

# Datenstrukturen

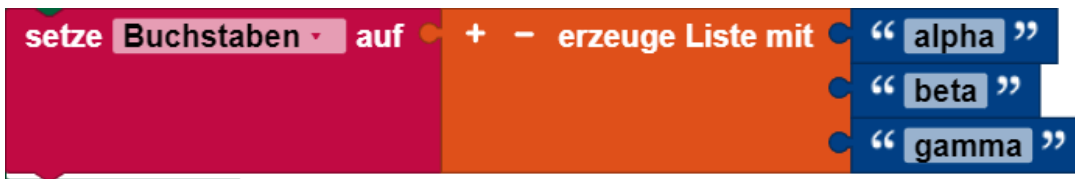
## Listen

Wie in der Alltagssprache ist auch in ROBO Pro Coding eine Liste eine geordnete Sammlung von Elementen, wie z. B. eine "To-do"-Liste oder eine Einkaufsliste. Elemente in einer Liste können von beliebigem Typ sein, und derselbe Wert kann mehrmals in einer Liste erscheinen.

## Erstellen einer Liste

### erzeuge Liste mit

Mit dem Block "erzeuge Liste mit" kann man die Anfangswerte in einer neuen Liste angeben. In diesem Beispiel wird eine Liste von Wörtern erstellt und in einer Variablen namens "Buchstaben" abgelegt:

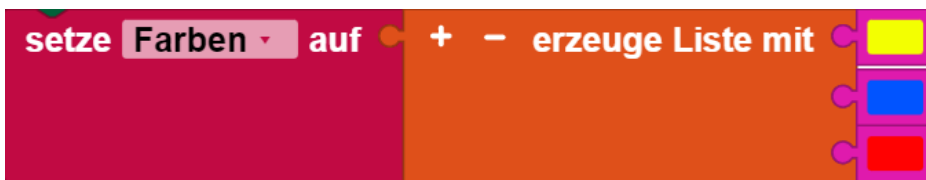


Wir bezeichnen diese Liste als ["alpha", "beta", "gamma"].

Dies zeigt die Erstellung einer Liste von "Zahlen":



So wird eine Liste von "Farben" erstellt:



Es ist weniger üblich, aber möglich, eine Liste mit Werten unterschiedlichen Typs zu erstellen:



## Anzahl der Eingänge ändern

Um die Anzahl der Eingänge zu ändern, klicke beziehungsweise tippe auf das (+) oder (-) Symbol. Dadurch werden neue Eingänge hinzugefügt oder wieder entfernt.

## Liste mit Element erstellen

Mit dem Block "erstelle Liste mit Element" kannst du eine Liste erstellen, die die angegebene Anzahl von Kopien eines Elements enthält. Die folgenden Blöcke setzen zum Beispiel die Variable "Wörter" auf die Liste ["sehr", "sehr", "sehr"].



## Prüfen der Länge einer Liste

### ist leer

Der Wert eines "ist leer"-Blocks ist "wahr", wenn seine Eingabe die leere Liste ist, und "falsch", wenn es irgendwas anderes ist. Ist diese Eingabe "wahr"? Der Wert des folgenden Blocks wäre "falsch", weil die Variable Farben nicht leer ist: Sie hat drei Elemente.



Beachte die Ähnlichkeit mit dem "ist leer"-Block für Text.

### Länge von

Der Wert des "Länge von"-Blocks ist die Anzahl der Elemente, die sich in der als Eingabe verwendeten Liste, befinden. Der Wert des folgenden Blocks wäre z. B. 3, da "Farbe" drei Elemente hat:



Beachte, dass der "Länge von"-Block angibt, wie viele Elemente in der Liste enthalten sind, und nicht, wie viele verschiedene Elemente in ihr enthalten sind. Zum Beispiel hat das Folgende den Wert 3, obwohl "Wörter" aus drei Kopien desselben Textes besteht:



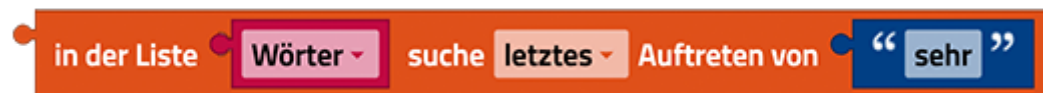
Beachte die Ähnlichkeit mit dem Block "Länge von" für Text.

# Suchen von Elementen in einer Liste

Diese Blöcke finden die Position eines Elements in einer Liste. Das folgende Beispiel hat den Wert 1, weil das erste Auftreten von "sehr" am Anfang der Wortliste steht (["sehr", "sehr", "sehr"]).



Das Ergebnis des Folgenden ist 3, weil das letzte Auftreten von "sehr" in "Wörter" an Position 3 ist.



Wenn das Element nirgendwo in der Liste vorkommt, ist das Ergebnis der Wert 0, wie in diesem Beispiel:



Diese Blöcke verhalten sich analog zu den Blöcken für das Finden von Buchstaben im Text.

# Abrufen von Elementen aus einer Liste

## Abrufen eines einzelnen Elements

Erinnere dich an die Definition der Liste "Farben":



Der folgende Block erhält die Farbe Blau, weil es das zweite Element in der Liste ist (von links beginnend gezählt):



Dieser erhält Grün, weil es das zweite Element ist (vom rechten Ende aus gezählt):



Dieser erhält das erste Element, Rot:



Dies erhält das letzte Element, Gelb:



Dies wählt zufällig ein Element aus der Liste aus, wobei mit gleicher Wahrscheinlichkeit eines der Elemente Rot, Blau, Grün oder Gelb zurückgegeben wird.

## Abrufen und Entfernen eines Elements

Mit dem Dropdown-Menü wird der Block "aus Liste ... abrufen" in den Block "aus Liste ... abrufen und entfernen" geändert, der die gleiche Ausgabe liefert, aber auch die Liste verändert:



Dieses Beispiel setzt die Variable "erster Buchstabe" auf "alpha" und lässt die restlichen Buchstaben (["beta", "gamma"]) in der Liste.



## Entfernen eines Eintrags

Wenn du im Dropdown-Menü "entfernen" wählst, verschwindet die Nase links vom Block:



Damit wird das erste Element aus "Buchstaben" entfernt.

## Eine Subliste abrufen

Der Block "aus Liste ... Subliste abrufen" ähnelt dem Block in "aus Liste ... abrufen" mit dem Unterschied, dass er eine Subliste extrahiert und nicht ein einzelnes Element. Es gibt mehrere Optionen, den Anfang und das Ende der Subliste anzugeben:



- ✓ nimm Teilliste ab
- nimm Teilliste ab von hinten
- nimm Teilliste ab erstes



- ✓ bis
- bis von hinten
- bis letztes

In diesem Beispiel wird eine neue Liste "erster Buchstabe" erstellt. Diese neue Liste hat zwei Elemente: ["alpha", "beta"].

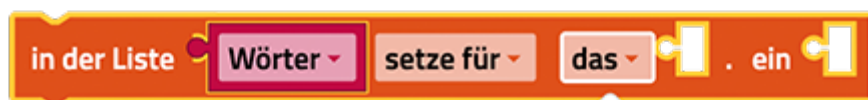


Beachte, dass dieser Block die ursprüngliche Liste nicht verändert.

## Hinzufügen von Elementen an eine Liste

### Elemente in einer Liste ersetzen

Der Block "in Liste ... ersetze" ersetzt das Element an einer bestimmten Stelle einer Liste durch ein anderes Element.



- ✓ das
- von hinten das
- Erste
- Letzte
- Zufällig

Die Bedeutung der einzelnen Dropdown-Optionen findest du im vorherigen Abschnitt.

Das folgende Beispiel bewirkt zwei Dinge:

1. Die Liste "Wörter" wird mit 3 Elementen erstellt: ["sehr", "sehr", "sehr"].
2. Das dritte Element in der Liste wird durch "gut" ersetzt. Der neue Wert von "Wörter" ist ["sehr", "sehr", "gut"]



## Elemente an einer bestimmten Stelle in eine Liste einfügen

Der "in Liste ... einfügen bei"-Block wird über das Dropdown-Menü des "in Liste ... ersetze"-Blocks aufgerufen:



Er fügt ein neues Element an der angegebenen Stelle in die Liste ein, und zwar vor dem Element, das sich zuvor an dieser Stelle befand. Das folgende Beispiel (das auf einem früheren Beispiel aufbaut) tut drei Dinge:

1. Die Liste "Wörter" wird mit 3 Elementen erstellt: ["sehr", "sehr", "sehr"].
2. Das dritte Element in der Liste wird durch "gut" ersetzt. Der neue Wert von "Wörter" ist somit ["sehr", "sehr", "gut"].
3. Das Wort "Sei" wird am Anfang der Liste eingefügt. Der endgültige Wert von "Wörter" ist somit ["Sei", "sehr", "sehr", "gut"].



## Zeichenketten aufteilen und Listen zusammenfügen

### Eine Liste aus einem Text erstellen

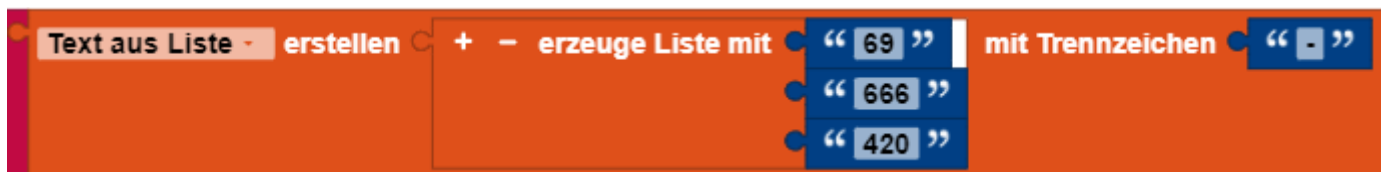
Der Baustein "erstelle Liste aus Text" zerlegt den angegebenen Text mithilfe eines Begrenzungszeichens in Teile:



Im obigen Beispiel wird eine neue Liste zurückgegeben, die drei Textstücke enthält: "311", "555" und "2368".

## Ein Text aus einer Liste erstellen

Der Baustein "erstelle Text aus Liste" fügt eine Liste mithilfe eines Trennzeichens zu einem einzigen Text zusammen:



Im obigen Beispiel wird ein neuer Text mit dem Wert zurückgegeben: "311-555-2368".

## Eine Liste sortieren

Der "Sortieren"-Block ist ein flexibles Werkzeug, das es ermöglicht, Listen anhand verschiedener Kriterien zu ordnen, die basierend auf dem Typ der Elemente in der Liste ausgewählt werden.

- Typ der Elemente:
  - Numerisch: Für Listen mit Zahlen. Sortiert die Elemente basierend auf ihrem numerischen Wert.
  - Alphabetisch: Für Listen mit Text. Sortiert die Elemente nach dem Alphabet. Die Sortierung berücksichtigt auch die Groß- und Kleinschreibung,
  - Alphabetisch, Großschreibung ignorieren: Sortiert Texte alphabetisch, wobei Groß- und Kleinschreibung nicht unterschieden wird.
- Sortierreihenfolge:
  - Aufsteigend: Sortiert die Liste von kleinsten zu größten Werten (oder von A bis Z).
  - Absteigend: Sortiert die Liste von größten zu kleinsten Werten (oder von Z bis A).

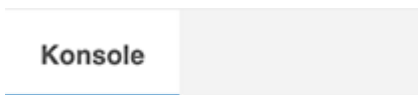
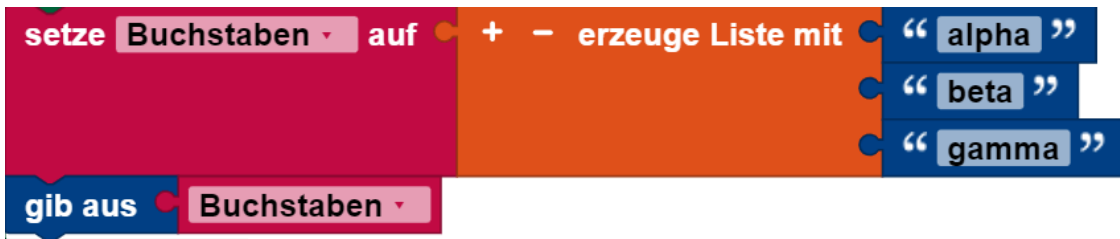
Im gezeigten Beispiel wird eine Liste mit den numerischen Werten 20, 10 und 100 erstellt und einer Variablen namens "liste" zugewiesen. Anschließend wird der "Sortieren"-Block verwendet, um die Daten numerisch in aufsteigender Reihenfolge zu ordnen. Die sortierten Daten werden dann einer neuen Variablen namens "sortiereListe" zugewiesen. Nach der Sortierung enthält "sortiereListe" die Werte in folgender Reihenfolge: 10, 20, 100. Die Reihenfolge der Liste "liste" ist unverändert.



## Verwandte Blöcke

### Drucken einer Liste

Der "drucken"-Baustein in der Kategorie Text kann Listen ausgeben. Das Ergebnis des folgenden Programms ist die abgebildete Konsolenausgabe:



Programm startet...

['apha', 'beta ', 'gamma']

Programm beendet.

## Etwas für jedes Element in einer Liste durchführen

Der "für-jeden"-Block in der Kategorie Steuerung führt eine Operation für jedes Element in einer Liste aus. Dieser Block druckt zum Beispiel jedes Element in der Liste einzeln aus:



Dadurch werden die Elemente nicht aus der ursprünglichen Liste entfernt.

# Map

Eine Map, auch bekannt als Dictionary oder Assoziatives Array, ist eine Sammlung von Schlüssel-Wert-Paaren, bei der jeder eindeutige Schlüssel direkt auf einen Wert verweist.

## Alle Schlüssel einer Map zurückgeben

Der Block "in der Map ... gib alle Schlüssel" kann dazu verwendet werden, alle Schlüssel aus einer Map zurückzugeben.

In dem Beispiel holt der Block aus der Map "map" alle Schlüssel und gibt jeden in einer Schleife aus. Dies könnte genutzt werden, um zu sehen, welche Schlüssel überhaupt in der Map gespeichert sind:

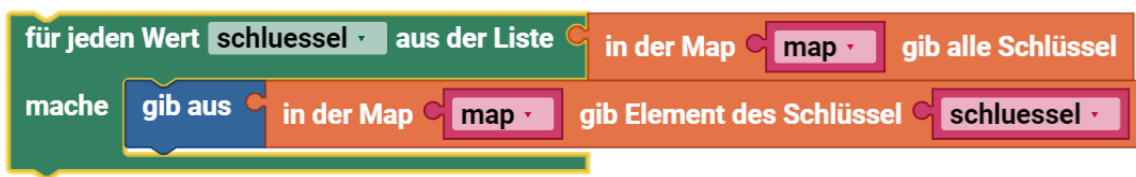




## Elements zurückgeben

Der Block "in der Map ... gib Element des Schlüssel ..." kann dazu verwendet werden, um den Wert eines bestimmten Schlüssels aus der Map zurückzugeben.

In dem Beispiel holt der Block, nachdem alle Schlüssel der Map "map" abgerufen wurden, für jeden Schlüssel den entsprechenden Wert. Dieser wird anschließend mit dem "gib aus"-Block ausgegeben. Dies demonstriert, wie man sequenziell durch eine Map iteriert und sowohl Schlüssel als auch Werte verarbeitet.



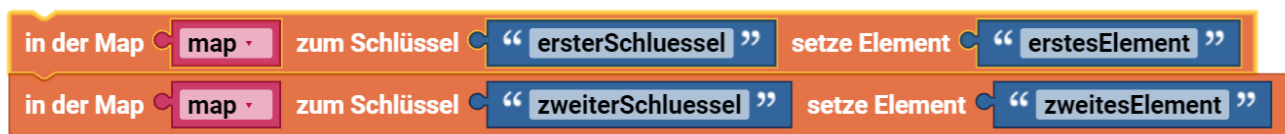
## Element zu einer Map setzen

Der Block "in der Map ... zum Schlüssel ... setze Element ..." fügt einer Map ein neues Schlüssel-Wert-Paar hinzu oder aktualisiert den Wert eines bestehenden Schlüssels. Dies ist grundlegend für das Hinzufügen oder Modifizieren von Daten in einer Map.

Im Beispiel werden zwei Schlüssel-Wert-Paare in die Map "map" eingefügt:

- "ersterSchlüssel" erhält den Zeichenkettenwert "erstesElement".
- "zweiterSchlüssel" erhält den Zeichenkettenwert "zweitesElement".

Falls diese Schlüssel bereits in der Map existieren, werden ihre Werte mit den neuen Einträgen überschrieben. Andernfalls werden die Schlüssel und ihre zugehörigen Werte neu hinzugefügt. Es ist wichtig zu beachten, dass die Werte in einer Map nicht auf Zeichenketten beschränkt sind; sie können vielfältige Datentypen umfassen, einschließlich, aber nicht beschränkt auf, Zahlen, Listen oder sogar andere Maps.



# JSON

JSON (JavaScript Object Notation) ist ein textbasiertes Datenformat, das für den Datenaustausch im Web genutzt wird. Es ist leicht lesbar und einfach zu handhaben. JSON strukturiert Daten als Sammlung von Schlüssel-Wert-Paaren, ähnlich wie Dictionaries oder Maps.

## Konvertierung JSON zu Map

Der "JSON zu Map"-Block nimmt einen JSON-String als Eingabe und konvertiert diesen in eine Map. Jedes Schlüssel-Wert-Paar im JSON-String wird zu einem Eintrag in der Map.

Im Beispiel wird der JSON-String `{"name": "Max", "age": 25}` in eine Map umgewandelt. Diese Map enthält dann die Schlüssel "name" mit dem Wert "Max" und "age" mit dem Wert "25".



## Konvertierung Map zu JSON

Der "Map zu JSON"-Block ermöglicht die Umwandlung von Map-Datenstrukturen in einen JSON-String. Dieser Vorgang ist besonders nützlich, wenn die strukturierten Daten einer Map für eine Datenübertragung oder als Konfigurationsdatei in einem standardisierten Format benötigt werden.



---

Revision #23

Created 21 February 2022 15:50:35 by Admin

Updated 26 September 2024 07:22:26 by Alexander Steiger