



# IRTlib Documentation

**Software for the administration of computer-based assessments**

Ulf Kroehne

Last Change: 3rd December, 2023

# Table of contents

<b>1 Einführung</b>	<b>4</b>
<b>2 Download &amp; Installation</b>	<b>4</b>
2.1 Offline (Windows) . . . . .	4
2.1.1 Studienvorbereitung mit Offline Editor . . . . .	5
2.1.2 Studiendurchführung mit Offline Player . . . . .	5
2.2 Online (Docker) . . . . .	6
<b>3 Übersicht: Schritte zur Verwendung von <i>IRTlib Editor</i> für die Studienvorbereitung</b>	<b>7</b>
3.1 Overview: Steps to use <i>IRTlib Editor</i> for Study Preparation . . . . .	7
<b>4 Vorbereitung: Studien</b>	<b>9</b>
<b>5 Vorbereitung: Studien</b>	<b>10</b>
5.1 Studienverwaltung . . . . .	10
5.1.1 Studien Anlegen . . . . .	10
5.1.2 Weitere Funktionen und Hinweise . . . . .	12
5.2 Grundlegende Konfigurationen . . . . .	13
5.2.1 Einstellungen zur Studie . . . . .	14
5.3 Zugang zu Studien (Login) . . . . .	15
5.3.1 Konfiguration der Anmeldung . . . . .	16
5.4 Anzeige von Assessment-Inhalten . . . . .	19
5.4.1 Anzeigeeinstellungen . . . . .	19
5.4.2 Skalierung und Ausrichtung . . . . .	22
5.5 Menü für Testadministratoren . . . . .	24
5.5.1 Konzept eines Testleitermenüs (Menü für Testadministratoren) . . . . .	24
5.6 Abschluss von Erhebungen . . . . .	27
5.6.1 Session und Session-Ende . . . . .	27
<b>6 Vorbereitung: Erhebungsteile</b>	<b>29</b>
<b>7 Vorbereitung: Erhebungsteile</b>	<b>30</b>
7.1 Erhebungsteilverwaltung . . . . .	30
7.1.1 Erhebungsteil Anlegen . . . . .	30
7.1.2 Grundkonfiguration für Erhebungsteile . . . . .	33

7.2	Assessmentinhalte (Items) Einfügen . . . . .	34
7.2.1	Items Konfigurieren . . . . .	35
7.3	Bearbeitungszeit . . . . .	37
7.3.1	Zeitbegrenzung Definieren . . . . .	37
7.3.2	Items nach einer Zeitebgrenzung . . . . .	39
7.3.3	Items vor einer Zeitebgrenzung . . . . .	40
7.4	Variablen . . . . .	41
7.5	Codebook . . . . .	41
7.6	ItemPool . . . . .	41
7.7	Routing innerhalb von Erhebungsteilen . . . . .	41
7.7.1	Zusammenfassung zu Routing innerhalb von Erhebungsteilen . . . . .	42
7.7.2	Verwendung von <i>Blockly</i> zur Ablaufsteuerung . . . . .	49
7.7.3	Fortgeschrittene <i>Blockly</i> -Verwendung . . . . .	54
7.7.4	Kommentieren von <i>Blockly</i> -Code . . . . .	79
7.7.5	Darstellung von <i>Blockly</i> -Code . . . . .	80
7.8	Routing zwischen Erhebungsteilen . . . . .	83
7.8.1	Zusammenfassung zu Routing zwischen Erhebungsteilen . . . . .	84

# 1 Einführung

(No translation available yet.)

## 2 Download & Installation

Die *IRTlib*-Software wird für die Offline-Nutzung (Windows) und für die Online-Nutzung (Docker Container) bereitgestellt.

### 2.1 Offline (Windows)

Die *IRTlib*-Software (*IRTlib Editor* und *IRTlib Player*) für die Offline-Nutzung kann aus dem Abschnitt [Releases] des Repository <https://github.com/DIPFtba/IRTlibDeploymentSoftware> bezogen und heruntergeladen werden.

Im Abschnitt [Releases](#) stehen die beiden Archive `TestApp.Editor.Desktop.zip` und `TestApp.Player.Desktop.zip` zum Download bereit.

#### Note

Beachten Sie, dass der aktuellste Build im Abschnitt [Preview](#) des *Release*-Abschnitts des [repository](#) zu finden ist.

#### Warning

Die automatisch erstellten Vorschauversionen des *IRTlib Editors* und *IRTlib Players* sind nicht signiert. Eine Warnmeldung des Betriebssystems muss akzeptiert werden, bevor die Programme ausgeführt werden können.

## 2.1.1 Studienvorbereitung mit Offline Editor

Der *IRTlib Editor* für die Offline-Nutzung wird als ZIP-Archiv (z.B. `TestApp.Editor.Desktop.zip`) bereitgestellt, das entpackt werden muss. Nach dem Entpacken des Editors kann die Anwendung `TestApp.Editor.Desktop.exe` auf einem Windows-Gerät gestartet werden.

In den Abschnitten [Vorbereitung > Übersicht](#), [Vorbereitung > Studien](#) und [Vorbereitung > Erhebungsteile](#) ist dokumentiert, wie man mit Hilfe von *CBA ItemBuilder*-Items Datenerhebungen vorbereitet und konfiguriert.

## 2.1.2 Studiendurchführung mit Offline Player

Der *IRTlib Player* ist auch als Windows-Anwendung für die Offline-Nutzung verfügbar und wird als ZIP-Archiv (z.B. `TestApp.Player.Desktop.zip`) bereitgestellt. Nach dem Entpacken des *IRTlib Player* ist eine veröffentlichte Studienkonfiguration erforderlich, die zur Datenerhebung verwendet werden soll.

Nach dem Hinzufügen der als Studienkonfiguration bereitgestellten Inhalte einer veröffentlichten Studie kann die ausführbare Datei `TestApp.Player.Desktop.exe` gestartet werden (entweder mit oder ohne Startparameter).

- **Kiosk Mode:** Der *IRTlib Player* kann über die ausführbare Datei `TestApp.Player.Desktop.exe` auf dem Computer, auf dem er lokal ausgeführt wird, direkt zur Datenerhebung verwendet werden. Die *Studie* kann dazu so konfiguriert werden, dass es in einem *Kiosk Mode* auf einem Bildschirm angezeigt wird und nur über den *Task Manager* oder das *Testleitermenü* beendet werden kann (siehe *Vollbildmodus* im Abschnitt [Konfiguration zur Anzeige](#)).
- **Lokaler Server:** Der *IRTlib Player* kann aber auch als lokaler Server ausgeführt werden. Nach dem Start des Programms `TestApp.Player.Server.exe` kann eine konfigurierte *Studie* auch über *Webbrowser* oder andere Browser mit *Kiosk Mode* ausgeliefert werden (z.B. den [Safe Exam Browser](#)). Mit dieser Konfiguration lassen sich Datenerhebungen bspw. in Schulen ohne Internetverbindung aber mit einem als *Bring-in-Server* fungierenden Notebook durchführen.

In den Abschnitten [Datenerhebung > Übersicht](#), [Datenerhebung > Veröffentlichen & Export](#) und [Datenerhebung > Integration & Auslieferung](#) ist dokumentiert, wie man mit Hilfe des *IRTlib Players* in den unterschiedlichen Konstellationen Datenerhebungen durchführen kann.

## 2.2 Online (Docker)

Die *IRTlib*-Software (*IRTlib Editor* und *IRTlib Player*) für die Online-Nutzung kann als *Docker*-Container bezogen werden. Ein Beispiel ist unter <https://github.com/DIPFtba/IRTlibDeploymentSoftware> zu finden.

Um den Docker-Container zu verwenden, wird empfohlen, das Repository auf dem Zielgerät auszuchecken (erfordert git) und den Befehl `./start.sh` im Ordner `docker` auszuführen (erfordert installiertes `docker` und `docker compose`), um die Software zu starten.

Wenn in der Datei `docker-compose.yml` nichts geändert wird, ist der Editor über Port `8002` und die Player-Software über Port `8001` erreichbar.

In dem Abschnitt [Datenerhebung > Integration & Auslieferung](#) finden sich weitere Informationen zur Verwendung der *Docker*-Container.

## 3 Übersicht: Schritte zur Verwendung von *IRTlib Editor* für die Studienvorbereitung

### 3.1 Overview: Steps to use *IRTlib Editor* for Study Preparation

Preparing a computer-based assessment based on CBA ItemBuilder items begins with using the *IRTlib Editor* to create a study configuration. This typically involves the following steps:



- **Pre-requisite:** Check the availability of the [Runtime](#). The *IRTlib Editor* can be used to prepare assessments using CBA ItemBuilder *Tasks*. Using CBA ItemBuilder *Tasks* are stored in *Project Files* require a *Runtime* (i.e., the files “main..js” and “main..css”) in the version corresponding exactly to the version of the CBA ItemBuilder that was used to generate the items (e.g., 9.9.0). Before using the IRTLib editor, ensure the required *Runtime* is included or import the runtime files (see here for details).

Note: This step is usually not necessary for the use of CBA ItemBuilder items from version 9.9.

- **Create a new “Study”:** The *IRTlib Editor* is used to configure so-called *Studies*. The version of studies can be tracked in the editor, and studies can be published (i.e., sealed to be used for data collection). To start creating content using the *IRTlib Editor*, a study must be created first (see section [Settings](#) for details).



Note that at least one study must be defined in the *IRTlib Editor*, before a study configuration can be used to collect data with an *IRTlib Player*.

- **Specify “Basic Configurations” for the Study (Info):** Basic configurations that apply to the content of a prepared study include the study label and description, the login mode, the display configuration, the menu for test administrators and the specification how to continue after completing all content defined in a study (see section [Studies](#) for details).

- **Create a new “Survey-Part”:** Each study is composed of one or multiple survey parts (see here for details). Parts are considered segments of an assessment that are administered together, like items from a particular domain. Survey parts of type “CBA ItemBuilder” can be used to administer CBA ItemBuilder tasks in a linear sequence or with Blockly-based routing (see Routing Across Survey Parts for details).

**i** Note

Note that each study needs at least one test-part defined in the *IRTlib Editor*, before a study configuration can be used to collect data with an *IRTlib Player*.

- **Configure “Basic Settings” for Survey Part (Info):** A study part of type “CBA ItemBuilder” is a set of CBA ItemBuilder projects that belong together. Each CBA ItemBuilder project requires at least one Task, but projects with multiple tasks are also supported. If items are administered with a common time limit, test parts allow for assigning tasks to a structure differentiating assessment content presented before a time-restricted section (“instructional items”), after a time-restricted section (“epilogue items”), and items in between with restricted time (“core items”, see here).
- **Define “Core Items” (Items):** To finalize the definition of a study-part, the CBA ItemBuilder tasks must be imported in the section “Items”. By default, the sequence of CBA ItemBuilder tasks is assumed to be linear. However, when routing is activated for a study part, Blockly-based routing (see Routing Within Survey Parts for details) can be used to implement various test designs (e.g., multiple booklets, multi-stage testing, etc.).

## **4 Vorbereitung: Studien**

# 5 Vorbereitung: Studien

Konfigurationen die mit dem *IRTlib-Editor* erstellt werden, werden in sogenannten *Studien* zusammengefasst. Eine *Studie* soll die Assessmentinhalte zusammenfassen, welche in einer Erhebung oder Sitzung administriert werden.

## 5.1 Studienverwaltung

Nach dem Start des *IRTlib-Editors* wird die Ansicht *Studien* angezeigt. In dieser Ansicht ist der erste Schritt zur Vorbereitung einer neuen Konfiguration das **Hinzufügen einer neuen Studie**:

<https://youtu.be/7VKf6U3oeM4>

Die erstellten *Studien* erscheinen als Karten in der Ansicht *Studien*. Beachten Sie, dass die Reihenfolge, in der die Studien in der *Studienansicht* angezeigt werden, keine Rolle spielt.

Eine detaillierte Anleitung zur Erstellung einer *Studie* findet sich hier in der eingebetteten Hilfe:

 Eingebettete Programmhilfe

### 5.1.1 Studien Anlegen

Mit dem *IRTLib Editor* werden Konfigurationen für Studien erstellt, welche dann in einem *IRTLib Player* zur Durchführung computerbasierter Assessments verwendet werden können.

#### 5.1.1.1 Wie geht's los?

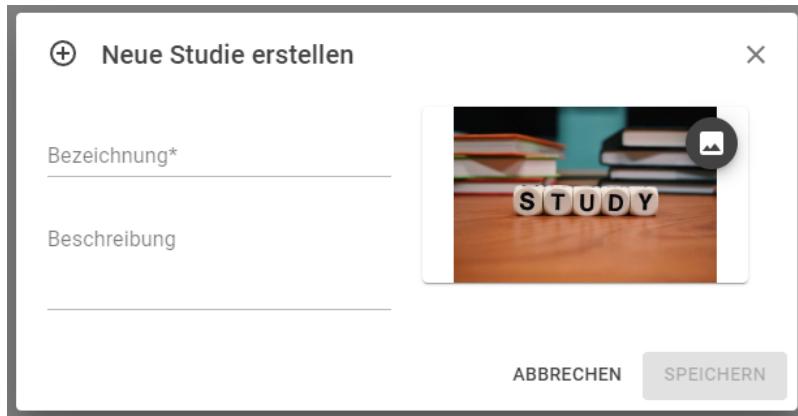
Um mit der Konfiguration einer Studie zu beginnen, klicken Sie auf das Plus-Icon unten rechts:



Danach geben Sie bitte in dem Dialog **Neue Studie erstellen** eine *Bezeichnung* und

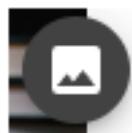
optional eine *Beschreibung* ein.

Achten Sie darauf, dass für die *Bezeichnung* nur Buchstaben (Groß und Kleinschreibung), Ziffer und ein \_ erlaubt sind.



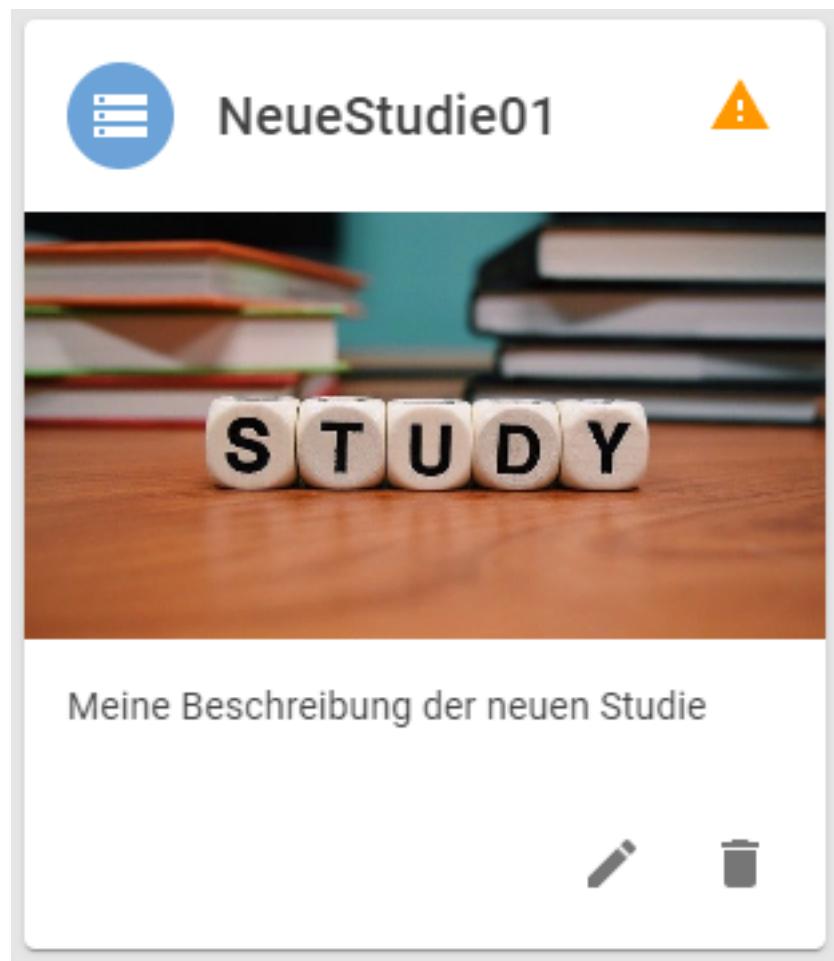
Klicken Sie anschließend auf *Speichern*.

Bei Bedarf können Sie über das folgende Icon einer Studie auch ein Bild zuordnen. Dieses Bild wird im *IRTLib Editor* für diese Studie verwendet:



#### 5.1.1.2 Wie geht's weiter?

Erstellten Studien werden in der Studienübersicht als Kacheln angezeigt:



Um nun mit der Erstellung und Konfiguration einer Studie fortzufahren, klicken Sie auf das kleine Bearbeiten-Icon:



### 5.1.2 Weitere Funktionen und Hinweise

- **Studie Löschen:** Mit dem Papierkorb-Icon können Sie Studien auch wieder löschen. Das Löschen von Studien kann nicht rückgängig gemacht werden:



- **Sprache Wechseln:** Über den Menüpunkt *Einstellungen* gelangen Sie zu dem Punkt *Allgemeine Einstellungen*, wo sie die Sprache des *IRTLib Editors* ändern können.



### Einstellungen

Über diesen Punkt erhalten Sie auch Zugriff auf die im *IRTLib Editor* verfügbaren *CBA ItemBuilder Runtimes* (Unterstützung für die Verwendung von *CBA ItemBuilder*-Inhalten, die mit unterschiedlichen Versionen des Programms erstellt wurden).

## 5.2 Grundlegende Konfigurationen

Die Konfigurationen einer bestimmten Studie, einschließlich Versionierung und Veröffentlichung, werden innerhalb von Studien verwaltet (d.h. nach dem Öffnen einer Studie zur Bearbeitung durch Klicken auf das Bearbeitungssymbol am unteren rechten Rand der Karte).

Erstellte Studien, die im *IRTLib Editor* in der Ansicht *Studien* angezeigt werden, können zur Bearbeitung geöffnet werden.

The screenshot shows the IRTLib Editor's study configuration interface. On the left, a sidebar lists 'My Study Title' under 'Studien' and other sections like 'Allgemein'. The main area has tabs for 'ÜBERSICHT', 'PARAMETER', 'LOGIN', 'ANZEIGE', 'TESTLEITERMENÜ', and 'SESSION-ENDE'. The 'ÜBERSICHT' tab is active, showing 'Bezeichnung\*' with 'My Study Title' and a 'Beschreibung' field. Below are two toggle switches: 'Bildschirmgröße überprüfen' (checked) and 'Routing für Erhebungsteile aktivieren' (checked). A preview image on the right shows a desk setup with a laptop displaying the word 'STUDY'.

Detaillierte Informationen zu der Grundkonfiguration einer Studie finden sich hier in der eingebetteten Hilfe:

#### Eingebettete Programmhilfe

##### 5.2.1 Einstellungen zur Studie

- **Bezeichnung:** Wie soll die *Studie* benannt werden? Achten Sie darauf, dass für die Bezeichnung nur Buchstaben (Groß- und Kleinschreibung), Ziffer und ein \_ erlaubt sind.
- **Beschreibung:** Um eine ausführliche Beschreibung der *Studie* hinterlegen zu können, ist dieses optionale Feld vorgesehen. Hier können auch Sonderzeichen und Umlaute usw. eingegeben werden.
- **Routing für Erhebungsteile aktivieren:** *Studien* bestehen aus einem oder mehreren *Erhebungsteilen*. Die *Erhebungsteile* werden per Default als lineare Abfolge administriert. Wenn die Option *Routing für Erhebungsteile aktivieren* ausgewählt ist, kann die Reihenfolge der *Erhebungsteile* mit einem Blockly-basierten Routing definiert werden. Dadurch sind dynamische Abfolgen von *Erhebungsteilen* möglich, wobei auch Aufrufparameter der Studie bspw. für die Zuordnung von unterschiedlichen Reihenfolgen genutzt werden können.
- **Bildschirmgröße überprüfen:** In Erhebungen, bei denen die Bildschirmgröße nicht bekannt ist, kann mit dieser Option ein Größenvergleich von Objekten (EC-Karte, Banknote, Personalausweis) mit Darstellungen auf dem Bildschirm vorgenommen werden.

Die Geräteprüfung erfolgt mit folgendem Dialog:

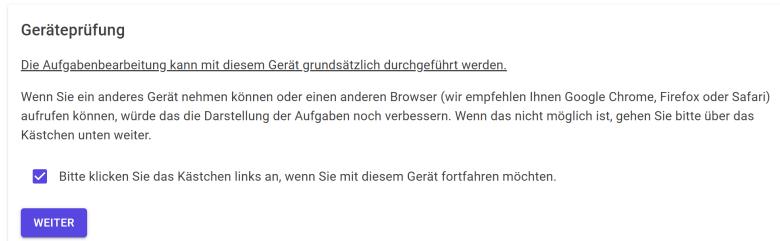
**Geräteprüfung**

Bitte stellen Sie die Größe Ihres Bildschirms mit dem unten abgebildeten Schieberegler ein. Dazu müssen Sie das angezeigte Objekt (Lineal, Personalausweis, Bankkarte o.ä.) als Vergleichsobjekt bei Ihnen vorliegen haben und mit der Anzeige abgleichen. Wählen Sie dazu unten das Ihnen vorliegende Objekt aus. Bitte bestätigen Sie Ihre Einstellung abschließend mit dem Klicken auf den „Weiter“-Button.

Lineal  Personalausweis, Bankkarte o.ä.  10-Euro-Banknote **WEITER**



Wenn die Option *Passende Bildschirmgröße erzwingen* (im Abschnitt *Anzeige*) nicht aktiviert ist, dann kann die Testbearbeitung dennoch begonnen werden. Es wird, wenn die Auflösung zu klein ist, folgender Dialog angezeigt:



Hinweis: Diese Option ist zur Zeit nicht weiter konfigurierbar.

Wenn geänderte Einstellungen erhalten bleiben sollen müssen die Änderungen über das Disketten-Symbol gespeichert werden. Andernfalls kann das Verwerfen-Symbol verwendet werden:



### 5.3 Zugang zu Studien (Login)

Die *IRTlib Software* unterstützt verschiedene Wege, wie sich Personen (Testteilnehmer, Befragte, ...) für ein Assessment Authentifizieren können. Die Konfigurationen umfassen zwei Aspekte:

- *Login-Modus*: Wird ein Zugang benötigt (Login, Login+Passwort, Passphrases/Token) oder nicht? Und wenn Zugangsdaten benötigt werden, was sind *gültige Werte*?
- *Loginquelle*: Wie wird die Login-Information abgefragt (direkte Eingabe auf der Plattform, CBA ItemBuilder Item, ....) oder übergeben (Login-Parameter oder Aufruf-Parameter)?

Detaillierte Informationen zu der Konfiguration der Anmeldung einer *Studie* finden sich hier in der eingebetteten Hilfe:

## Eingebettete Programmhilfe

### 5.3.1 Konfiguration der Anmeldung

Im Abschnitt *Login* kann konfiguriert werden, wie Testteilnehmer, die ein Assessment starten (entweder durch Aufruf eines Links in einem Browser der auf den Online-*IRTlib-Player* verweist oder durch Start des Offline-*IRTlib-Players*), identifiziert oder authentifiziert werden sollen.

- **Login-Modus:** Die *IRTlib-Software* unterstützt verschiedene *Login-Modi*.
  - *Zufälliger Inditifikator*: Wenn eine Sitzung zum ersten Mal gestartet wird, wird in diesem *Login-Modus* ein Identifikator generiert. Diese zufällige, aber eindeutige Zeichenkette (eine sogenannte *UUID*, d.h. ein *Universally Unique Identifier*) wird als Personenidentifikator in allen Daten (d.h. Ergebnisdaten) und allen anderen gespeicherten Daten (z.B. Log-Daten/Trace-Daten, Snapshot-Daten, etc.) verwendet.
  - *Benutzername*: Wenn von den Testteilnehmern erwartet wird, dass sie sich durch eine eindeutige Zeichenfolge identifizieren (z.B. eine Zahl oder ein Text, der als Zugangskennung verwendet wird), kann eine *Studie* mit dem *Login-Modus Benutzername* konfiguriert werden. Der Zugang zum Assessment ist dann nur möglich, wenn die als *Benutzername* eingegebene Zeichenkette gültig ist. Die zugrundeliegende Idee ist, dass die Studienkonfiguration mit einer Liste gültiger Benutzernamen geladen wird und dass ein Testteilnehmer einen gültigen Benutzernamen eingeben muss, bevor er oder sie das Assessment starten kann. Nur authentifizierte Testteilnehmer können auf die als *Studie* definierten Assessmentinhalte zugreifen und die Aufgaben bzw. Fragen beantworten.
  - *Benutzername und Passwort*: Wenn in einer *Studie* nicht nur gültige Benutzernamen, sondern ein Passwort zur Authentifizierung der Testpersonen verwendet werden sollen, ermöglicht der *Login-Modus Benutzername und Passwort* eine Eingabe von Benutzernamen und Passwort. Analog zu *Benutzername* müssen dann beide Informationen in der Studienkonfiguration hinterlegt sein.
  - *Zugriffstoken*: Wenn in der Studienkonfiguration die gültigen Benutzernamen nicht gespeichert werden sollen, kann die Option *Zugriffstoken* verwendet werden. Jedes Token, das einem definierten Schema entspricht, wird dann akzeptiert und als Identifikator für die Testteilnehmer verwendet.
- **Speicher für Sessiondaten:** Bei Onlineauslieferungen kann nach einer Unterbrechung ein Assessment forgesetzt werden. Diese Funktionalität wird bspw. auch benötigt, wenn im Browser die Seite neu geladen wird (bspw. durch Erzwingen eines *Reload/F5*, oder durch Schließen und erneutes Aufrufen der URL). Um

sicherzustellen, dass Sitzungen, die von derselben Person (d.h. vom selben Browser) stammen, auch fortgesetzt werden können, kann die Software so konfiguriert werden, dass der Identifikator im Client gespeichert wird.

- **Gültige Werte:** Die *IRTlib-Software* bietet für die *Login-Modu Benutzername*, *Benutzername + Passwort* und *Zugriffstoken* folgende Mechanismen zur Validierung von Anmeldeinformationen:
  - *Liste*: Eine Liste gültiger Berechtigungen (Benutzername oder Benutzername und Passwort, je nach Konfiguration des *Login-Modus*) kann als Teil der Studienkonfiguration definiert werden. Die Informationen können entweder im *IRTlib-Editor* bearbeitet oder aus einer CSV-Datei importiert werden. Definierte Werte können auch als CSV-Datei exportiert werden.
  - *Code zur Prüfung*: Es kann eine *Blockly*-Funktion angegeben werden, welche *Wahr* zurückmeldet, wenn die übergebenen Anmeldedaten gültig sind (sonst *Falsch*).
- **Gruppenlogin:** Je nach *Login-Modus* dienen Benutzername oder Zugriffstoken als Personenidentifikator. Wenn die Option *Gruppenanmeldung* aktiviert wird, dann werden diese übergebenen Anmeldedaten zur Authentifizierung verwendet, um den Testteilnehmer als Mitglied einer Gruppe zu identifizieren (d. h. nur Testteilnehmer, die den Benutzernamen kennen, können sich als Teil der Gruppe authentifizieren). Innerhalb der Gruppe wird dann ein zusätzlicher Zufallsidentifikator generiert, um verschiedene Personen aus einer Gruppe zu unterscheiden.
- **Loginquelle:** Die *IRTlib-Software* unterstützt verschiedenen möglichen Optionen, wie für Anmeldeinformationen bereitgestellt werden können.
  - *Plattform*: Ein Anmeldedialog wird vom *IRTlib-Player* (d.h. der Plattform) angezeigt. Die Überschrift zur Eingabe der Zugangsdaten, die Beschriftung der Eingabe für Benutzernamen und Passwort, die Beschriftung des Weiter-Buttons, dein Begrüßungs- und ein Instruktionstext sowie ein Fehlertext bei fehlgeschlagenen Login-Versuchen können konfiguriert werden.
  - *Parameter*: Gültige Anmeldedaten für Testteilnehmer können auch über die *Befehlszeile* (d.h. Parameter beim Aufruf der Offline-Version des *IRTlib Players*) oder über URL-Parameter (d.h. Parameter beim Aufruf der *Studie* über einen Link auf eine Online-Version des *IRTlib Players*) bereitgestellt werden. In diesem Fall wird kein Anmeldedialog oder Loginitem angezeigt.
  - *Item*: Alternativ zu einem Dialog des *IRLLib-Players* kann auch ein *CBA Item-Builder*-Task konfiguriert werden, der als Login-Eingabemaske dient. Innerhalb des Items wird ein sogenannter *ExternalPageFrame* verwendet, um zur Validierung einer Eingabe einen bestimmten JavaScript-Befehl an den *IRLLib-Player* zu senden (ein Beispiel finden Sie [hier](#)).

Das Login Item muss als *CBA ItemBuilder* Projektdatei für die konfigurierte Laufzeitumgebung (Runtime) verfügbar sein und der Studienkonfiguration hinzugefügt werden. Um ein Login Item zur Studienkonfiguration hinzuzufügen, kann der integrierte Importdialog verwendet werden. Mehr Informationen zum Importieren von *CBA ItemBuilder*-Projekten findet sich in der Hilfe zum Abschnitt *Items* eines *Erhebungsteils*.

- **Zusätzliche Parameter:** Neben der *Authentifizierung* von Testteilnehmern können die Anmeldeinformationen in der *IRTlib-Software* auch als zusätzlicher Parameter hinterlegt werden, die dann bspw. in der Ablaufsteuerung verwendet werden können.
  - Parameter für Dateinamen: Der `RawDataPath` (d.h. der relative Pfad unter dem der Offline-*IRTlib-Player* die Ergebnisdaten speichert) und der `MonitoringFile` (d.h. der Name der Datei in welcher der Offline-*IRTlib-Player* Informationen fürs Studienmonitoring schreibt) können als Teil der Anmeldedaten konfiguriert werden.
  - *Blockly*-Variablen: Zusätzliche Parameter können auch als sogenannte *Preload*-Variablen mit den Anmeldeinformationen hinterlegt werden.

Table 5.1: Zusammenfassung der Optionen, die als *Konfiguration der Anmeldung* kombiniert werden können

Login Modus	Speicher für				
	Session-daten	Gruppenlogin	Gültige Werte	Loginquelle	Zusätzliche Parameter
Zufälliger Inditifikator	ja	nein	nein	keine	nein
Benutzername	ja	ja	Liste oder Code	Plattform, Item + Parameter	Werte oder Parameter
Benutzername und Passwort	ja	ja	Liste oder Code	Plattform, Item + Parameter	Werte oder Parameter
Zugriffstoken	ja	ja	Schema oder Code	Plattform, Item + Parameter	Parameter

## 5.4 Anzeige von Assessment-Inhalten

Studien können festlegen, wie der *CBA ItemBuilder*-Inhalt dargestellt werden soll. Die Einstellungen im Abschnitt *Anzeige* können sich auf die Skalierung und die Ausrichtung der Inhalte sowie auf das Verhalten der *IRTlib Player*-Anwendung beziehen.

Detaillierte Informationen zu der Konfiguration der *Anzeige* einer *Studie* finden sich hier in der eingebetteten Hilfe:



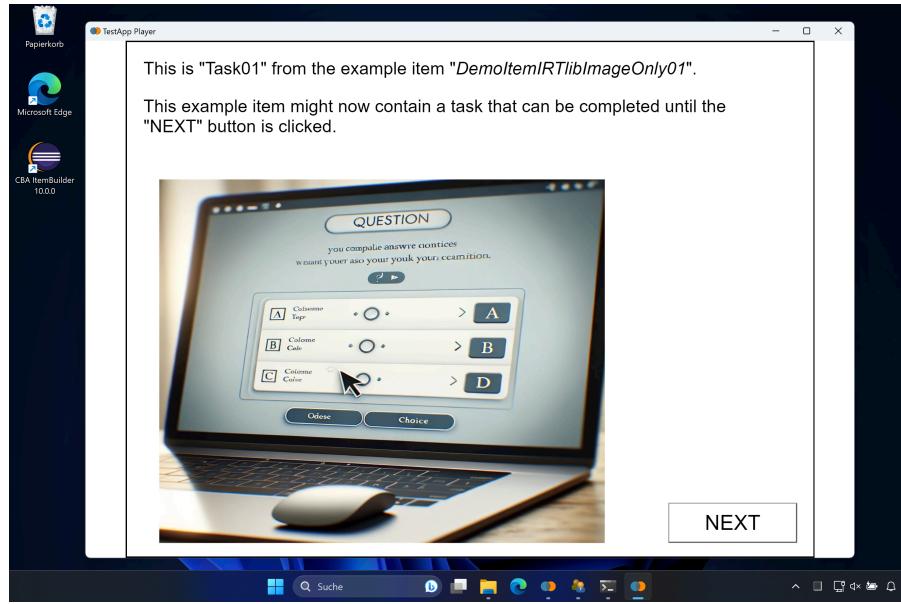
### 5.4.1 Anzeigeeinstellungen

Für die Konfiguration der Anzeige stehen ausgewählte Optionen zur Verfügung, die sich auf die Darstellung der Assessmentinhalte und die Verwendung von *CBA ItemBuilder*-Inhalten beziehen, welche mit einem festgelegten Seitenverhältnis (Breite und Höhe) erstellt werden.

#### 5.4.1.1 Fenstermodus

In der Auswahl **Fenstermodus** kann konfiguriert werden, ob ein zusätzliches Fenster im *IRTlib Player* angezeigt wird. Die Konfiguration wird je nach Umgebung unterschiedlich umgesetzt:

- *Window*: In diesem *Fenstermodus* wird im Offline-*IRTlib Player* ein reguläres Programmfenster verwendet, im Online-*IRTlib Player* wird der Assessmentinhalt im normalen Browserbereich angezeigt, und die Adressleiste und die Navigationsschaltflächen des Browsers sind in diesem Modus sichtbar.



- **Vollbild:** Der Offline-*IRTlib Player* startet direkt im Vollbildmodus, wenn diese Option konfiguriert ist. Damit verbunden ist auch ein *Kiosk-Modus*, d.h. der Zugriff auf andere Programme und das (versehentliche) Beenden des Programms ist nur über den *Task Manager* möglich. Soll die Möglichkeit zum Beenden der Testung bspw. für einen Testleiter möglich sein, muss ein *Testleitermenü* konfiguriert sein.

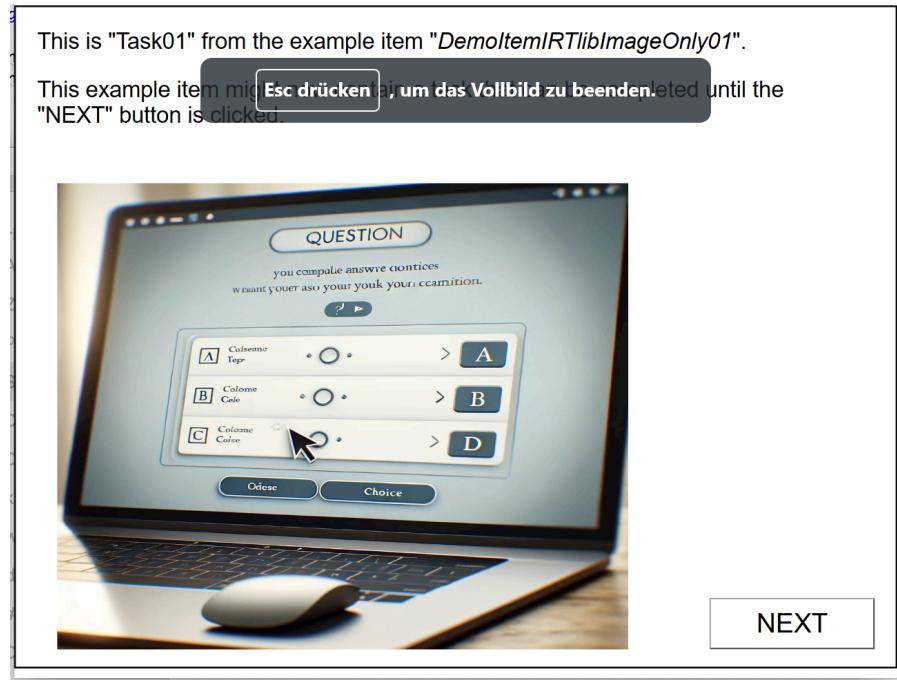
Der Online-*IRLlib Player* kann Assessmentinhalte auch im Vollbildmodus anzeigen, wenn diese Option gewählt ist. Wird der Vollbildmodus im Browser verlassen, wird dann der Assessmentinhalt ausgeblendet. Da automatisiert nicht in einem Browser in den Vollbildmodus gewechselt werden kann, sieht die Zielperson zunächst folgende Nachricht der Plattform:

## Achtung - Wichtiger Hinweis

Der Test kann nur im Vollbild durchgeführt werden. Wenn Sie im Vollbildmodus ihres Browsers sind, beenden Sie diesen (z.B. mit ESC oder F11) und klicken Sie dann auf den folgenden Button, um mit dem Test fortfahren zu können.

Durch klick auf den Button *Vollbild Aktivieren* wird der Vollbildmodus gestartet und der Assessmentinhalt dann ohne Fensterrahmen und Navigationsflchen des Browsers dargestellt. Fr kurze Zeit wird dann von den Browser

typischerweise ein Hinweis eingeblendet dass mit Esc der Vollbildmodus wieder beendet werden kann.



Beachten Sie, dass diese Funktion nur in Browsern zur Verfügung steht, die den Vollbildmodus unterstützen (insbesondere auf älteren mobilen Geräten wird der Vollbildmodus nicht vollständig unterstützt; siehe für Details z.B. auf [caniuse.com](#)).

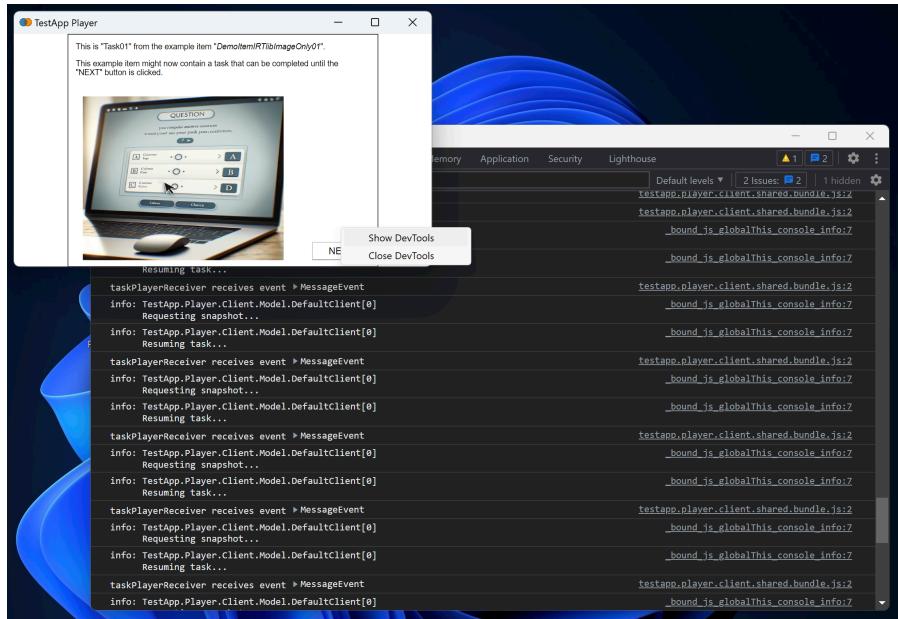
- *Vollbild, wenn unterstützt:* In diesem Modus wird das Assessment im Online-*IRTlib Player* nur dann im Vollbildmodus angezeigt, wenn der Browser den Vollbildmodus unterstützt. Der Inhalt des computerbasierten Assessments wird jedoch im Fenstermodus angezeigt, wenn eine Studie online bereitgestellt wird und ein Browser verwendet wird, der den Vollbildmodus nicht unterstützt. Für den *IRLLib Player* offline ist diese Konfiguration identisch mit *Vollbild*.

#### Achtung - Wichtiger Hinweis

Der Test kann nur im Vollbild durchgeführt werden. Leider kann die Aufgabenbearbeitung auf diesem Gerät nicht durchgeführt werden, da der Browser kein Vollbildmodus unterstützt. Bitte prüfen Sie, ob Sie ein anderes Gerät (Computer oder Laptop) verwenden können!

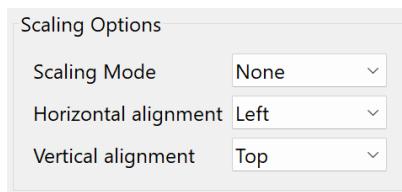
- *Debug*: Dieser Modus ermöglicht während der Testausführung den Zugriff auf die Entwicklerwerkzeuge des Browsers, die für die Fehlersuche von Softwareentwicklern vorgesehen sind.

Wenn der Offline-*IRTlib Player* mit einer Studie gestartet wird, welche als *Festermodus* den Eintrag *Debug* konfiguriert hat, dann lassen sich über die rechte Maustaste während der Aufgabenpräsentation die sogenannten Entwickertools (*DevTools*) abrufen.



#### 5.4.2 Skalierung und Ausrichtung

Assessmentinhalte, die mit dem *CBA ItemBuilder* erstellt werden, werden für eine konkrete Größe in Pixeln (Breite mal Höhe) erstellt. Für die Darstellung auf Geräten mit unterschiedlichen Bildschirmgrößen und Bildschirmauflösungen können die Inhalte dann skaliert werden. Im *CBA ItemBuilder* stehen deshalb in der *Preview* die Optionen unter *Scaling Options* zur Verfügung:



Im *IRTlib Editor* können analoge Einstellung vorgenommen werden.

- **Skalierung:** Einstellung wie Inhalte angepasst werden sollen, wenn verfügbarer Platz und Größe der Items abweichen (*Scaling Mode*).
  - *keine*: Die Inhalte werden ohne Anpassung an die verfügbare Fenster- bzw. Bildschirmgröße angezeigt (entspricht *None*).
  - *hochskalieren*: Inhalte werden vergrößert, damit der verfügbare Platz ausgenutzt wird (entspricht *Up*).
  - *runterskalieren*: Inhalte werden verkleinert, damit sie auf den Bildschirm/ins Fenster passen (entspricht *Down*).
  - *Fenstergröße*: Inhalte werden vergrößert und verkleinert (entspricht *Both*).
- **Horizontale Ausrichtung:** Die Optionen *zentriert / links / rechts* werden genutzt um Iteminhalte horizontal auszurichten, wenn die Breite von Fenster oder Bildschirm größer ist als die Breite des Inhalts.
- **Vertikale Ausrichtung:** Die Optionen *zentriert / oben / unten* werden genutzt um Iteminhalte vertikal auszurichten, wenn die Höhe von Fenster oder Bildschirm größer ist als die Höhe des Inhalts.

#### 5.4.2.1 Weitere Einstellungen

- **Passende Bildschirmgröße erzwingen:** Wenn bei *Skalierung* nicht *runterskalieren* oder *Fenstergröße* ausgewählt ist, kann über diese Option erzwungen werden, dass man nur dann mit der Aufgabenbearbeitung starten kann, wenn die verfügbare Größe des Fensters bzw. Bildschirms größer ist als die benötigte Breite/Höhe der Items. Andernfalls wird folgende Nachricht angezeigt:

Achtung - Wichtiger Hinweis  
 Das Fenster ist zu klein um den Test durchzuführen. Um mit dem Test fortfahren zu können, vergrößern Sie das Fenster.  
 Die aktuelle Fenstergröße ist 1075 x 717. Der Test erfordert eine Größe von 1024 x 768.

**VOLLBILD AKTIVIEREN**

Hinweis: Die Einstellungen zur Anzeige beziehen sich auf alle *Erhebungsteile* innerhalb einer *Studie*. Werden in einem *IRTlib-Player* mehrere Studien konfiguriert, müssen die Einstellungen zueinander passen, d.h. es ist nicht mögliche mit einer Instanz eines *IRTlib-Players* gleichzeitig eine Studie im *Fenstermodus: Fenster* oder im *Fenstermodus: Vollbild* zu administrieren.

Wenn geänderte Einstellungen erhalten bleiben sollen müssen die Änderungen über das Disketten-Symbol gespeichert werden. Andernfalls kann das Verwerfen-Symbol verwendet werden:



## 5.5 Menü für Testadministratoren

Wenn die Durchführung von Assessments begleitet durch Testleiter oder Interviewer erfolgt, können Funktionen passwortgeschützt für Testleiter definiert werden.

### ⚠ Warning

Auch wenn Sie die Funktionalität eines Testadministratormenüs für die Durchführung Ihrer Datenerfassung nicht benötigen, sollten Sie dennoch ein Testadministratormenü definieren, wenn Sie planen, mit dem *IRTlib Player* Daten offline zu erfassen. Nur so können Sie sicherstellen, dass Sie die Anwendung im Falle unvorhergesehener Ereignisse ohne den Task-Manager (und ohne möglichen Datenverlust) beenden können.

Detaillierte Informationen zu der Konfiguration des *Testleitermenüs* finden sich hier in der eingebetteten Hilfe:

### 💡 Eingebettete Programmhilfe

#### 5.5.1 Konzept eines Testleitermenüs (Menü für Testadministratoren)

Die Konfiguration des Menü für Testadministratoren erfolgt in zwei Schritten. Zunächst muss eine Tastenkombination definiert werden, mit der das Testmanager-Menü aufgerufen werden kann. Wird diese Tastenkombination während der Testbearbeitung gedrückt, erscheint ein Fenster zur Passworteingabe. Testadministratoren geben das (nur) ihnen bekannte Passwort ein und erhalten so Zugriff auf ausgewählte Funktionen. Zu diesem Zweck müssen in einem zweiten Schritt eine oder mehrere Rollen im *IRTlib Editor* definiert werden.

##### 5.5.1.1 Zugriff für Testleitung

Zunächst muss eine Tastenkombination definiert werden.

- **Taste:** Die Konfiguration der Tastenkombination für das Testleitermenü erfordert zunächst die Definition einer Taste. Um eine Taste festzulegen klickt man in das Feld und drückt die Taste, welche für das Testleitermenü verwendet werden soll.
- **Modifikatoren** (Alt, Strg und Shift): Für eine Taste kann dann zusätzlich fest-

gelegt werden, ob eine oder mehrere Modifikatoren gedrückt werden müssen damit das Testleitermenü geöffnet wird.

Beispiel:

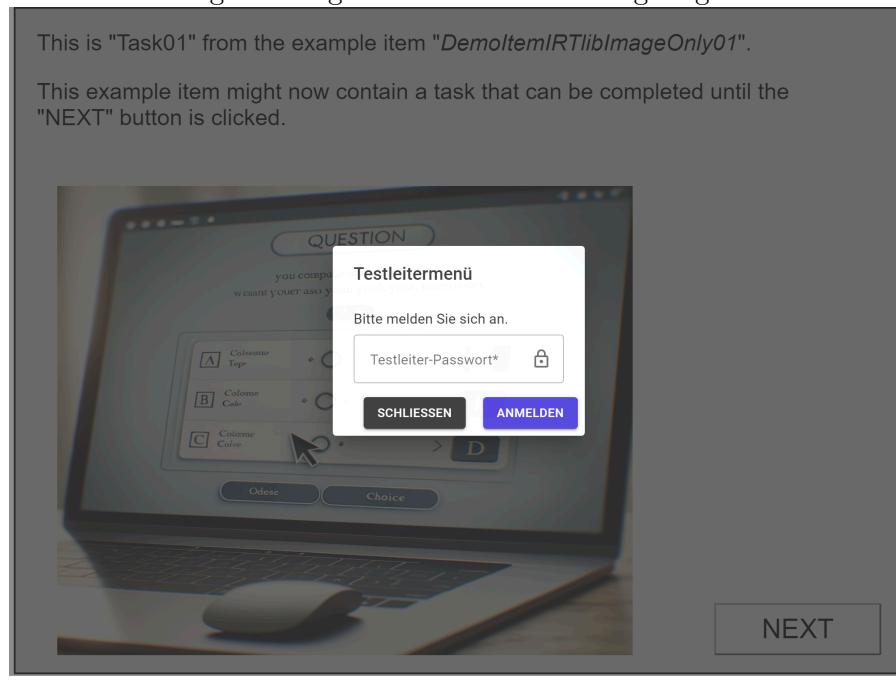
- Die folgende Konfiguration definiert die Tastenkombination **Strg + Shift + X**:

Taste*	X
	<input type="checkbox"/> Alt
	<input checked="" type="checkbox"/> Strg
	<input checked="" type="checkbox"/> Shift

Die definierte Tastenkombination öffnet nur die Option zur Eingabe eines Passworts für Testleiter während der Testbearbeitung im *IRTLib Player*. Um die Funktion zu nutzen ist ein Passwort notwendig, welches zusammen mit einer Rolle im zweiten Schritt definiert wird.

#### 5.5.1.2 Rollen

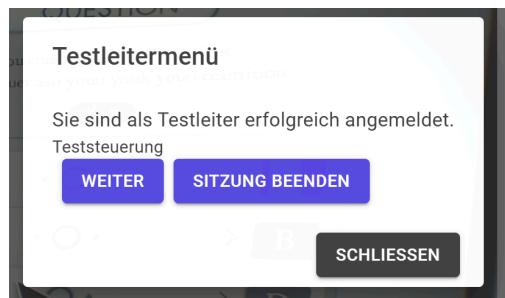
Nach dem Aufruf der definierten Tastenkombination wird während der Testbearbeitung die Aufforderung zur Eingabe eines Passworts angezeigt:



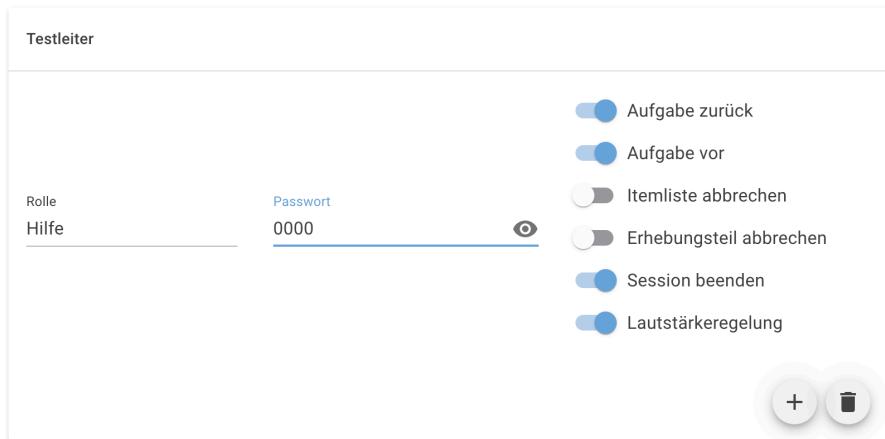
Welche Funktionen nun wirklich zugreifbar sind wird dadurch gesteuert, welches Passwort eingegeben wird. Nur wenn ein gültige Passwort bekannt ist, können Funktionen der Testleitung aufgerufen werden.

Beispiel:

- In der folgenden Konfiguration können Testleiter mit diesem Passwort zur nächsten Aufgaben springen (*Weiter*) oder die Anwendung beenden (*Sitzung beenden*):



Um eine Rolle zu definieren, muss zunächst das +-Symbol unten rechts geklickt werden. Danach kann der Name einer Rolle und ein Passwort definiert:



Der Name der Rolle dient nur der Dokumentation. Entscheidend für die Funktionalität ist die Vergabe eines eindeutigen Passwortes und die Auswahl einer oder mehrerer der folgenden Funktionen:

- **Aufgabe zurück:** Ermöglicht die Navigation zur vorherigen Aufgabe.
- **Aufgabe vor:** Ermöglicht die Navigation zur nächsten Aufgabe an.
- **Itemliste abbrechen:** Ermöglicht die Abarbeitung der aktuellen Itemliste abzubrechen. Diese Option ist insbesondere sinnvoll, wenn in einem *Erhebungsteil* die Option *Routing* aktiviert ist und die Definition von *CBA ItemBuilder Tasks*

mit Hilfe von Itemlisten umgesetzt ist.

- **Erhebungsteil abbrechen:** Ermöglicht den Abbruch des aktuellen Erhebungsteils.
- **Sitzung beenden:** Ermöglicht das Beenden der aktuellen Sitzung.
- **Lautstärkeregelung:** Ermöglicht die Änderung der Lautstärke.

Die Audiodatei, welche zur Kontrolle der Audioausgabe abgespielt wird nachdem die Lautstärke verändert wurde, kann im Abschnitt *Audio für Soundtest* eingefügt und in der Studienkonfiguration hinterlegt werden.

Wenn geänderte Einstellungen erhalten bleiben sollen müssen die Änderungen über das Disketten-Symbol gespeichert werden. Andernfalls kann das Verwerfen-Symbol verwendet werden:



## 5.6 Abschluss von Erhebungen

Für die Integration von Assessments in externe Abläufe besteht die Möglichkeit zu konfigurieren, wie nach der Bearbeitung der Assessmentinhalte in einer *Session* vorgegangen werden soll, was also am *Session-Ende* geschehen wird.

### Eingebettete Programmhilfe

#### 5.6.1 Session und Session-Ende

Eine *Session* bezieht sich auf die Durchführung einer Erhebung mit *einer* Person zu *einem* bestimmten Zeitpunkt. Der in einer Session angezeigte Inhalt entspricht einer konfigurierten *Studie*, wie sie im *IRTlib Editor* erstellt werden kann. Nachdem alle in einer *Studie* definierten Erhebungsteile durchgeführt worden sind, wird das *Session-Ende* erreicht.

##### 5.6.1.1 Konfiguration des Session-Endes

Was nach einem *Session-Ende* erfolgt, d.h. wie sich der *IRTlib Players* am Ende einer Sitzung verhält, kann mit den folgenden Optionen festgelegt werden:

- **Neue Session starten:** Es wird eine neue Sitzung gestartet. Dieses Verhalten ist nicht sinnvoll, wenn die Anmelde Daten übergeben werden (entweder als *Startup-*

Parameter oder als *URL-Parameter*).

- **Endtext anzeigen:** Wenn diese Option ausgewählt ist, zeigt die Plattform den konfigurierten Text an. Der Text kann als *Nachricht auf Endseite* konfiguriert werden.
- **End-Item anzeigen:** Analog zu einem *Login-Item* kann auch ein *CBA Item-Builder-Item* definiert werden, das am Ende einer Sitzung angezeigt wird.

Das *End-Item* kann schließlich das Beenden des Offline-*IRTlib Players* anstoßen. Ein Beispiel für ein *End-Item* mit dem notwendigen JavaScript-Aufruf findet sich [hier](#).

- **Redirect to Exit URL (Redirect zu Exit-Url):** Bei Online-Lieferungen mit dem *IRTlib Player* ist es möglich, auf eine URL umzuleiten. Die *Weiterleitungs-URL* kann dann konfiguriert werden.

#### 5.6.1.2 Weitere Optionen

- **Session ID kann wiederverwendet werden:** Wenn diese Option aktiviert ist, können mehrere Datenerfassungen mit einer Session-ID administriert werden.

Wenn geänderte Einstellungen erhalten bleiben sollen müssen die Änderungen über das Disketten-Symbol gespeichert werden. Andernfalls kann das Verwerfen-Symbol verwendet werden:



## **6 Vorbereitung: Erhebungsteile**

# 7 Vorbereitung: Erhebungsteile

Assessments, die mit der *IRTLib Software* administriert werden bestehen aus sogenannten *Erhebungsteilen*. Nach der Konfiguration einer Studie muss zumindest ein *Erhebungsteil* angelegt werden.

## 7.1 Erhebungsteilverwaltung

Nach dem Erstellen einer *Studie* erfolgt in der Ansicht *Erhebungsteile* als nächster Schritt zur Vorbereitung einer Testausfließierung das **Hinzufügen eines neuen Erhebungsteils**:

<https://youtu.be/YFgu8uz8nkc>

Die erstellten *Erhebungsteile* erscheinen als Karten in der Ansicht *Erhebungsteile*. Wenn Studien aus mehreren Erhebungsteilen bestehen, kann für *lineare Abläufe* die Reihenfolge der Erhebungsteile in der Ansicht *Erhebungsteile / Übersicht* angepasst werden. Sollen *Erhebungsteile* in Abhängigkeit von Variablen (z.B. übergebene *Preload*-Variablen oder andere *Blockly*-Variablen gesteuert werden, kann alternativ ein Routing zwischen Erhebungsteilen konfiguriert werden.

Eine detaillierte Anleitung zur Erstellung von *Erhebungsteilen* findet sich hier in der eingebetteten Hilfe:

 Eingebettete Programmhilfe

### 7.1.1 Erhebungsteil Anlegen

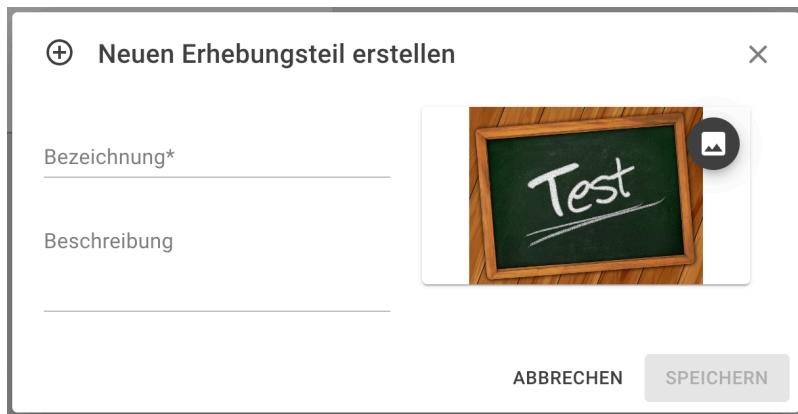
Mit dem *IRTLib Editor* werden Konfigurationen für *Studien* erstellt, welche dann in einem *IRTLib Player* zur Durchführung computerbasierter Assessments verwendet werden können. *Studie* bestehen aus einem oder mehreren *Erhebungsteilen*.

#### 7.1.1.1 Wie geht's?

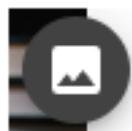
Nachdem eine *Studie* angelegt ist, kann nun über Plus-Icon unten rechts ein *Erhebungsteil* hinzugefügt werden:



Danach geben Sie bitte in dem Dialog **Neuen Erhebungsteil erstellen** eine *Bezeichnung* und optional eine *Beschreibung* ein.  
Achten Sie darauf, dass für die *Bezeichnung* wieder nur Buchstaben (Groß und Klein-schreibung), Ziffer und ein \_ erlaubt sind.  
Klicken Sie anschließend auf *Speichern*.

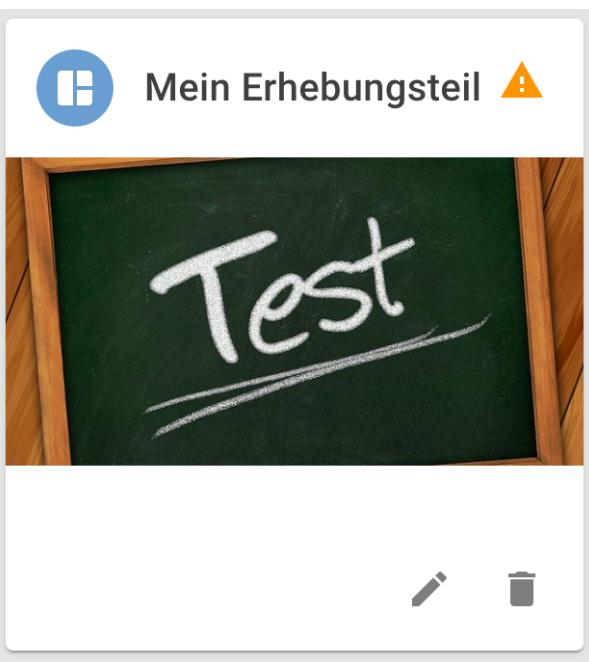


Bei Bedarf können Sie über das folgende Icon einem *Erhebungsteil* auch ein Bild zuordnen.  
Dieses Bild wird im *IRTLib Editor* für diesen Erhebungsteil verwendet:



#### 7.1.1.2 Erhebungsteil Bearbeiten

Erstellten *Erhebungsteile* werden in der Erhebungsteilübersicht als Kacheln angezeigt:



- Um nun mit der Konfiguration eines *Erhebungsteils* fortzufahren, klicken Sie auf das kleine Bearbeiten-Icon:



- Erhebungsteil Löschen:** Mit dem Papierkorb-Icon können Sie *Erhebungsteile* auch wieder löschen. Das Löschen von *Erhebungsteilen* kann nicht rückgängig gemacht werden:



#### 7.1.1.3 Erhebungsteile Sortieren

Wenn in der Konfiguration einer *Studie* in der Ansicht *Info* (Abschnitt *Übersicht*) die Option *Routing für Erhebungsteile aktivieren* nicht ausgewählt ist, dann werden *Erhebungsteile* in der Reihenfolge administriert, in welcher sie in der Erhebungsteilverwaltung angezeigt werden.

- **Erhebungsteile Verschieben:** Um per drag-and-drop die Reihenfolge von *Erhebungsteilen* zu verändern, muss zunächst über folgendes Toggle-Icon der Modus *Reihenfolge Ändern* aktiviert werden:



Danach können die Kacheln in die gewünschte Reihenfolge gebracht werden. Der Modus *Reihenfolge Ändern* wird beendet, wenn das Disketten-Symbol angeklickt oder die Änderungen verworfen werden:



Die Reihenfolge von *Erhebungsteilen* kann in der Ansicht *Erhebungsteile* verändert werden:

<https://youtu.be/Ag0IcETZTdM>

Vor dem Hinzufügen bzw. Auswählen von CBA ItemBuilder Projekten, sie im Abschnitt Assessmentinhalte (Items) beschrieben, können in der Ansicht *Info* ausgewählte Einstellungen konfiguriert werden.

Eine detaillierte Beschreibung der findet sich hier in der eingebetteten Hilfe:

 Eingebettete Programmhilfe

## 7.1.2 Grundkonfiguration für Erhebungsteile

### 7.1.2.1 Bezeichnung und Beschreibung

- **Bezeichnung:** Die interne Bezeichnung des Erhebungsteils, welche im *IRTlib Editor* zur Bearbeitung und zur Definition des Ablaufs angezeigt wird. Bezeichnungen dürfen keine Sonderzeichen, Leerzeichen und Umlaute enthalten und nicht mit einer Ziffer beginnen.
- **Beschreibung:** Optionale zusätzliche Beschreibung eines Erhebungsteils.

### 7.1.2.2 Routing innerhalb von Erhebungsteilen

- **Routing aktivieren:** Die konfigurierten Assessmentinhalte im Abschnitt *Items* können als lineare Sequenz, d.h. in der konfigurierten Reihenfolge administriert

werden. Soll eine davon abweichende Reihenfolge verwendet werden, kann hier die Option *Routing aktivieren* gewählt werden. Die Reihenfolge kann dann im Abschnitt *Routing* als visuelles Programm spezifiziert werden.

#### 7.1.2.3 Weitere Einstellungen

- **Snapshot verwenden:** Damit CBA ItemBuilder *Tasks* mehrfach besucht werden können, wird deren *Inhalt* beim Verlassen des Items in sogenannten *Snapshots* gespeichert. *Snapshots* können auch dazu verwendet werden, um die Inhalte eines CBA ItemBuilder *Tasks* zu einem späteren Zeitpunkt erneut darzustellen. Diese Option sollte nur dann deaktiviert werden, wenn es einen gewichtigen Grund gibt und die Konsequenzen (d.h. die dann nicht gespeicherten *Snapshot*-Daten) sorgfältig bedacht wurden.

Das Hinzufügen und Verwalten von CBA ItemBuilder Projekte innerhalb des *IRTlib-Editors* erfolgt im Abschnitt Items.

##### Hinweis zur Zeitbegrenzung

Für die Administration von zeitbegrenzten Erhebungsteilen kann unter Bearbeitungszeit ein Zeitlimit definiert werden. Wenn die Option *Bearbeitungszeit begrenzen* aktiviert ist, können ein oder mehrere *Tasks* definiert werden, welche bei einem *Timeout* angezeigt werden. Außerdem können im Abschnitt Vorspann-Items(s) und Nachspann-Item(s) Inhalte definiert werden, welche vor bzw. nach dem zeitbegrenzten Teil administriert werden.

## 7.2 Assessmentinhalte (Items) Einfügen

Die Inhalte, welche in einem Erhebungsteil vom Typ *CBA ItemBuilder* verwendet werden sollen, werden über den *IRTlib Editor* in die Konfiguration übernommen, d.h. die mit dem *IRTlib Editor* erstellte Konfiguration enthält auch die *CBA ItemBuilder Project Files*. Für das Hinzufügen oder Aktualisierung von *CBA ItemBuilder* Projekten steht die Ansicht *Items* zur Verfügung.

Eine detaillierte Beschreibung der findet sich hier in der eingebetteten Hilfe:

 Eingebettete Programmhilfe

## 7.2.1 Items Konfigurieren

### 7.2.1.1 Grundfunktionen

- **Importieren von CBA ItemBuilder-Projektdateien:** Der *IRTlib Editor* pflegt eine *Liste bekannter Items*, zu denen CBA ItemBuilder-Projektdateien die noch nicht bekannt sind hinzugefügt werden können. Um eine Projektdatei hinzuzufügen, öffnet man zunächst mit dem +-Symbol die *Liste bekannter Items* und wählt dann die Schaltfläche *Importieren* aus.



 IMPORTIEREN

- **Aktualisieren bereits importierter CBA ItemBuilder-Projektdateien:** Wenn eine CBA ItemBuilder-Projektdatei bereits in der *Liste der bekannten Items* enthalten ist, können die Projektdateien aktualisiert werden. Sie werden dann nicht zusätzlich zur *Liste der bekannten Items* hinzugefügt, sondern die bereits vorhandene CBA ItemBuilder-Projektdatei wird in einer neueren Version hinterlegt. Um ein Item zu aktualisieren muss es zunächst in der Liste der Items eines *Erhebungsteils* ausgewählt werden. Dadurch wird das Aktualisieren-Symbol aktiv. In dem sich dann öffnenden Dialog *Item aktualisieren* kann über die Schaltfläche *Importieren* eine aktualisierte Version einer CBA ItemBuilder-Projektdatei hinzugefügt werden.



 IMPORTIEREN

- **Vorschau von CBA ItemBuilder-Projektdateien:** Die in einem *Erhebungsteil* hinzugefügten Items können direkt im *IRTlib Editor* in einer eingebauten Preview-Funktion angesehen werden. Um ein Item zu anzusehen muss es zunächst in der Liste der Items eines *Erhebungsteils* ausgewählt werden. Danach kann die *Vorschau* über das Augen-Symbol aufgerufen werden:



- **Exportieren von CBA ItemBuilder-Projektdateien:** CBA ItemBuilder-Projektdateien die in den *IRTlib Editor* importiert wurden, können zur weiteren Bearbeitung mit dem *CBA ItemBuilder* exportiert werden. Um ein ausgewähltes Item aus der Liste der Items eines *Erhebungsteils* zu exportieren, kann das Download-Symbol augerufen werden:



- **Löschen von CBA ItemBuilder-Projektdateien:** Die in *Erhebungsteilen* eingefügten Items können aus einem *Erhebungsteil* wieder gelöscht werden. Durch das Löschen-Symbol wird das Item aus einem *Erhebungsteil* entfernt, es verbleibt aber in der *Liste bekannter Items*:



Hinweis: Es ist bisher nicht möglich, CBA ItemBuilder-Projektdateien aus der *Liste bekannter Items* wieder zu löschen. Diese Funktionalität ist nicht notwendig, da CBA ItemBuilder-Projektdateien vom *IRTlib-Editor* nur dann in die Konfiguration einer *Studie* übernommen werden, wenn *Tasks* aus einer CBA ItemBuilder-Projektdatei in einem *Erhebungsteil* verwendet werden.

#### 7.2.1.2 Sortieren von Items (Linearer Ablauf)

- **Sortieren CBA ItemBuilder-Projektdateien:** Wenn für einen *Erhebungsteil* die Option *Routing aktivieren* nicht ausgewählt ist, dann kann in der Liste der Items die Reihenfolge über die folgenden Schaltflächen angepasst werden:



Die Items dann exakt so administriert, wie sie für einen *Erhebungsteil* in dieser Liste aufgeführt sind.

Hinweis: Änderungen an der Sicht *Items* müssen über das Disketten-Symbol gespeichert oder mit dem Rückgängig-Symbol verworfen werden:



## 7.3 Bearbeitungszeit

Wenn die Administration einer linearen Folge von *CBA ItemBuilder Tasks* mit einer begrenzten Bearbeitungszeit administriert werden sollen, lässt sich dies durch Definieren einer maximalen Bearbeitungszeit (in Sekunden) umsetzen. Soll bspw. ein Testinhalt maximal 28 min. administriert werden, wird als *Bearbeitungszeit* eine Zeit von 1680 Sekunden definiert. Die Nachricht, welche beim Ablaufen der Bearbeitungszeit angezeigt werden soll, lässt sich als ein (oder mehrere) *CBA ItemBuilder Tasks* definieren.

Eine detaillierte Beschreibung der findet sich hier in der eingebetteten Hilfe:

### Eingebettete Programmhilfe

#### 7.3.1 Zeitbegrenzung Definieren

*Erhebungsteile* ohne *Routing* können auf einfache Weise einen zeitbegrenzten Abschnitt enthalten. Dafür wird in der Sicht *Bearbeitungszeit* die Option *Bearbeitungszeit begrenzen* aktiviert und ein Zeitlimit in Sekunden (>0) eingetragen.

Für eine Zeitbegrenzung werden vier Gruppen von *CBA ItemBuilder Tasks* unterschieden, die an unterschiedlichen Stellen im *IRTlib Editor* definiert werden. In der Sicht *Items* (analog zu nicht zeitbegrenzten *Erhebungsteilen*) werden die Items für welche die Zeitbegrenzung gelten soll definiert:

- **Items:** Items die so lange angezeigt werden, bis das Zeitlimit erreicht wurde.

In der Ansicht *Bearbeitungszeit* kann zusätzlich definiert werden:

- **Timeout-Items:** Items die nur angezeigt werden, wenn die zeitbegrenzten Items nicht in der begrenzten Bearbeitungszeit abgeschlossen wurden.

Als einzelne Sichten der Konfiguration von *Erhebungsteile* können schließlich folgende Tasks definiert werden:

- **Vorspann-Items:** Items die vor dem zeitbegrenzten Abschnitt angezeigt werden.

- **Nachspann-Items:** Items die nach dem zeitbegrenzten Abschnitt angezeigt werden.

In allen genannten Dialogen stehen die Symbole für folgende Operationen zur Verfügung:

- Hinzufügen:
- Aktualisieren:
- Vorschau/Preview:
- Download/Export:
- Löschen:
- Sortieren:

Hinweis: Komplexere Designs mit ggf. mehreren Timern lassen sich mit dem *IRTlib Editor* umsetzen, wenn die Option *Routing aktivieren* in der Übersichtsansicht zu einem *Erhebungsteil* aktiviert ist.

Hinweis: Änderungen an der Sicht *Bearbeitungszeit* müssen über das Disketten-Symbol gespeichert oder mit dem Rückgängig-Symbol verworfen werden:



### Vorspann-/Nachspann-Items

Ein zentrales Konzept für die Umsetzung von Zeitbegrenzungen in der *IRTlib Software* ist die Trennung der zeitbegrenzten Items, und zusätzlicher Assessmentinhalte, die *vor* oder *nach* dem zeitbegrenzten Teil administriert werden.

- Items die *nach* einem potentiell Zeitbegrenzten Abschnitt eines Erhebungsteils administriert werden, werden als *Nachspann-Items* bezeichnet.

## Eingebettete Programmhilfe

### 7.3.2 Items nach einer Zeitebengrenzung

Der *Erhebungsteile* erlauben die Definition von Items in verschiedenen Abschnitten. Items in diesem Abschnitt *Nachspann-Item(s)* werden nach den Items angezeigt, welche im Abschnitt *Items* eines *Erhebungsteils* definiert sind. Die Trennung in *Nachspann-Item(s)* und *Items* ist besonders sinnvoll, wenn unter *Bearbeitungszeit* eine Zeitbegrenzung aktiviert ist.

Um Items in dem Abschnitt *Nachspann-Item(s)* zu konfigurieren stehen die folgenden Optionen zur Verfügung:

- Hinzufügen: 
- Aktualisieren: 
- Vorschau/Preview: 
- Download/Export: 
- Löschen: 
- Sortieren: 

Hinweis: Änderungen an der Sicht *Nachspann-Item(s)* müssen über das Disketten-Symbol gespeichert oder mit dem Rückgängig-Symbol verworfen werden:



- Items die *vor* einem potentiell Zeitbegrenzten Abschnitt eines Erhebungsteils administriert werde, werden als *Vorspann-Items* bezeichnet.

## Eingebettete Programmhilfe

### 7.3.3 Items vor einer Zeitebengrenzung

Der *Erhebungsteile* erlauben die Definition von Items in verschiedenen Abschnitten. Items in diesem Abschnitt *Vorspann-Item(s)* werden vor den Items angezeigt, welche im Abschnitt *Items* eines *Erhebungsteils* definiert sind. Die Trennung in *Vorspann-Item(s)* und *Items* ist besonders sinnvoll, wenn unter *Bearbeitungszeit* eine Zeitbegrenzung aktiviert ist.

Um Items in dem Abschnitt *Vorspann-Item(s)* zu konfigurieren stehen die folgenden Optionen zur Verfügung:

- Hinzufügen: 
- Aktualisieren: 
- Vorschau/Preview: 
- Download/Export: 
- Löschen: 
- Sortieren: 

Hinweis: Änderungen an der Sicht *Vorspann-Item(s)* müssen über das Disketten-Symbol gespeichert oder mit dem Rückgängig-Symbol verwerfen werden:



## 7.4 Variablen

! Under Development

Diese Funktion ist gerade in Entwicklung.

💡 Eingebettete Programmhilfe

(Diese Funktionalität ist gerade noch in *Entwicklung*.)

## 7.5 Codebook

! Under Development

Diese Funktion ist gerade in Entwicklung.

💡 Eingebettete Programmhilfe

(Diese Funktionalität ist gerade noch in *Entwicklung*.)

## 7.6 ItemPool

! Under Development

Diese Funktion ist gerade in Entwicklung.

💡 Eingebettete Programmhilfe

(Diese Funktionalität ist gerade noch in *Entwicklung*.)

## 7.7 Routing innerhalb von Erhebungsteilen

Wenn *CBA ItemBuilder*-Tasks nicht in einer lineare Abfolge administriert werden sollen, die im Vorhinein feststeht und für alle Testpersonen identisch ist, dann kann die Funktion *Routing* der *IRTlib Software* verwendet werden.

Eine detaillierte Beschreibung zum *Routing innerhalb von Erhebungsteilen* findet sich hier in der eingebetteten Hilfe:

#### Eingebettete Programmhilfe

### 7.7.1 Zusammenfassung zu Routing innerhalb von Erhebungsteilen

Die Reihenfolge von *CBA ItemBuilder*-Aufgaben kann hier mit Hilfe von *Blockly* (also einer Form des visuellen Programmierens) definiert werden. *Blockly*-basierte Ablaufsteuerung ist verfügbar, wenn bei einem Erhebungsteil die Option *Routing aktivieren* ausgewählt ist. Die Option ist im Abschnitt *Info* eines Erhebungsteils zu finden. Ist sie aktiviert, enthält der Erhebungsteil den Eintrag *Routing*.

#### 7.7.1.1 Beispiele

Die Grundidee zur Verwendung von *Blockly* für die Definition von Abläufen in *computerbasierten Assessments* soll zunächst mit einigen Beispielen illustriert werden.

- **Beispiel für linearen Ablauf**

Basierend auf den einem Erhebungsteil hinzugefügten *CBA ItemBuilder Tasks* in der Ansicht *Items* entspricht eine lineare Folge der *Tasks* der folgenden *Blockly*-Definition:



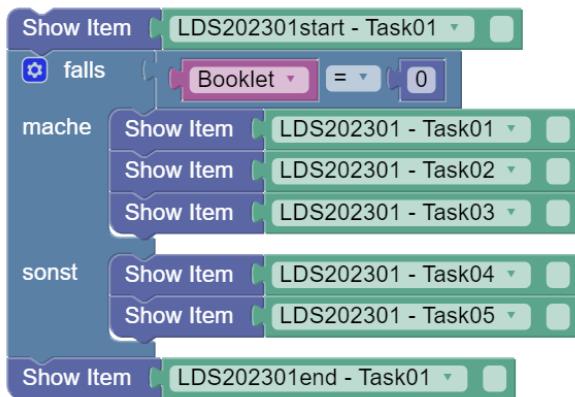
Dem *Blockly*-Element *Show Items* wird eine Liste der *CBA ItemBuilder Tasks* übergeben, die mit dem Operator *erzeuge Liste mit* erstellt wird. Die Liste wird in der dargestellten Reihenfolge abgearbeitet, wobei jeder *CBA ItemBuilder Tasks* solange dargestellt wird, bis das *NEXT\_TASK-Command* ausgeführt wird.

Eine äquivalente Formulierung einer linearen Sequenz kann auch mit mehreren *Show Items*-Blöcken erfolgen, wenn keine Zurücknavigation notwendig ist:



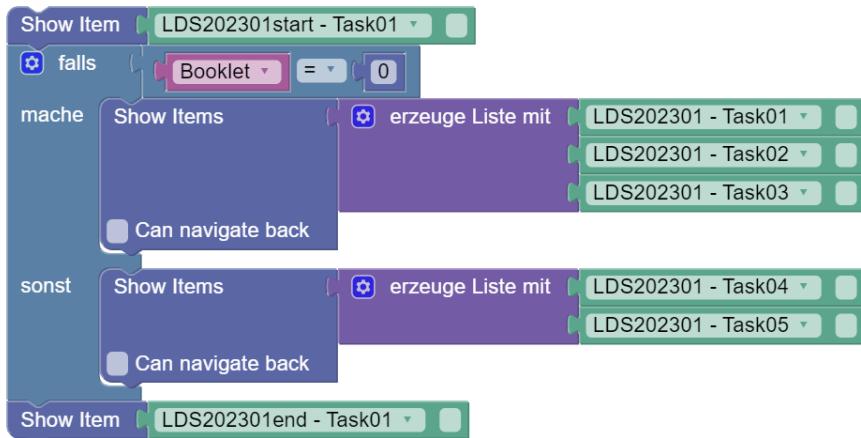
- **Beispiel für einfache Testhefte**

Mit Hilfe einer Variable (hier: *Booklet*) und einer einfachen *falls/mache-* Bedingung lässt sich daraus nun ein Ablauf definieren, welcher je nach Wert der Variable unterschiedliche Items administriert:



Die Items für Start und Ende werden immer administriert, die Tasks 1-3 nur, wenn die Variable *Booklet* der Wert *0* hat, die Tasks 4 und 5, wenn die Variable *Booklet* einen von *0* verschiedenen Wert hat.

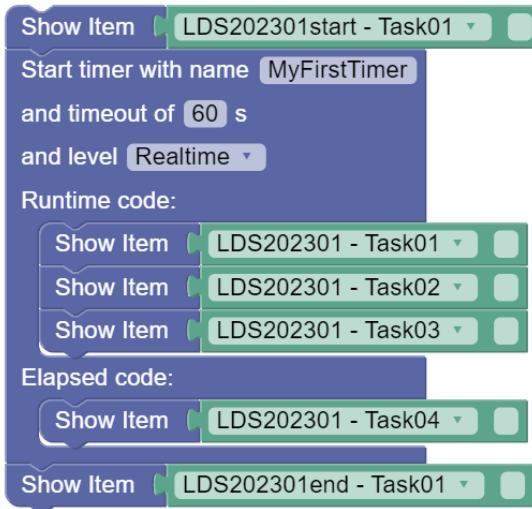
Der identische Ablauf lässt sich alternativ auch unter Verwendung des *Blockly*-Operators für das Anzeigen von Itemlisten erstellen:



Beide Varianten sind bzgl. der Funktionalität völlig äquivalent, die zweite Vorgehensweise mit Listen erlaubt aber die Verwendung der Option Zurück-navigation innerhalb der booklet-spezifischen Tasks.

- **Beispiel für Ablauf mit Zeitbegrenzung**

Um mit Hilfe der *Blockly*-Konfiguration zeitbegrenzte Abschnitte innerhalb eines Erhebungsteils umzusetzen, kann die folgende *Blockly*-Komponente verwendet werden:



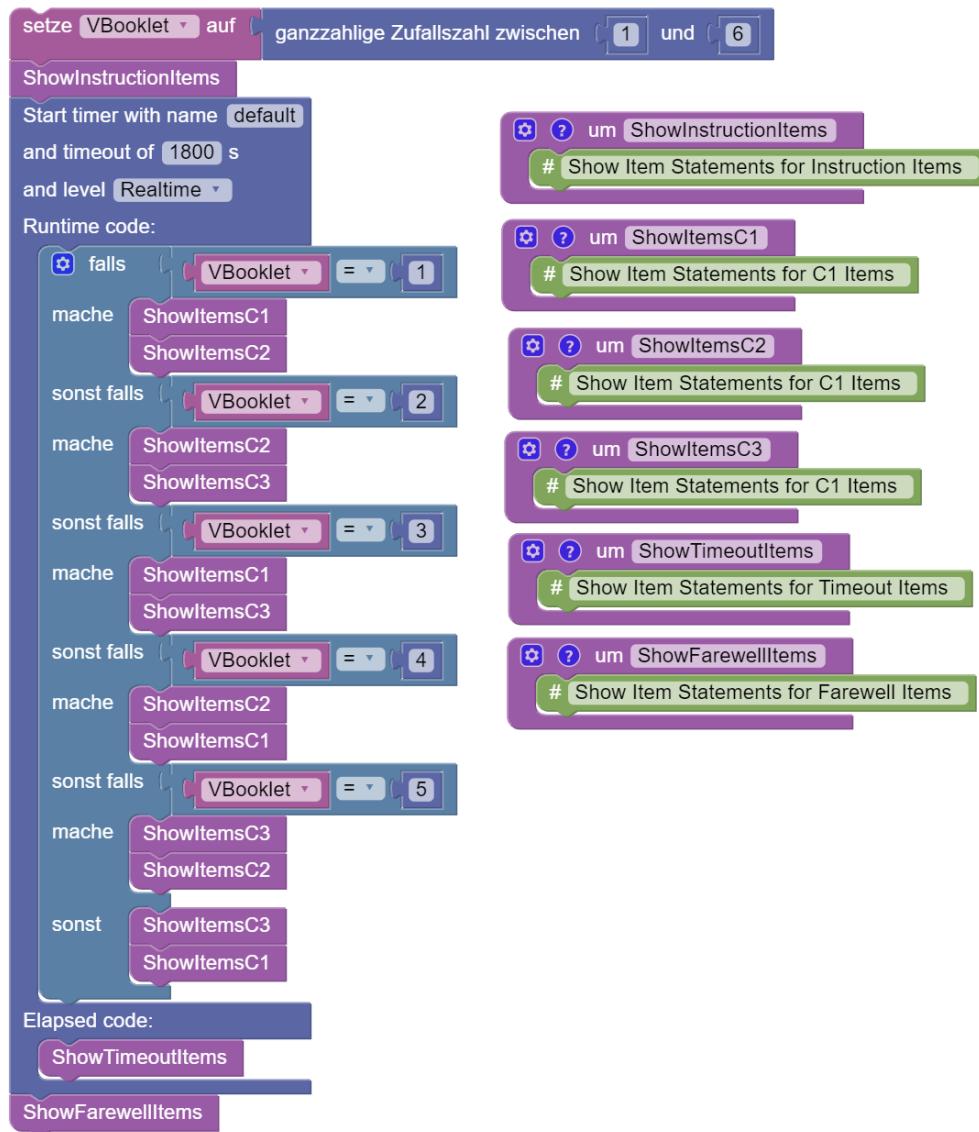
Jeder Ablauf beginnt mit einer nicht zeitbegrenzten Start-Tasks und endet mit einem ebenfalls nicht zeitbegrenzten End-Tasks. Dazwischen läuft eine Zeitbegrenzung für einen Abschnitt mit der Bezeichnung *MyFirstTimer*, der eine Zeitbegrenzung für 60 Sekunden hat.

Die Tasks 1, 2 und 3 werden in dem Abschnitt *Runtime code* mit einer Zeitbegrenzung angezeigt. Tritt ein Timeout auf, d.h. werden die drei Tasks nicht innerhalb der 60 Sekunden bearbeitet, wird (ebenfalls ohne Zeitbegrenzung) der Task 4 angezeigt.

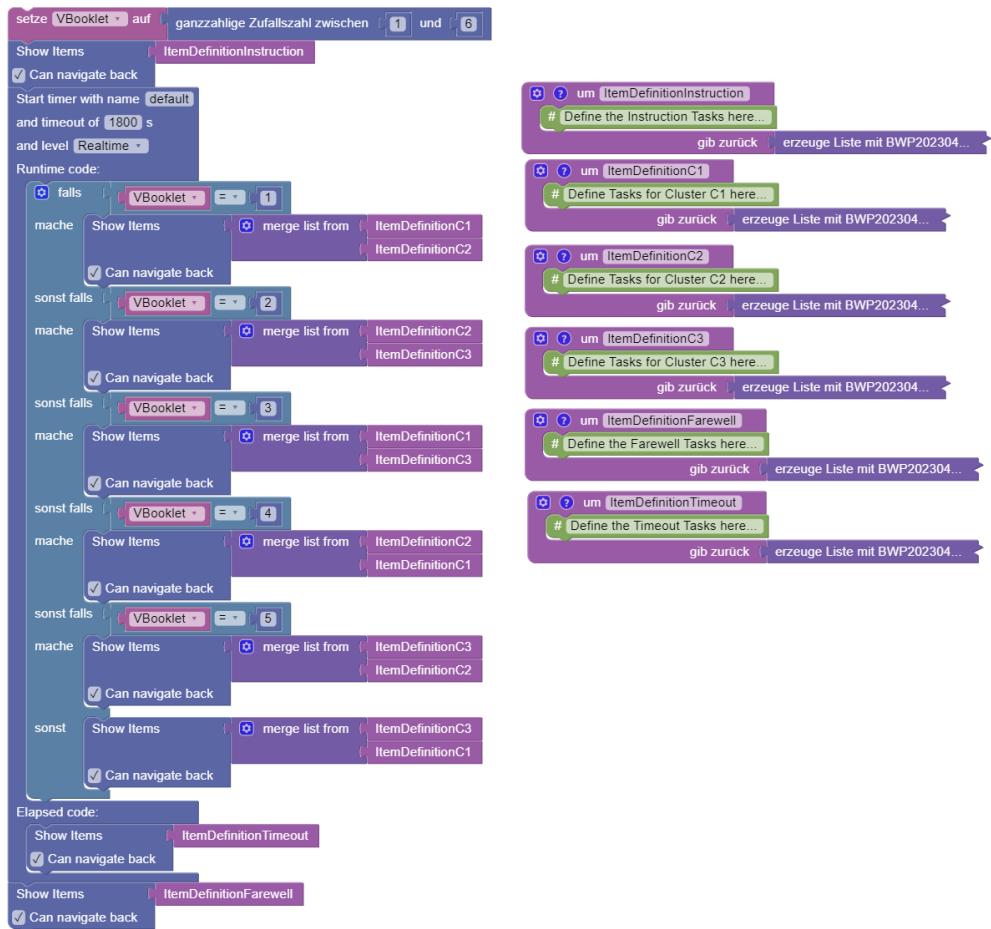
- **Beispiel für einfaches Booklet-Design mit Zeitbegrenzung**

Bei vielen Items kann die Definition von *Booklet Designs*, d.h. Taskreihenfolgen mit balancierten Positionen, durch Funktionen bzw. Listen vereinfacht werden.

Wenn keine Zurücknavigation notwendig ist, können Funktionen für die Definition von Clustern verwendet werden:



Mit Zurücknavigation können die Funktionen Listen von Tasks zurückgeben:



Weitere Informationen siehe [hier](#).

#### 7.7.1.2 Hinweise zur Verwendung des *Blockly*-Editors

Die Definition von Abläufen erfolgt in dem visuellen *Blockly*-Editor. Die Ausführung beginnt mit dem Element, welches am weitesten oben ausgerichtet ist. Wenn nötig, kann der Arbeitsbereich mit der Funktion Aufräumen automatisch ausgerichtet werden. Zum Hinzufügen von *Blockly*-Operatoren können diese per Drag-and-Drop aus der Palette gezogen werden.

- **Löschen:** Zum Löschen von Operatoren können diese auf den Papierkorb gezogen werden. Ausgewählte *Blockly*-Elemente können auch über die Taste *Entf(ernen)* gelöscht werden. Alternativ können ausgewählte *Blockly*-Elemente auch über Kontextmenü gelöscht werden.

- **Redo-/Undo:** Innerhalb des *Blockly*-Editor können einzelne Aktionen rückgängig gemacht werden. Dafür kann die Tastenkombination **Strg + Z** verwendet werden. Mit **Strg + Y** wird eine Aktion wiederholt. Durch einen Klick in einen leeren Bereich des *Blockly*-Editors ist der Zugriff auf ein Kontextmenü möglich, welches ebenfalls die Optionen für *Rückgängig* und *Wiederholen* bereithält:



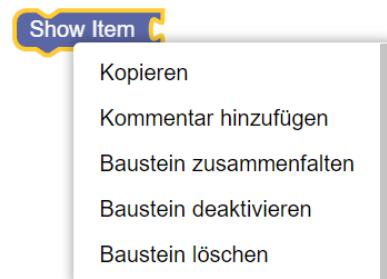
- **Speichern:** Anpassungen im *Blockly*-Editor müssen gespeichert werden. Dafür steht unten rechts das Disketten-Symbol zur Verfügung:



Sollen die Änderung (insgesamt) verworfen werden, kann unten rechts die das Verwerfen-Symbol verwendet werden.

- **Zoom:** Die Ansicht im Arbeitsbereich kann mit den Icons + vergrößert und mit - verkleinert werden.
- **Kontextmenü:** Weitere Optionen sind über die rechte Maustaste (Kontextmenü) im *Blockly*-Editor verfügbar. Um diese Funktionen aufzurufen, muss auf ein *Blockly*-Element ein Sekundärklick (rechte Maustaste) durchgeführt werden:
  - Kopieren dupliziert das ausgewählte *Blockly*-Element, inklusive aller verbunder Elemente.

- Kommentieren von Blöcken ist möglich.
- Blöcke können deaktiviert/aktiviert werden.
- Manche Block-Typen erlauben die Darstellungsform extern/intern zu wechseln.
- Blöcke, welche weitere Blöcke enthalten, können zusammengefalten/entfaltet werden.
- Das Löschen von Blöcken ist auch über das Kontextmenü möglich.



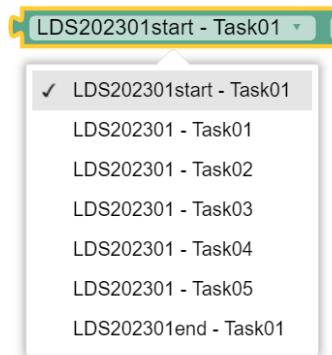
Einige *Blockly*-Elemente stellen im Kontextmenü auch einen Eintrag *Hilfe* zur Verfügung, welcher auf die allgemein zugängliche *Blockly*-Dokumenten (<https://github.com/google/blockly/wiki/>) verweist.

### 7.7.2 Verwendung von *Blockly* zur Ablaufsteuerung

Die Grundfunktionen für die Nutzung der *Blockly*-Umgebung zur Steuerung von Assessments finden sich im Abschnitt *Session*.

#### 7.7.2.1 Einzelne Items anzeigen

Auf *CBA ItemBuilder*-Tasks, die in der Ansicht *Items* für einen Erhebungsteil importiert wurden, kann in der Ablaufsteuerung wie in den Beispielen oben gezeigt mit Hilfe des folgenden *Blockly*-Elements für *Tasks* zugegriffen werden:



Das Element, welches im Abschnitt *Session* der Palette des *Blockly*-Editors zu finden ist, kann durch die Auswahlliste konfiguriert werden. Jedes *Blockly*-Elements für Tasks kann auf genau einen konkreten Task verweisen, d.h. in der Regel besteht eine Ablaufdefinition aus mehrerer solcher Elemente.

*Blockly*-Elemente für Tasks können nicht direkt in den Ablauf eingefügt werden, sondern werden zusammen mit einem *Show Item*-Element verwendet:



Das Beispiel für einfache Testhefte illustriert, dass Abläufe in der *Blockly*-Definition häufig durch eine Abfolge von mehreren *Show Item*-Operatoren definiert werden. *Show Item*-Operatoren können dabei in Bedingungen und Schleifen, sowohl innerhalb des Hauptablaufs als auch innerhalb von Funktionen eingefügt werden.

#### 7.7.2.2 Verwendung von Geltungsbereichen (Scopes)

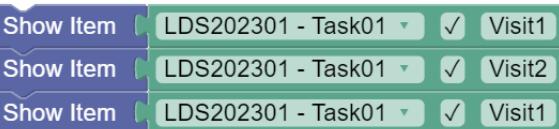
Mit Hilfe der *Blockly*-basierten Ablaufsteuerung ist es auch möglich, *CBA ItemBuilder*-Tasks mehrfach innerhalb eines Ablaufs zu administrieren:



Dabe wird beim erneuten Aufruf eines Items der Zustand aus dem letzten Besuch wiederhergestellt, d.h. die Bearbeitung wird fortgesetzt. Sollen Items mehrfach neu, d.h. unbearbeitet vorgelegt werden, kann das automatische wiederherstellen nicht gewünscht sein. Dafür kann optional die Checkbox für die Angabe eines *Scopes* (Geltungsbereich) aktiviert werden:



Wird nichts weiter angegeben, wird das Item im "Default"-Scope administriert. Alternativ kann ein Text definiert werden, wie in folgendem Beispiel zu sehen:

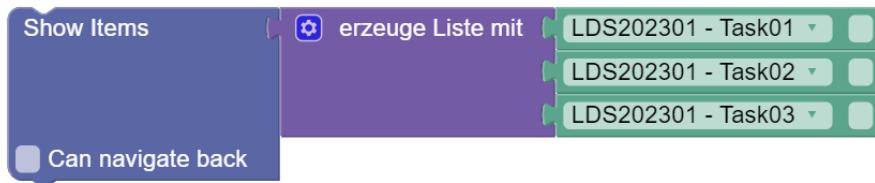


Beim ersten Besuch wird der Task in dem *Scope* "Visit1" dargestellt. Danach folgt eine neue, unabhängige Darstellung des Tasks in einem anderen *Scope* ("Visit2"). Im dritten Aufruf wird der Task weder mit den Daten dargestellt, die beim ersten Besuch bereits gesammelt wurde (d.h. der *Scope* "Visit1" wird erneut verwendet).

### 7.7.2.3 Anzeigen mehrerer Items (Itemlisten)

Wie im Beispiel für linearen Ablauf zu sehen, können lineare Tests auch über Listen von Tasks dargestellt werden.

Listen können mit dem *Blockly*-Operator *Show Items* verwendet werden:



- **Zurücknavigation:** Das *Show Items*-Element für Listen kann über die Eigenschaft *Can navigate back* konfiguriert werden. Ist diese Eigenschaft ausgewählt, dann können *CBA ItemBuilder-Tasks* mit dem *Command BACK\_TASK* eine Navigation zum vorherigen *CBA ItemBuilder Tasks* anfordern.
- **Abbrechen von Listen:** Die Verwendung von Listen erlaubt auch das Abbrechen von Listen. Listen können über zwei Wege abgebrochen werden:
  - Das *Command CANCEL\_TASK*, welches innerhalb von CBA ItemBuilder Tasks verwendet werden kann, wird aufgerufen.
  - Im Testleitermenü, welches für die Studie konfiguriert und ggf. über den *Blockly*-Operator Testleitermenü bearbeiten angepasst wurde, wird die Funktion *Itemliste abbrechen* aufgerufen.

Die Adminsitration einer Itemliste wird dadurch abgebrochen, und die Abarbeitung des *Blockly*-Ablaufs nach dem *Show Items*-Block fortgesetzt.

### 7.7.2.4 Anzeige von Items mit Speicherung der Ergebnisse

Die Operatoren *Show Item* (für einzelne Items) und *Show Items* (für Itemlisten) sind auch als Operatoren für Wertzuweisungen verfügbar:

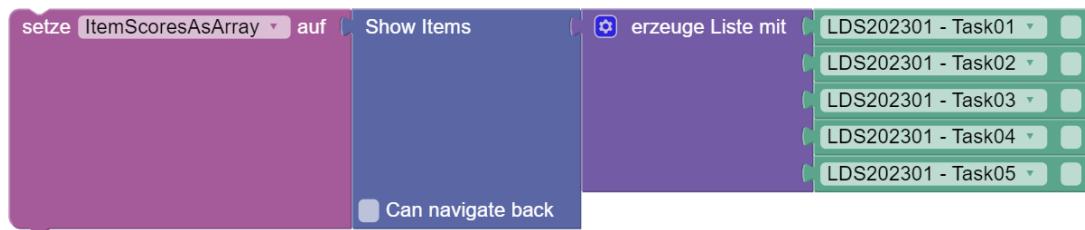


Mit deren Hilfe lassen sich Ergebnisse der Itembearbeitung zu Variablen (String oder Array) zuweisen, und dann für die Ablaufsteuerng auswerten.

- Einzelner Task:



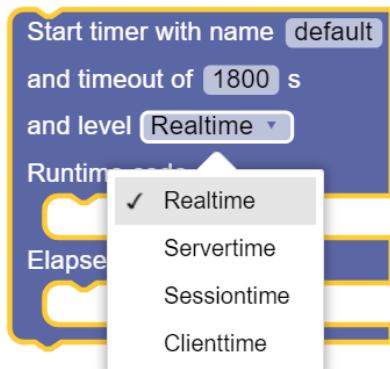
- Liste von Tasks:



#### 7.7.2.5 Definition von Zeitbegrenzungen

Wie im Beispiel Ablauf mit Zeitbegrenzung bereits illustriert, kann mit dem *Blockly*-Block *Start time with name* die Zeitbegrenzte Administration von Items umgesetzt werden.

Das *Blockly*-Element *Start timer with name* erlaubt die Definition von Zeitbegrenzungen. Jede Zeitbegrenzung kann einen eigenen Namen haben. Zusätzlich muss die Zeit in Sekunden angegeben werden. Darüber kann definiert werden, welche Art von Zeit verwendet werden soll:



- Realtime:
- Servertime:
- Sessiontime:
- Clienttime:

Schließlich können zwei Stellen mit weiteren *Blockly*-Operatoren (wie bspw. ein oder mehrere Show Item-Blöcke zum Anzeigen einzelner Items oder ein oder mehrere Show Items-Blöcke zum Anzeigen von Listen) gefüllt werden:

- Runtime code: Diese Blöcke werden ausgefüllt, bis die definierte Zeit abgelaufen ist.
- Elapsed code: Diese Blöcke werden nur ausgefüllt, wenn der *Runtime code* nicht innerhalb der Zeit beendet wurde.

#### 7.7.2.6 Blockly-Operatoren für das Testleitermenü

In der Studiendefinition können Funktionen des Testleitermenüs für eine oder mehrere Rollen angelegt werden. Rollen stellen unterschiedliche Funktionen zusammen, die mit Hilfe des vom Testleiter einzugebenden Passworts unterschieden werden können.

**Anpassen von Standardfunktionen:** Folgende Standardfunktionen können für eine Studie im Abschnitt *Info / Testleitermenü* definiert werden:

- *Navigation:* Aufgabe vor / Aufgabe zurück
- *Listen:* Itemliste abbrechen
- *Beenden:* Erhebungsteil beenden und Session beenden
- *Lautstärkeregelung:* Einstellen der Audiolautstärke während des Assessments

Während der Bearbeitung eines Erhebungsteils kann in der Ablaufsteuerung mit Hilfe des folgenden *Blockly*-Operators das Testleitermenü kontextspezifisch angepasst werden:



Das Testleitermenü kann für jede der Standardfunktionen (im Bereich *Funktion*) für eine Rolle (im Bereich *Gruppe*) sowohl die Beschriftung der Schaltfläche (im Bereich *Label*) geändert werden:

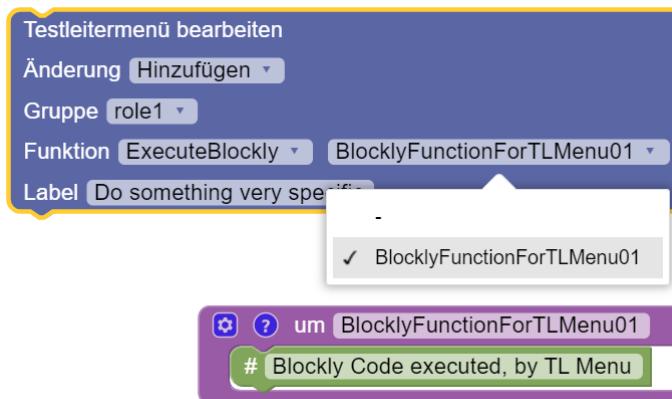
- *Hinzufügen:* Funktion wird im Testleitermenü ergänzt
- *Entfernen:* Funktion wird aus dem Testleitermenü entfernt
- *Deaktivieren:* Funktion wird im Testleitermenü deaktiviert
- *Aktivieren:* Funktion wird im Testleitermenü aktiviert



Der Aufruf dieses *Blockly*-Operators im Testablauf definiert das Verhalten des Testleitermenüs im weiteren Testablauf. Im Unterschied zu *Entfernen*

bleiben *deaktivierte* Funktionen im Testleitermenü sichtbar, können aber (bis sie wieder *aktiviert* werden) nicht ausgeführt werden.

**Verwenden von *Blockly*-Funktionen im Testleitermenü:** Der *Blockly*-Operator für das Bearbeiten des Testleitermenüs enthält im Abschnitt *Funktion* auch die Option zum Ausführen von *Blockly*-Code (*ExecuteBlockly*):

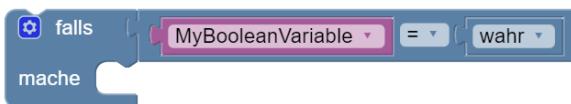


Wenn als *ExecuteBlockly* ausgewählt ist, dann kann in dem *Blockly*-Element *Testleitermenü bearbeiten* eine innerhalb des *Blockly*-Editors definierte Funktionen ausgewählt werden. Die in dieser Funktion definierten *Blockly*-Operatoren werden dann ausgeführt, wenn ein Testleiter die entsprechende Schaltfläche zur Laufzeit im Testleitermenü auswählt.

### 7.7.3 Fortgeschrittene *Blockly*-Verwendung

#### 7.7.3.1 Ablaufsteuerung mit Bedingungen

Der Abschnitt *Logic* enthält den *Blockly*-Operator *falls/mache*, welcher zur Umsetzung von Bedingungen im Ablauf verwendet werden kann. Bedingungen sind logische Ausdrücke, bspw. die Prüfung ob eine Preload-Variable einen bestimmten Wert hat:



Nur wenn die Bedingung (*falls*) erfüllt ist, werden die *Blockly*-Operatoren ausgeführt, welche innerhalb des Bedingungsblocks definiert sind (d.h. neben *mache*). In dem Beispiel wird geprüft, ob eine boolsche Variable den Wert *wahr* hat.

Die Bedingung wird dabei als separater Block definiert, die mit dem *Blockly*-Operator *falls/mache* verbunden ist. Hier die beiden Komponenten separat:

- Bedingung:



- Logischer Ausdruck:



#### 7.7.3.2 Verwendung logischer Ausdrücke

Logische Ausdrücke in Bedingungen basieren entweder auf Wertevergleichen oder Rückgaben von Funktionen. Wertevergleiche können mit folgendem Blockly-Element realisiert werden:



Die beiden Slots können mit Werten gefüllt werden. Für boolsche Werte (wahr/falsch) steht ein entsprechendes Blockly-Element im Abschnitt *Logic* bereit:



Bedingungen sind auch mit Variablen von anderem Datentyp möglich:

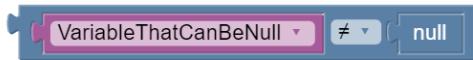


Für numerische Werte gibt es ein entsprechendes Blockly-Element im Abschnitt *Math*, welcher Operatoren für Zahlen und einfache mathematische Operationen enthält:

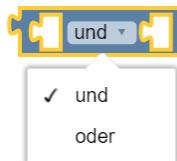


Mit dessen Hilfe und einer numerischen Variable lässt sich folgende Bedingung formulieren:

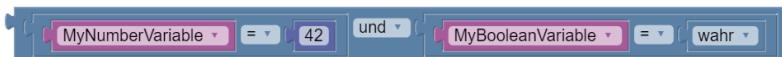
Aus technischen Gründen kann es auch notwendig sein zu prüfen, ob eine Variable noch gar keinen Wert hat. Das kann durch Verwendung der Blockly-Komponente **null** umgesetzt werden:



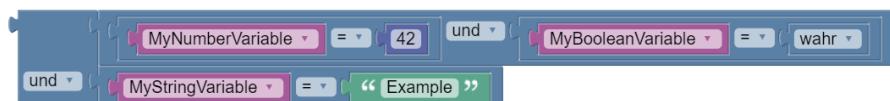
**Kombination von logischen Ausdrücken:** Einzelne Bedingungen oder logische Ausdrücke können mit folgendem *Blockly*-Element aus dem Abschnitt *Logik* verbunden werden:



Dabei steht eine *und* sowie eine *oder*-Verknüpfung der Aussagen zur Auswahl.  
Die freien *Eingänge*

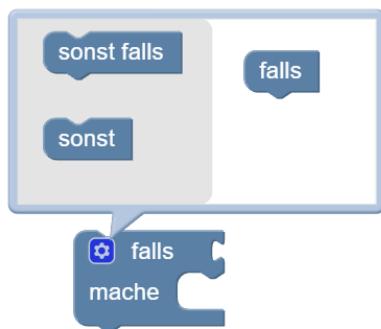


Mehrere *logische Ausdrücke* können in einander geschachtelt werden:



Hinweis: Für eine übersichtlichere Darstellung ist bei der äußeren *und*-Verknüpfung die externe Darstellung gewählt.

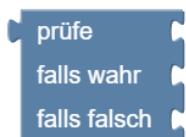
**Mehrere Bedingungen (*sonst falls / sonst*):** Durch Klick auf das kleine Zahnrad-Symbol eines Bedingungsblocks (*falls/mache*) kann dieser konfiguriert werden:



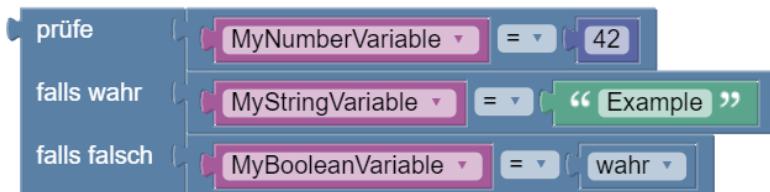
Durch das Hinzufügen eines Abschnitts *sonst falls* können kann eine weitere Bedingung hinzugefügt werden. Die in einem *sonst falls* Abschnitt definierte Bedingung wird geprüft, wenn die vorherigen Bedingungen (*falls*) nicht erfüllt sind. Ist eine Bedingung erfüllt, werden die definierten Blockly-Operatoren ausgeführt.

Durch das Hinzufügen eines Abschnitts *sonst* können Blöcke hinzugefügt werden, welche dann ausgeführt werden wenn keine der Bedingungen erfüllt ist.

**Spezialfall: *prüfe*-Operator für drei Bedingungen:** Für drei Bedingungen stellt der *Blockly*-Editor eine speziellen Operator *prüfe-falls wahr-falls falsch* zur Verfügung:



Der Operator kombiniert zwei logische Ausdrücke, z.B.:



Das Konstrukt ist eine Kurzform für folgende Prüfung, wie sie in folgender Tabelle dargestellt ist:

MyNumberVariable	MyStringVariable	MyBooleanVariable	Ergebnis
= 42	= Example	(any)	true
= 42	≠ Example	(any)	false
≠ 42	(any)	true	true
≠ 42	(any)	false	false

Ohne den Operator für drei Bedingungen könnte die gleiche Prüfung mit folgender Kombination umgesetzt werden:



**Negation:** Um einen logischen Ausdruck umzukehren (Negation) steht folgender *Blockly*-Operator zur Verfügung:



### 7.7.3.3 Ablaufsteuerung mit Schleifen

Die mehrfache Ausführung von *Blockly*-Operatoren (und der damit darstellbaren Aktionen) ist mit Schleifen möglich. Der Abschnitt *Loops* der *Palette* enthält die dafür notwendigen *Blockly*-Elemente.

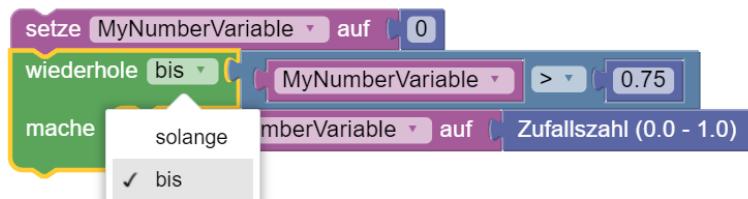
**Wiederhole n-mal:** Der folgende *Blockly*-Operator kann verwendet werden, um die Ausführung von Blöcke n-mal zu wiederholen:



**Wiederhole solange:** Schleifen können auch solange wiederholt werden *bis* eine Bedingung zutrifft (oder *solange* eine Bedingung zutrifft):



Beispiel:



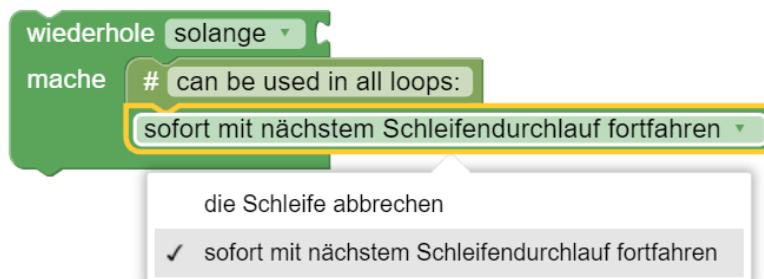
**Zähle von/bis:** Schleife mit Hilfsvariablen:



**Für jeden Wert aus Liste:** Schleife über alle Werte einer Liste:



**Schleifen vorzeitig abbrechen:** Folgendes *Blockly*-Element kann genutzt werden, um eine Schleife (vorzeitig) abzubrechen oder um vorzeitig mit dem nächsten Schleifendurchlauf zu beginnen:



#### 7.7.3.4 Operatoren für Zahlen und einfache mathematische Funktionen

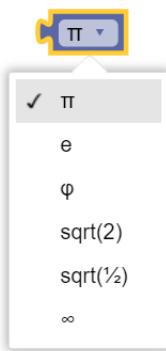
Der Abschnitt *Math* der *Palette* enthält *Blockly*-Elemente zur Verwendung von Zahlen und einfachen mathematischen Funktionen.

##### Ausdrücke

- Zahlen: Ganzzahlen / Dezimalzahlen

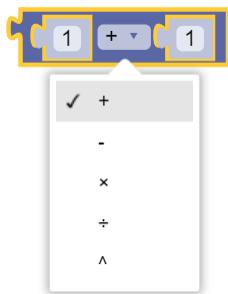


- Symbole: Spezielle Symbole oder Konstanten:



## Basale Funktionen

- Addition, Subtraktion, Multiplikation, Division und Potenzfunktion von zwei Argumenten:



Schachtelung ist möglich, z.B.:



- Division mit Rest:



- Ob eine Zahl gerade ist, kann mit diesem *Blockly*-Element geprüft werden:



- Mit dem folgenden *Blockly*-Element, kann eine Zahl auf einen Bereich begrenzt werden:



## Eingebaute Funktionen

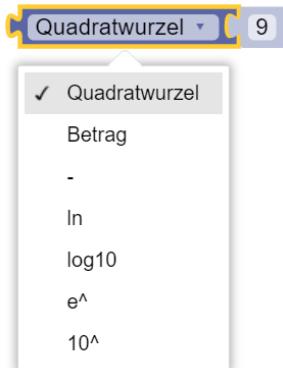
- Trigonometrische Funktionen:



- Runden von Werten:



- Weitere Funktionen:



**Erzeugung von Zufallszahlen:** Für die Erstellung von Zufallszahlen stehen zwei *Blocky*-Elemente zur Verfügung:

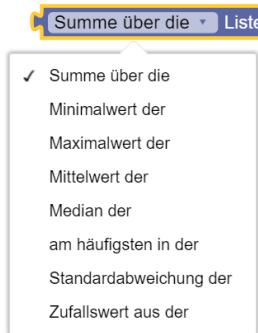
- Ganzzahlen (in Wertebereich):



- Zufallszahl zwischen 0 und 1:



**Numerische Funktionen für Listen:** Vordefinierte Funktionen für Listen umfassen:



Hinweise:

- Weitere Funktionen lassen sich, wenn benötigt, mit Schleifen für Listen umsetzen.
- Bei der Verwendung der Funktionen ist darauf zu achten, dass die ausgewählte Funktion für die Datentypen der List anwendbar ist.

#### 7.7.3.5 Operatoren für Text und einfache String-Operationen

Der Abschnitt *Text* der *Palette* enthält *Blockly*-Elemente zur Verwendung Zeichenketten. **Ausdrücke:** Zum erstellen von Text steht folgender Operator zur Verfügung:

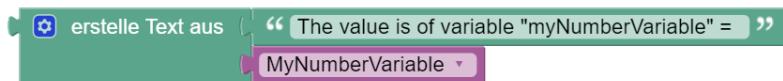


**Verketten:** Verschiedene Operatoren können verwendet werden, um Text zusammenzufügen und zu Variablen zuzuweisen:

- Einen Text an eine Variable anfügen:



- Texte (und Variablenwerte) verketten und an andere *Blockly*-Operatoren weitergeben:



- Zusammengefügten Texte einer Variable zuweisen:



**Textlänge:** Die Länge einer Zeichenkette kann mit folgendem *Blockly*-Operator ermittelt werden:



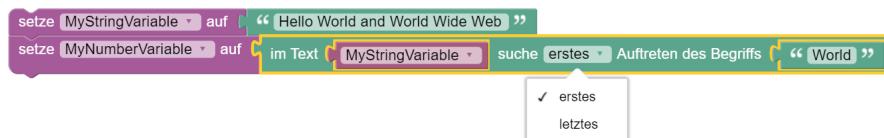
**Prüfung auf leeren String:** Leere String-Variablen können daran erkannt werden, dass die Anzahl der Zeichen 0 ist.



Alternativ kann dafür auch der folgende *Blockly*-Operator verwendet werden:



**Position in String finden:** Ein Operator, der *im Text* (der per Variable oder als Ausdruck übergeben wird) das *erste* oder *letzte* Auftreten eines *Begriffs* sucht, kann wie folgt verwendet werden:



Zurückgegeben wird dabei die Position des *Begriffs* innerhalb der Zeichenkette (d.h. *im Text*).

**Teilzeichenketten bilden:** Der folgende Operator nimmt aus der übergebenen Zeichenkette *im Text* die ersten Buchstaben. Die Anzahl der Buchstaben wird dabei ebenfalls übergeben.

- Beispiel (hier wird, wenn die Option *nimm ersten* ausgewählt ist der Variable `MyStringVariable` der Text ABC, d.h. die ersten drei Buchstaben der Zeichenkette ABCDEFG) zugewiesen:



Buchstaben aus einer Zeichenkette kann man auch mit folgendem Operator entnehmen, und bspw. einer Variablen zuweisen:

- Beispiel (hier können bspw. die Zeichen 3 bis 5 aus einer Zeichenkette entnommen werden):

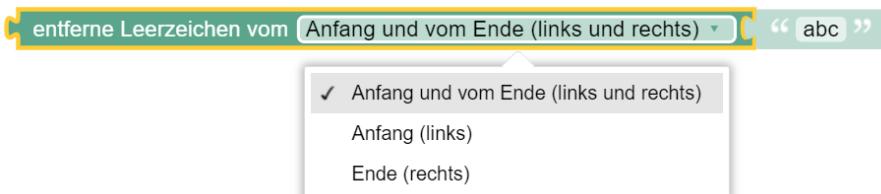


**Texte Verändern:** Vorhandene Texte (entweder als Ausdrücke oder aus Variablen vom Datentyp String) können durch die Anwendung von Operatoren verändert werden.

- Folgender Operator kann verwendet werden, um Text in Grossbuchstaben oder in Kleinbuchstaben umzuwandeln:



- Führende, abschließende oder führende und abschließende Leerzeichen können durch folgenden Operator entfernt werden:



### 7.7.3.6 Operatoren für Zeiten und einfache Zeit-Operationen

Der Abschnitt *Date & Time* der *Palette* enthält *Blockly*-Elemente zur Verwendung Zeiten innerhalb von Ablaufdefinitionen.

**Festhalten von Zeitpunkten:** Variablen vom Datentyp *DateTime* können Zeitstempel zugewiesen werden.

**Ermitteln von Zeitdifferenzen:** Vollständiges Beispiel: Folgender *Blockly*-Code misst die Zeit für die Bearbeitung von Task 1 bis 4. Dafür wird zunächst der Startzeitpunkt festgehalten, und nach der Bearbeitung der Aufgaben wird die Zeitdifferenz ermittelt und in Sekunden umgewandelt:



### Umrechnen von Zeitmaßen



### 7.7.3.7 Operatoren für Listen

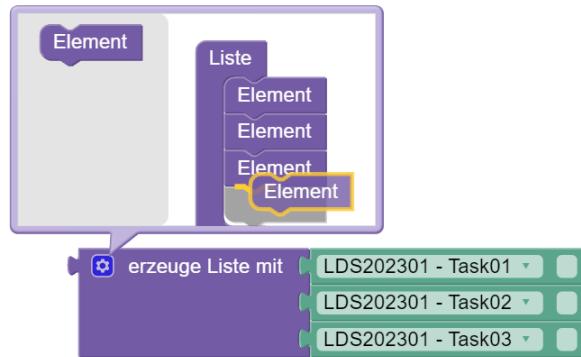
Der Abschnitt *Lists* der *Palette* enthält *Blockly*-Elemente zur Erstellung und Verwendung von Listen.

**Liste erstellen:** Es stehen verschiedene Optionen zur Verfügung, wie Listen erstellt werden können.

- Listen können aus bestehenden Elementen erstellt werden:



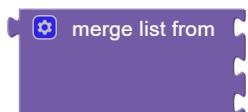
Die Anzahl der Elemente des Operators *erzeuge Liste mit* kann per Drag-and-Drop konfiguriert werden, nachdem Zahnrad-Symbol angeklickt wurde:



- Listen können durch Wiederholung eines Elements erstellt werden:



**Verbinden von Listen:** Bestehende Listen können zusammengeführt werden mit folgenden Operator:



**Teillisten:** Aus Listen kann mit folgendem Operator eine Teilliste ausgewählt werden:



Weitere Optionen des Operators für *bis*: *bis von hinten* und *bis letztes*.

**Eigenschaften von Listen:** Folgende Operatoren stehen zur Verfügung um Eigenschaften einer Liste abzufragen:

- Folgender Operator gibt *wahr* zurück, wenn die verbundene Liste leer ist:



- Folgender Operator gibt die Länge der Liste zurück:

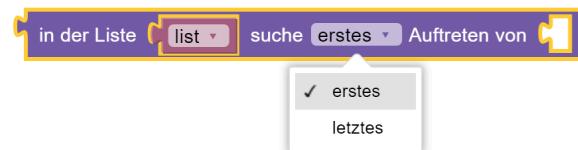


- Folgender Operator gibt die distinkte Elemente einer Liste zurück



**Suchen und Ersetzen:** Folgende Operatoren stehen zum Suchen und Ersetzen von Elementen in Listen zur Verfügung:

- Folgender Operator findet Elemente in Listen:



- Folgender Operator gibt / entfernt oder ersetzt in einer Liste und gibt das Element zurück:



Weitere Optionen des Operators für *das*: *von hinten das* / *Erste* / *Letzte* und *Zufällig*.

- Folgender Operator ersetzt und fügt in einer Liste ein:



Weitere Optionen des Operators für *das*: *von hinten das* / *Erste* / *Letzte* und *Zufällig*.

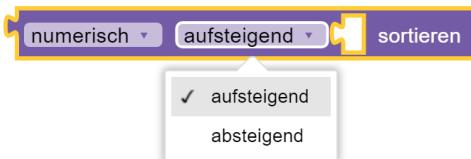
**Umwandlung von Listen und Text:** Liste und Text können über Trennzeichen umgewandelt werden.

- Folgender Operator erstellt einen Text aus einer Liste oder eine Liste aus einem Text:



**Listen Sortieren:** Elemente in Listen können auch sortiert werden.

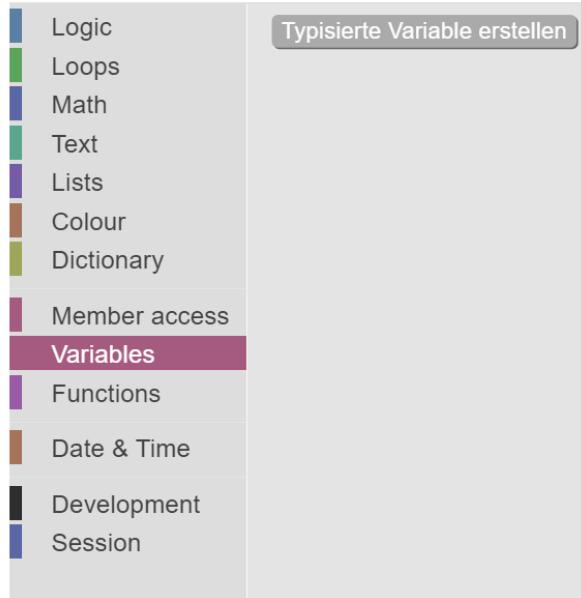
- Folgender Operator gibt die distinkte Elemente einer Liste zurück:



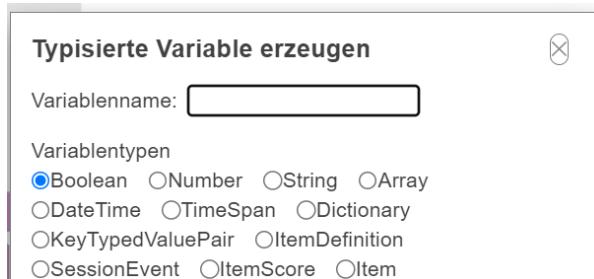
#### 7.7.3.8 Blockly-Variablen

Der Abschnitt *Variables* der *Palette* enthält *Blockly*-Elemente zur Erstellung und Verwendung von Variablen.

**Variable Erstellen:** Um eine *Blockly*-Variable zu erstellen enthält die *Palette* die *Typisierte Variable Erstellen*:



- *Blockly*-Variablen haben immer einen *Variablenname* und *Datentyp*:



**Einfache Datentypen und Wertzuweisungen:** Folgende basale Datentypen werden unterstützt:

- *Boolean*: Logische Wahrheitswerte und Logische Ausdrücke (`wahr` oder `falsch`)

`set MyBooleanVariable to wahr`

- *Number*: Datentyp für Zahlenwerte (mit und ohne Dezimalstelle)

`set MyNumberVariable to 22`

- *String*: Textwerte bzw. Zeichenketten

```
set [MyStringVariable v] to ["New Value"]
```

Für Zeiten werden folgende Datentypen bereitgestellt:

- *DateTime*: Datum und Uhrzeit

```
setze [StartTimeSection1 v] auf [Date 1 . 1 . 2020 Time 12 : 0 : 0]
```

- *TimeSpan*: Zeitspanne

```
setze [TimeSpanSection1 v] auf []
```

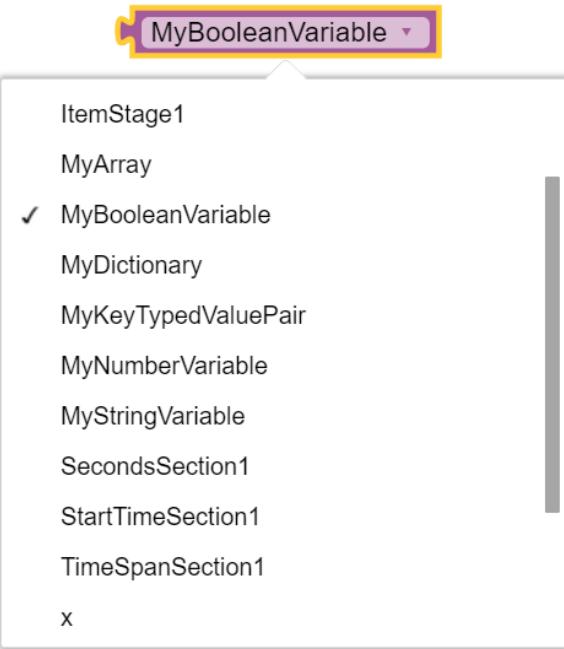
**Datentypen für mehrere Werte:** Neben den basalen Datentypen werden auch Datentypen für mehrere Werte unterstützt:

- *Array*: Datentyp für Listen

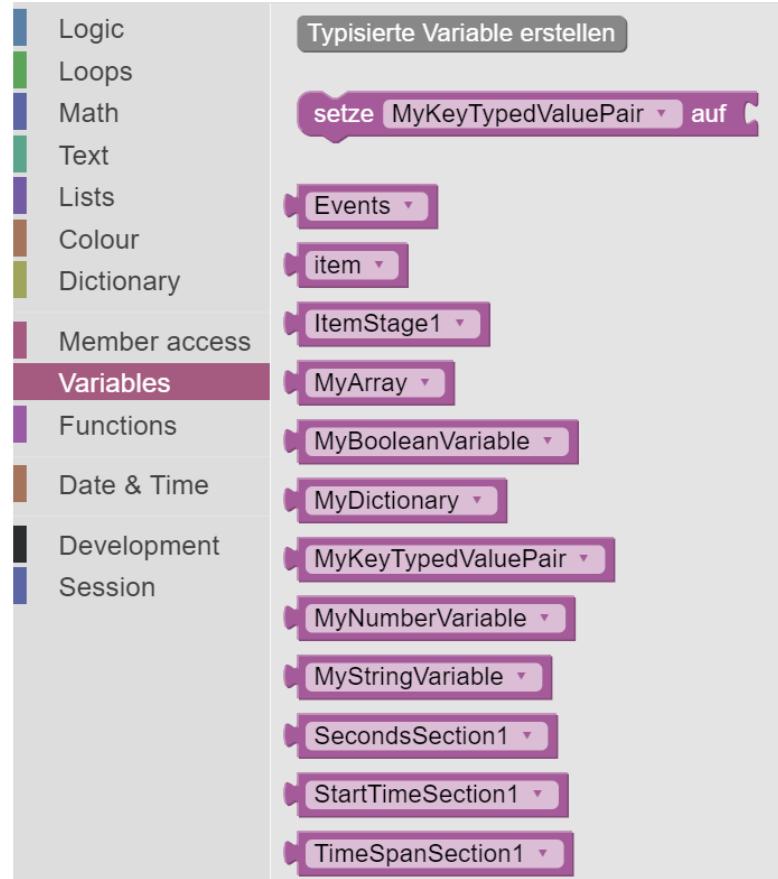
```
setze [MyArray v] auf [erzeuge Liste mit [0 , 8 , 15]]
```

- *Dictionary*: (Dokumentation fehlt)
- *KeyTypedValuePairs*: (Dokumentation fehlt)

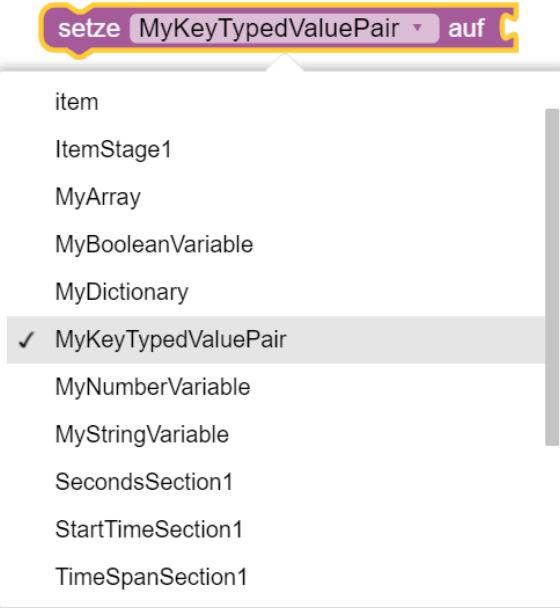
**Variablenwerte Verwenden:** Für die Verwendung von Variablenwerten, können Blockly-Elemente mit *Eingängen* folgende Komponente aufnehmen:



- Welche Variable verwendet wird, kann dabei ausgewählt werden. Für definierte Variablen findet sich dafür jeweils auch ein *Blockly*-Element im Abschnitt *Variables* der *Palette*:



- In der Palette findet sich auch ein *Blockly*-Element vom Typ *setze ... auf*. In diesem kann ebenfalls ausgewählt werden, den Wert welcher Variable es setzt:

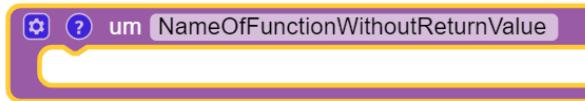


#### 7.7.3.9 Blockly-Funktionen

Der Abschnitt *Functions* der *Palette* enthält *Blockly*-Elemente zur Verwendung von Funktionen innerhalb von Ablaufdefinitionen. Funktionen kombinieren *Blockly*-Code, so dass dieser nur einmal definiert aber mehrfach verwendet werden kann.

**Definieren von Funktionen:** Es können zwei verschiedene Formen von Funktionen definiert werden.

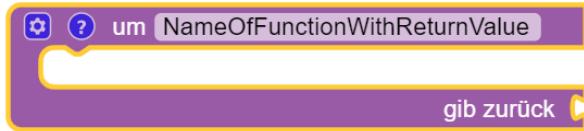
- Funktionen ohne Rückgabewert:



Funktionen ohne Rückgabewert können, damit sie augerufen werden, im Ablauf einfach mit vorherigen und nachfolgenden *Blockly*-Elementen verbunden werden (d.h. sie haben eine Verbindung nach oben und unten):

NameOfFunctionWithoutReturnValue

- Funktionen mit Rückgabewert:

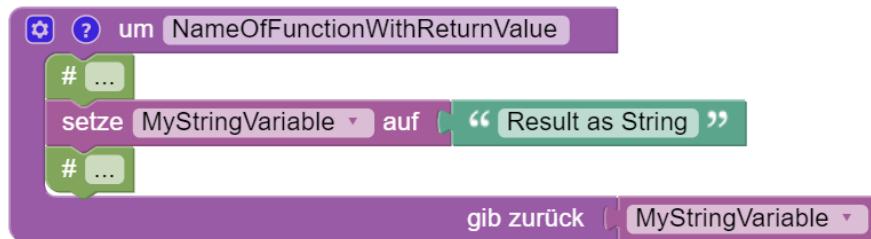


Funktionen mit Rückgabewert können in einem Zuweisungs-Block aufgerufen werden (d.h. sie haben eine Verbindung nach links):



Zu welchem Typ eine Zuweisung sinnvoll ist, hängt von dem Typ des Rückgabewerts ab.  
**Definieren von Rückgabewerten von Funktionen:** Funktionen werden durch speziellen *Blockly*-Elemente definiert, die an einer beliebigen Stelle im Code-Editor eingefügt werden können.

- *Rückgabewerte* können für Funktionen mit Rückgabewert definiert werden. Der Rückgabewert kann direkt an die Funktionsdefinition neben *gib zurück* angefügt werden:



Ergänzend stehen die folgenden zwei *Blockly*-Elemente zur Verfügung, die nur innerhalb einer Funktiondefintion (mit Rückgabewert) verwendet werden können:

- Der Operator *gib zurück* erlaubt die Rückgabe eines Wertes. Danach können innerhalb der Funktion keine weiteren *Blockly*-Elemente in den Ablauf platziert werden (d.h. der *gib zurück*-Operator hat keine Verbindung nach unten):



- Der Operator *falls gib zurück* Operator gibt eine Wert nur dann zurück, wenn eine Bedingungen erfüllt ist. Ist die Bedingung erfüllt, endet die Abarbeitung des Ablaufs in der Funktion, ist die Bedingung nicht erfüllt, wird die Bearbeitung fortgesetzt (d.h. der *falls gib zurück*-Operator hat eine Verbindung nach unten):



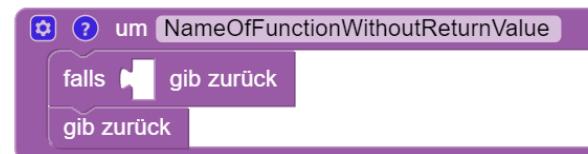
- Der *falls gib zurück* Operator ist also identisch mit folgender Kombination von Operatoren:



- Beide Operatoren (*falls gibt zurück* und *gib zurück*) können nicht außerhalb von Funktionen verwendet werden:

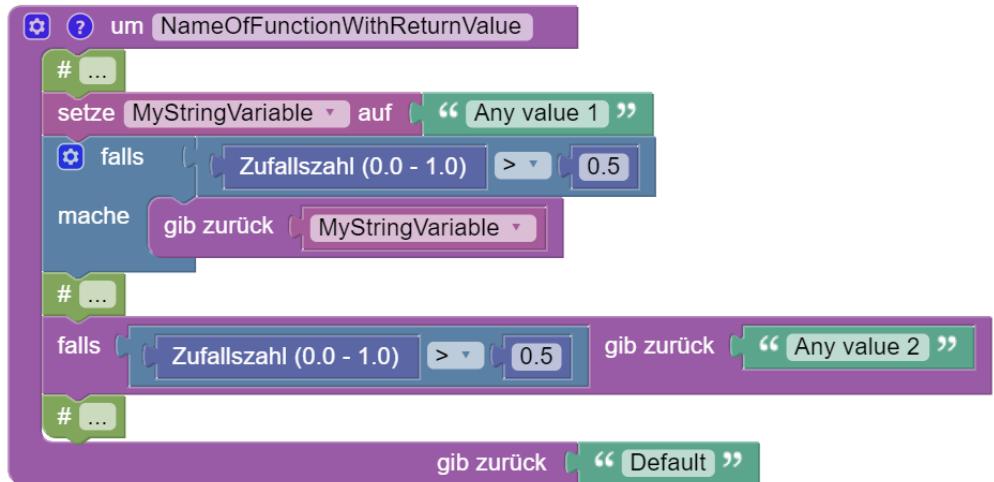


- Die beiden Operatoren (*falls gibt zurück* und *gib zurück*) können innerhalb von Funktionen ohne Rückgabewert verwendet werden, um die Abarbeitung von Funktionen zu beenden (aber nicht zum Rückgeben von Werten):



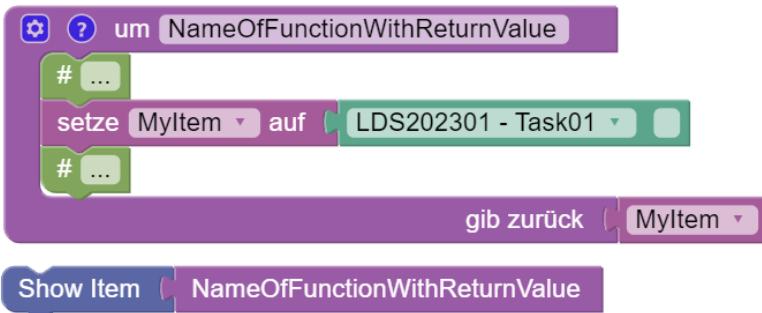
Beispiel:

- Die folgende Funktion gibt den Wert der Variablen *MyStringVariable* (*Any value 1*) in 50% der Fälle zurück (d.h. wenn eine erste gezogene Zufallsvariable größer 0.5 ist). In den anderen 50% der Fälle, wird eine weitere Zufallsvariable gezogen, und wenn diese größer 0.5 ist, dann wird der Text *Any value 2* zurückgegeben. Ist auch dies nicht der Fall, dann wird der Text *Default* zurückgegeben:

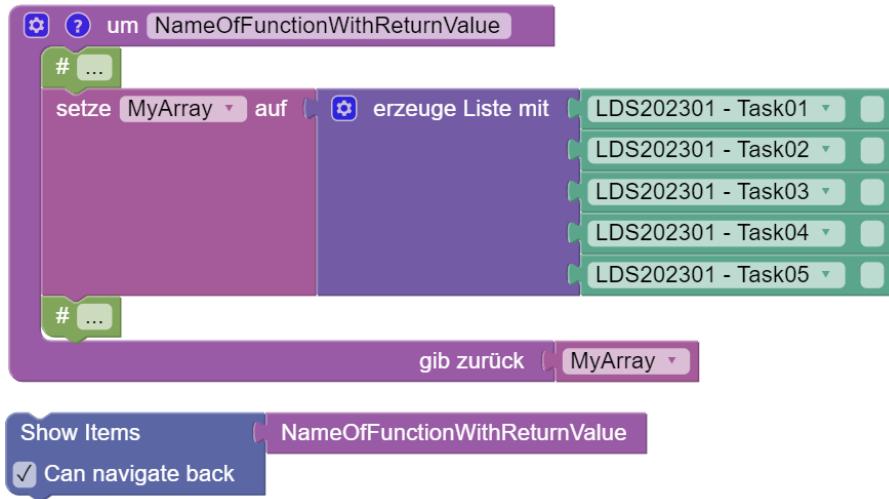


Rückgabewerte sind typisiert. Die Ablaufsteuerung unterstützt auch Funktionen, die ...

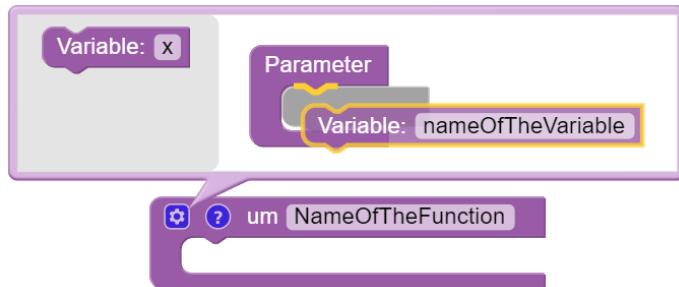
- ... einzelne *Tasks* zurückgeben:



- ... Listen von *Tasks* zurückgeben:

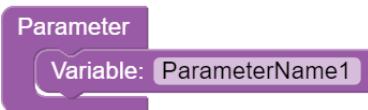


**Definieren von Aufrufparametern von Funktionen:** Funktionen können auch Parameter verwenden, die beim Aufruf der Funktion zu übergeben sind (*Aufrufparametern*). Die Definition von Aufrufparametern ist nach einem Klick auf das kleine Zahnrad-Symbol eines Funktions-Blocks möglich:



Passend zur Definition der Parameter, erfolgt dann der Aufruf der Funktion durch Übergabe:

- Definition eines Parameters

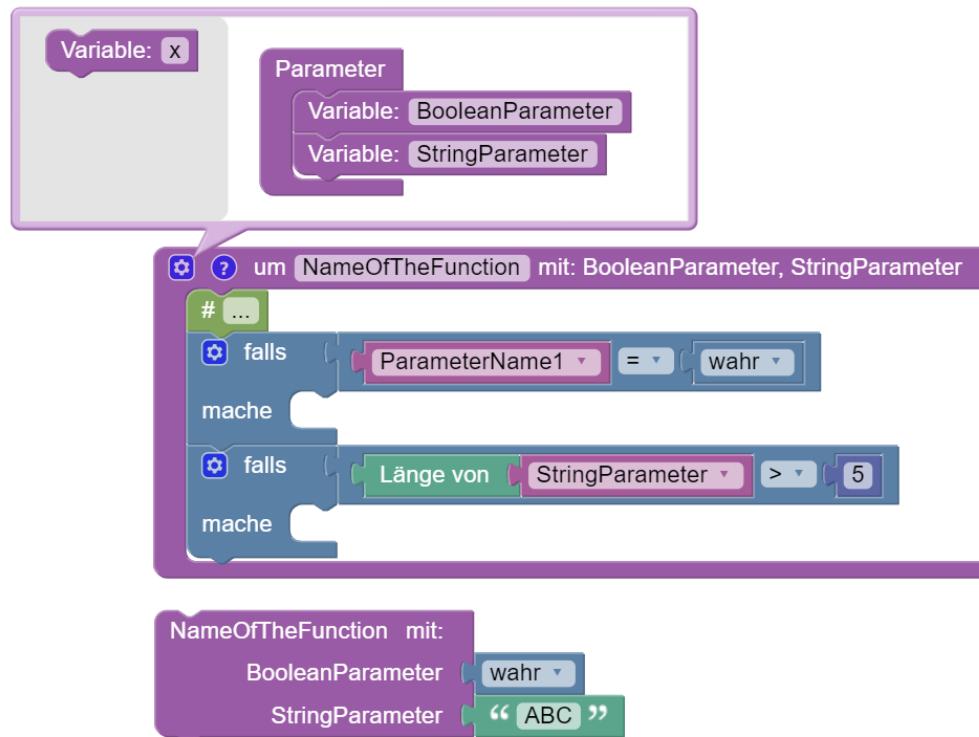


- Aufruf der Funktion mit Angabe von Wert:



Beispiel:

- Das folgende Beispiel zeigt eine Funktion mit zwei Parametern, deren Verwendung innerhalb der Funktion am Beispiel von Bedingungen und den Aufruf der Funktion mit festen Werten:



- Alternativ kann die Funktion natürlich auch mit Variablen aufgerufen werden:



#### 7.7.3.10 Nutzung von Itemergebnissen in der Ablaufsteuerung

(Dokumentation)

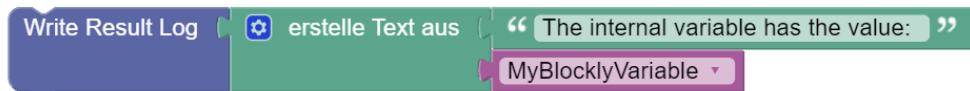
### 7.7.3.11 Blockly-Operatoren zum Kodieren fehlender Werte

(Dokumentation folgt)

### 7.7.3.12 Blockly-Operatoren zum schreiben von Daten

(Dokumentation folgt)

**Log-Daten:** Folgender Operator kann genutzt werden, um Informationen direkt in die Log-Daten zu speichern:



**Ergebnis-Daten:** (Dokumentation folgt)

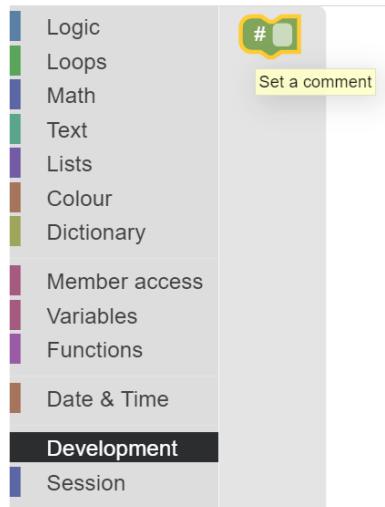
**Monitoring-Daten:** (Dokumentation folgt)

## 7.7.4 Kommentieren von Blockly-Code

Der *IRTLib Editor* unterstützt zwei verschiedene Optionen zur Kommentierung von Blockly-Code.

### 7.7.4.1 Kommentare als Blockly-Elemente

Kommentare, die im Ablauf dauerhaft sichtbar sein sollen, können über die Plaette im Abschnitt *Development* hinzugefügt werden:

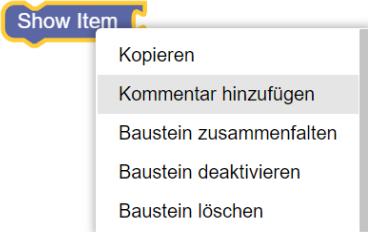


Diese Kommentare können wie Blockly-Operatoren verschoben werden und zeigen einzeiligen Kommentartext.

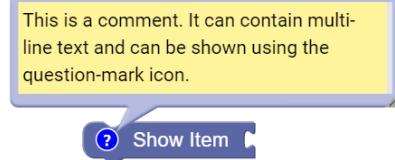
# Permanent single-line comments can also be inclu...

#### 7.7.4.2 Ausführliche Kommentare an *Blockly*-Elementen

Für ausführlichere Kommentare kann über das Kontextmenü jeder Block mit einem Kommentar hinzugefügt (und wenn vorhanden gelöscht) werden:



Diese Kommentare können mehrere Zeilen umfassen und werden dargestellt, wenn auf das kleine ?-icon eines Blocks geklickt wurde.

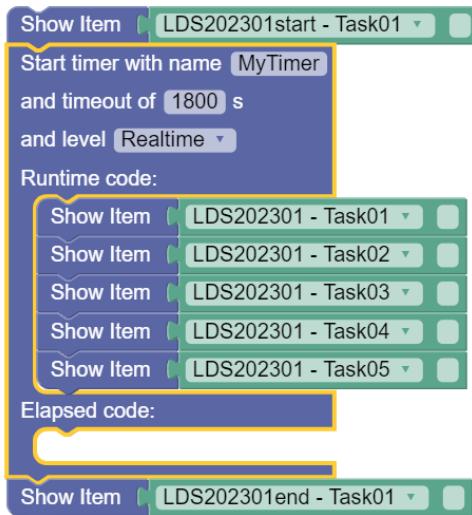


#### 7.7.5 Darstellung von *Blockly*-Code

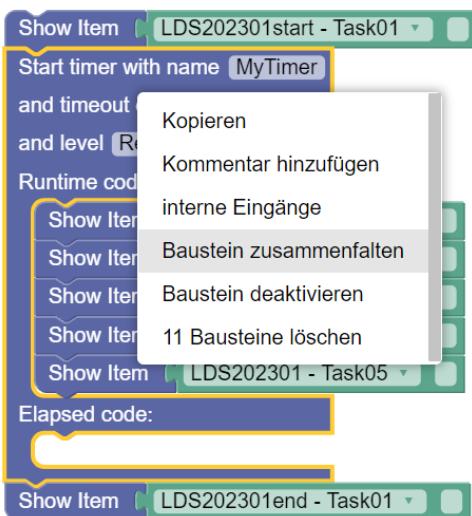
##### 7.7.5.1 Entfalten / Zusammenfalten

Große und komplexe Abläufe können im *Blockly*-Editor unter Umständen unübersichtlich werden. Um für eine Betrachtung nicht benötigte *Blockly*-Elemente auszublenden, ohne die Funktion des Ablaufdefinition zu verändern, können Blöcke *zusammengefaltet* werden: Das wird in folgendem Beispiel illustriert:

- Entfaltete (d.h. vollständige) Darstellung des markierten Blocks:



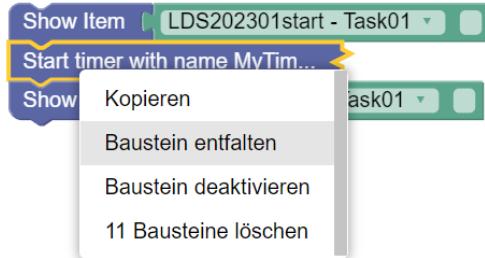
- Option zum *zusammenfalten* des Blocks im Kontextmenü:



- *Zusammengefaltete* Darstellung des Blocks innerhalb der Ablaufdefinition:



- Option zum *entfalten* des Blocks im Kontextmenü:

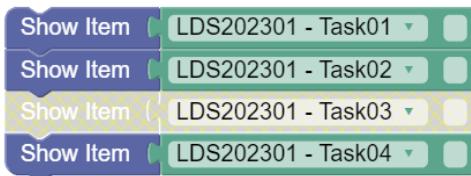


Das *zusammenfalten / entfalten* von *Blockly*-Elementen ändert nichts an der Funktion einer Ablaufdefinition und dient nur der übersichtlicheren Anordnung von komplexen Ablaufdefinitionen.

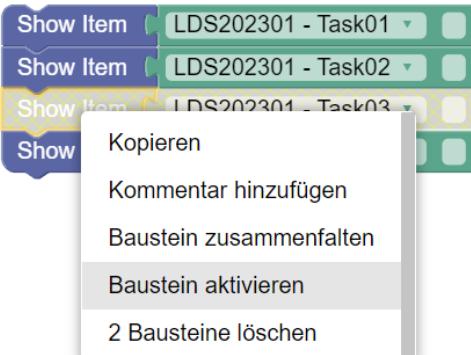
#### 7.7.5.2 Deaktivieren / Aktivieren

Der *Blockly*-Editor bietet die Option, *Blockly*-Element statt sie zu Löschen nur zu deaktivieren. Deaktivierte *Blockly*-Elemente bleiben in der Ablaufdefinition enthalten, werden aber nicht ausgeführt.

In folgendem Beispiel ist der Block zum Anzeigen des Tasks 3 deaktiviert, d.h. es werden nur Task 1, 2 und 4 angezeigt:



Aktivieren bzw. Deaktivieren von *Blockly*-Elementen erfolgt über das Kontextmenü:



**Internal / External:** Einige *Blockly*-Elemente mit *Eingängen* (d.h. Stellen, an denen man weitere Blöcke verbinden kann) erlauben zwischen zwei Darstellungsformen zu wechseln.

- Internal: Die *Eingänge* sind innerhalb der Blöcke angeordnet.



- External: Die *Eingänge* sind an der Seite der Blöcke angeordnet.



Beide Darstellungsformen sind bzgl. der Funktionalität äquivalent.

**Aufräumen:** Im Kontextmenü des *Blockly*-Editors, welches durch Klick in einen leeren Bereich geöffnet werden kann, ist die Funktion *Bausteine aufräumen* enthalten:

Rückgängig  
Wiederholen  
Bausteine aufräumen  
Alle Bausteine zusammenfalten  
Alle Bausteine entfalten  
32 Bausteine löschen

Durch Aufruf von *Bausteine aufräumen* werden alle *Blockly*-Elemente im *Blockly*-Editor vertikal untereinander ausgerichtet.

## 7.8 Routing zwischen Erhebungsteilen

Wenn mehrere *Erhebungsteile* für eine *Studie* definiert sind kann die Abfolge von Erhebungsteilen definiert werden, in welcher Befragte oder Testpersonen die Inhalte der *Erhebungsteile* präsentiert bekommen.

Neben einfachen linearen Abläufen können Abläufe von mehreren Erhebungsteilen auch mit einer *Blockly*-basierten Routing konfiguriert werden.

Eine detaillierte Beschreibung zum *Routing zwischen Erhebungsteilen* findet sich hier in der eingebetteten Hilfe:

## Eingebettete Programmhilfe

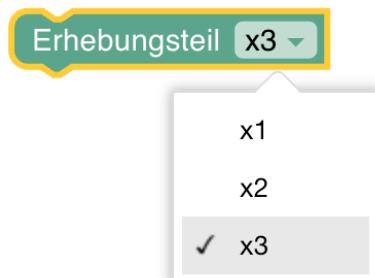
### 7.8.1 Zusammenfassung zu Routing zwischen Erhebungsteilen

Die Reihenfolge von *Erhebungsteilen* kann mit Hilfe von *Blockly* definiert werden (analog zur Definition der Reihenfolge von *Items* innerhalb von *Erhebungsteilen*). Diese Option ist verfügbar, wenn in der Grundkonfiguration zu einer Studie (in der Ansicht *Übersicht*) die Option *Routing für Erhebungsteile aktivieren* gewählt ist.

Für die allgemeinen Grundlagen zur Verwendung von *Blockly* im *IRTlib Editor* siehe die Hilfe zum *Routing innerhalb von Erhebungsteilen*.

Funktionen die nur im *Routing zwischen Erhebungsteilen* zur Verfügung stehen sind:

- Erhebungsteil Anzeigen



Dieser *Blockly*-Operator ersetzt das *Show Item* innerhalb von Erhebungsteilen.

- Erfolgreiches Login



Dieser *Blockly*-Operator hat den Wert *wahr*, wenn vor der Anzahl der maximalen Versuche (hier: unendlich, d.h. unbegrenzt) gültige Login-Informationen angegeben wurden.

Hinweis: Änderungen an der Sicht *Routing zwischen Erhebungsteilen* müssen über das Disketten-Symbol gespeichert oder mit dem Rückgängig-Symbol verworfen werden:

