

## 6 Kontrollstrukturen

# 6.1 Sequenz

Mehrere nacheinander ausgeführte Anweisungen, die in Java mit einem Semikolon (;) abgeschlossen werden, werden als **Sequenz** bezeichnet.

```
wash();
brushTeeth();
wash
brush teeth
```

# 6.2 Entscheidungsanweisung

Unter einer Entscheidungsanweisung versteht man auch eine *bedingte Anweisung*. Diese gibt es zum einen **mit** und zum anderen **ohne** Alternative.

#### Mit Alternative:

```
if(<Bedinung>){
     <Anweisungen>
}
else{
     <Anweisungen>
}
```

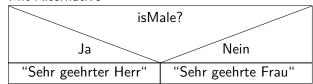
## Beispiel mit Alternative:

```
if(gender == 'm') {
    System.out.println("Sehr geehrter Herr");
}
else {
    System.out.println("Sehr geehrte Frau");
}
```

## Ohne Alternative:

```
if(<Bedingung>) {
     <Anweisungen>
}
```

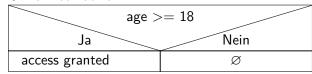
#### Mit Alternative



## Beispiel ohne Alternative:

```
if(age >= 18) {
   access = true;
}
```

### Ohne Alternative



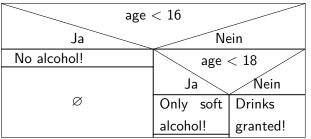
Bei der Fallunterscheidung kann es passieren, dass für verschiedene Bedingungen verschiedene Anweisungen definiert werden. Bei der Überprüfung werden diese der Reihe nach überprüft, bis die erste erfüllt ist. Hier ist zu beachten, dass die entsprechenden Bedingungen definiert sein müssen.



### Beispiel mehrere Bedingungen mit Alternative:

```
if(age < 16) {
    lightDrinks = false;
    hardDrinks = false;
}
else if(age < 18){
    lightDrinks = true;
    hardDrinks = false;
}
else{
    lightDrinks = true;
    hardDrinks = true;
    hardDrinks = true;
}</pre>
```

## Ohne Alternative



### 6.3 Mehrfachauswahl

Im Gegensatz zur Entscheidungsanweisung wird bei der der Mehrfachauswahl auf bestimmte Werte geprüft. So dass man hiermit beliebig viele Fälle ohne größeren Aufwand unterscheiden kann.

Die Java-Anweisung für die Mehrfachauswahl ist die switch-Anweisung. Dabei muss aber beachtet werden, dass die zu überprüfende Variable vom Typ byte, short, int oder char sein muss.

Das bedeutet, dass bei switch nicht nach Ungleichheit oder größer bzw. kleiner gleich entschiede werden kann. Beispiel Mehrfachauswahl:

```
switch(<Variable>){
  case <Wert1>: <Anweisung 1>; break;
  case <Wert2>: <Anweisung 2>; break;
  ...
  default: <Anweisung n>; break;
}
```

Die Anweisung break erzwingt das Verlassen der gesamten switch-Anweisung. Die nachfolgenden case-Bedingungen werden dadurch übersprungen.

