Kommunikation

- Fernbedienung
- Sprachsteuerung
- Cloud / MQTT
- HTTP

Fernbedienung

Wenn ein Bedienfeld erstellt wurde, kann man mit den Blöcken des Abschnitts "Fernbedienung" Manipulationen an diesem Bedienfeld vornehmen und damit operieren.

Wichtige Hinweise:

- Der Abschnitt Fernbedienung steht in keinem Zusammenhang mit dem Control Set Baukasten
- Um ein Bedienfeld zu erstellen, wählen Sie im Menü die Option "Neue Datei" und dann "Bedienfeld".
- Die Fernsteuerung wird erst sichtbar, wenn das Programm gestartet wird.

Blöcke

Beschriftungsfeld setzen

Der Block "setze Beschriftungsfeld - Text" ermöglicht es, den Text in einem Beschriftungsfeld auf deinem Bildschirm zu ändern. Er ermöglicht es also, den Text während des Programms zu aktualisieren.

Diagramm setzen

Der Block „setze Diagramm“ in deinem Beispiel wird verwendet, um ein Diagramm auf dem Bildschirm zu zeichnen, indem Punkte in definierten Koordinaten platziert werden

Im Beispiel wird der Block „setze Diagramm“ verwendet, um ein Diagramm zu zeichnen, indem Punkte systematisch platziert werden. In einer Schleife wird für jede Iteration das Diagramm aktualisiert. Dabei wird der aktuelle Wert von x und sein Quadrat (angegeben als "y: Quadrieren x") als Koordinaten für die Punkte auf dem Diagramm verwendet. Dadurch werden die Punkte in einer parabolischen Form dargestellt, die die Beziehung zwischen x und y visualisiert.

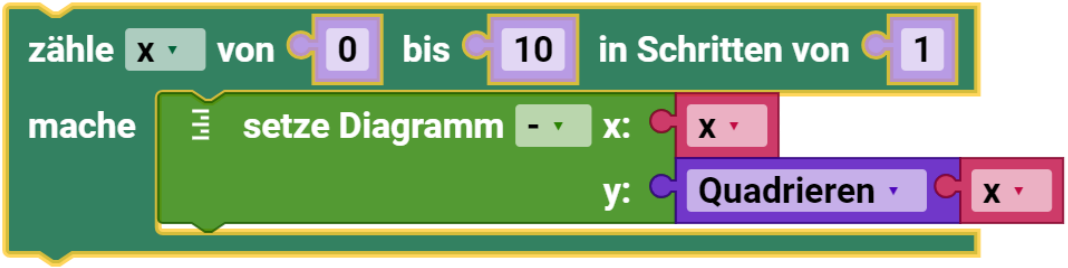


Bild setzen

Der Block "setze Bild - Base64-Bild" wird verwendet, um ein Bild auf einer Benutzeroberfläche zu setzen oder zu aktualisieren. Dies geschieht durch Angabe eines Base64-codierten Bildstrings. Der Block ermöglicht es, flexibel

verschiedene Bilder zur Laufzeit des Programms anzuzeigen, was besonders nützlich ist, wenn die Bilder dynamisch basierend auf Programmbedingungen geändert werden sollen.

Statusanzeige setzen

Der Block "setze Statusanze

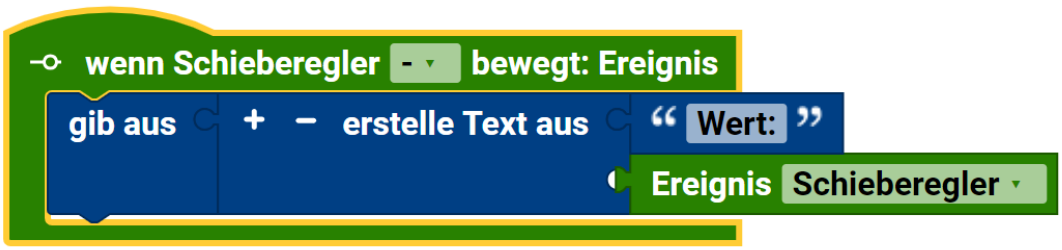
aktivieren oder deaktivieren.

Ereignis Schaltfläche angeklickt

Dieser Block wird verwendet, um auf das Klicken einer Schaltfläche zu reagieren. Dies kann nützlich sein, um Benutzerinteraktionen zu verfolgen oder bestimmte Aktionen auszulösen. Im Beispiel wird dann eine Meldung "Geklickt!" ausgegeben.

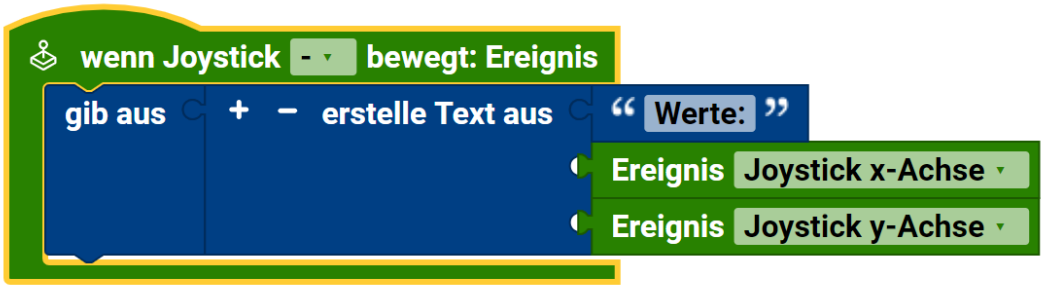
Ereignis Schieberegler angeklickt

Dieser Block wird verwendet, um auf Bewegungen eines Schiebereglers zu reagieren und den aktuellen Wert des Schiebereglers auszugeben. Dies ist nützlich, um kontinuierliche Anpassungen zu verfolgen und darauf zu reagieren, wie beispielsweise das Steuern der Lautstärke oder anderer variabler Parameter. Im Beispiel wird der aktuelle Wert des Reglers ausgegeben.



Ereignis Joystick angeklickt

Dieser Block wird verwendet, um auf Bewegungen eines Joysticks zu reagieren. Dies ist nützlich für Steuerungsanwendungen, bei denen die Richtung überwacht und verwendet werden müssen, wie z.B. bei der Steuerung eines Fahrzeugs. Im Beispiel wird die aktuelle Positionen der x- und y-Achse ausgegeben.


wenn Joystick - bewegt: Ereignis
gib aus
+
−
erstelle Text aus
“ Werte: ”
Ereignis Joystick x-Achse
Ereignis Joystick y-Achse

Sprachsteuerung

Blöcke für die die Kommunikation mit der App Voice Control.

wenn Befehl empfangen: Text

Wird ausgeführt, wenn ein neuer Text von App "Voice Control" empfangen wurde.

Text

Text ist der erkannte Sprachbefehl.

Cloud / MQTT

fischertechnik Cloud

Blöcke für die Kommunikation mit der fischertechnik Cloud.

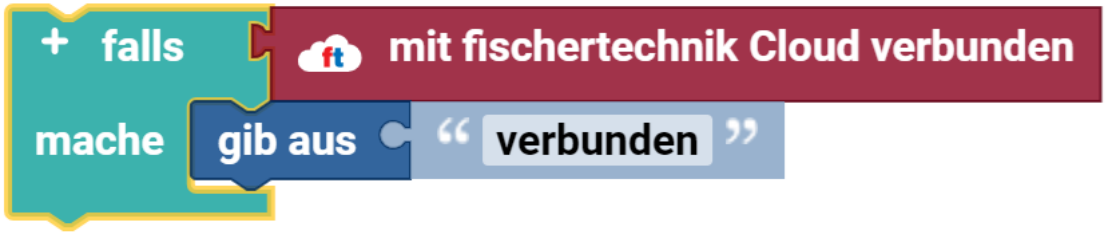
Diese Blöcke werden für "Sensorstation IoT" und "Lernfabrik 4.0" verwendet.

Cloud verbinden

Dieser Block stellt eine Verbindung zur fischertechnik Cloud her. Er wird verwendet, um eine Kommunikationsverbindung zwischen dem Controller und der Cloud zu initialisieren.

Cloud verbunden

Dieser Block prüft, ob eine Verbindung zur fischertechnik Cloud besteht. Er kann verwendet werden, um zu verifizieren, dass die Verbindung aktiv ist, bevor andere Cloud-Operationen ausgeführt werden.



Cloud trennen

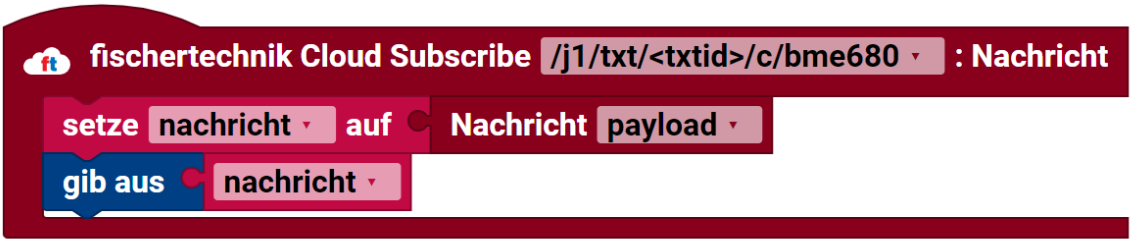
Dieser Block trennt die bestehende Verbindung zur fischertechnik Cloud. Er sollte verwendet werden, um die Kommunikation mit der Cloud ordnungsgemäß zu beenden.

Cloud Publish

Mit diesem Block können Textnachrichten an die fischertechnik Cloud gesendet werden. Der spezifische Pfad und die Nachricht, die veröffentlicht werden sollen, können konfiguriert werden.

Cloud Subscribe

Der Block "fischertechnik Cloud Subscribe" ermöglicht es, spezifische Nachrichten von der fischertechnik Cloud zu abonnieren. Wenn eine Nachricht am angegebenen Pfad empfangen wird, wird sie verarbeitet. Das Beispiel zeigt, wie man Nachrichten abonnieren, empfangen und verarbeiten kann.



MQTT Client

MQTT steht für "Message Queuing Telemetry Transport" Protokoll und wird oft bei IoT (Internet of Things) Anwendungen eingesetzt.

MQTT client create: websockets

Erstellt einen MQTT Client, mit dem Nachrichten empfangen und gesendet werden können. Es wird empfohlen die Ausgabe des Blocks in eine Variable zu schreiben, um den Client später in anderen Blöcken mehrfach verwenden zu können.

MQTT client ... connect

Verbindet einen MQTT Client mit einem MQTT Broker mit den angegebenen Einstellungen. Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Host gibt die Adresse des MQTT Brokers an. Um den lokalen MQTT Broker zu verwenden, wird als Wert localhost oder 127.0.0.1 eingetragen. Um externe MQTT Broker zu verwenden, muss dessen IP-Adresse oder Hostname verwendet werden. Port gibt den Port an, an welchem der MQTT Broker verfügbar ist. Standard ist 1883 für MQTT Broker (fischertechnik Cloud) oder 2883 (GUI Applikation). Bei Verwendung externer MQTT Broker muss dessen Port eingetragen werden. Username und Password sind standardmäßig leer, bei externen Servern müssen dessen Anmeldeinformationen hier eingetragen werden.

MQTT client ... is connected

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Gibt „true“ aus, wenn der angegebene MQTT Client mit einem MQTT Broker verbunden ist. Wenn der MQTT Client nicht verbunden ist, wird „false“ zurückgegeben.

MQTT client ... disconnect

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Trennt den angegebenen MQTT Client vom MQTT Broker.

MQTT client ... publish

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Veröffentlicht mit dem angegebenen MQTT Client eine Nachricht „payload“ in einen angegebenen Kanal „topic“. Zusätzlich kann "qos" und "retain" angegeben werden, also ob neu verbundene Clients die Nachricht nach dem Senden empfangen sollen und mit welchem Sicherheitslevel die Nachricht gesendet werden soll.

MQTT client ... publish (mit Rückgabewert)

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Veröffentlicht mit dem angegebenen MQTT Client eine Nachricht „payload“ in einen angegebenen Kanal „topic“. Zusätzlich kann "qos" und "retain" angegeben werden, also ob neu verbundene Clients die Nachricht nach dem Senden empfangen sollen und mit welchem Sicherheitslevel die Nachricht gesendet werden soll. Beim erfolgreichen Versenden wird „true“ ansonsten „false“ zurückgegeben.

MQTT client ... will set

MQTT Client kann der Block „MQTT client create“ sein, oder dessen Ausgabe in einer Variablen. Setzt die Nachricht „payload“, welche nach dem Trennen des MQTT Clients in einem angegebenen Kanal „topic“ versandt werden soll, und mit welchem Sicherheitslevel die Nachricht gesendet werden soll.

MQTT client ... subscribe

Abonniert mit einem MQTT Client einen Kanal. im Callback Argument wird die Funktion angegeben, welche beim Empfangen einer Nachricht ausgeführt werden soll. "Qos" gibt an, mit welchem Sicherheitslevel die Nachricht versendet werden soll.

subscribe callback ... : message

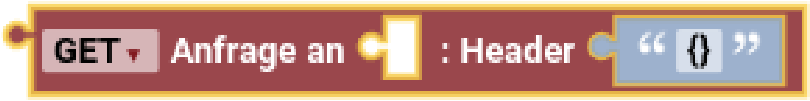
Definiert eine Funktion, welche durch Empfangen einer Nachricht ausgeführt werden soll. Message enthält die empfangene Nachricht.

HTTP

Blöcke zur Erstellung von HTTP-Anfragen

GET

Dieser Block ermöglicht das Senden einer GET-Anfrage an eine spezifische URL. Es können Header-Informationen hinzugefügt werden, um zusätzliche Metadaten mit der Anfrage zu senden.



POST, PUT, DELETE, PATCH

Dieser Block ermöglicht das Senden einer POST-, PUT-, DELETE- oder PATCH-Anfrage an eine spezifische URL. Neben Header-Informationen kann auch ein Payload, also die zu sendenden Daten, angegeben werden.