

HomeKit32 is a personal hobby project turning the Keyestudio KS5009 smart-home kit into a clean, modular automation platform using **ESP32, BLE, MQTT and B4X**.

The ESP32 runs **B4R firmware** acting as a BLE GATT server and optional MQTT gateway. Clients exist for **B4J, B4A, Python**, and a **Blockly visual programming interface** integrated into B4J.

All 13 KS5009 devices are supported (LEDs, relays, servos, sensors, LCD, RFID, etc.). Communication uses structured BLE packets and JSON/MQTT payloads. Focus has been mostly on BLE development.

Why B4R + BLE?

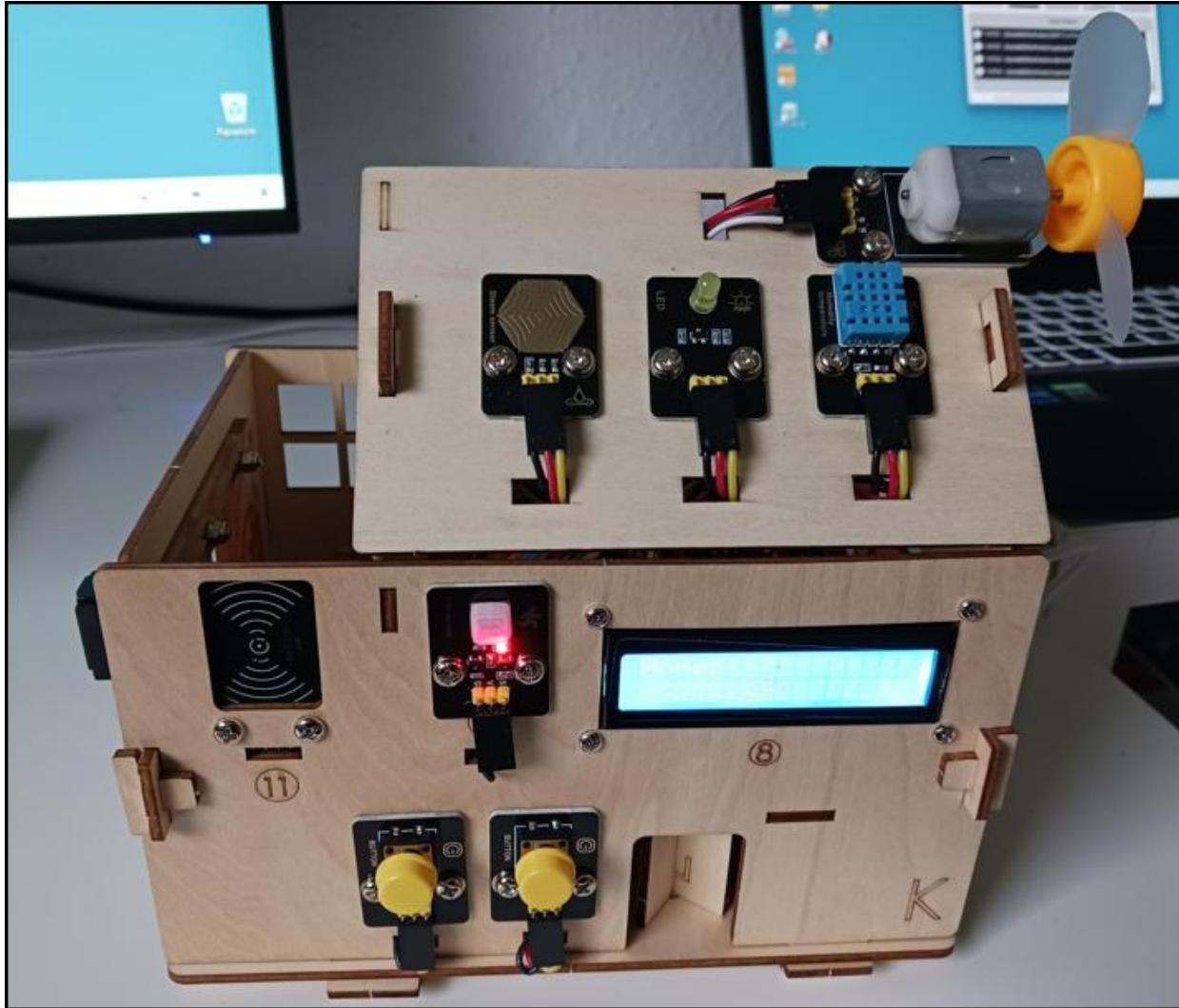
- B4R gives native-speed embedded code with clean structure
- BLE provides low-latency, cable-free control without Wi-Fi dependency
- Perfect for education, experimentation, and protocol design

Why B4J + B4A?

- Same language across firmware, desktop, and Android
- Rapid UI development with minimal glue code

This is a **non-commercial learning project** shared for inspiration.

Keyestudio Smart Home Kit ESP32



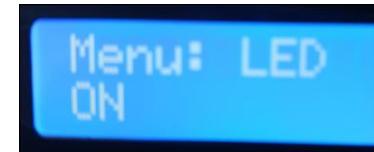
ESP32 Firmware: B4R
Kit fully assembled: device 13 sensors and actuators.
Project modular structure with dedicated component libraries (C++ wrapped or b4xlib).
Communication modes with clients BLE or WiFi + MQTT.

Integrated Menu

Button left select menu item displayed on the LCD1602 (top row).
Button right select menu item state (bottom row).

Example:

Menu: LED, state ON or OFF

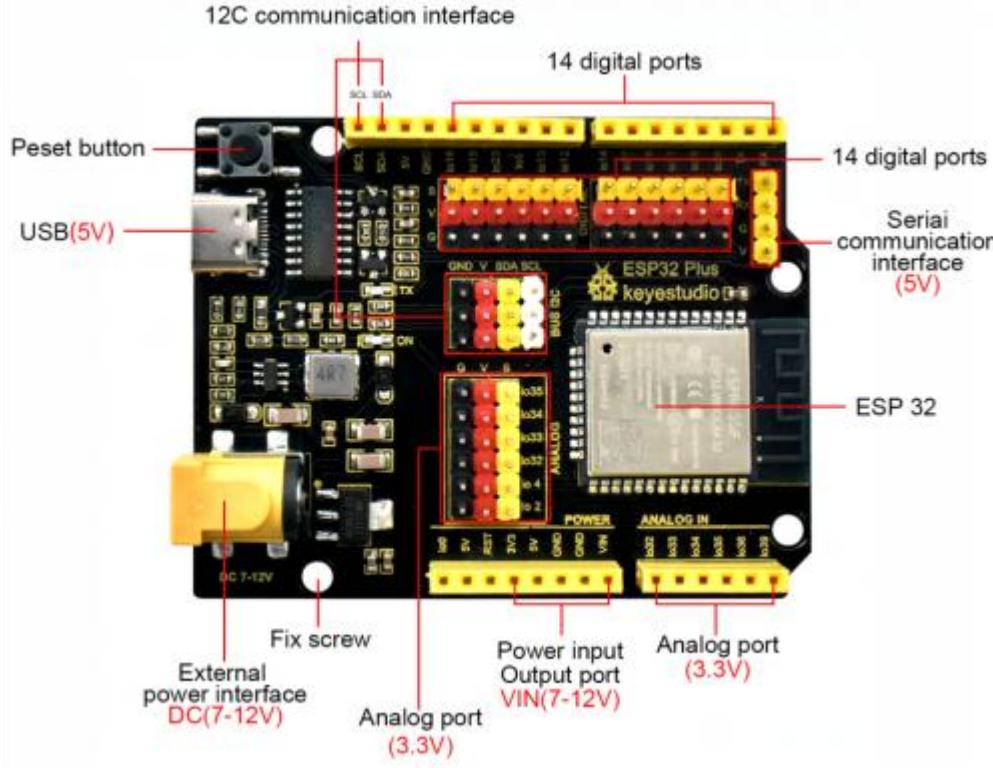


RFID

Open or closed door when touched with tag holding group 2 and command 4.



Keyestudio ESP32 Plus Pinout



Communication

WiFi, MQTT
BLE (UART)
Serial (Arduino, B4R upload)

Pin mapping

===== LEDs =====

Public ONBOARDLED_PIN As Byte = 2

Public YELLOW_LED_PIN As Int = 12

Public RGB_LED_PIN As Int = 26

===== Buttons =====

Public BTN_LEFT_PIN As Int = 16

Public BTN_RIGHT_PIN As Int = 27

===== PIR Sensor (Motion) =====

Public PIR_SENSOR_PIN As Int = 14

===== DHT11 Temp + Hum =====

Public DHT11_PIN As Int = 17

===== Moisture Sensor (Analog) =====

Public MOISTURE_SENSOR_PIN As Int = 34

===== Gas Sensor (Analog) =====

Public GAS_SENSOR_PIN As Int = 23

===== Audio =====

Public BUZZER_PIN As Int = 25

===== Fan =====

Public FAN_DIRECTION_PIN As Int = 19

Public FAN_SPEED_PIN As Int = 18

===== Servos =====

Public SERVO_WINDOW_PIN As Int = 5

Public SERVO_DOOR_PIN As Int = 13

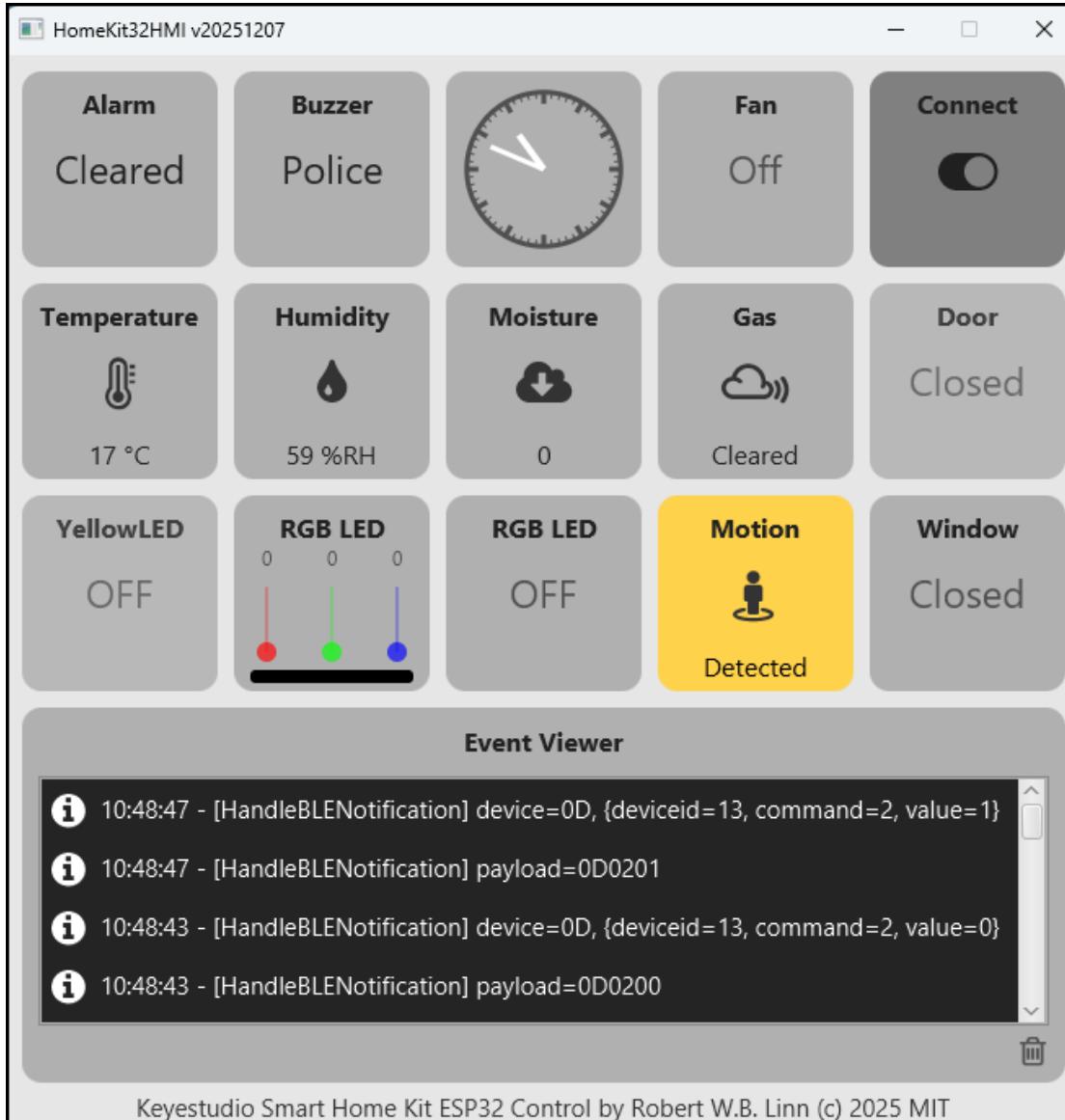
===== I2C Devices =====

Public RFID_I2C_ADDRESS As Byte = 0x28 ' RFID Mifare

Public LCD_I2C_ADDRESS As Byte = 0x27 ' LCD1602

[Source Keyestudio](#)

B4J BLE Client HK32HMI



HMI to control devices over BLE.

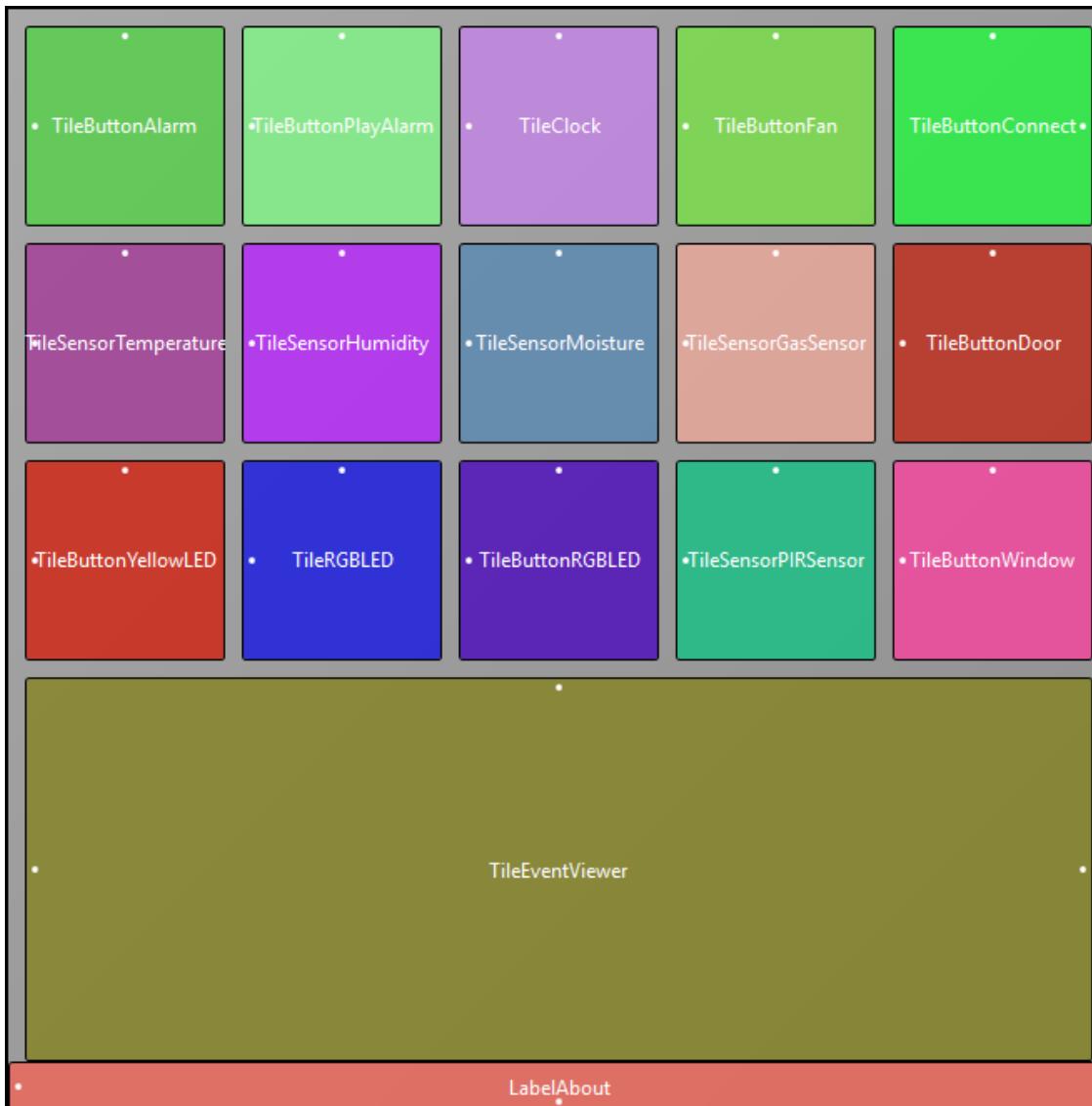
B4J Project

- B4X Pages Project
- B4J Layout Single Page
- HMI Tiles (Custom Views)
(Human-Machine Interface)
Default 120px x 120px
- Event log (CustomListView)
- ISA-101 Guidelines aligned
(Human-Machine Interface Design)

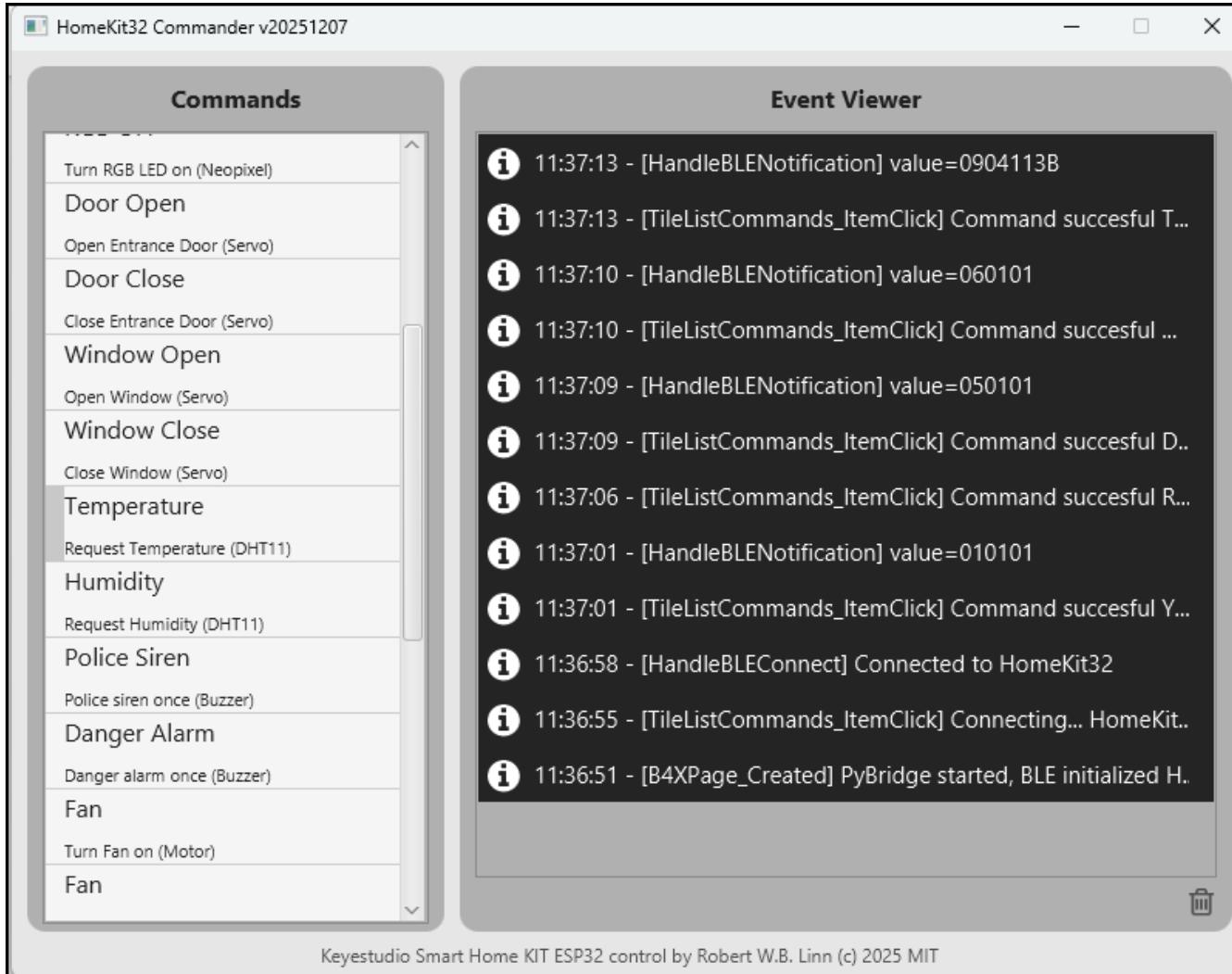
Communication Mode

- BLE
- BLE Frame:
[DeviceID][Command][Payload...]
Optional response
(GET/request commands):
[DeviceID][Command][Response...]
- Example Payload switch YellowLED ON:
Byte Array with 3 bytes:
0x01 0x01 0x01
Device-ID Command-ID Payload

B4J BLE Client HK32HMI Layout Single Page



B4J BLE Client HK32Commander

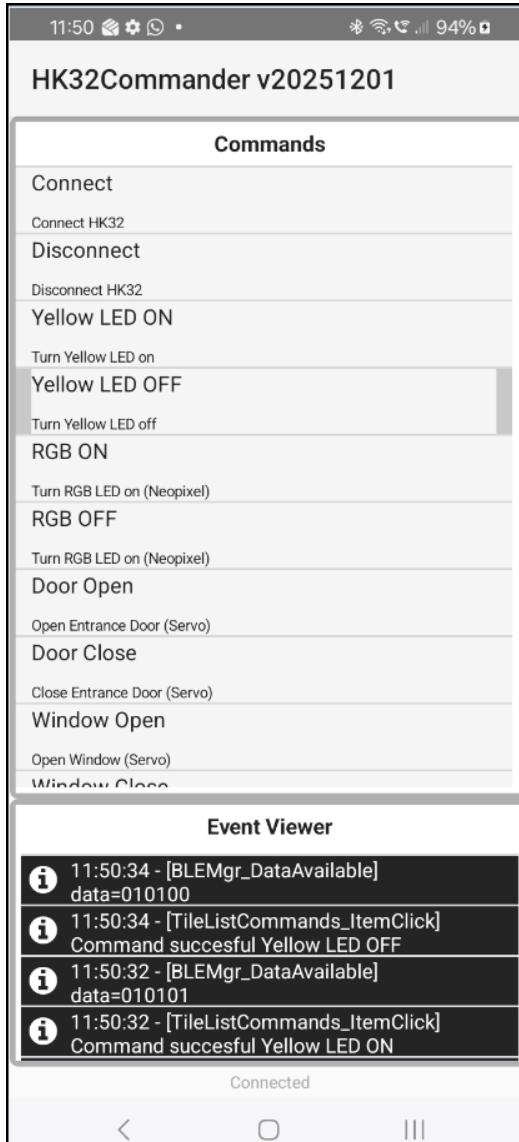


Test Commands to control devices over BLE.

B4J Project

- B4X Pages Project
- B4J Layout Single Page
- Commands List (CustomListView)
- Event Viewer (CustomListView)

B4A BLE Client HK32Commander



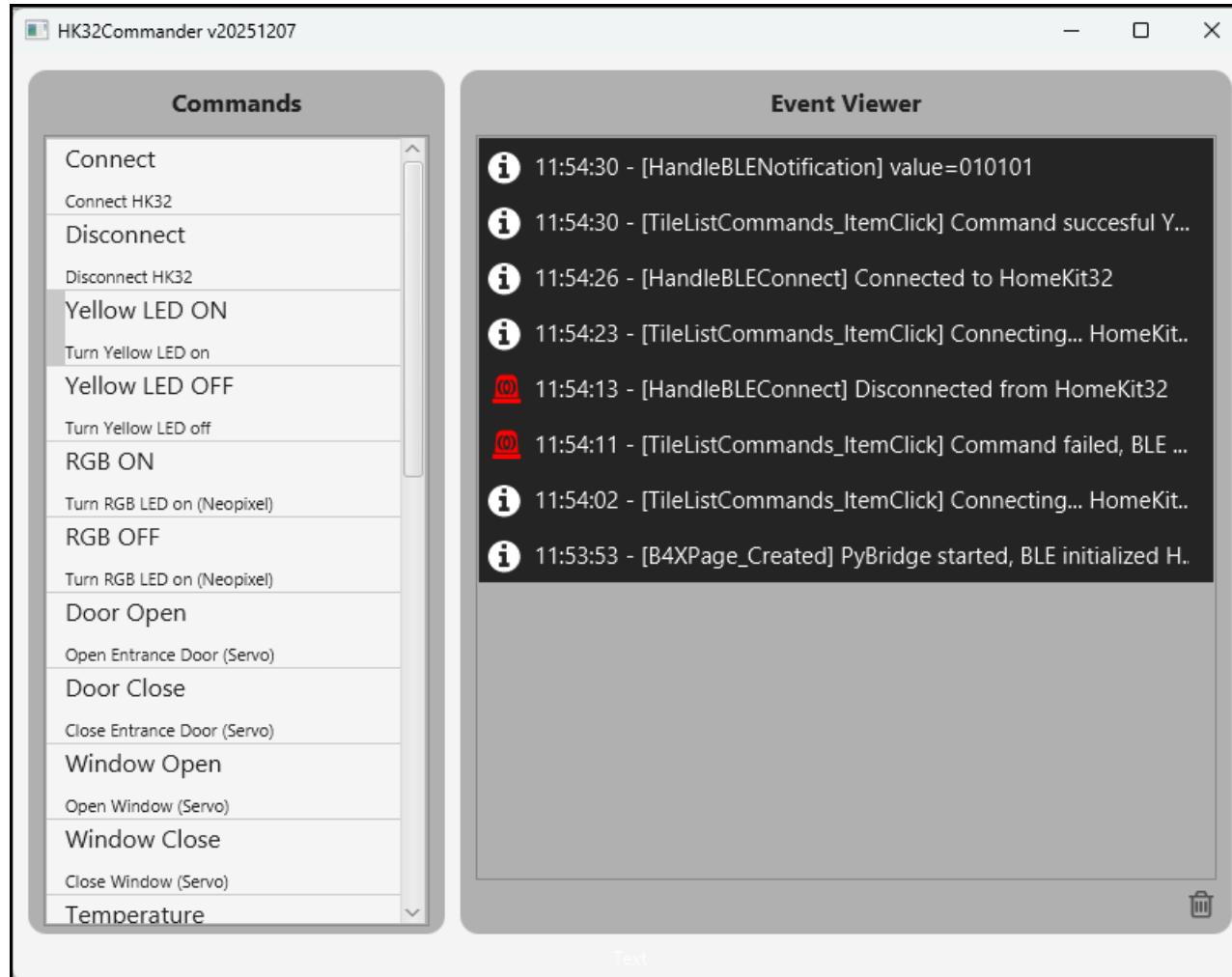
Test Commands to control devices over BLE.

B4A Project

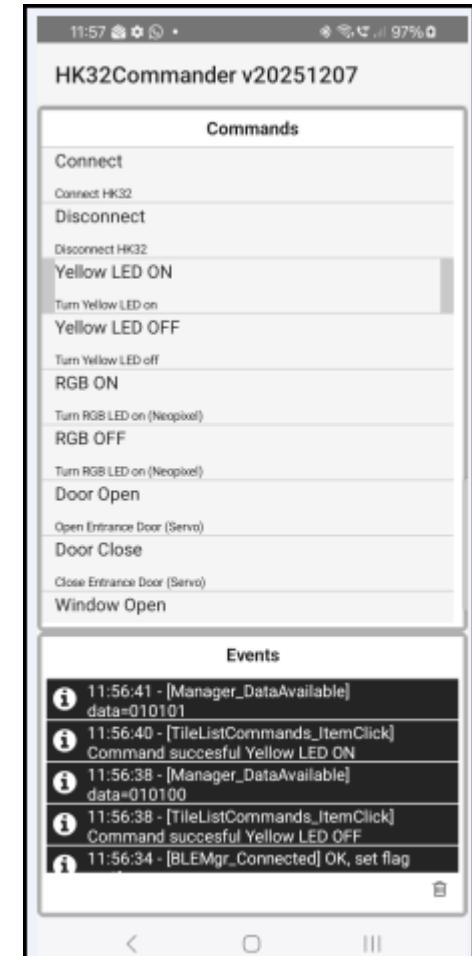
- B4X Pages Project
- B4A Layout Single Page
- Commands List
(CustomListView)
- Event Viewer
(CustomListView)

B4X BLE Clients HK32Commander

B4A & B4J BLE clients. B4X pages project with HMI Tiles to test direct commands to control devices.



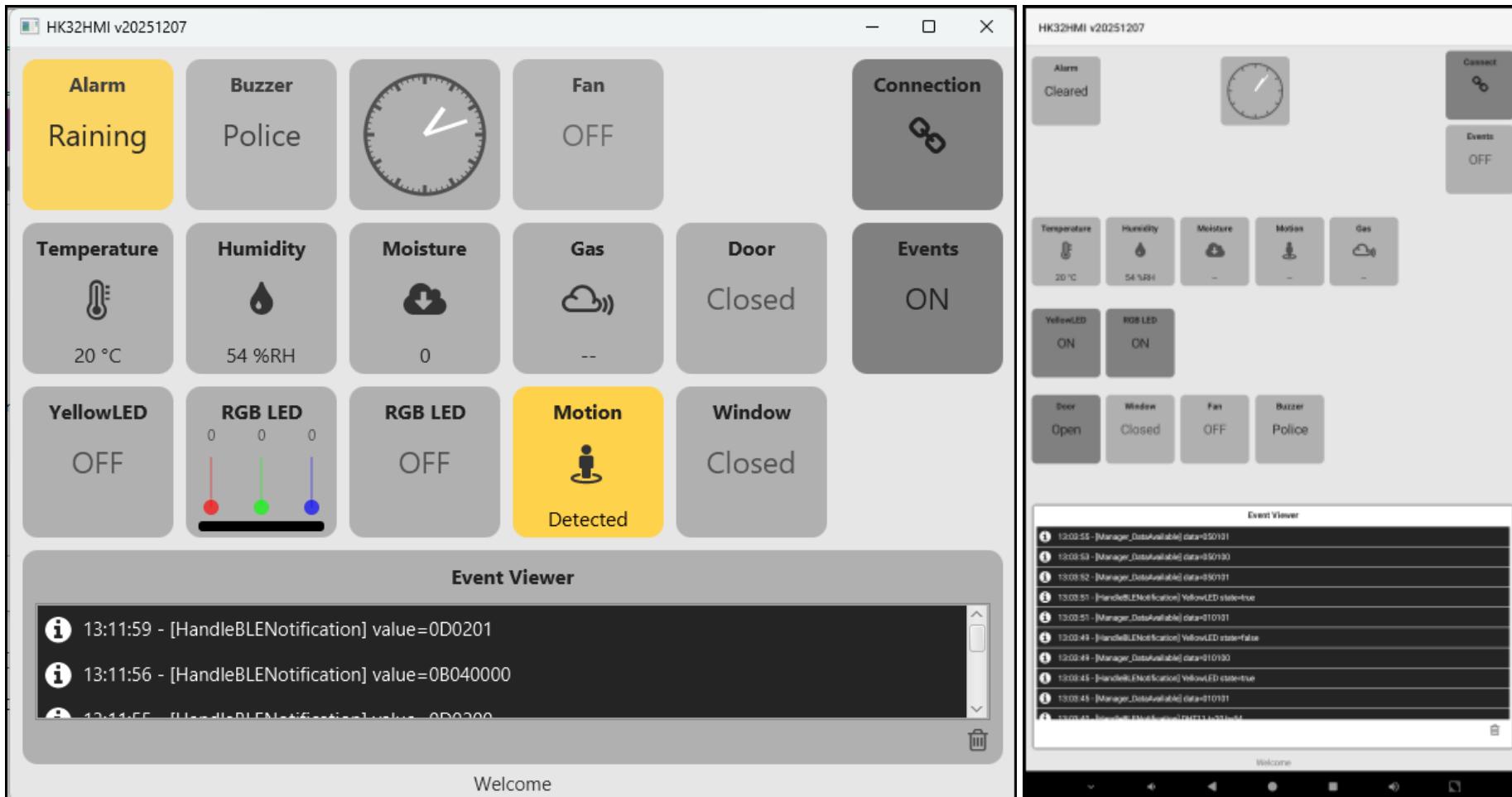
B4J Windows 11 Desktop



B4A Smart Phone

B4X BLE Clients HK32HMI

B4A & B4J BLE clients. B4X pages with HMI Tiles project HMI.



B4J Windows 11 Desktop

12/14/2025

Project make-homekit32

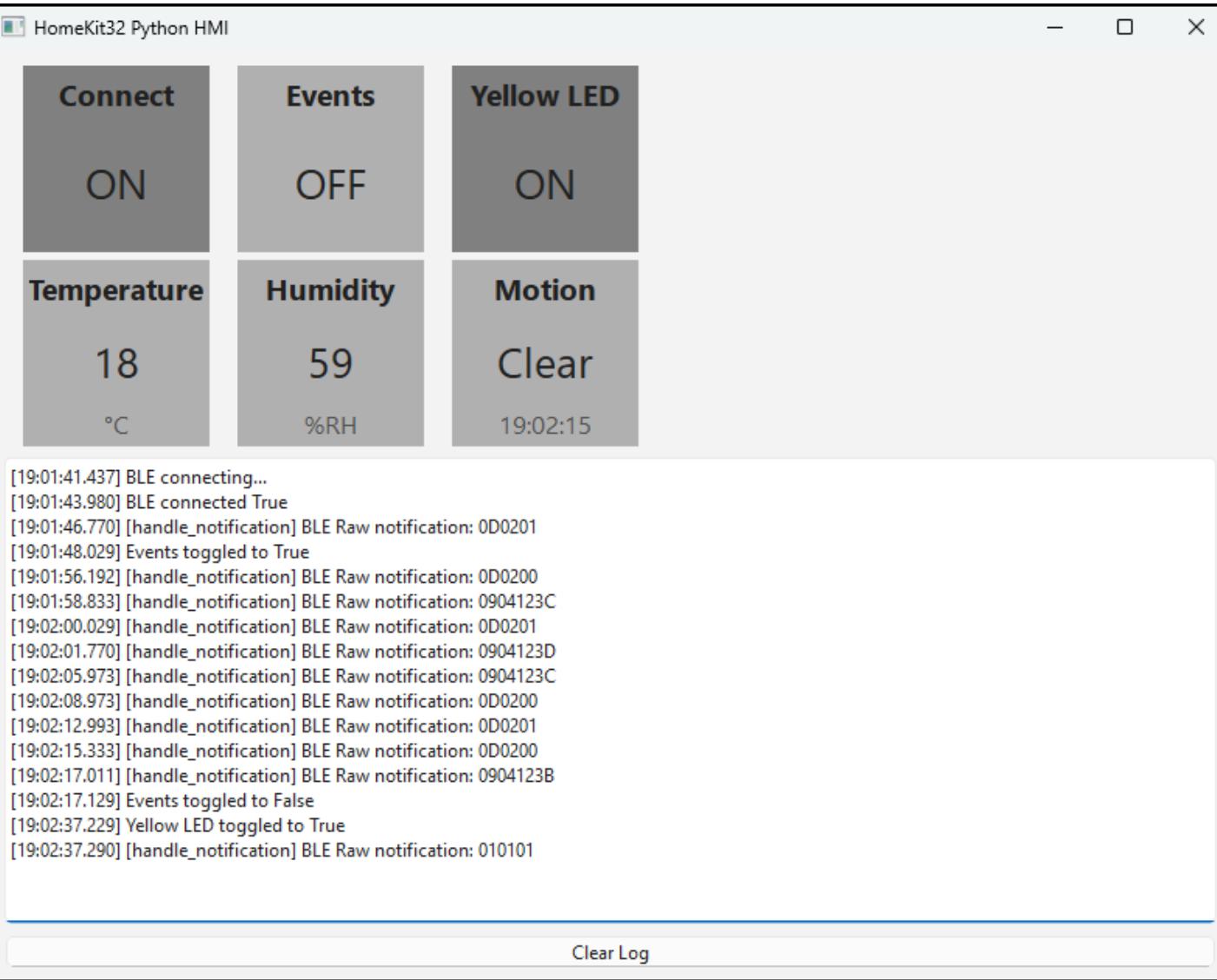
B4A Tablet

9

Python BLE Client HK32HMI

Python with PySide6 framework example using same HMITile concept as B4X – selective TileButton & TileReadout.

TileButton



MQTT - mosquitto

On Windows 11 device started the **mosquitto broker** and subscribed to topics **homekit32/home1/#**.
Submitted test commands using the mosquitto client **mosquitto_pub**. Example Yellow LED on:
`mosquitto_pub.exe -t homekit32/home1/yellow_led/set -m {"s": "on"}`

Mosquitto Broker

```
C:\WINDOWS\system32\cmd. x + □ ×

1765183477: Received PUBLISH from auto-0B9C9BF3-AEE1-054C-52B0-701E2F9613B6 (d0, q0
, r0, m0, 'homekit32/home1/yellow_led/set', ... (10 bytes))
1765183477: Sending PUBLISH to homekit32 (d0, q0, r0, m0, 'homekit32/home1/yellow_l
ed/set', ... (10 bytes))
1765183477: Sending PUBLISH to auto-1116E697-CB28-CE36-3F15-41507BCAE531 (d0, q0, r
0, m0, 'homekit32/home1/yellow_led/set', ... (10 bytes))
1765183477: Received DISCONNECT from auto-0B9C9BF3-AEE1-054C-52B0-701E2F9613B6
1765183477: Client auto-0B9C9BF3-AEE1-054C-52B0-701E2F9613B6 disconnected.
1765183477: Received PUBLISH from homekit32 (d0, q0, r1, m0, 'homekit32/home1/yello
w_led/status', ... (10 bytes))
1765183477: Sending PUBLISH to auto-1116E697-CB28-CE36-3F15-41507BCAE531 (d0, q0, r
0, m0, 'homekit32/home1/yellow_led/status', ... (10 bytes))
1765183484: New connection from ::1:55980 on port 1883.
1765183484: New client connected from ::1:55980 as auto-CAA9919F-9091-CCED-C24B-FA3
FA1606BA1 (p2, c1, k60).
1765183484: No will message specified.
1765183484: Sending CONNACK to auto-CAA9919F-9091-CCED-C24B-FA3FA1606BA1 (0, 0)
1765183484: Received PUBLISH from auto-CAA9919F-9091-CCED-C24B-FA3FA1606BA1 (d0, q0
, r0, m0, 'homekit32/home1/yellow_led/set', ... (11 bytes))
1765183484: Sending PUBLISH to homekit32 (d0, q0, r0, m0, 'homekit32/home1/yellow_l
ed/set', ... (11 bytes))
1765183484: Sending PUBLISH to auto-1116E697-CB28-CE36-3F15-41507BCAE531 (d0, q0, r
0, m0, 'homekit32/home1/yellow_led/set', ... (11 bytes))
1765183484: Received DISCONNECT from auto-CAA9919F-9091-CCED-C24B-FA3FA1606BA1
1765183484: Client auto-CAA9919F-9091-CCED-C24B-FA3FA1606BA1 disconnected.
1765183484: Received PUBLISH from homekit32 (d0, q0, r1, m0, 'homekit32/home1/yello
w_led/status', ... (11 bytes))
1765183484: Sending PUBLISH to auto-1116E697-CB28-CE36-3F15-41507BCAE531 (d0, q0, r
0, m0, 'homekit32/home1/yellow_led/status', ... (11 bytes))
|
```

Subscribe Topics

```
C:\WINDOWS\system32\cmd. x + □ ×

c:\Daten\projects\make\make-homekit32\servers\mosquitto>REM Subscribe to mosquitto payload homekit
32/home1/#

c:\Daten\projects\make\make-homekit32\servers\mosquitto>c:\Prog\mosquitto\mosquitto_sub.exe -v -t
homekit32/home1/#

homekit32/home1/yellow_led/set {"s": "on"}
homekit32/home1/yellow_led/status {"s": "on"}
homekit32/home1/yellow_led/set {"s": "off"}
homekit32/home1/yellow_led/status {"s": "off"}|
```

Publish Topics

```
C:\WINDOWS\system32\cmd. x + □ ×

Microsoft Windows [Version 10.0.26200.7309]
(c) Microsoft Corporation. Alle Rechte vorbehalten.

c:\Daten\projects\make\make-homekit32\servers\mosquitto>c:\Prog\mosquitto\mosquitto_pub.exe -t homekit32/home1/yellow_led/set -m {"s": "on"}

c:\Daten\projects\make\make-homekit32\servers\mosquitto>c:\Prog\mosquitto\mosquitto_pub.exe -t homekit32/home1/yellow_led/set -m {"s": "off"}|
```

B4R IDE Log

```
GlobalStoreHandler.Put][I] slot=0, bufferlength=31, Max=200, data size=10, hex=7B2273223A226F6E227D
[CommMQTT.MQTT_MessageArrived][I] topic=homekit32/home1/yellow_led/set, index=1, payload={"s": "on"}, storeindex=0
[DevYellowLed.ProcessMQTT] storeindex=0, payload={"s": "on"}
```

B4J MQTT Broker (console app)

B4J console application acting as an MQTT broker.

Running on Windows 11 the MQTT broker us started, the MQTT internal client is connected and **subscribed** to topics **homekit32/#**.

Submitted test commands using the mosquitto client **mosquitto_pub**.

Example Yellow LED on / off:

```
mosquitto_pub.exe -i hk32client -t homekit32/home1/yellow_led/set -m {"s":"on"}
```

```
mosquitto_pub.exe -i hk32client -t homekit32/home1/yellow_led/set -m {"s":"off"}
```

Notes

Mandatory to set the MQTT client ID, like hk32client (or any other name)

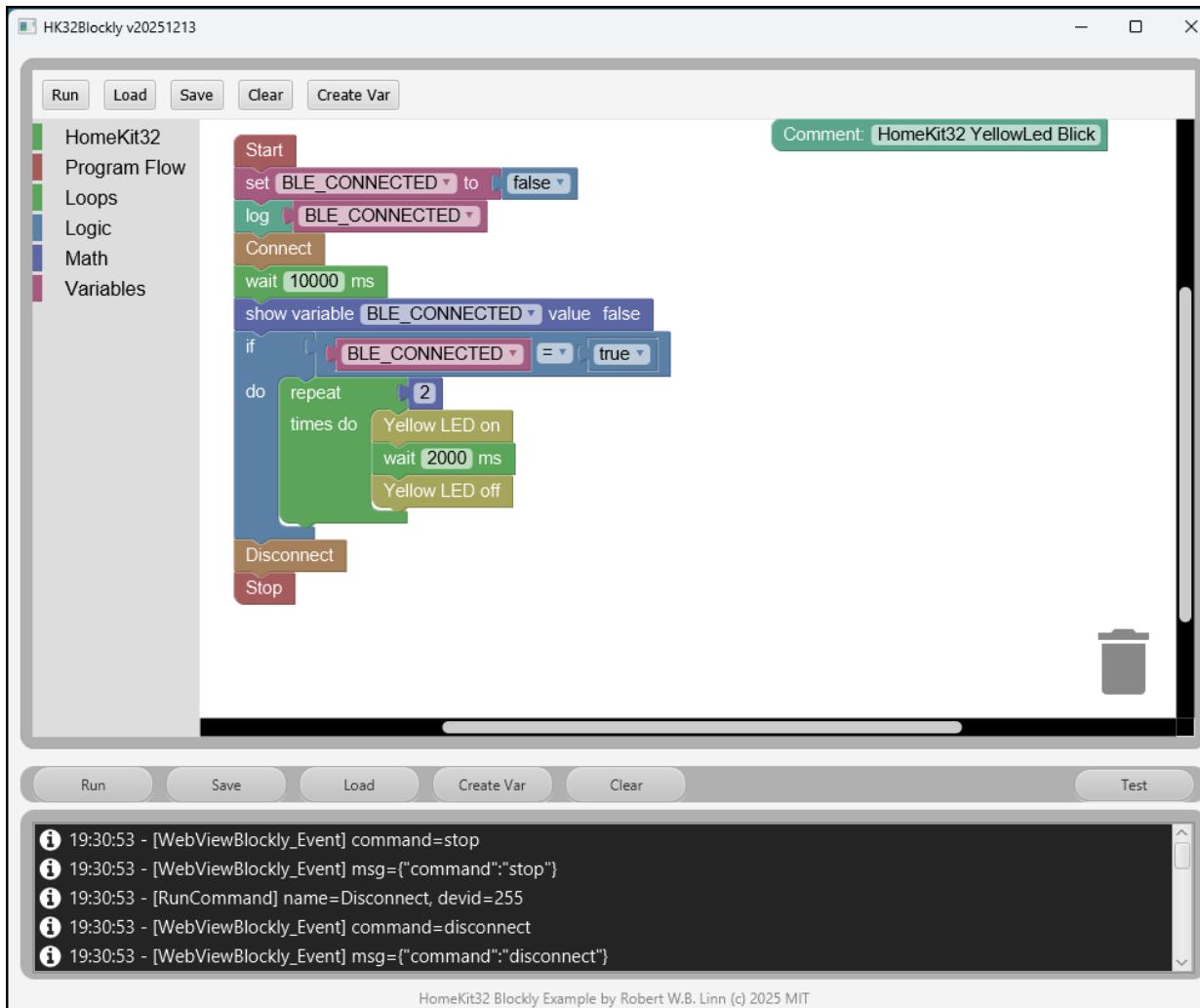
On Windows 11 the “ characters must be escaped \”

MQTT Broker B4J IDE Log

```
[Main.AppStart][I] HomeKit32 Broker Starting
[Main.AppStart][I] Broker IP 192.168.1.94
[Main.AppStart][I] Broker initialize on port 1883
[Main.AppStart][I] Broker started
[Main.AppStart][I] MQTT internal client connected
[Main.MQTT_Connected][I] OK. Subscribed to homekit32/# 
[MQTT_MessageArrived][I] Topic=homekit32/home1/yellow_led/set, Payload={"s":"off"}
[MQTT_MessageArrived][I] Topic=homekit32/home1/yellow_led/set, Payload={"s":"on"}
[MQTT_MessageArrived][I] Topic=homekit32/home1/yellow_led/set, Payload={"s":"on"}
[MQTT_MessageArrived][I] Topic=homekit32/home1/yellow_led/status, Payload={"s":"on"}
[MQTT_MessageArrived][I] Topic=homekit32/home1/yellow_led/set, Payload={"s":"off"}
[MQTT_MessageArrived][I] Topic=homekit32/home1/yellow_led/status, Payload={"s":"off"}
```

B4J BLE Client Blockly (Experimental)

Example custom **Blockly** integration inside a **B4J WebView**, enabling visual programming of **HomeKit32** devices.
Blockly programs are executed in **JavaScript** and communicate back to **B4J** via `executeScript` / **WebView** callbacks
The project is experimental and optimized for Blockly v12.x.



The screenshot shows the B4J BLE Client Blockly workspace. The main area contains a Blockly script:

```
Comment: HomeKit32 YellowLed Blink
Start
set [BLE_CONNECTED] to [false]
log [BLE_CONNECTED]
Connect
wait [10000] ms
show variable [BLE_CONNECTED] value [false]
if [BLE_CONNECTED] = [true]
do
repeat [2]
times do
Yellow LED on
wait [2000] ms
Yellow LED off
Disconnect
Stop
```

The script starts by setting `BLE_CONNECTED` to `false`, logging its value, connecting to a device, and waiting 10 seconds. It then checks if `BLE_CONNECTED` is `true`. If so, it enters a loop that repeats twice, each time turning the yellow LED on for 2 seconds and then off. Finally, it disconnects and stops the program.

At the bottom, a log window displays the following messages:

- i 19:30:53 - [WebViewBlockly_Event] command=stop
- i 19:30:53 - [WebViewBlockly_Event] msg={"command":"stop"}
- i 19:30:53 - [RunCommand] name=Disconnect, devid=255
- i 19:30:53 - [WebViewBlockly_Event] command=disconnect
- i 19:30:53 - [WebViewBlockly_Event] msg={"command":"disconnect"}

At the very bottom, it says "HomeKit32 Blockly Example by Robert W.B. Linn (c) 2025 MIT".

Example Blockly Workspace

Let the Yellow LED blink 2 times.

For testing 2 button panels:

- **Webview panel** to test the Javascript modules in a web browser without B4J connection, because the debug & console windows are required.
- **B4X page panel** has been initially used to test within B4J – but can be removed, because the Webview buttons work also under B4J.

To Do

- Add B4X pages menu toolbar
- Add more Blockly Custom Blocks.
- Make this application generic without dedicated HomeKit32 features.