

HomeKit32

A modular smart-home automation framework for the Keyestudio Smart Home Kit (KS5009), powered by ESP32, BLE, MQTT and B4X.

Overview

HomeKit32 is a modular smart-home control system based on the Keyestudio Smart Home Kit (KS5009) and the Keyestudio ESP32 Plus microcontroller.

It provides real-time communication via BLE and MQTT, integrates 13 sensors and actuators, and exposes a clean, structured protocol for external applications.

The firmware is written in B4R, with performance-critical functionality implemented in wrapped C++ libraries.

Client applications currently exist for B4J, B4A and Python to complete the cross-platform ecosystem.

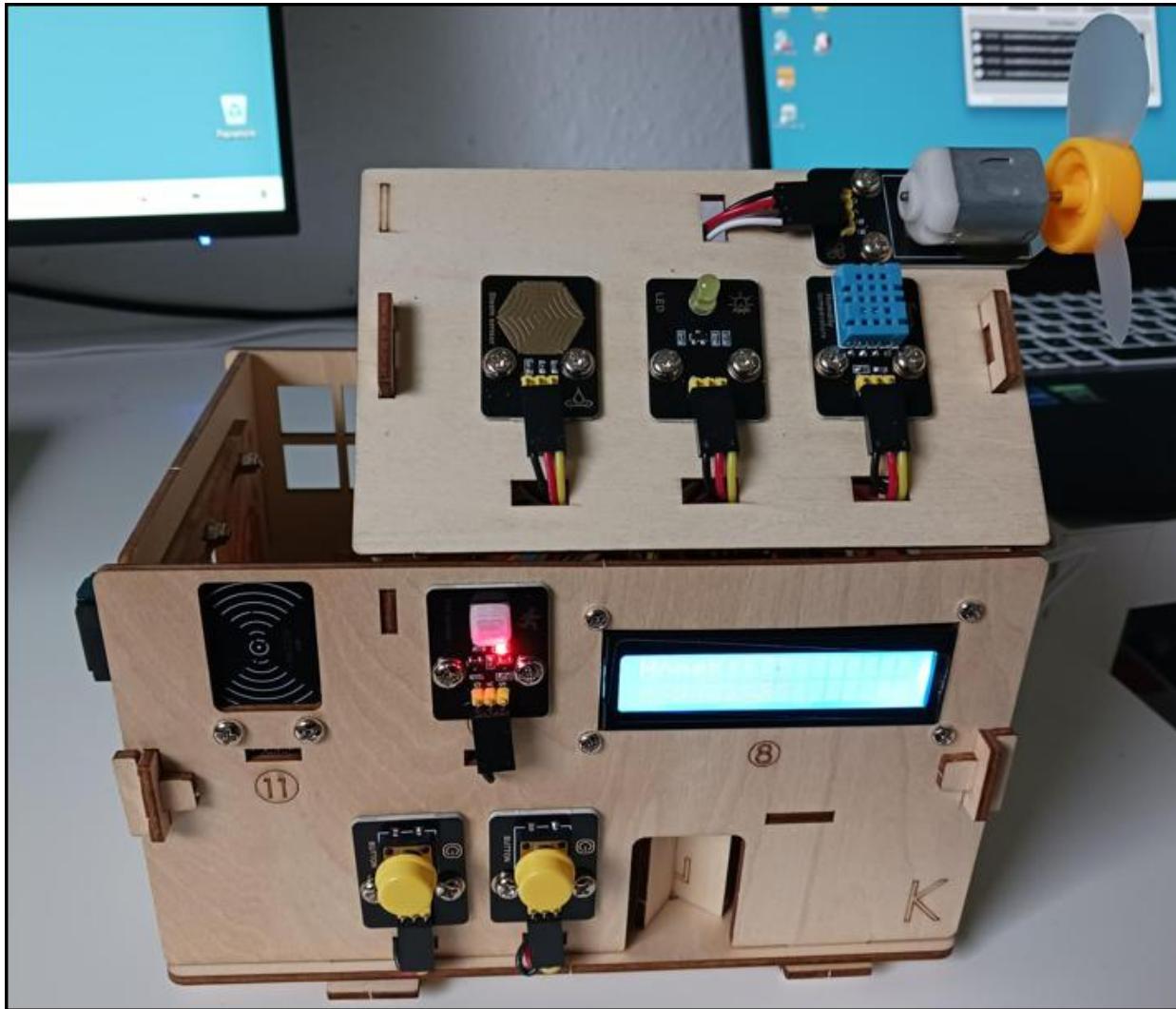
HomeKit32 is ideal for education, prototyping, IoT experiments, and as a reference architecture for a clean, extensible, multi-protocol smart-home system.

IMPORTANT

This example is for educational & personal use only.

It is NOT meant to be used for commercial purposes, nor to replace the original device.

Keyestudio Smart Home Kit ESP32



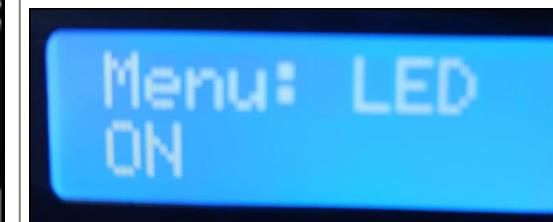
Kit assembled.

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

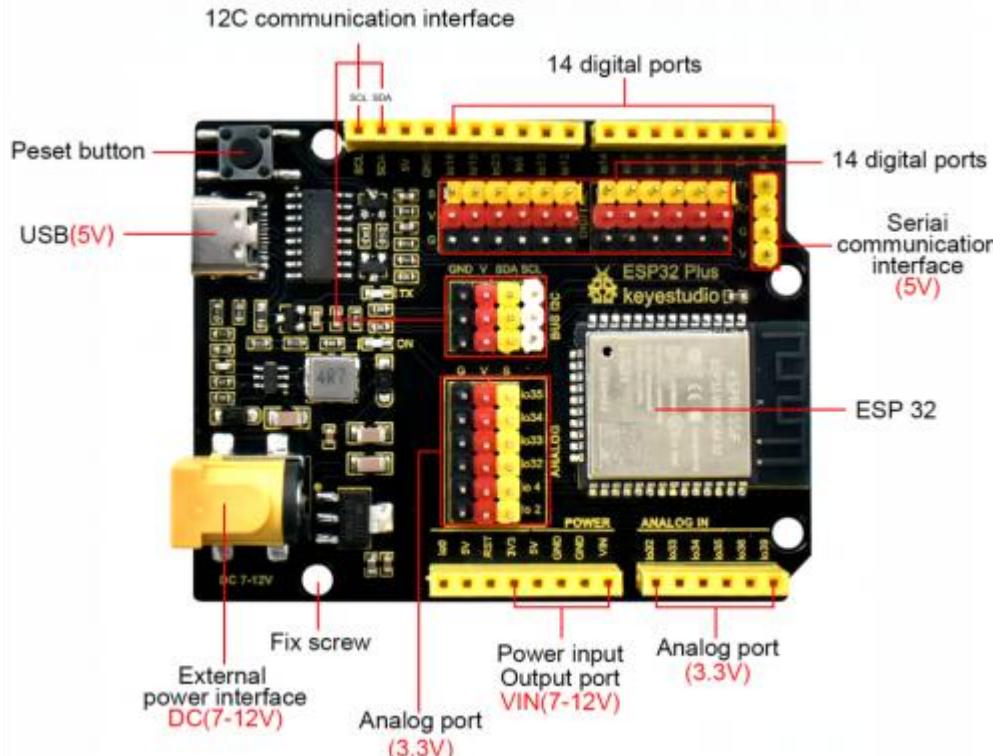


RFID

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



Keyestudio ESP32 Plus Pinout



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 Client HK32HMI

The screenshot shows a user interface for controlling various smart home devices. The top section contains several tiles:

- Alarm: Cleared
- Buzzer: Police
- Clock: Shows the time as approximately 10:10.
- Fan: Off
- Connect: A toggle switch icon.
- Temperature: 17 °C
- Humidity: 59 %RH
- Moisture: 0
- Gas: Cleared
- Door: Closed
- YellowLED: OFF
- RGB LED: A slider with three colored dots (red, green, blue) set to 0, 0, 0.
- RGB LED: OFF
- Motion: Detected (highlighted in yellow)
- Window: Closed

Below this is an "Event Viewer" section displaying the following log entries:

- 10:48:47 - [HandleBLENotification] device=0D, {deviceid=13, command=2, value=1}
- 10:48:47 - [HandleBLENotification] payload=0D0201
- 10:48:43 - [HandleBLENotification] device=0D, {deviceid=13, command=2, value=0}
- 10:48:43 - [HandleBLENotification] payload=0D0200

At the bottom, a footer bar reads: Keyestudio Smart Home Kit ESP32 Control by Robert W.B. Linn (c) 2025 MIT

HMI to control devices over BLE.

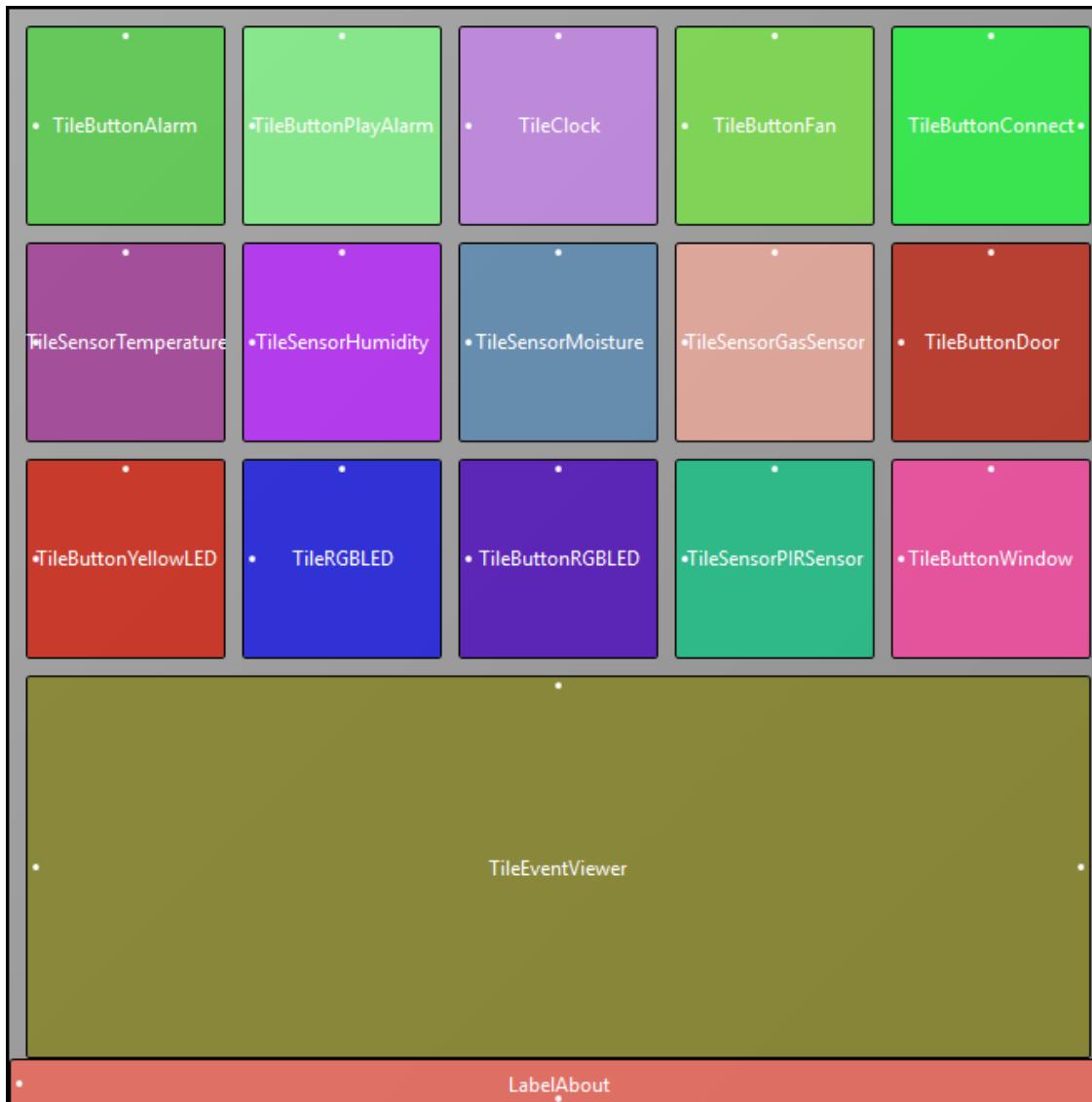
B4J Project

- 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)

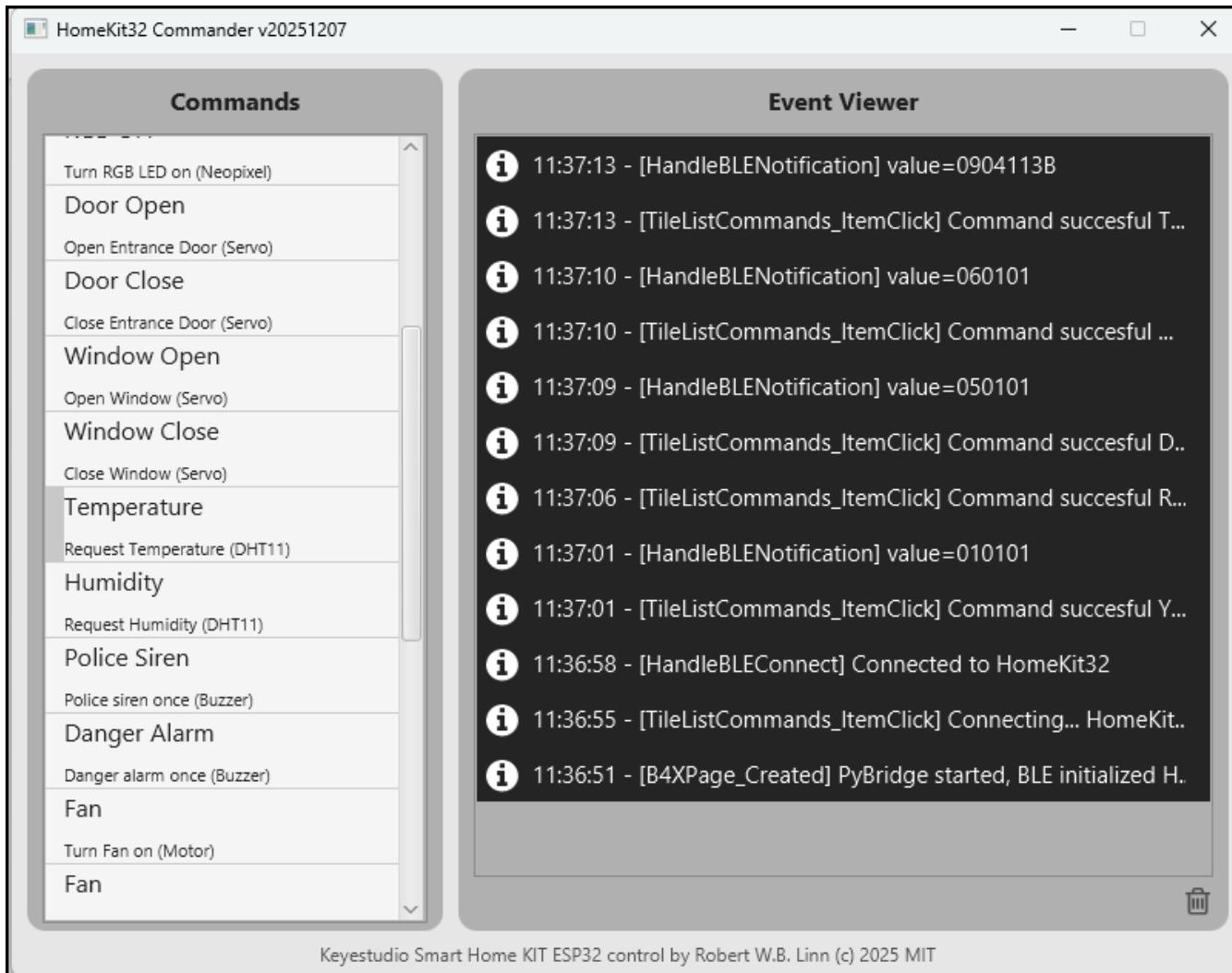
Communication Mode

- 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

HK32HMI – B4J Layout Single Page



B4J Client HK32Commander

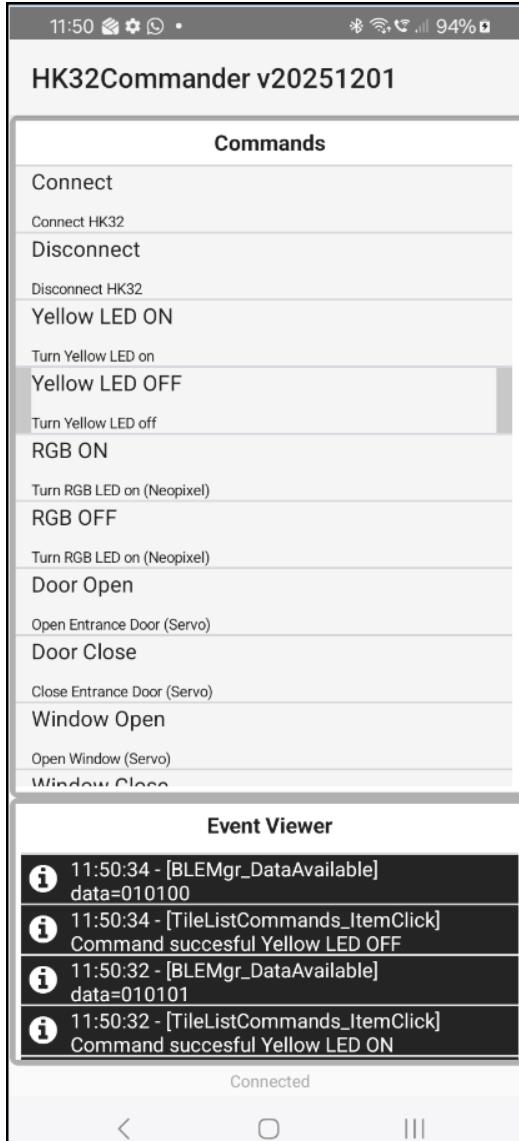


Test Commands to control devices over BLE.

B4J Project

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

B4A Client HK32Commander



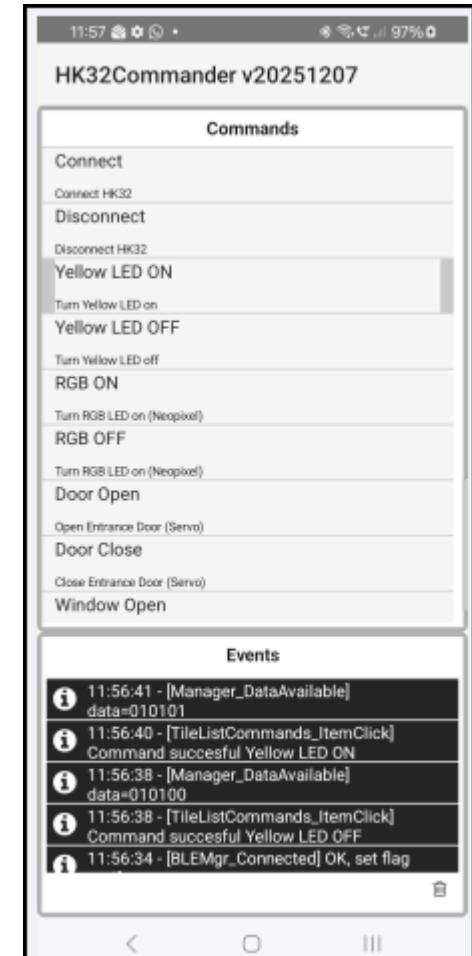
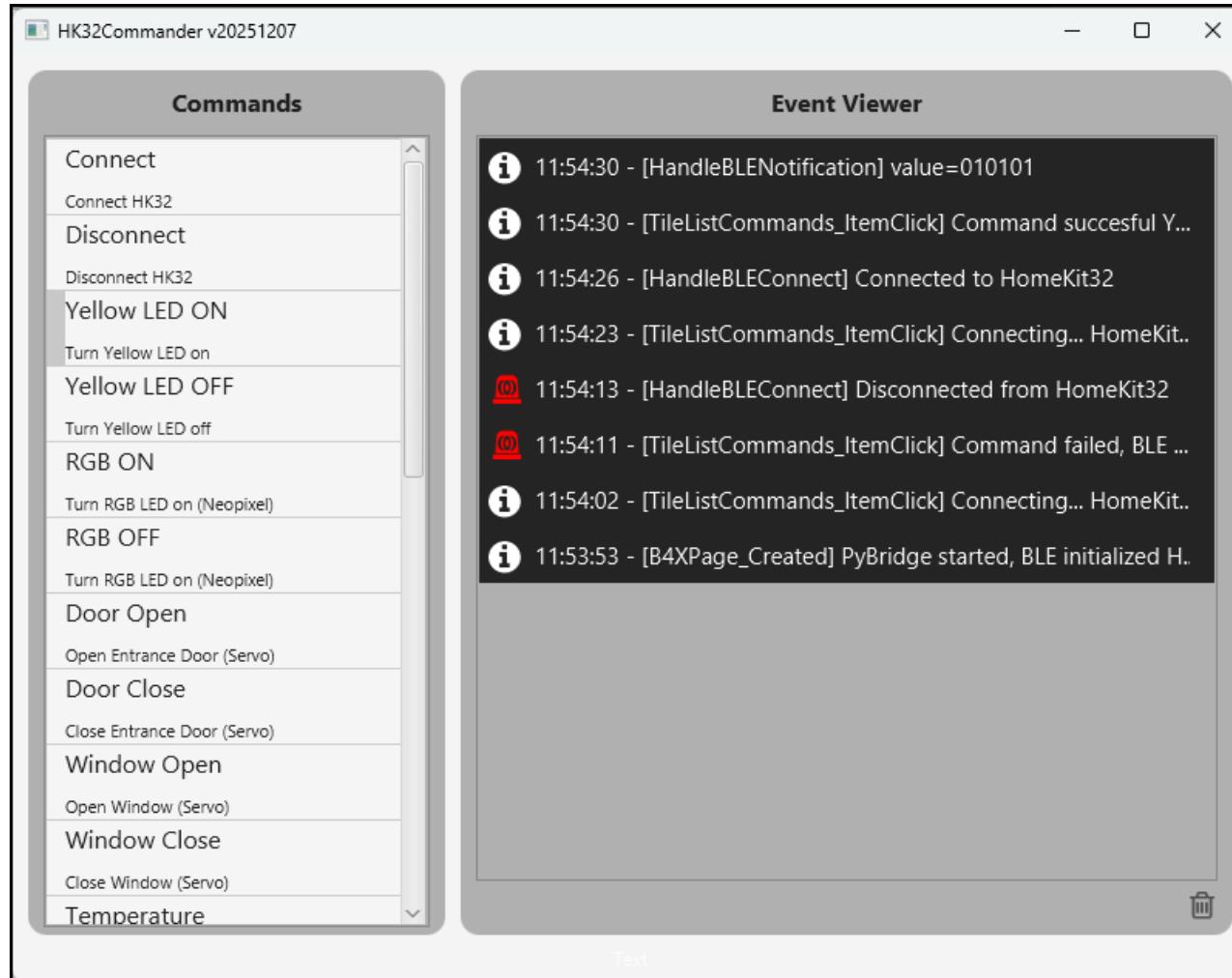
Test Commands to control devices over BLE.

B4A Project

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

B4X Clients HK32Commander

B4A & B4J BLE clients (B4X project) to test direct commands to control devices.



B4X Clients HK32HMI

B4A & B4J BLE clients (B4X project) HMI.

The image displays two side-by-side screenshots of the HK32HMI mobile application interface, version v20251207.

Left Screenshot: This screenshot shows a grid of 12 cards representing various sensors and controls:

- Row 1:** Alarm (Yellow background, labeled "Raining"), Buzzer (Police), Clock icon, Fan (OFF).
- Row 2:** Temperature (20 °C), Humidity (54 %RH), Moisture (0), Gas (--), Door (Closed), Events (ON).
- Row 3:** YellowLED (OFF), RGB LED (0 0 0), RGB LED (OFF), Motion (Detected), Window (Closed).

Event Viewer: A scrollable list of recent events:

- 13:11:59 - [HandleBLENotification] value=0D0201
- 13:11:56 - [HandleBLENotification] value=0B040000
- 13:11:55 - [HandleBLENotification] value=0D0200

Welcome: A simple text message at the bottom of the screen.

Right Screenshot: This screenshot shows a simplified view of the same data, likely representing a different client or a simplified UI:

- Alarm Cleared (Grey card).
- Temperature (20 °C), Humidity (54 %RH), Moisture (0), Motion (--), Gas (--).
- YellowLED (ON), RGB LED (ON).
- Door (Open), Window (Closed), Fan (OFF), Buzzer (Police).

Event Viewer: A scrollable list of recent events (identical to the left screen):

- 13:03:55 - [Manager_DataAvailable] data=0S0101
- 13:03:53 - [Manager_DataAvailable] data=0S0100
- 13:03:52 - [Manager_DataAvailable] data=0S0101
- 13:03:51 - [HandleBLENotification] YellowLED state=true
- 13:03:51 - [Manager_DataAvailable] data=010101
- 13:03:49 - [HandleBLENotification] YellowLED state=false
- 13:03:49 - [Manager_DataAvailable] data=010100
- 13:03:48 - [HandleBLENotification] Window state=true
- 13:03:48 - [Manager_DataAvailable] data=010101
- 13:03:47 - [HandleBLENotification] Window state=false

Python Client HK32HMI

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

TileButton

