Quick-start user manual

createq

BRIQX
The complete retail solution in a box

208 controller & elements

V1.0 June 2024

createq

INDEX

Table of Contents

The Script

Building a script is simple. We will explain step by step how it is constructed. To do this, we will first explain the structure of a script.

A script is composed of three categories:

1. Setup
2. Main
3. State

Setup

In the setup, initializations take place. These actions are executed once and first after starting up the device. This includes for example: controlling the LED patterns of the buttons and configuring the settings of the connected sensors.

Main

In the main section, all commands that need to perform an action when a message is received are placed. For example, pressing a button results in receiving a message (recv:), and then an action is sent to execute a command (send), such as playing a movie.

State

This indicates the current state of the system, such as the status of playing a movie and turning on the LED light in the push button. The state defines what action to execute. To transition from one state to another, a transition is needed which occurs after a command or a notification.

Example Script:

setup

```
send: [U0500>OVL00]    # A command is sent during initialization.
transition: state00    # Once executed, the system transitions to another state.
```

main

```
state00:                  # In this state, it waits for possible commands (recv:).
    recv: [X0150<PUSH]    # A command to capture a PUSH, after which an action is executed.
        send: [U0100>V01]  # Command (send) to play video V01.
        transition: state01 # Once the command is executed, transition to state01.
```

State01

```
recv: [U0100<A01]      # Wait for a message indicating the video is done playing.
    transition: state00 # When this message is received, return to state00.
```

Structure of a Script

To understand the script, we will explain the structure of a command. An instruction consists of an input or an output followed by conditions in parentheses that specify the task or the incoming notification linked to a unique ID.

1. Explanation of the Symbols < and > in the Script

Example of input and output in a script:

```
recv: [X0152<PUSH]   # where the symbol < indicates the incoming notification.
send: [U0100>V01]    # where the symbol > indicates the command to execute.
```

2. The Structure of a Unique ID

An ID is composed of several parts. We will break down its structure below, focusing on what appears before the < and > symbols.

```
recv: [X0152<PUSH]
```

A. Communication Ports on a 208 Controller

A 208 controller has three different ports for communication:

- **X port**: Used for communication with buttons and elements. (In this example a X)
- **U port**: Used for communication with USB devices.
- **Q bus**: Used for communication between different controllers.

B. Unique Controller ID

A controller can be connected to other controllers and all-in-one products; therefore, each controller requires a unique number, which can be set in the WiFi Setup menu. In this example, the box ID is: 01

C. Port Number on a Controller

Each port on a controller has a number, which corresponds to the buttons and X ports. In this example, it is Xport 5.

D. Xport Hub Connection

An Xport Hub can be connected to an Xport (for expanding buttons, for example). In the overview below, a hub is connected to port ID 5, and this command refers to position 2 on this hub.

Overview variable input:

A: [X0152<PUSH]	B: [X0152<PUSH]	C: X0152<PUSH]	D: X0152<PUSH]
Q = Qbus	Controller ID: 01	Port ID: 1	Hub port ID: 1
X = Port	Controller ID: 02	Port ID: 2	Hub port ID: 2
U = USB	Controller ID: 03, etc	Port ID: 3, etc	Hub port ID: 3, etc

3. Commands and Notifications

To the right of the < and > symbols are the commands and notifications.

Pushbutton Commands	
PUSH	Notification: the button is pressed
PUSH% (PUSH1, PUSH2, etc)	Notification: the button is pressed for a number of (%) seconds
RELEASED	Notification: the button is released
INDON	Command: turn on the LED indicator of port % or 0 for all Xports
INDOFF	Command: turn off the LED indicator on Xport % or 0 for all Xports
INDBREATH	Command: enable breathing on Xport %, or 0 for all Xports
INDBLINK%	Command: enable blinking on Xport % for (%) number of times

4K30 Media Player Commands	
AUDIO	Media player, change to source x
A01	Command: play attract loop video A01, A02 in a loop
V%% (V01, V02, V03, etc)	Command: play a video with the name V%%
OVL%% (OVL01, OVL02, etc)	Command: display overlay with the name OVL%%
VOL%%	Command: adjust the volume to 0~99 percent

Example Script with Setup for 4K30 Player with 208 Controller and Pushbutton:

Main

```
Main
state00:
   recv: [X0150<PUSH1]  # Notification received: button pressed for 1 second from position Xport 05 of the controller with ID 1.
      send: [U0100>V01]      # Command via USB to player: start playing video V01.
      send: [X0100>INDOFF]   # Command: turn off all LEDs of all connected buttons.
      send: [X0150>INDON]    # Command: turn on the LED of the button on Xport 05.
      transition: state01    # Once executed, go to state01.
```

Element: Human Presence (XP-100)	
VERSION?	Query: what is the software version?
NAME?	Query: what is the name of the sensor?
PRESENCE?	Query: is a human detected?
PRESENCE	Notification: a human is detected
ABSENCE	Notification: human no longer detected
ZONE=% (ZONE=1, ZONE=2, etc)	Configuration: ZONE= detection (1 to 8) 1 = 75cm
TIME=% (TIME=1, TIME=2, etc)	Configuration: TIME= debounce time (0 to 9) per second

Element: Recoiler	
EXTRACT	Notification: the cable is pulled out

Element: Gesture	
UP	Notification: upward movement detected
DOWN	Notification: downward movement detected
LEFT	Notification: leftward movement detected
RIGHT	Notification: rightward movement detected
FORW	Notification: forward movement detected
BACK	Notification: backward movement detected
CW	Notification: clockwise movement detected
CCW	Notification: counter-clockwise movement detected
WAVE	Notification: waving movement detected

Element: NFC	
VERSION?	Query: what is the software version?
NAME?	Query: what is the name of the sensor?
TAGDET%% (TAGDET01, etc)	Notification: a tag with ID%% is detected

Element IR Distance Sensor	
ADC=%%% (ADC=100, etc)	Notification: the distance is 1 to 254 cm

All-in-One eARC Controller	
PUSH	Notification: the button is pressed
PUSH%	Notification: the button pressed for % seconds
RELEASED	Notification: the button released
ON: Command	turn on HDMI output %
OFF: Command	turn off all HDMI ports
INDON	Command: turn on the LED of the pushbutton on Xport %
INDOFF	Command: turn off all LEDs of all connected pushbuttons
LEDSTRIPON	Command: turn on the LED strip connected to Xport %
LEDSTRIPOFF	Command: turn off all connected LED strips
VOLUP	Command: increase the volume of HDMI output %
VOLDN	Command: decrease the volume of HDMI output %
UNMUTE	Command: unmute the last selected HDMI output
NEXT	Command: play the next track of the media player
AUDIO	Command: switch the source to % on the media player
A%% (A01, A02, etc)	Command: play the attract loop named A01, A02 in a loop
V%% (V01, V02, etc)	Command: play the video named V%%
OVL%% (OVL01, OVL02, etc)	Command: show the overlay named OVL%%

Example Scripts for (1x) 208 Controller with 4 Pushbuttons and a 4K Media Player

This script configures a 208 Controller with 4 pushbuttons and a 4K media player.
In idle mode, the LEDs of the pushbuttons will 'breathe'. When a button is pressed, the LED of that specific button remains on, and a unique video for each button is played.

setup

```
send: [X0100>INDBREATH]
transition: state00
```

Main

```
state00:
    recv: [X0150<PUSH]
        send: [U0500>V01]
        send: [X0100>INDOFF]
        send: [X0150>INDON]
        transition: state01

    recv: [X0160<PUSH]
        send: [U0500>V02]
        send: [X0100>INDOFF]
        send: [X0160>INDON]
        transition: state01

    recv: [X0170<PUSH]
        send: [U0500>V03]
        send: [X0100>INDOFF]
        send: [X0170>INDON]
        transition: state01

    recv: [X0180<PUSH]
        send: [U0100>V04]
        send: [X0100>INDOFF]
        send: [X0180>INDON]
        transition: state01

state02:
    recv: [U0100<A01]
        transition: state00
```

Example Script for (1x) 208 Controller with 4 Pushbuttons and a 4K Media Player

This script configures a 208 Controller with 4 pushbuttons and a 4K media player. In idle mode, the LEDs of the pushbuttons will 'breathe'. When a button is pressed, the LED of that specific button remains on, and a unique video for each button is played. When the push button is pressed again, an overlay with product information, for example, is shown.

setup

```
send: [X0100>INDBREATH]   # Command: all pushbuttons breathe.
send: [U0500>OVL00]       # Command: do not display any overlays.
transition: state00       # After execution, go to the next state (state00).
```

main

```
state00:
    recv: [X0150<PUSH]     # Notification: push received on controller ID 1 and on xport 5.
        send: [U0100>V01]  # Command: send USB command to play video V01.
        send: [X0100>INDOFF]  # Command: turn off all LEDs of the connected pushbuttons.
        send: [X0150>INDON]   # Command: turn on the LED of the pushbutton on controller ID 1 on Xport 5.
        transition: state01   # After execution, go to the next state (state01).

    recv: [X0160<PUSH]
        send: [U0500>V02]
        send: [X0100>INDOFF]
        send: [X0160>INDON]
        transition: state01

    recv: [X0170<PUSH]
        send: [U0500>V03]
        send: [X0100>INDOFF]
        send: [X0170>INDON]
        transition: state01

    recv: [X0580<PUSH]
        send: [U0500>V04]
        send: [X0100>INDOFF]
        send: [X0180>INDON]
        transition: state01

state01:
    recv: [X0150<PUSH]   # Notification: PUSH received from controller ID 1 and Xport 5.
        send: [U0100>OVL01]  # Command: send USB command to play overlay OVL01.
        transition: state01

    recv: [X0160<PUSH]
        send: [U0500>OVL02]
        transition: state01

    recv: [X0170<PUSH]
        send: [U0500>OVL03]
        transition: state01

    recv: [X0180<PUSH]
        send: [U0100>OVL04]
        transition: state01

    recv: [U0100<A01]  # Notification received via USB to play attract loop.
        transition: state00  # After execution, return to state00.
```

How to add Timers

The setup:

Initializes the timer and turns off all LEDs in the setup state.
Cycles through four states (state01 to state04), where each state waits for the timer to trigger, turns off all LEDs, and then turns on a specific LED before transitioning to the next state.
Continuously loops through the states, maintaining a 100 ms interval for each state change.

setup

```
setup:
    set: TMR01 5            # Enable timer 1 and trigger in 500 ms
    send: [X0200>INDOFF]    # Turn every indicator LED off
    transition: state01     # Go to the main loop
```

- Initializes the timer (TMR01) to trigger after 500 ms.
- Turns off all indicator LEDs.
- Transitions to state01.

main
Each state in the main loop does the following:

1. Waits for the timer (TMR01) to trigger.
2. Resets the timer to trigger in 100 ms.
3. Turns off all indicator LEDs.
4. Turns on a specific indicator LED.
5. Transitions to the next state.

```
state01:
    time: TMR01             # If timer 1 is triggered
    set: TMR01 1            # Reset timer to 100 ms
    send: [X0200>INDOFF]    # Turn every indicator LED off
    send: [X0250>INDON]     # Turn indicator LED on of button 5
    transition: state02     # Go to the next state
```

Waits for the timer to trigger.

- Resets the timer to 100 ms.
- Turns off all indicator LEDs.
- Turns on the indicator LED for button 5.
- Transitions to state02.

This pattern repeats for state02, state03, and state04, each turning on a different indicator LED.

```
state02:
    time: TMR01
    set: TMR01 1
    send: [X0200>INDOFF]
    send: [X0260>INDON]
    transition: state03

state03:
    time: TMR01
    set: TMR01 1
    send: [X0200>INDOFF]
    send: [X0270>INDON]
    transition: state04

state04:
    time: TMR01
    set: TMR01 1
    send: [X0200>INDOFF]
    send: [X0280>INDON]
    transition: state01     # Go back to state 01 so it will keep looping
```

Note:

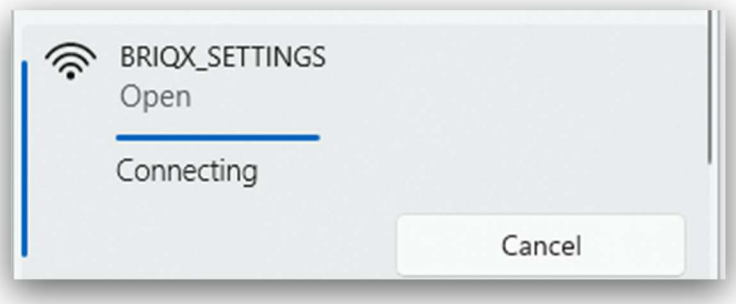
In the script, annotations can be made by placing a # and the text of the annotation after it.

Update

To download the script and upload it to the 208 controller, we first need to connect to the Wi-Fi setup menu of the controller.

Connect the power to the BRIQX Controller. You now have 2 seconds to press the small button next to the USB-C port. When you press the button, the blue LED light next to the power connector will start flashing. This confirms that the BRIQX Controller has created a Wi-Fi access point.

Connect to BRIQX_SETTINGS.



Go to 10.10.10.10/settings in your web browser.

At the bottom of the page, you will see buttons to upload, remove, and download a script. Next to 'Remove script' is the name of the currently uploaded script. Press 'Download Script' to download the current script.

Click 'Choose File' to directly upload the script. Name the script you upload 'yourname.txt', as only filenames with the .txt extension are accepted.

You can then disconnect and reconnect the power to the BRIQX Controller to start the script.