Dokumentation M226a

# Lerndoku

# Kapitel 1

**Wichtige Begriffe / Funktionen**

**Klassen** = Im Greenfoot die Akteure oder die Welten die rechts aufgelistet sind

**Objekt/Instanzen** = Die in die Welt gesetzten Klassen

**Methode** = Aktion die ein Objekt ausführen kann

**Methodenaufruf** = eine Methode die ausgeführt wird

**Rückgabetyp** = (Bsp. void) Rückgabetypen bestimmen was nach der Methode gemacht wird, im Falle void wird die Methode nur ausgeführt und beendet. Als Rückgabetyp kann auch int(integer) oder Boolean verwendet werden, wie beim Wombat.

**Integer(int)** = Zahlen ohne Dezimalstellen rausgeben (Bsp. -3, 120039)

**Boolean** = kann nur **true** oder **false** wiedergeben

**Ein Bild, das Text enthält.

Automatisch generierte BeschreibungParameter** = Die Klammer nach der Methode «void move**()**» wenn in der Klammer eine oder mehrere Informationen stehen erwartet die Methode zusätzliche Informationen die wir bereitstellen müssen. Im Screenshot rechts steht also int direction. Dort wird also ein Dialogfeld aufgehen in welchem wir Zahlenmässig die richtung vorgeben müssen bevor die Methode ausgeführt wird.

**Methodensignatur** = Ausführliche beschreibung der ganzen Methode mit Parameter

**Klassendiagramm** = Mover, Actor, World. Dies können sich unterordnen oder überordnen. Per Pfeil wird gezeigt welche Klasse mit welcher über oder unterklasse eine IST= Beziehung hat. Unterklasse ist eine Klasse, die eine spezialisierung einer anderen Klasse ist.

**Hilfsklasse** = Wird von den Aktiven Klassen gebraucht und kann nicht in die Welt gesetzt werden.

**Quelltext** = Die Programmierung eines Objektes/einer Klasse was kann/darf sie und wie wird sie gesteuert

**Kompiler/Kompilieren** = Übersetzung von Quellcode zu Computersprache. Damit der Computer die Programmierung versteht. Computer verstehen keinen Quellcode.

# Kapitel 2

**Wichtige Begriffe / Funktionen**

**Move(«Zahl»)** = wird verwendet, um ein Objekt nach einem Tastendruck oder einem Act zu befehlen die in Klammern gesetzte Zahl sich zu bewegen. (Auch Minuszahlen für Gegenübergesetzte richtung)

**If-Anweisung** = Wenn eine if-Bedingung erfüllt wird werden die nächsten Parameter ausgeführt. Ansonsten wird die Methode beendet. (Gilt nur wenn es nur eine if Bedingung gibt, falls es noch ein else gibt kann die Methode trotzdem ausgeführt werden)

**Ein Bild, das Text enthält.

Automatisch generierte BeschreibungFolgen** = die Reihenfolge (mit Klammern definiert) bestimmt welche Methoden, wie, und wann ausgeführt werden

# Kapitel 3

**Input 02.09.21
KELWichtige Begriffe / Funktionen**

**Klassenvariabel** = Ist in der Klasse und alle Objekte haben darauf Zugriff (Methode)

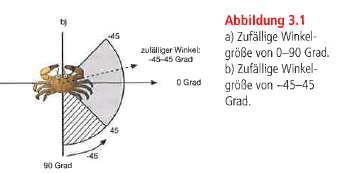
**Instanzvariable** = wird bei jeder Instanz individuell erzeugt. Die versch. Objekte kennen die Instanzvariablen voneinander nicht!

**GetRandomNumber** = liefert eine Zufallsgenerierte Zahl welche zwischen dem Grenzwert und 0 (Null) liegt / Greenfoot.GetRandomNumber(20). Somit bekommt man eine Zahl zwischen 0-19.

**(Static) Klassenmethoden** = (Objekt/Klassenname.Methodenname) Der Objekt- oder Klassenname entscheidet wo Greenfoot nach der Methode nach dem Punkt sucht. Methoden die zu Klassen und nicht zu Objekten gehören werden mit **static** gekennzeichnet.

**Ein Bild, das Text enthält.

Automatisch generierte BeschreibungZufallsturn** = Objekt bewegt sich Zufällig nach links oder Rechts



**Neue Methode erstellen**Ein Bild, das Text enthält.

Automatisch generierte Beschreibung



**static boolean isKeyDown(String key) =** Prüft ob die Taste XY gedrückt wurde.

**Ref: 3.9 Eigenen Sound herstellen**

**Auto Fill-In =** Strg + Leertaste (es öffnet sich ein Menü, wo man die Befehle auswählen kann)

# Kapitel 4

**Wichtige Begriffe / Funktionen**

**Konstruktoren =** Methode /Keinen Rückgabetyp, Name lautet gleich wie der der Klasse

**Zustand =**

**Variablen =** Instanzvariable, Variable worauf nur eine bestimmte Instanz Zugriff hat

Klassenvariable (Lokale Variable), Variable worauf alle Instanzen einer Klasse Zugriff haben.

Variablen besitzen einen Typ z.B. Boolean. Man kann sich Variablen als Container vorstellen.

**Variablendeklaration =** z.B.

**Zuweisung =** Der Inhalt einer Variable. Was wird in Ihr gespeichert.

**New =** erzeugt ein neues Objekt einer Klasse (Quellcode) wird gefolgt vom Klassennamen.

**Referenz =** Variablen können Referenzen zu einer Klasse beinhalten, welche danach ein Objekt davon erstellen kann oder darauf zugreifen kannn.

**Prepare() =** speichert die Welt zum Zeitpunkt wo man speichern gedrückt hat.

# Kapitel 5

**Wichtige Begriffe / Funktionen**

**Strings =**

**String-Verknüpfung =**

**Abstraktion =**

**Casting =**

**This =** verweist auf das aktuelle Objekt (auf sich selbst)

**setLocation(inr x, int y) =** erwartet 2 Parameter X und Y Koordinaten, platziert Akteur an dieser Stelle.

**getX() und getY() =** geben die aktuellen X und Y Koordinaten als integer Werte zurück

**Private und Public =** public, wenn Methoden von anderen Klassen aufgerufen werden soll und private, wenn die Methode für den internen Aufruf gedacht ist.

**Ein Bild, das Text enthält.

Automatisch generierte BeschreibungObjekt in unregelmässigen Abständen erzeugen =**

# Kapitel 6

**Wichtige Begriffe / Funktionen**

**Abstraktion =** löst eine ganze Klasse von Problemen anstatt nur ein einzelnes Problem z.B.das Abspielen von einer Reihe von Klaviertönen)

**Schleifen =** z.B. while, Schleifen sind sich repetierende Anweisung welche so lange durchgeführt wird bis eine Bedingung erfüllt ist.

**Felder =** Objekt welches mehrere Variablen hält, zugegriffen wird über ein Index

**Elemente =** Zugriff auf einzelne Elemente erfolgt über eine eckige Klammer/  **[]**

# Kapitel 7

**Wichtige Begriffe / Funktionen**

**null =** Ein Wert welcher Objektvariablen zugewiesen werden kann. Wenn eine Variable null enthält dann hat sie aktuell keine Referenz auf ein Objekt.

**Java-Klassenbibliothek =** umfangreiche Sammlung vordefinierter Klassen. Man kann mit Hilfe einer import Anweisung darauf zugreifen.

**Sammlung (Collection) =** Spezielle Objektart welche viele andere Objekte halten kann

**LIST =** Beispiel für eine Sammlung

**foreach-Schleife =** **Ein Bild, das Text enthält.

Automatisch generierte Beschreibung**

**getOneIntersectingObject =** Referenz auf Akteur mit dem man sich überschneidet

**generischer Typ =** Typ der zweiten Typ Namen als Parameter übernimmt. Z.B. List<Actor>

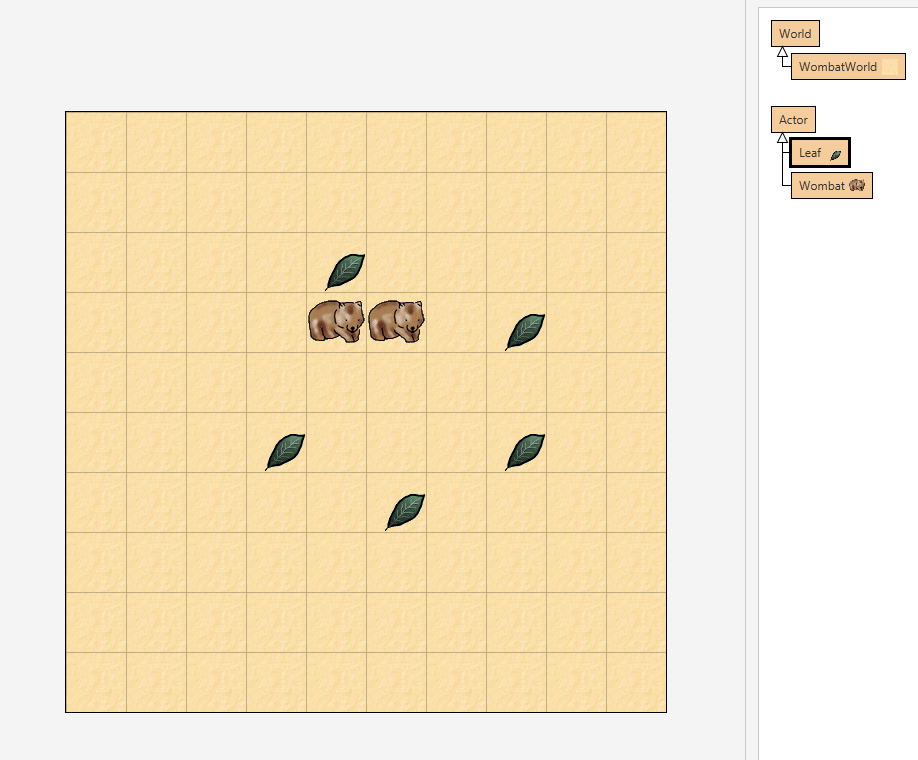
List<String>

# Aufgaben (Kapitel 1-7)

## Übung 1.1

Erzeuge weitere Wombats in der Welt. Erzeuge einige Blätter (/leaf).

* Bei dieser Aufgabe muss aus den rechts aufgeführten Klassen, Instanzen in die Welt gesetzt werden.



## Übung 1.2

Rufe für einen Wombat die Methode **move()** auf. Welche Operation

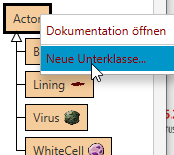
ist damit verbunden? Wiederhole diesen Schritt mehrmals. Rufe dann

die Methode **turnleft()** auf. Platziere zwei Wombats in deiner Welt und

sorge dafür, dass sie sich anschauen.

* mit **move()** bewegt sich der Wombat nach rechts. Nachdem ich ihn mit **turnleft()** 90 Grad im Gegenurzeigersinn bewegt habe und dann nochmals **move()** befohlen habe. Konnte ich feststellen das er sich in die Richtung, wo er schaut, bewegt.

# Übung 5.9

Erzeuge eine neue Klasse für einen Virus in deinem Szenario (als Unterklasse von Actor) mit dem Namen Virus. Du findest ein vorbereitetes Bild im Szenario, das du direkt verwenden kannst (Abbildung 5.2).

# Übung 5.10

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungErweitere die act-Methode von Bloodstream, um neue Virus-Objekte am rechten Bildschirmrand hinzuzufügen. Die y-Koordinate sollzufällig gewählt sein. Setze die Wahrscheinlichkeit von neuen Objekten auf 1% (einmal in 100 act-Schritten). Teste deinen Code. (Virus-Objekte sollten gelegentlich am rechten Rand auftauchen, sich aber noch nicht bewegen.)

# Übung 5.11

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungLasse die Virus-Objekte eine Linksbewegung und eine Rotation ausführen (wie Bakterien). Die Viren sollen sich jedoch vier Zellen pro act-schritt (nicht zwei) bewegen und gegen den Uhrzeigersinn rotieren.

# Übung 5.12

Kommentiere deine Virus-Klasse. (Das heißt: fülle den Klassenkommentar ganz oben sowie den Methodenkommentar über der act-Methode aus.)

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Übung 5.13

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungErzeuge eine neue private Methode in der Klasse WhiteCell mit Namen checkCollision. Der Methodenrumpf kann anfänglich leer sein. Rufe diese Methode aus deiner act-Methode auf.

# Übung 5.14

Füge Code in der Methode checkCollision ein, der Bakterien entfernt, wenn wir sie berührt haben. (Verwende die Methoden isTouchingo und removeTouching(. .) analog dem Krabben-Szenario.)

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Übung 5.15

Spiele einen Sound ab, wenn Bakterien entfernt werden. Den Sound,, slurp.wav" findest du auch wieder im Szenario - du kannst ihn verwenden.

# Übung 5.16

Denselben Soundeffekt noch einmal zu benutzen, ist ein bisschen zu bequem. Erzeuge und verwende einen neuen Sound für das Löschen von Bakterien.

# Übung 5.17

Füge wieder ähnlichen Code zur Methode checkCollision hinzu (eine weitere if-Anweisung). um zu prüfen, ob wir einen Virus berühren. Falls ja, spiele einen Sound ab (es gibt einen Sound namens ,,gameover. wav" für diesen Zweck) und beende die Ausführung.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Übung 6.1

Öffne das Szenario piano-1 und rufe den Quelltext für die beiden bestehenden Klassen Piano und Key auf. informiere dich darüber; welcher Code bereits vorhanden ist und was er macht.

# Übung 6.2

Erzeuge ein Objekt der Klasse Key und füge es in die Welt ein. Erzeuge mehrere Objekte und setze sie nebeneinander.

# Ein Bild, das Text enthält. Automatisch generierte BeschreibungÜbung 7.2

# Übung 7.3

# Ein Bild, das Text enthält. Automatisch generierte Beschreibung

# Übung 7.4

Die Methode getOneIntersectingObject gehört zu der Klasse Actor

# Ein Bild, das Tisch enthält. Automatisch generierte BeschreibungÜbung 7.5

# Übung 7.6

# Übung 7.7

Ein Bild, das Text enthält.

Automatisch generierte BeschreibungMit einer While Schlaufe 18 Blätter erzeugen, welche in der setup() Methode in der Klasse World ist.

# Übung 7.8

Ein Bild, das Text, Vogel enthält.

Automatisch generierte BeschreibungchangeImage Methode ausprobieren.

# Übung 7.10

7 konstruktoren

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Übung 7.11

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

# Übung 7.14

