Indy test application for Android

Indy test application for Android is a Delphi-based application designed using the IDE RAD Studio Tokyo 10.3. It presents a user-friendly interface which communicates with an IoT board via a WiFi connection. The IoT board acts as a server, whereas the Android device is a client. With this application, IoT (internet of things) projects can be easily designed. The IoT board can be anyone based on the ESP8266 circuit, including the Wemos D1 R2 and the nodeMCU 1.0 boards.

The IoT module must have the sketch test\_indy.ino previously stored in it. This sketch is based on the library esp8266wifi.h. It configurates the internet connection using a local fixed IP address which must be previously known. This address must be a free one available by the router (see intructions in the sketch). The state of the connection and the WiFi signal strength are reported to the user via the serial monitor. In the loop routine, it waits for a client to connect, reads a command sent by the client (if no command is sent, it reads an empty string), and prints the command on the serial monitor.

The Indy test application is based on the use of the IdTCPClient component and the IdThreadComponent. The Indy (internet direct) components are open source easily found in the internet and they are installed in the standard version of the RAD Studio IDE. These components allows to state an internet connection very easily.

In the Android application, the user must first introduce the IP address (local or public) and the port number of the IoT device. The sketch test\_indy.ino sets the local IP address to 192.168.0.200 and the port number to 100. These are default values which can be easily changed in the sketch.

If a public IP address is used in order to state a remote connection, it must be the public IP address of the router to which the IoT device is connected, and this router must have the port number 100 open (port forwarding operation).

Once the IP address and the port number are introduced, the device can be connected pressing the Connect button.

A led drawn on the screen shows the connection status.

A memo log window shows the state of the connection and shows the sentences sent by the server. These sentences can be introduced by writing on the serial monitor.

Sentences can also be written in an edit box in the application screen and sent to de IoT device by pressing the Send button. The IoT device prints them on the serial monitor (if a serial monitor is connected).

These words can be the commands

/LED=ON

The IoT device turns on the builtin led on the board and sends its status to the client

/LED=OFF

The IoT device turns off the builtin led on the board and sends its status to the client

/STATEBUTTON

This is the command for the client to require the state of the builtin button (pressed or unpressed) which will be printed on the Memo Log window.

(This command only works with the nodeMCU board because the Wemos board does not have a builtin button).

/STOPCLIENT

Command to close the connection, which also happens when the disconnect button is pressed.

Other commands can be easily added in the sketch test\_indy.ino in order to control other devices (sensors and actuators) connected to the IoT board.

Android Indy test screen:

