

## OUTPUT:

WOKWI SAVE SHARE Exp 5 REST API with the esp32 Docs FACTS FOR TODAY

sketch.ino • diagram.json • Library Manager • Simulation 00:18.882 (19%)

```
1 // WiFi library for ESP32
2 #include <WiFi.h>
3 #include <HTTPClient.h> // HTTP client library
4
5 // Replace these with your WiFi credentials
6 const char* ssid = "Wokwi-GUEST"; // Wokwi virtual WiFi
7 const char* password = ""; // No password
8
9 void setup() {
10 // Initialize Serial Monitor for debugging
11 Serial.begin(115200);
12 delay(1000);
13
14 // Start connecting to WiFi
15 Serial.print("Connecting to WiFi: ");
16 Serial.println(ssid);
17 WiFi.begin(ssid, password);
18
19 // Wait until the ESP32 connects to WiFi
20 while (WiFi.status() != WL_CONNECTED) {
21 delay(1000);
22 Serial.println("Connecting...");
23 }
24 Serial.println("WiFi Connected!");
25 Serial.print("IP Address: ");
26 Serial.println(WiFi.localIP()); // Print local IP
27
28 // -----
29 // HTTP GET Request
30 // -----
31 HTTPClient http;
32 http.begin("http://jsonplaceholder.typicode.com/posts/1");
33 // Example test API for GET
```

Connecting to WiFi: Wokwi-GUEST  
Connecting...  
Connecting...  
WiFi Connected!  
IP Address: 10.10.0.2  
Sending HTTP GET request...  
GET Response Code: 200



WOKWI SAVE SHARE Exp 5 REST API with the esp32 Docs FACTS FOR TODAY

sketch.ino • diagram.json • Library Manager • Simulation 00:28.948 (10%)

```
9 void setup() {
10 // -----
11 // HTTP GET Request
12 // -----
13 HTTPClient http;
14 http.begin("http://jsonplaceholder.typicode.com/posts/1");
15 // Example test API for GET
16
17 Serial.println("Sending HTTP GET request...");
18 int httpCode = http.GET(); // Perform GET request
19
20 if (httpCode > 0) {
21 // Check for valid response
22 Serial.printf("GET Response Code: %d\n", httpCode);
23 String payload = http.getString();
24 Serial.println(payload);
25 } else {
26 Serial.printf("GET request failed, error: %s\n", http.errorToString());
27 }
28 http.end(); // Free resources
29
30 // -----
31 // HTTP POST Request
32 // -----
33 HTTPClient httpPost;
34 httpPost.begin("http://jsonplaceholder.typicode.com/posts");
35 // Example test API for POST
36 httpPost.addHeader("Content-Type", "application/json"); // Set header
37
38 // Example JSON payload to send
39 String postData = "{\"title\":\"ESP32\", \"body\":\"Hello from ESP32!\"}";
40 Serial.println("Sending HTTP POST request...")
```

GET Response:  
{  
    "userId": 1,  
    "id": 1,  
    "title": "sunt aut facere repellat provident occaecati excepturi optio reprehenderit",  
    "body": "quia et suscipit\\nsuscipit recusandae consequuntur expedita et  
cum\\nreprehenderit molestiae ut ut quas totam\\nnostrum rerum est autem sunt rem eveniet  
architecto"  
}  
Sending HTTP POST request...  
POST Response Code: 201  
POST Response:  
{  
    "title": "ESP32",  
    "body": "Hello from ESP32!",  
    "userId": 1,  
    "id": 101

## OUTPUT:

WOKWI • SAVE ▾ • SHARE • EXP 7 Docs

sketch.ino • diagram.json • Library Manager

```
1 #include <Arduino.h> // Required for ESP32 Arduino framework
2
3 // -----
4 // Task 1: Runs every 1 second
5 // -----
6 void Task1(void *pvParameters) {
7     while (1) {
8         Serial.println("Task 1 is running");
9         vTaskDelay(1000 / portTICK_PERIOD_MS); // Delay 1 second
10    }
11 }
12
13 // -----
14 // Task 2: Runs every 0.5 seconds
15 // -----
16 void Task2(void *pvParameters) {
17     while (1) {
18         Serial.println("Task 2 is running");
19         vTaskDelay(500 / portTICK_PERIOD_MS); // Delay 0.5 seconds
20    }
21 }
22
23 void setup() {
24     // Initialize Serial Monitor
25     Serial.begin(115200);
26     delay(1000); // Give Serial time to start
27
28 // -----
29 // Create FreeRTOS Tasks
30 // -----
31 // Parameters:
32 // 1. Task function
33 // 2. Task name (for debugging)
34 }
```

Simulation

00:06.849 20%



Task 1 is running  
Task 2 is running  
Task 2 is running  
Task 1 is running  
Task 2 is running  
Task 1 is running

WOKWI • SAVE ▾ • SHARE • EXP 7 Docs

sketch.ino • diagram.json • Library Manager

```
16 void Task2(void *pvParameters) {
17 }
18
19 void setup() {
20     // Initialize Serial Monitor
21     Serial.begin(115200);
22     delay(1000); // Give Serial time to start
23
24 // -----
25 // Create FreeRTOS Tasks
26 // -----
27 // Parameters:
28 // 1. Task function
29 // 2. Task name (for debugging)
30 // 3. Stack size (in words, not bytes)
31 // 4. Task input parameters (NULL if not needed)
32 // 5. Priority (higher number = higher priority)
33 // 6. Task handle (NULL if not needed)
34
35 xTaskCreate(Task1, "Task 1", 1000, NULL, 1, NULL); // Create Task1
36 xTaskCreate(Task2, "Task 2", 1000, NULL, 1, NULL); // Create Task2
37
38 }
39
40 void loop() {
41     // Empty - FreeRTOS handles task scheduling
42 }
```

Simulation

00:15.816 17%



Task 2 is running  
Task 1 is running  
Task 2 is running  
Task 2 is running  
Task 1 is running  
Task 2 is running  
Task 2 is running  
Task 1 is running  
Task 2 is running  
Task 2 is running  
Task 1 is running  
Task 2 is running  
Task 2 is running  
Task 1 is running

## OUTPUT:

WOKWI SAVE SHARE wifi-scan.ino by urish Docs FACTS DATA

wifi-scan.ino • diagram.json Library Manager

```
1 #include <Arduino.h>
2
3 // -----
4 // Flags for simulation
5 // -----
6 bool deviceConnected = false; // Simulated connection status
7 unsigned long lastToggle = 0; // For toggling connection
8
9 // -----
10 // Task 1: Simulate BLE notification every 1 second
11 // -----
12 void Task1(void *pvParameters) {
13     while (1) {
14         if (deviceConnected) {
15             String msg = "Hello from ESP32 (simulated BLE notification)";
16             Serial.println("[Task1] Sent: " + msg);
17         } else {
18             Serial.println("[Task1] No device connected");
19         }
20         vTaskDelay(1000 / portTICK_PERIOD_MS); // Delay 1 sec
21     }
22 }
23
24 // -----
25 // Task 2: Background serial prints every 0.5 seconds
26 // -----
27 void Task2(void *pvParameters) {
28     while (1) {
29         Serial.println("[Task2] Running background task");
30         vTaskDelay(500 / portTICK_PERIOD_MS); // Delay 0.5 sec
31     }
32 }
33
```

Simulation ↻ ▢ || 00:26.066 ⚡20%



load:0x40088400,len:2972  
entry 0x400880dc  
[Setup] ESP32 BLE Simulation (Serial Monitor Only)  
[Task1] No device connected  
[Task2] Running background task  
[Task2] Running background task  
[Task1] No device connected

WOKWI SAVE SHARE wifi-scan.ino by urish Docs FACTS DATA

wifi-scan.ino • diagram.json Library Manager

```
1 #include <Arduino.h>
2
3 // -----
4 // Flags for simulation
5 // -----
6 bool deviceConnected = false; // Simulated connection status
7 unsigned long lastToggle = 0; // For toggling connection
8
9 // -----
10 // Task 1: Simulate BLE notification every 1 second
11 // -----
12 void Task1(void *pvParameters) {
13     while (1) {
14         if (deviceConnected) {
15             String msg = "Hello from ESP32 (simulated BLE notification)";
16             Serial.println("[Task1] Sent: " + msg);
17         } else {
18             Serial.println("[Task1] No device connected");
19         }
20         vTaskDelay(1000 / portTICK_PERIOD_MS); // Delay 1 sec
21     }
22 }
23
24 // -----
25 // Task 2: Background serial prints every 0.5 seconds
26 // -----
27 void Task2(void *pvParameters) {
28     while (1) {
29         Serial.println("[Task2] Running background task");
30         vTaskDelay(500 / portTICK_PERIOD_MS); // Delay 0.5 sec
31     }
32 }
33
```

Simulation ↻ ▢ || 00:33.833 ⚡30%



[Task2] Running background task  
[Task1] Sent: Hello from ESP32 (simulated BLE notification)  
[Task2] Running background task  
[Task2] Running background task  
[Task1] Sent: Hello from ESP32 (simulated BLE notification)  
[Task2] Running background task  
[Task2] Running background task  
[Task1] Sent: Hello from ESP32 (simulated BLE notification)  
[Task2] Running background task  
[Task2] Running background task  
[Task1] Sent: Hello from ESP32 (simulated BLE notification)  
[Task2] Running background task  
[Task2] Running background task  
[BLE] Device disconnected (simulated)  
[Task2] Running background task  
[Task1] No device connected  
[Task2] Running background task  
[Task2] Running background task