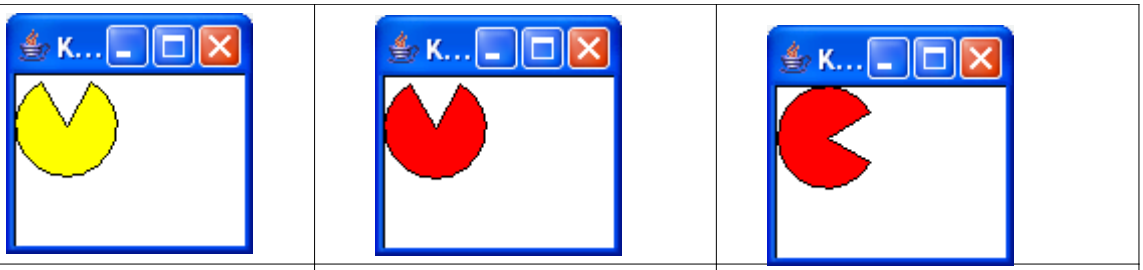
© brichzin@tcs.ifi.lmu.de, sebastian.zinkhahn@web.de

Aufgabe 5.3
Spiele die Objektkommunikation beim Aufruf der Methode *VerwundbarSetzen* als Rollenspiel nach. Zur Kennzeichnung des Werts des Attributs farbe des Objekts form wären verschiedenfarbige Mützen des Schauspielers des Objekts form eine elegante Version.
Beachte auch, dass die Anmerkung im Hinweis im Rollenspiel umgesetzt wird.

Aufgabe 5.4
Erkläre Unterschiede bzw. Gemeinsamkeiten der Sequenzdiagramme in der Abbildung 5. Verwende dabei Fachausdrücke!

5.3 Objektzustand

Je nach Blickrichtung und Verwundbarkeit befindet sich mampfi in einem anderen **Zustand.** In Abbildung 6 sind drei verschiedene Zustände abgebildet.



Zustand Z1	Zustand Z2	Zustand Z3

Abbildung 7: Unterschiedliche Zustände des Objekts mampfi, sichtbar an den unterschiedlichen Attributwerten in den Objektkarten (Die Attribute positionX und positionY werden in die Überlegungen zunächst nicht mit einbezogen)
Zur Veranschaulichung ist zu jedem Objektzustand das passende Objekt der Klasse KREISFORM abgebildet.

Aufgabe 5.5
Wie viele verschiedene, sinnvolle Objektzustände von mampfi gibt es, wenn man nur die Attribute verwundbar und blickrichtung betrachtet?

© brichzin@tcs.ifi.lmu.de, sebastian.zinkhahn@web.de

Für das Attribut blickrichtung sind vier verschiedene sinnvolle Werte möglich ('O', 'S', 'W', 'N') sind, für das Attribut verwundbar zwei (true, false). So kann mampfi 4 * 2 = 8 verschiedene Zustände annehmen, wenn man nur diese beiden Attribute betrachtet. Betrachtet man noch zusätzlich die Attribute positionX und PositionY, so gibt es noch deutlich mehr verschiedene Zustände.

Aufgabe 5.6
Wie viele verschiedene, sinnvolle Objektzustände gibt es für Objekte der Klasse KREISFORM? Die Frage ist nur sehr schwer vollständig zu beantworten. Notiere alle Überlegungen zu einer Antwort in dein Heft, auch wenn noch offene Punkte dabei sind.

Ändern sich mindestens ein Attributwert, so ändert sich auch der Zustand eines Objekts. Man spricht hier auch von einem **Zustandsübergang**. Stimmen in zwei Situationen alle Attributwerte überein, dann befindet sich das Objekt im selben Zustand.
Attributwertänderungen werden durch Zuweisungen bzw. Methodenaufrufe erreicht. Teilweise sind die Wertzuweisungen auch innerhalb einer Methode „verborgen“.

In der Informatik wird der Begriff Zustand wie folgt definiert:
Der Zustand eines Objekts wird durch die Gesamtheit der Werte aller seiner Attribute festgelegt. Ein Objekt ändert seinen Zustand, wenn sich der Wert mindestens einer seiner Attribute ändert.

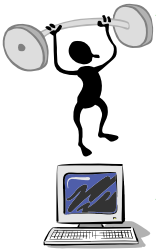
Aufgabe 5.7
Schreibe die Methodenaufrufe auf, die in Abbildung 5 für einen Zustandsübergang von Z1 nach Z2 bzw. Z2 nach Z3 sorgen. Welche Zuweisungen sind für die Zustandsübergänge entscheidend?

Aufgabe 5.8
a) Mampfi ist im Zustand Z3 in Abbildung 5. In welchen Zustand ist er, wenn der Methodenaufruf mampfi.VerwundbarSetzen(„false“) ausgeführt wurde.
b) Mampfi ist im Zustand Z3 in Abbildung 5. Es werden finden nun folgende Methodenaufrufe statt:
mampfi.NachSuedenBlicken()
mampfi.VerwundbarSetzen(„true“)
mampfi.NachWestenBlicken()
Zeichne ein Objektdiagramm von dem zustand, in dem sich mampfi nun befindet.

5.3 Zusammenfassung

Aufgabe 5.9
Fasse die wesentlichen Inhalte dieses Kapitels in deinem Heft zusammen. Folgende wichtigen Begriffe sollten dabei enthalten sein:
bedingte Anweisung, Bedingung, Zustand, Zustandsübergang

© brichzin@tcs.ifi.lmu.de, sebastian.zinkhahn@web.de



Aufgabe 5.10
Mit der bedingten Anweisung ist eine im Vergleich zu Kapitel 4 unterschiedliche Bewegung von mampfi möglich. Formuliere zusätzlich zu den Methoden *NachOstenBlicken, NachSuedenBlicken, NachWestenBlicken* und *NachNordenBlicken* neue Methoden *RechtsDrehen* und *LinksDrehen*, die eine Änderung der Blickrichtung abhängig von der aktuellen Blickrichtung umsetzen.

© brichzin@tcs.ifi.lmu.de, sebastian.zinkhahn@web.de