3.1.5 Einführung in die Bedienung der Python-Programmierumgebung

Python ist eine textbasierte Programmiersprache, die der Kategorie der höheren Programmiersprachen zugeordnet wird. In den meisten Fällen werden Python-Programme interpretiert. Dies bedeutet, dass kein Compiler vorab das Programm in Maschinencode übersetzt, sondern erst zur Laufzeit die Übersetzung passiert. Dies hat den Vorteil, dass Skripte direkt ausgeführt werden können, was auch bei LEGO der Fall ist. Nachteilig an diesem Vorgehen ist, dass die korrekte Ausführbarkeit erst während der Laufzeit geprüft wird und somit Fehler im Programmcode nicht von der LEGO-Programmierumgebung beim Erstellen der Programme geprüft und gemeldet werden können (z. B. fehlerhaft genutzte Variablen). Python hat das Ziel, einen gut lesbaren, knappen Programmierstil zu unterstützen, was beispielsweise dazu führt, dass Einrückungen anstelle von geschweiften Klammern für die Strukturierung von Blöcken genutzt werden müssen.

Python ist eine objektorientierte Programmiersprache, unterstützt aber auch andere Programmierparadigmen wie die aspektorientierte und die funktionale Programmierung. Variablen erhalten in Python keinen festen Typ, vielmehr bietet Python eine dynamische Typisierung an.

Python bietet wie Java für die Behandlung von Fehlern die Ausnahmebehandlung (Englisch: exception handling), was kurz in Kapitel 6.6 aufgegriffen wird.

Für die Ansteuerung des Hubs, der Motoren und Sensoren liefert LEGO eine vollständige aus verschiedenen Modulen bestehende Python-Bibliothek mit. Damit lassen sich die verfügbaren Komponenten vollständig ansteuern und verwalten.

Die folgenden Klassen spiegeln dabei die wichtigsten Bereiche wider.

Python-Klassen	Kurzerläuterung
App	Wiedergabe von Musik auf dem verbundenen Computer/Tablet
Button	Abfrage des Status der linken oder rechten Taste des Hubs, Zugriff über die Hub-Klasse (MSHub bzw. PrimeHub)
ColorSensor	Zugriff auf den Farbsensor zum Auslesen der Sensorwerte und Warten auf Ereignisse
ForceSensor	Zugriff auf den Kraftsensor (nur LEGO Spike Prime) zum Auslesen der Sensorwerte und Warten auf Ereignisse
DistanceSensor	Zugriff auf den Ultraschallsensor zum Auslesen der Sensorwerte und Warten auf Ereignisse
LightMatrix	Ansteuerung der Lichtmatrix, Zugriff über die Hub-Klasse (MSHub bzw. PrimeHub)
MotionSensor	Abfrage des Status des Kreiselsensors (Drehwinkel, Orientierung, Gesten o. ä.), Zugriff über die Hub-Klasse (MSHub bzw. PrimeHub)
Motor	Zugriff auf einen einzelnen Motor zum Auslesen des Drehwinkels und zum Starten und Stoppen des Motors
MotorPair	Zugriff auf zwei Motoren zur gleichzeitigen Ansteuerung

Python-Klassen	Kurzerläuterung
PrimeHub/MSHub	Zugriff auf den Hub und die darin integrierten Komponenten (Lichtmatrix, Kreiselsensor, Tasten, Lautsprecher)
Speaker	Zugriff auf den Lautsprecher und Ausgabe von Tönen
StatusLight	Ansteuerung des Statuslichts des Hubs in der Zentraltaste, Zugriff über die Hub-Klasse (MSHub bzw. PrimeHub)
Timer	Zugriff auf den Zeitgeber zur Abfrage von Systemstartzeiten

Die Ansteuerung der Motoren und Sensoren wird durch Übergabe bestimmter Parameter geregelt, die den jeweiligen Port bestimmen. Dabei können der Klasse `MotorPair` auch mehrere Ports übergeben werden, um zum Beispiel zwei Motoren gleichzeitig zu starten.

Beispielcode	Beschreibung
`motor_pair = MotorPair('A', 'B')` `motor_pair.move(10, ,cm')`	Initialisiert ein Motor-Paar, welches an die Ports A und B angeschlossen ist, startet beide Motoren für eine Fahrt von 10 cm (bezogen auf die Größe der Standard-Räder des Sets)
`motor = Motor('A')` `motor.run_for_degrees(90)`	Initialisiert den Motor an Port A und lässt diesen um 90 Grad im Uhrzeigersinn drehen

Die Python-Programmierumgebung von LEGO erzeugt beim Anlegen eines neuen Python-Projekts automatisch ein Grundgerüst eines Python-Programms. Damit die Klassen für die Programmierung des Hubs genutzt werden können, müssen diese in das Programm importiert werden. Details zum Thema Import werden noch u. a. in Abschnitt 3.1.7 behandelt. Das Grundgerüst enthält bereits die wichtigsten Imports von Klassen, die für die gewöhnliche Programmierung benötigt werden. Für die Aufgaben in diesem Buch sind nicht alle Imports notwendig, können aber einfach bestehen bleiben. Wenn weitere Module, Klassen oder Funktionen benötigt werden, wird dies bei der jeweiligen Aufgabe behandelt. An dieser Stelle besteht allerdings ein Unterschied zwischen LEGO Mindstorms und LEGO Spike Prime. Die relevanten Python-Klassen sind bei LEGO Mindstorms im Modul `mindstorms` zu finden, bei LEGO Spike Prime im Modul `spike`. Außerdem ist die Klasse zum Ansteuern des Hubs in den beiden Modulen unterschiedlich benannt. Bei LEGO Mindstorms heißt die Klasse `MSHub` bei LEGO Spike Prime `PrimeHub`.

Bei LEGO Mindstorms sieht das Grundgerüst eines Python-Programms, welches die Programmierumgebung erzeugt, folgendermaßen aus:

```
from mindstorms import MSHub, Motor, MotorPair, ColorSensor, DistanceSensor, App
from mindstorms.control import wait_for_seconds, wait_until, Timer
from mindstorms.operator import greater_than, greater_than_or_equal_to, less_
than, less_than_or_equal_to, equal_to, not_equal_to
import math
```

```
# Create your objects here.
hub = MSHub()

# Write your program here.
hub.speaker.beep()
```

Bei LEGO Spike Prime sieht das Grundgerüst eines Python-Programms, welches die Programmierumgebung erzeugt, folgendermaßen aus:

```
from spike import PrimeHub, LightMatrix, Button, StatusLight, ForceSensor,
MotionSensor, Speaker, ColorSensor, App, DistanceSensor, Motor, MotorPair
from spike.control import wait_for_seconds, wait_until, Timer
from math import *

hub = PrimeHub()

hub.light_matrix.show_image('HAPPY')
```

Hinweis für die weitere Nutzung der Python-Beispielprogramme

Im weiteren Verlauf des Buchs werden bei den Beispielprogrammen die notwendigen Import-Anweisungen nicht mehr explizit mit abgedruckt, da von den Standard-Importen ausgegangen wird. Zusätzlich notwendige Importe werden bei den jeweiligen Aufgaben erläutert.

Außerdem wird bei Zugriff auf den Hub die Klasse des LEGO Mindstorms `MSHub` abgedruckt. Alle Nutzer des LEGO Spike Prime müssen diese durch `PrimeHub` in den Beispielprogrammen ersetzen.

Alle Beispielprogramme stehen sowohl für den LEGO Mindstorms als auch LEGO Spike Prime auf der Verlagsseite zum Download zur Verfügung.

Die Onlinehilfe für die Python-Programmierumgebung kann über das Symbol auf der rechten Bildschirmseite eingeblendet werden. Über dieses Symbol lässt sich die Hilfe auch vergrößern und verkleinern. Bei LEGO Mindstorms wird das Symbol  genutzt, bei LEGO Spike Prime  . Es öffnet sich daraufhin die Wissensdatenbank.

In der Software LEGO SPIKE 1.3.5 liegt diese in deutscher Sprache vor. In LEGO Mindstorms 10.1.0 wird sie nur in englischer Sprache angeboten und sieht folgendermaßen aus:

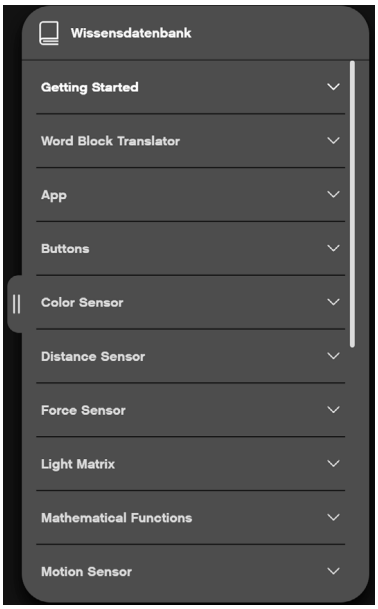

Abb. 3–10 // Python-Wissensdatenbank und Onlinehilfe beim LEGO Mindstorms

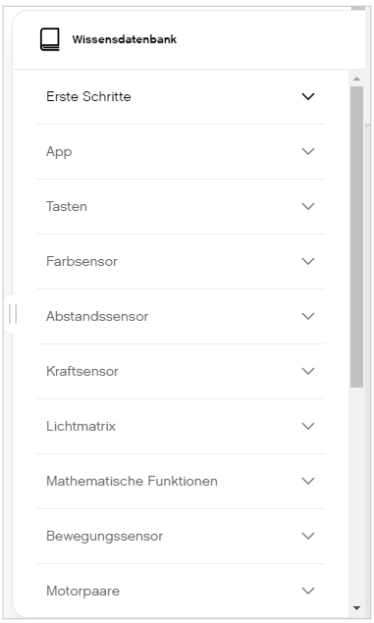

Abb. 3–11 // Python-Wissensdatenbank und Onlinehilfe beim LEGO Spike Prime

Einen sehr wichtigen Bereich der Programmierumgebung stellt die Konsolenausgabe dar. Bei einem Fehler im Programm werden dort die relevanten Fehler ausgegeben. Weiterhin kann das Programm auf der Konsole Statusinformationen oder Meldungen ausgeben, indem es die Funktion `print` nutzt. Ohne die Konsole sind aufgrund des fehlenden Displays im Hub Fehler in einem Python-Programm schwer bis gar nicht zu lokalisieren. Die Konsole wird durch das Symbol in der Mitte im unteren Bereich sichtbar gemacht.

Bei LEGO Mindstorms wird dabei das Symbol genutzt, bei LEGO Spike Prime das Symbol . Das folgende einfache Beispielprogramm gibt »Hallo Welt« auf der Konsole aus.

```
print('Hallo Welt')
```

In der Konsole wird die aktuelle Uhrzeit jeweils mit ausgegeben, was somit zu folgender Ausgabe führt:

Abb. 3–12 // Darstellung der Python-Konsole mit Ausgabe

Im Gegensatz zur Programmieroberfläche mit den Textblöcken ist ein Kopieren des Quellcodes mit Strg+C und das Einfügen mit Strg+V im Python-Editor möglich, wodurch Programmteile aus einem Programm in ein anderes Programm problemlos kopiert werden können.

Ein Kopieren der Konsolenausgabe ist zum Zeitpunkt der Drucklegung nur in der Programmierumgebung für den LEGO Spike Prime möglich.

3.1.6 Kurze Einführung in die Objektorientierung

In diesem Abschnitt wird eine kurze Einführung zur objektorientierten Programmierung gegeben. Dabei wird auf einige wenige der wichtigsten Begriffe eingegangen. Es sollen hier nur die wichtigsten Grundlagen vermittelt werden, die für die direkte Programmierung des LEGO Mindstorms bzw. LEGO Spike Prime notwendig sind. Es ist nicht das Ziel, eine vollständige Vermittlung der Prinzipien der Objektorientierung zu versuchen, da hierfür ganze Bücher gefüllt werden.

Bei der prozeduralen Programmierung, wie sie zum Beispiel bei Basic verwendet wird, wird das Programm inhaltlich durch Prozeduren (sprich Unterprogramme) strukturiert, die nacheinander gerufen werden können.

Im Gegensatz dazu geht die Objektorientierung davon aus, dass ein Gesamtsystem sich durch Objekte beschreiben lässt, die sich durch Eigenschaften und Aktivitäten/Funktionen auszeichnen. Durch die Verknüpfung von Objekten wird dabei ein Gesamtsystem aufgebaut.

Beispiel

Ein sehr beliebtes Beispiel ist dabei ein Auto. Ein Auto besteht aus verschiedenen Teilen, die Eigenschaften haben und durch Aktivitäten das Gesamtsystem beeinflussen. Das Auto hat dabei einen Motor und mehrere Räder. Ein Rad kann sich drehen und wird durch die Aktivität des Motors beeinflusst. Die Türen haben die Möglichkeit, sich zu öffnen, und haben als sichtbare Eigenschaften eine Farbe. Diese wenigen Beispiele sollen verdeutlichen, dass sich im Grunde alle Systeme als Verbindungen von Objekten darstellen lassen, die sich gegenseitig beeinflussen beziehungsweise sich gegenseitig bedingen.

Die Objektorientierung erfordert somit auch eine andere Art der Herangehensweise an die Erstellung eines Programms im Vergleich zu einer prozeduralen Programmiersprache.

In Python ist auch ein prozeduraler Programmierstil möglich. Bei der Umsetzung mit Python wird in diesem Buch sogar öfter explizit dieser Stil angewendet, um die Vergleichbarkeit der Lösungen in den beiden Programmiersprachen besser zu verdeutlichen. Erst in den letzten Kapiteln wird stärker auf die Objektorientierung eingegangen, dabei werden die vorgestellten Lösungen mit entsprechenden Diagrammen erläutert.

In der Objektorientierung sind einige wichtige Kernbegriffe vorhanden, die die Grundlagen bestimmen.

- Klasse

  Eine Klasse stellt die allgemeine Definition eines Objekts dar und beinhaltet die Eigenschaftsbeschreibungen und Aktivitäten in verallgemeinerter Form, ohne dass diesen konkrete Werte zugewiesen sein müssen. Diese sind als Programmcode vorhanden. Eine Klasse kann nicht direkt ausgeführt werden.

  Ein Auto stellt dabei eine Beschreibung und Definition dar und kann sowohl Eigenschaften (zum Beispiel Farbe) als auch Aktivitäten (zum Beispiel Starten des Motors) haben. Eine Klasse sollte in Python mit einem Großbuchstaben (»UpperCaseCamelCase«) beginnen.

- Objekt

  Ein Objekt ist eine konkrete Instanz einer Klasse und repräsentiert eine konkrete Ausprägung. Ein Objekt wird während der Ausführung des Programms benötigt. Dabei erhalten Eigenschaften konkrete Werte.

  Im Fall des Autos stellt dies ein fertig produziertes, fahrbereites, konkretes Fahrzeug in der Farbe Rot dar.

- Kapselung

  Ein Grundkonzept der Objektorientierung ist die Kapselung von Daten und Eigenschaften. Ein Objekt kapselt dabei den vollständigen Zugriff auf dessen Inhalt und erlaubt nur auf die Inhalte Zugriff, die von außen erreichbar sein sollen.

- Methode
  Eine Methode ist ein von außen erreichbarer Kommunikationspunkt, der entweder Zugriff auf Daten oder die Ausführung von Programmlogik auf Basis eines konkreten Objekts (= Instanz einer Klasse) erlaubt. Damit wird eine Aktivität oder Funktion auf Basis eines Objekts gerufen.
  Der Name einer Methode sollte in Python nur aus Kleinbuchstaben bestehen. Mehrere Wörter in einer Methode sollten dann per Unterstrich getrennt werden.

- Vererbung
  Die Objektorientierung definiert eine Vererbungshierarchie. Dabei können konkretere Klassen existieren, die eine definierte Klasse genauer beschreibt. Ein Auto ist beispielsweise ein Pkw. Dies bedeutet, dass eine Klasse Pkw die Oberklasse eines Autos darstellt. Ein Pkw kann bereits Eigenschaften haben (zum Beispiel eine Geschwindigkeit), kennt aber noch keine Definition bezüglich der Reifen, die erst durch das Auto dazukommen.

- Pakete und Module
  Pakete und Module werden in Python verwendet, um Klassen in einer Struktur abzulegen.

3.1.7 Weiterführende Informationen zu Python

In Python gibt es eine Vielzahl von Schlüsselwörtern beziehungsweise Ausdrücken, die unveränderlich feststehen. Diese können nicht für andere Zwecke (zum Beispiel als Variablennamen) verwendet werden, sondern dienen einem bestimmten Zweck und bilden damit die Grundlage der Programmierung. Eine Auflistung wird im Anhang 8.2 zur Verfügung gestellt.

Bibliotheken

Python verfügt neben den Schlüsselwörtern über eine sehr große Anzahl an Standardbibliotheken mit Zusatzfunktionen, die für die Programmierung genutzt werden können. Diese Bibliotheken sind entweder Bestandteil der Python-Umsetzung und lassen sich direkt verwenden (zum Beispiel Klassen für den Zugriff auf Dateien) oder können als Zusatzbibliotheken hinzugenommen werden (zum Beispiel Klassen für ein erweitertes Logging). Für viele Bibliotheken existieren meist ausführliche Dokumentationen. Im Normalfall ist diese Dokumentation auf Englisch. Eine Übersetzung in andere Sprachen existiert häufig nicht, die Inhalte werden auch eher über Bücher oder Tutorials vermittelt.

Programmierbefehle ausführen

Ein Befehl in Python entspricht einem Methodenaufruf oder einer mathematischen Operation. Eine Zeile stellt dabei einen Befehl dar.

Bei Python ist hierbei wichtig, dass nicht alle Parameter einer Methode übergeben werden müssen (im Gegensatz zu Java). Zusätzlich kann ein Parameter mit seinem Namen übergeben werden, sodass z. B. nur der erste und dritte Parameter übergeben werden muss. Bei einer Übergabe ohne