Softwarentwicklung 1 - Glossar

Livi Franke

elementare Befehle	`moveForward();` , `turnLeft();` , `turnAround();` , `turnRight();` , `pickBeeper();` , `dropBeeper();`
zusammengesetzte Befehle	Kombination aus mehreren Befehlen, in Karel mit `void`
Zählschleifen	`repeat(n){}`
Fallunterscheidung	`if(Bedingung1) else if(Bedingung2)else`
logische Verknüpfungen	`&&` (Konjunktion) sowie `||` (Disjunktion) und `!` (Negation) können genutzt werden um logische Aussagen zu verknüpfen
bedingte Schleife	`while(Bedingung)`

Compiler	Übersetzt Quelltext in maschinenausführbare Form
Anweisungen	Aktionen eines Programms
Ausnahme (Exception)	Wird geworfen, wenn während der Ausführung eines Programms ein Fehler festgestellt wird
Klassen	Definiert welche Methoden auf Objekten und Exemplaren der Klasse aufrufbar sind
Interfaces	benannte Schnittstellen; definieren die Schnittstelle einer Klasse (die durch ihre öffentlichen Methoden definiert ist) ohne sie zu implementieren
Operationen	Methodenköpfe in Interfaces
Objekte	Oft synonym mit Exemplare; eine Klasse selbst ist aber auch ein Objekt
Exemplare	Instanzen einer Klasse
Felder	Interner Zustand eines Exemplars
Prozeduren	Bestehen aus Rückgabetyp, Bezeichner und optional einer Liste von Parametern: `Rueckgabetyp Bezeichner(Parametern)/*Prozedurrumpf*`

Ergebnisprozedur	Prozedur mit Rückgabetyp anders als `void`
Rückgabetypen	Gibt an, was eine Prozedur zurückgibt; z.B.: `void` , `int` , `boolean`
Bezeichner	Name einer Prozedur mit welcher diese aufgerufen werden kann, dabei ist die Nutzung eines Bezeichners für mehrere Prozeduren nur unter bestimmten Voraussetzungen möglich (Überladen/Overloading)
Parameter	Möglichkeit um einer Prozedur Werte mitzugeben
Signatur	Legt die Anzahl, Reihenfolge und jeweiligen Typ von Parametern einer Prozedur fest
Formale Parameter	Im Kopf einer Prozedur deklarierte Variablen; im Rumpf sichtbar
Aktuelle Parameter	Parameter, die beim Aufruf einer Prozedur übergeben werden
Konstruktor	Ausgeführt beim Erzeugen eines Exemplars; bringt Exemplar auf einen Anfangszustand
Methoden	Variante von Prozeduren in von Java-Klassen

sondierende Methoden	Geben Variablenwerte zurück ohne sie zu verändern
verändernde Methoden	Verändern Variablen
öffentliche Methoden	Mit `public` deklarierte Methoden; außerhalb einer Klasse auf ein Objekt anwendbar
private Methoden	Mit `private` deklarierte Methoden; nicht von außerhalb nutzbar
Methodenkopf	Bestehen aus Rückgabetyp, Bezeichner und optional einer Liste von Parametern
Methodenrumpf	Sequenz an Anweisungen, die von einer Methode ausgeführt wird
Überladen (Overloading)	Wenn mehrere Methoden den gleichen Methodennamen haben, aber unterschiedliche Signaturen haben
Klient	Sind Objekte, die Methoden an anderen Objekten aufrufen
Dienstleister	Sind Objekte, deren Methoden von anderen Objekten aufgerufen werden

Punktnotation	Erlaubt Zugriff auf öffentliche Methoden und Variablen eines Objektes durch `Objekt.Methode()` bzw. `Objekt.Variable`
Literale	Zeichenfolge im Quelltext, welche einen Wert repräsentiert
Variablen	Speicherplätze für Werte, müssen in Java deklariert werden mit Typ und Bezeichner
lokale Variablen	eine lokale Variable innerhalb einer Prozedur existiert nur während der Ausführung dieser Prozedur
Belegung	Wert einer Variable
Deklarationen	Reservieren von Speicher unter einem Variablennamen
Variablentyp	Legt fest, welche Werte eine Variable annehmen kann, man unterscheidet zwischen primitiven Datentypen und Referenztypen
Konstanten	Mit `final` deklarierte Variablen können nur einmal initialisiert werden und danach nicht mehr überschrieben werden

primitive Datentypen	`int` (Ganzzahl), `float` (32 Bit), `double` (64 Bit), etc.
boolesche Ausdrücke	Ausdrücke, die als Ergebnis einen booleschen Wert `true` oder `false` liefern
Zeichenketten (Strings)	Eine Zeichenfolge; prinzipiell ein `char`-Array; sind nicht veränderbar, stattdessen liefert jede Änderung ein neues Objekt; Literale durch doppelte Anführungszeichen
Typumwandlungen	Entweder implizite (cast) Konversion oder explizite Konvertierung
Sichtbarkeitmodifikator	`public` oder `private`
Operatoren	Verknüpfen Operanden zu Ausdrücken, z.B. Addition, Subtraktion, etc.
dynamischer Typ	Typ einer Objekts, das zur Laufzeit in einer Variable gehalten wird
statischer Typ	Typ mit dem eine Variable im Quelltext deklariert wurde

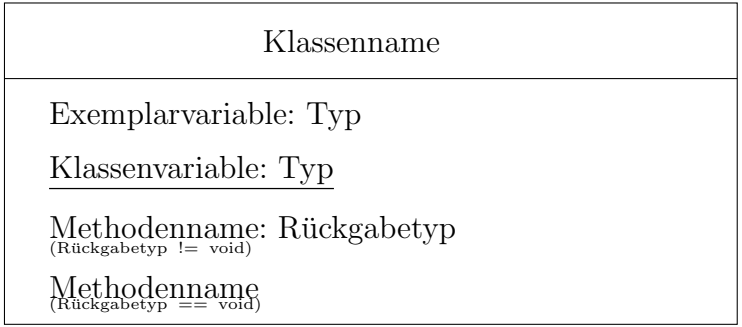
Null-Typ	Typ einer Referenzvariable, die an kein Objekt gebunden ist
Sichtbarkeitsbereich	Bereich in dem eine Variable benutzt werden kann; Variablen sind nicht außerhalb einer Programmeinheit sichtbar
Lebensdauer	Zeit während der Ausführung eines Programmes, in der einer Variable oder einem Objekt Speicher zugeteilt ist
Klassenvariablen und Klassenoperationen	Mit `static` deklarierte Variablen und Operationen; kein Klassenexemplar nötig um diese aufzurufen
Arrays	Form von Sammlungen gleichartiger Elemente; indexbasierter Zugriff; in der Größe fest (Java): `<Typ>[] Variablenname = new <Typ>[Laenge];`
Sammlungen	Meist dynamische Behälter mit dynamischem Platz; Arrays sind nicht dynamisch Unterschied in Relevanz von Position und Erlaubnis von Duplikaten
Elemente	Inhalt von Sammlungen

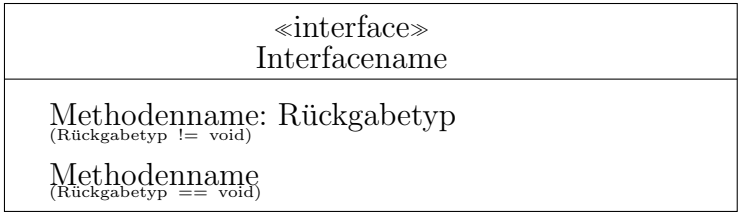
Gleichheit	Etwa wenn zwei Objekte einer Klasse die gleichen Werte in ihren Exemplarvariablen haben
Identität	Ein Objekt ist dies nur zu sich selbst; Impliziert Gleichheit
Sammlungsbibliothek (JCF)	Bietet verschiedene Interfaces und Klassen für verschiedenartige Sammlungen; z.B.: `List<String> stringList;`
Pakete (packages)	Sind Bibliotheken der Sprache; müssen vor Verwendung importiert werden; z.B.: `import java.util.*;`
Referenztypen	Eine Variable mit dem Typ einer Klasse, sie enthält kein Objekt sondern eine Referenz auf ein Objekt
Programmierschnittstelle (API)	In der API Specification beschrieben, welche u.a. Dokumentationen beinhaltet

Kontrollstrukturen	Bestimmen die Ausführungsreihenfolge von Anweisungen; Sequenzen, Fallunterscheidungen, etc.
Sequenzen	Abfolge von Anweisungen; in Java in chronologischer Reihenfolge, kann Unterbrechungen / Wiederholungen haben
Block	Fasst eine Sequenz an Anweisungen mit geschweiften Klammern zusammen; Rümpfe
Auswahl	Ausführen einer Sequenz in Abhängigkeit einer Bedingung; ein- bis mehrseitig (`if`, `else if`, `else`)
Wiederholung	Sequenzen wiederholende Kontrollstrukturen (manchmal mit veränderten Werten); auch Schleifenkonstrukte; Zählschleifen und bedingte Schleifen in Java
Zählschleife (Java)	`for(Startwert, Gueltigkeit, Regel){/*Rumpf*/}`
Erweiterte Zählschleife (Java)	`for(Element: Sammlung){/*Rumpf*/}`
bedingte Schleife (Java)	`while(Schleifenbedingung){/*Rumpf*/}`

abweisend / kopfgesteuert	Wenn es dazu kommen kann, dass der Schleifenrumpf nicht ausgeführt wird
nicht abweisend / endgesteuert	Wenn der Schleifenrump grundsätzlich mindestens einmal ausgeführt wird
positiv bedingt	Wenn die (jeweils nächste) Ausführung des Schleifenrumpfes von einer Bedingung abhängt; in Java bei allen Schleifen
zielorientiert bedingt	Wenn der Schleifenrumpf solange ausgeführt wird, bis die Bedingung zutrifft
Rekursion	Um Rekursion zu verstehen muss man Rekursion verstehen
Aufruf-Stack	Last In Fisrt Out (LIFO) Prinzip; dient zum Ablegen von lokalen Variablen bei Methodenaufruf; entfernen durch `return`
Heap	Zum Verwalten von Objekten; Objekte verbleiben mindestens bis es keine Referenz mehr auf sie gibt

Unified Modeling Language	Graphische Sprache zur Beschreibung von Softwaresystemen
Klassendiagramm (Klassen)	Beschreibt die statische Struktur einer Klasse:
Klassendiagramm (Interfaces)	Beschreibt die statische Struktur eines Interfaces:





Objektdiagramm

Beschreibt einen Schnappschuss eines Objekts:

:Exemplartyp
Exemplarvariable = Belegung