RunCode – Bedienungsanleitung

Stand: 23.02.2019, Verfasser: Christoph Gräßl (graessl@hotmail.de)

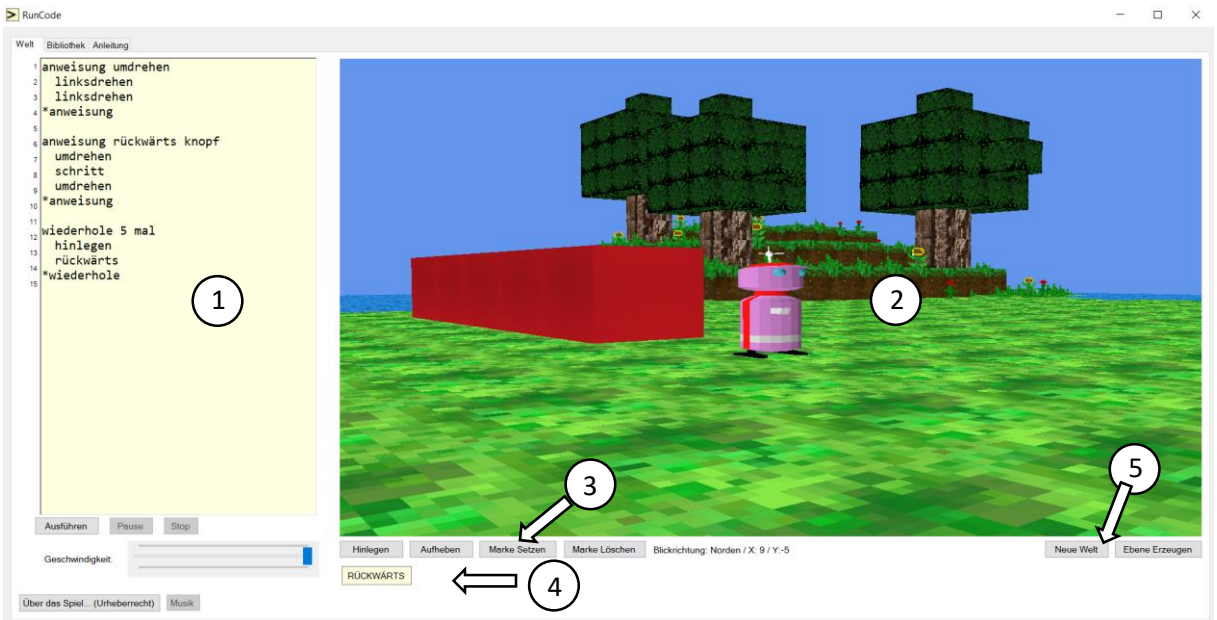
1) Allgemeines

Bei RunCode handelt es sich um eine Lernumgebung, in der Schüler einen virtuellen Roboter in einer dreidimensionalen Welt programmieren können um einfache Aufgaben zu lösen. Diese Idee stammt aus dem Buch „Karel the Robot: A Gentle Introduction to The Art of Programming“ von Richard E. Pattis (1981). Damit kann beispielsweise in der siebten Jahrgangsstufe in Informatik der Lehrplanbereich „Algorithmen“ an bayerischen Gymnasien umgesetzt werden.

Neu ist, dass die Welt nicht nur aus einem einfachen Schachbrettmuster mit begrenzter Größe besteht, sondern eine prozedurale (zufallsgesteuerte) dreidimensionale Insellandschaft ist, mit (zumindest theoretisch) unendlicher Größe. Die Minimalsprache, mit der der Roboter programmiert werden kann, orientiert sich an der, die auch in den einschlägigen Schulbüchern zu finden ist, besitzt aber noch zusätzliche Elemente.

2) Grafische Benutzeroberfläche

Zuerst kann der Schüler (gemeint sind natürlich auch unsere Schülerinnen) sich zwischen zwei verschiedenen Roboter entscheiden und mit einem Klick auf „Schulmodus“ das Programm öffnen. Der Abenteuermodus ist noch in Entwicklung und soll interessierten Schülern zusätzlich spielerische Elemente liefern, die aber für den Unterricht nicht geeignet sind.



Im linken Bereich ist der Editor (1), in dem das Programm eingegeben werden kann. Die Minisprache wird weiter unten beschrieben, sie kann aber auch vom Benutzer unter dem Punkt „Anleitung“ im Programm angesehen werden. Dort sind alle Befehle und auch Kontrollstrukturen aufgeführt.

Ein Programm wird durch einen Klick auf „Ausführen“ gestartet, kann durch einen Klick auf „Pause“ angehalten und durch Betätigung der „Stop“-Taste beendet werden. Um die Programmausführung

besser nachvollziehen zu können, wird die aktuell ausgeführte Zeile farblich hervorgehoben. Die Geschwindigkeit kann mit einem Regler eingestellt werden.

Um zu verhindern, dass der Editor zu unübersichtlich wird, können benutzerdefinierte Anweisungen in die Code-Bibliothek verschoben werden. Hierzu muss sich der Cursor sich in einer benutzerdefinierten Methode befinden und ein Rechtklick ausgeführt werden. Der entsprechende Programmcode verschwindet dann zwar aus dem Editor, steht aber nach wie vor zur Verfügung. Im Bereich „Bibliothek“ können die Anweisungen betrachtet und wieder in den Editor verschoben werden. Außerdem ist es möglich eine Bibliothek in eine Datei zu schreiben und von dort wieder einzulesen.

Prinzipiell soll der Roboter durch ein Programm kontrolliert werden. Dennoch kann er vom Benutzer auch direkt gesteuert werden. Wenn man im Welt-Fenster (2) die linke Maustaste gedrückt hält, so kann man den Blickwinkel auf den Roboter ändern. Mit dem Rad der Maus kann man die Entfernung der virtuellen Kamera zum Roboter einstellen. Durch Drücken der `W`-Taste führt der Roboter einen Schritt nach vorne aus, mit `A` und `D` dreht sich der Roboter um 90 Grad nach links beziehungsweise rechts. Damit die Tastenanschläge nicht im Editor landen, muss vorher auf das Weltfenster geklickt werden. Außerdem können die Standardbefehle „Hinlegen“, „Aufheben“, „Marke Setzen“ und „Marke Löschen“ vom Benutzer ebenfalls direkt im Bereich (3) ausgeführt werden.

Werden benutzerdefinierte Anweisungen, die mit dem Schlüsselwort „knopf“ versehen sind, programmiert, so erscheinen sie im Bereich (4) und können dann ebenfalls direkt vom Benutzer ausgeführt werden. Diese Knöpfe erscheinen aber erst, wenn nach der Programmierung „Ausführen“ angeklickt wurde.

Im Bereich (5) kann die Welt direkt manipuliert werden. Mit „Neue Welt“ wird die Welt in den Startzustand zurückversetzt. Durch Ebene-Erzeugen kann ein Bereich erstellt werden, der komplett eben, ohne Wasser und Berge, Bäume und ähnlichen Elementen ist. Die Größe kann eingestellt werden.

3) Minisprache

a) Befehle / Methoden

SCHRITT ( )
Der Roboter macht einen Schritt nach vorne. Vorne ist die Richtung, in die der Roboter blickt.

SCHRITT (*Anzahl* **)**
Der Roboter macht eine Anzahl an Schritten nach vorne. Vorne ist die Richtung, in die der Roboter blickt. Der Parameter 'Anzahl' gibt an, wie viel Schritte der Roboter machen soll.

HINLEGEN ( )
Der Roboter legt vor sich einen Ziegelstein ab. Der Roboter kann die Ziegelsteine selber erzeugen und hat somit einen unerschöpflichen Vorrat.

HINLEGEN (*Anzahl* **)**
Der Roboter legt vor sich eine Anzahl an Ziegelstein ab. Der Parameter 'Anzahl' gibt an, wie viele Ziegelsteine vor dem Roboter gestapelt werden sollen.

AUFHEBEN ( )
Der Roboter entfernt vor sich einen Ziegelstein. Die Ziegelsteine werden danach zerstört.

AUFHEBEN (*Anzahl* **)**
Der Roboter entfernt vor sich auf einmal eine Anzahl an Ziegelsteinen. Der Parameter 'Anzahl' gibt an, wie viele Ziegelsteine entfernt werden sollen.

LINKSDREHEN ( )
Der Roboter dreht sich (von sich aus gesehen) um 90 Grad nach links.

RECHTSDREHEN ( )
Der Roboter dreht sich (von sich aus gesehen) um 90 Grad nach rechts.

MARKESETZEN ( )
Auf der Stelle, wo sich der Roboter befindet (also unter seinen Füßen), wird eine gelbe Marke platziert.

MARKELÖSCHEN ( )
Auf der Stelle, wo sich der Roboter befindet (also unter seinen Füßen), wird eine gelbe Marke entfernt, falls sich dort eine befindet.

ABBAUEN (*Richtung* **)**
Der Roboter baut vor sich einen Block ab. Die Methode benötigt einen Parameter, der einen von drei Werten annehmen muss: Oben, Mitte oder Unten. Der Parameter gibt also an, ob der Block vor ihm schräg unten, mittig oder vor ihm schräg oben abgebaut werden soll. Natürlich muss der Roboter auch drauf zugreifen können. Die dabei gewonnenen Rohstoffe werden in das Inventar gelegt.

b) Kontrollstrukturen

Wiederholung mit fester Anzahl

```
Wiederhole x mal
   Anweisung1
   Anweisung2
   …
*Wiederhole
```

Die Anweisungssequenz im Wiederhole-Block wird *x*-Mal wiederholt. Dabei ist *x* eine ganze Zahl größer 0.

Wiederholung mit (Anfangs-) Bedingung

```
Wiederhole solange Bedingung
   Anweisung1
   Anweisung2
   …
*Wiederhole
```

Die Anweisungssequenz im Wiederhole-Block wird immer dann wiederholt, wenn die *Bedingung* wahr ist.

Bedingte Anweisung

```
Wenn [nicht] Bedingung dann
   Anweisung1
   Anweisung2
   …
```

```
*Wiederhole
```

Die Anweisungssequenz im Wenn-Block wird dann ausgeführt, wenn die *Bedingung* wahr ist (oder nicht wahr ist). Man kann einen Sonst-Zweig einfügen, der eine alternative Anweisungssequenz enthält.

Eigene Methoden

Anweisung *MeineMethode* [Knopf]
    Anweisung1
    Anweisung2
    …
*Anweisung

Die Anweisungssequenz innerhalb des Blocks wird aufgerufen, wenn die Anweisung *MeineMethode* im Programmcode erscheint. Durch das optionale Wort Knopf wird nach Ausführung des Programms ein zusätzlicher Knopf auf das Fenster gelegt, der es ermöglicht MeineMethode durch einen Klick auszuführen.

c) Bedingungen

ISTZIEGEL
Wenn direkt vor dem Roboter ein Ziegel liegt, dann ist die Bedingung 'wahr'.

NICHTISTZIEGEL
Wenn direkt vor dem Roboter ein Ziegel liegt, dann ist die Bedingung 'falsch'.

BETRETBAR
Ergibt 'wahr', wenn der Roboter in der Lage ist, erfolgreich einen Schritt nach vorne durchzuführen.

ISTMARKE
Wenn der Roboter auf einer Marke steht, ist die Bedingung 'wahr'

NICHTISTMARKE
Wenn der Roboter auf einer Marke steht, ist die Bedingung 'falsch'

ISTNORDEN
Die Bedingung ist 'wahr', wenn der Roboter in Richtung Norden blickt.

ISTSÜDEN
Die Bedingung ist 'wahr', wenn der Roboter in Richtung Süden blickt.

ISTWESTEN
Die Bedingung ist 'wahr', wenn der Roboter in Richtung Westen blickt.

ISTOSTEN
Die Bedingung ist 'wahr', wenn der Roboter in Richtung Osten blickt.

ISTWAND
Ergibt 'falsch', wenn der Roboter in der Lage ist, erfolgreich einen Schritt nach vorne durchzuführen.

NICHTISTWAND
Ergibt 'wahr', wenn der Roboter in der Lage ist, erfolgreich einen Schritt nach vorne durchzuführen.

4) Systemvoraussetzung und Installation

RunCode arbeitet auf Windowsrechnern und setzt eine Installation von .NET 4.5 voraus. Auf Windows 10 ist bereits .NET installiert. Bei Windows 7 kann .NET 4.5 nachinstalliert werden. Auf anderen Betriebssystemen mit .NET-Installation (beispielsweise Mono) ist die Ausführung nicht ohne weiteres möglich, da die Grafik auf Microsofts Direct-X-Framework basiert.

Die Software benötigt keine spezielle Setup-Routine, da gerade in Computerräumen die Installation neuer Software sich als aufwändig zeigt. Es genügt, den Ordner in ein beliebiges Verzeichnis zu entpacken. Von dort aus kann man die Programmdatei „RunCode.exe“ einfach starten.

Leider besitzt RunCode keine digitale Signatur. Da die Software frei verfügbar ist, waren die Kosten für solche eine Signatur zu hoch. Eventuell könnte daher ein Virenscanner warnen, dass die Software nicht vertrauenswürdig erscheint.

5) Lizenzen

RunCode kann kostenlos von der Internetseite http://klassenkarte.de bezogen und kostenlos für Bildungszwecke genutzt werden. Die Software darf ohne schriftliche Erlaubnis des Autors nicht verändert werden.

RunCode setzt auf der Monogame-Plattform (http://www.monogame.net/) auf, die im Programmverzeichnis mitgeliefert wird. Monogame steht unter der „Microsoft Public License“.

```
Microsoft Public License (Ms-PL)
MonoGame - Copyright © 2009-2019 The MonoGame Team

All rights reserved.

This license governs use of the accompanying software. If you use the software,
you accept this license. If you do not accept the license, do not use the
software.

1. Definitions

The terms "reproduce," "reproduction," "derivative works," and "distribution"
have the same meaning here as under U.S. copyright law.

A "contribution" is the original software, or any additions or changes to the
software.

A "contributor" is any person that distributes its contribution under this
license.

"Licensed patents" are a contributor's patent claims that read directly on its
contribution.

2. Grant of Rights

(A) Copyright Grant- Subject to the terms of this license, including the
license conditions and limitations in section 3, each contributor grants you a
non-exclusive, worldwide, royalty-free copyright license to reproduce its
```

contribution, prepare derivative works of its contribution, and distribute its contribution or any derivative works that you create.

(B) Patent Grant- Subject to the terms of this license, including the license conditions and limitations in section 3, each contributor grants you a non-exclusive, worldwide, royalty-free license under its licensed patents to make, have made, use, sell, offer for sale, import, and/or otherwise dispose of its contribution in the software or derivative works of the contribution in the software.

3. Conditions and Limitations

(A) No Trademark License- This license does not grant you rights to use any contributors' name, logo, or trademarks.

(B) If you bring a patent claim against any contributor over patents that you claim are infringed by the software, your patent license from such contributor to the software ends automatically.

(C) If you distribute any portion of the software, you must retain all copyright, patent, trademark, and attribution notices that are present in the software.

(D) If you distribute any portion of the software in source code form, you may do so only under this license by including a complete copy of this license with your distribution. If you distribute any portion of the software in compiled or object code form, you may only do so under a license that complies with this license.

(E) The software is licensed "as-is." You bear the risk of using it. The contributors give no express warranties, guarantees or conditions. You may have additional consumer rights under your local laws which this license cannot change. To the extent permitted under your local laws, the contributors exclude the implied warranties of merchantability, fitness for a particular purpose and non-infringement.

The MIT License (MIT)
Portions Copyright © The Mono.Xna Team

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.