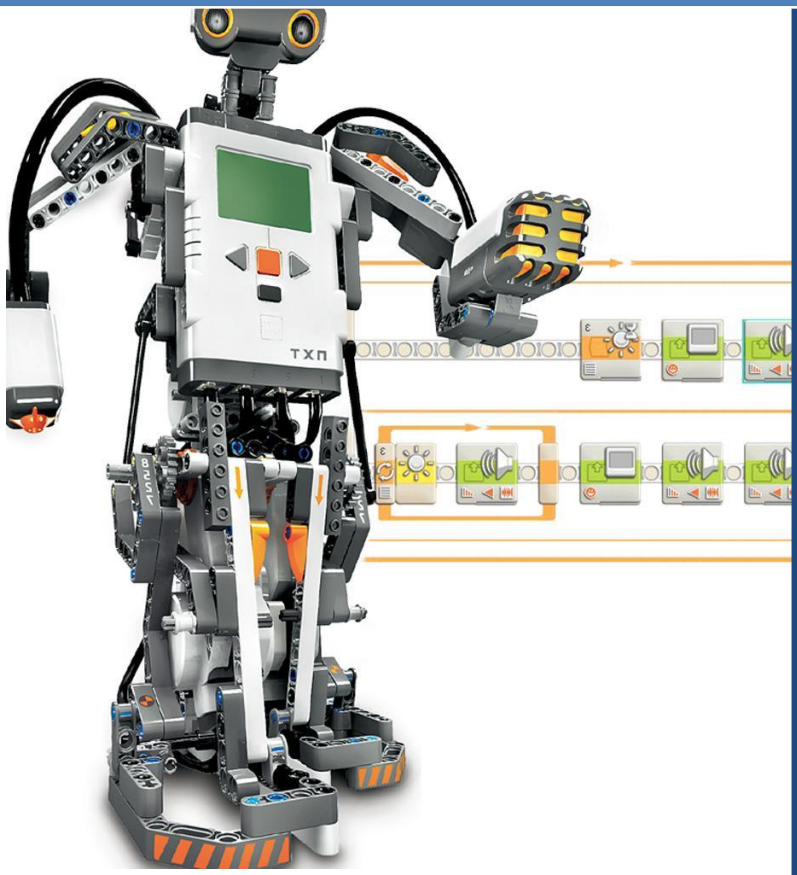
ReferenceGuide

Einführungspraktikum

Roboterprogrammierung



Fachhochschule Südwestfalen
University of Applied Sciences

Autor:
Fachhochschule Südwestfalen
Fachbereich Elektrische Energietechnik
Campus Soest

Hinweise zur Benutzung

Dieses Dokument soll die Arbeit während des Praktikums erleichtern, hierzu steht eine systematische Sammlung der wichtigsten Befehle aus RobotC zur Verfügung. Jeder Befehl wird dabei durch ein kurzes Beispiel illustriert. Außerdem wird jeweils angegeben in welchem Praktikum und innerhalb welcher Aufgabe die jeweilige Funktion zum ersten Mal benutzt wurde. Die Sortierung der einzelnen Befehle orientiert sich dabei in den meisten Fällen an der Aufteilung in der RobotC-Bibliothek. Die meisten Teile dieses ReferenceGuides basieren auf den Dokumentationen des National Robotics Engineering Center an der Carnegie Mellon University (siehe www.robotc.net).

A Generelle C-Syntax

Die folgenden Unterpunkte enthalten Übersichten zu den wichtigsten Elementen der generellen Syntax von C. Hierbei werden natürlich in dieser Referenz nur die Elemente angegeben, welche auch zur Bearbeitung der Praktika benötigt werden.

A1 Datentypen und Deklarationen von Variablen

A1.1 Deklaration von Variablen → Aufgabe 2.2

```
1  int distance;                    // Deklaration einer Ganzzahlvariablen
   Syntax: Datentyp Variablenname;
2  float distance = 0;              // Gleitkommavariable + Zuweisung Startwert
   Syntax: Datentyp Variablenname = Startwert;
```

A1.2 Übersicht der Datentypen → Aufgabe 2.2

Datentyp	Beschreibung	Beispiele	Code
Integer (Ganzzahl)	Positive und negative Ganze Zahlen und Null	-35, -1, 0, 33, 100, 345	`int`
Float (Fließkommazahl)	Numerische Werte mit Dezimalpunkten	-0.123; 0.56; 3.0; 1000.07	`float`
Boolesche (wahr / falsch)	Zur Auswertung von logischen Ausdrücken	`true` / `false`	`bool`

A1.3 Felder → Aufgabe 4.1

```
1  int feld[5];                  // Initialisieren Feld mit 5 Integer Variablen
2  int feld[5] = {1,2,3,4,5};    // Initialisieren Feld mit gleichzeitiger
3                                // Definition der Elemente
4  A = feld[4]                   // Zugriff auf den Wert in der 5. Zelle des
                                 Feldes
5  B = feld[0]                   // Index des ersten Elementes des Feldes: 0 !!!
```

A2 Operatoren

A2.1 Arithmetische Operationen → Aufgabe 3.2

```
1  A++;          // Erhöhe den Wert von A um eins
2  B--;          // Erniedrige den Wert von B um eins
3  C=A;          // Weise C den Wert von A zu
4  A+B;          // Addition von A und B
5  D-C;          // Subtraktion von D und C
6  B/D;          // Dividiere B durch D
7  C*A;          // Multipliziere C und A
```

A2.2 Vergleichsoperationen → Aufgabe 2.3

```
1  A==B;         // Ist A gleich B?
2  A!=B;         // Ist A ungleich B?
3  C>A;          // Ist C größer als A?
4  A<B;          // Ist A kleiner als B?
5  D<=C;         // Ist D kleiner oder gleich C?
6  B>=D;         // Ist B größer oder gleich D?
```

A2.3 Logische Verknüpfungen → Aufgabe 3.1

```
1  A&&B;         // True wenn A und B true sind
2  A||B;         // True wenn A oder B oder beide true sind
3  !A;           // True wenn A false ist
```

A3 Programmflusssteuerung (C_Constructs)

A3.1 while-Schleife → Aufgabe 2.3

```
1  while ( Bedingung )
2  {
3        Anweisungen;
4  }
5
6  // Beispiel:
7  while ( a < 10 )              // Solange a kleiner ist als 10…
8  {
9        a++;                    // … erhöhe a um eins
10 }
```

A3.2 for-Schleife → Aufgabe 4.3.1

```
1  for ( Initialisierung; Bedingung; Aktion )     // Initialisierung: Zuweisung
2  {                                              // eines Startwertes für die
3        Anweisungen;                             // Zählvariable
4  }                                              // Bedingung: Verbleibe
5                                                 // solange in der Schleife bis
6                                                 // die Bedingung nicht mehr
7                                                 // erfüllt ist
8                                                 // Aktion: Veränderung der
9  // Beispiel:
10 int a=1;                                       // Zählvariable nach Iteration
11 int i;                                         // Deklaration Zählvariable
12 for ( i=0; i<9; i++ )                          // Erhöhe i solange i kleiner
13 {                                              // als neun ist um eins und
14       a=a+i;                                   // berechne a = a + i
15 }                                              // 1,2,4,7,11,16,22,29,37
16                                                // Ergebnis: 37
```

A3.3 if-Struktur → Aufgabe 4.3.1

```
1  if ( Bedingung )          // Wenn Bedingung wahr ist, dann…
2  {
3        Anweisungen 1;      // …führe Anweisungen 1 aus,
4  }
5  else                      // sonst…
6  {
7        Anweisungen 2;      // … führe Anweisungen 2 aus.
8  }
9
10 // Beispiel:
11 if ( a < 10 )             // Wenn a kleiner ist als 10, dann…
12 {
13       b = 1;              // … weise b den Wert 1 zu,
14 }
15 else                      // sonst…
16 {
17       b = 2;              // … weise b den Wert 2 zu.
18 }
```

B Buttons des NXT-Bricks

B1 Abfragen ob Taste auf NXT-Brick gedrückt → Aufgabe 2.2

```
1  if (nNxtButtonPressed == 1)                    // Variable nNxtButtonPressed
2  {                                              // enthält einen Wert der die
3     // Pfeil nach rechts wurde gedrückt         // gedrückte Taste
4  }                                              // repräsentiert
```

Button	Wert von `nNxtButtonPressed`
Kein Button gedrückt	-1
	3
	2
	1

C Display des NXT-Bricks

C 1	Texte auf dem LCD ausgeben	→ Aufgabe 2.1

```
1  nxtDisplayTextLine(nLineNumber, sString);    // In der Zeile nLineNumber wird
2                                               // der Text in sString angezeigt
3
4  // Beispiel:
5  nxtDisplayTextLine(1, "Hallo Leute !");      // In Zeile 1 wird der Text „Hallo
6                                               // Leute !“ ausgegeben
```

C 2	Texte und Werte von Variablen auf dem LCD ausgeben	→ Aufgabe 2.2

```
1  nxtDisplayTextLine(nLineNumber, sString, parm1);  // In der Zeile nLineNumber
2                                                    // wird der Text in sString
3                                                    // angezeigt, welcher einen
4                                                    // Platzhalter für den Wert
5                                                    // in parm1 enthält
6
7  int wert = 5;                                     // Anlegen Variable
8  nxtDisplayTextLine(1, "Nummer %i lebt !", wert);  // Darstellung auf LCD
9
```

D. Motoren

D1	Zuweisung eines Geschwindigkeitswertes	→ Aufgabe 2.4

```
1  motor[motorA] = 100;     // Weise dem Motor an Port A einen Geschwindigkeits-
2                           // wert von 100 zu. Motor dreht linksrum.
3
4  motor[motorC] = -50;     // Weise dem Motor an Port C einen Geschwindigkeits-
5                           // wert von 50 zu. Motor dreht rechtsrum.
```

E. Sensoren

F1	Auslesen eines Sensorwertes	→ Aufgabe 2.2

```
1  int distance;                        // Definition einer Variablen zur
2                                       // Speicherung des Sensorwertes
3
4  distance = SensorValue[S4];          // Auslesen des Sensors an Port 4
5                                       // Sensor muss vorher in RobotC
6                                       // konfiguriert werden !!!
```

F. Sounds

F1	Abspielen eines vordefinierten Soundfiles	→ Aufgabe 2.2

```
1  PlaySound(sound);    // Spielt das Soundfile, welches durch sound definiert
2                       // wird auf dem NXT-Brick ab
3                       // Es stehen folgende vordefinierte Sounds zur
4                       // Verfügung:
5                       // - soundBeepBeep
6                       // - soundBlip
7                       // - soundDownwardTones
8                       // - soundException
9                       // - soundFastUpwardTones
10                      // - soundLast
11                      // - soundLowBuzz
12                      // - soundLowBuzzShort
13                      // - soundShortBlip
14                      // - soundUpwardTones
```

G. Timing

G1 Verzögerungen → Aufgabe 2.1

```
1  wait10Msec(nTenMSec);        // Verzögerung von nTenMSec*10ms
2  wait1Msec(nMSec);            // Verzögerung von nMSec*1ms
3
4  wait10Msec(10);              // Verzögerung von 10*10ms = 100ms
5  wait1Msec(1000);             // Verzögerung von 1000*1ms = 1000ms
```

Ansprechpartner

Bei weiteren Fragen zur Organisation des Einführungspraktikums wenden Sie sich bitte jederzeit an:

Antonius Schmidt, M.Sc

Raum 04.105

Tel: +49 (0)29 21 / 378-**452**

E-Mail: schmidt.antonius@fh-swf.de