

Read querystring

Alter the config, so the uri_match_fn is wildcard match

```
httpd_config_t config = HTTPD_DEFAULT_CONFIG();
config.server_port = 80;
config.uri_match_fn = httpd_uri_match_wildcard;

httpd_uri_t index_uri = {
    .uri      = "/*",
    .method   = HTTP_GET,
    .handler  = file_handler,
    .user_ctx = NULL
};
```

In the file handler, one can now start to retrieve the query string url.com?test=anser

```
char query[100];
size_t query_len = sizeof(query);
// Get the query string
if (httpd_req_get_url_query_str(req, query, query_len) == ESP_OK) {
    ESP_LOGI(TAG, "Found URL query => %s", query);

    // Parse specific query parameters
    char param[32];
    if (httpd_query_key_value(query, "param", param, sizeof(param)) == ESP_OK) {
        ESP_LOGI(TAG, "Found URL query parameter => param=%s", param);
    }
}
```

This comes together in this sample:

```
#include "esp_http_server.h"
#include
static const char *TAG = "example";
// Replace with your network credentials
const char* ssid = "YourWifi";
const char* password = "YourPass";

httpd_handle_t stream_httpd = NULL;
```

```

httpd_handle_t camera_httpd = NULL;

static esp_err_t file_handler(httpd_req_t *req) {

char query[100];
size_t query_len = sizeof(query);
// Get the query string
if (httpd_req_get_url_query_str(req, query, query_len) == ESP_OK) {
    ESP_LOGI(TAG, "Found URL query => %s", query);

    // Parse specific query parameters
    char param[32];
    if (httpd_query_key_value(query, "param", param, sizeof(param)) == ESP_OK) {
        ESP_LOGI(TAG, "Found URL query parameter => param=%s", param);
    }
}

Serial.printf("File handler");

char ret_homepage[255] = "";

strcpy(ret_homepage, "Hello world");

/* Set some custom headers */
httpd_resp_set_hdr(req, "Connection", "close");
httpd_resp_set_hdr(req, "Cache-Control", "no-cache");

/* Send response with custom headers and body set as the
 * string passed in user context*/
const char *resp_str = (const char*) ret_homepage;

Serial.print(ret_homepage);
httpd_resp_send(req, resp_str, HTTPD_RESP_USE_STRLEN);
ESP_LOGI(TAG, "Response sent for home page request.Time:%s", esp_log_system_timestamp());
return ESP_OK;
}

void StartServer(){
    httpd_config_t config = HTTPD_DEFAULT_CONFIG();
    config.server_port = 80;
    config.uri_match_fn = httpd_uri_match_wildcard;

    httpd_uri_t index_uri = {
        .uri      = "/*",
        .method    = HTTP_GET,
        .handler   = file_handler,
        .user_ctx  = NULL
    };

    Serial.printf("Starting server on port: '%d'\n", config.server_port);
    if (httpd_start(&stream_httpd, &config) == ESP_OK) {
        httpd_register_uri_handler(stream_httpd, &index_uri);
    }
}

```

```
void setup() {  
  Serial.begin(115200);  
  Serial.setDebugOutput(true);  
  Serial.println("Started");  
  
  Serial.println("Wifi");  
  // Wi-Fi connection  
  WiFi.begin(ssid, password);  
  while (WiFi.status() != WL_CONNECTED) {  
    delay(500);  
    Serial.print(".");  
  }  
  Serial.println("");  
  Serial.println("WiFi connected");  
  
  Serial.print("Camera Stream Ready! Go to: http://");  
  Serial.println(WiFi.localIP());  
  
  Serial.println("Start Server");  
  StartServer();  
  
}  
  
void loop() {  
  Serial.println("Loop");  
  //s_listFiles("/sdcard");  
  // captureImage();  
  delay(10000);  
  
}
```

Revision #1

Created 25 September 2024 21:22:05 by Tim

Updated 25 September 2024 21:24:33 by Tim