D01 - Esempio programma C - API HTTP REST libmicrohttpd

Esempio pratico e funzionante di programma in C che espone una `API HTTP REST` (es. `/hello`) usando una libreria leggera e molto usata: `libmicrohttpd`.

È ideale per:

- servizi REST semplici
- API interne di sistema
- embedded / IoT
- interfaccia con script o sistemi esterni

? Esempio completo: API in C (GET /hello)

? Requisiti

Installa la libreria:

```
sudo apt-get install libmicrohttpd-dev
```

? api_server.c

```
#include <microhttpd.h>
#include <string.h>
#include <stdlib.h>

#define PORT 8080

// Funzione che gestisce le richieste HTTP
static int answer_to_connection(
    void *cls,
    struct MHD_Connection *connection,
    const char *url,
    const char *method,
    const char *version,
    const char *upload_data,
    size_t *upload_data_size,
    void **con_cls)
{
    const char *response_msg;
```

```
    if (strcmp(url, "/hello") == 0 && strcmp(method, "GET") == 0) {
        response_msg = "{\"message\": \"Hello from API in C!\"}";
    }
    else {
        response_msg = "{\"error\": \"Endpoint not found\"}";
    }

    struct MHD_Response *response;
    int ret;

    response = MHD_create_response_from_buffer(
        strlen(response_msg),
        (void*) response_msg,
        MHD_RESPMEM_PERSISTENT);

    MHD_add_response_header(response, "Content-Type", "application/json");
    ret = MHD_queue_response(connection, MHD_HTTP_OK, response);
    MHD_destroy_response(response);

    return ret;
}

int main() {
    struct MHD_Daemon *daemon;

    daemon = MHD_start_daemon(
        MHD_USE_INTERNAL_POLLING_THREAD, PORT, NULL, NULL,
        &answer_to_connection, NULL, MHD_OPTION_END);

    if (daemon == NULL) {
        fprintf(stderr, "Error starting server\n");
        return 1;
    }

    printf("Server running on port %d\n", PORT);
    printf("Try: curl http://localhost:%d/hello\n", PORT);

    getchar();  // Ferma il server quando premi INVIO

    MHD_stop_daemon(daemon);
    return 0;
}
```

? Compilazione

```
gcc api_server.c -o api_server -lmicrohttpd
```

▶ Esecuzione

```
./api_server
```

? Test con curl

curl http://localhost:8080/hello

Output:

```
{"message": "Hello from API in C!"}
```

? Struttura del Progetto

```
rest_api_c/
??? src/
?   ??? api_server.c
?   ??? router.c
?   ??? router.h
?   ??? utils.c
??? include/
?   ??? router.h
?   ??? utils.h
??? Makefile
??? config.json   (esempio)
```

? 1. src/api_server.c — Avvio server REST

```
#include <microhttpd.h>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "router.h"

#define PORT 8080

static int answer_to_connection(
    void *cls,
    struct MHD_Connection *connection,
    const char *url,
    const char *method,
    const char *version,
    const char *upload_data,
    size_t *upload_data_size,
```

```
    void **con_cls)
{
    return route_request(connection, url, method);
}

int main() {
    struct MHD_Daemon *daemon;

    daemon = MHD_start_daemon(
        MHD_USE_INTERNAL_POLLING_THREAD, PORT, NULL, NULL,
        &answer_to_connection, NULL, MHD_OPTION_END);

    if (daemon == NULL) {
        fprintf(stderr, "Errore avvio server\n");
        return 1;
    }

    printf("Server REST avviato su porta %d\n", PORT);
    printf("Esempio: curl http://localhost:%d/hello\n", PORT);

    getchar();   // fermare server con INVIO
    MHD_stop_daemon(daemon);
    return 0;
}
```

? 2. src/router.c — Gestione endpoint REST

```
#include <microhttpd.h>
#include <string.h>
#include <stdio.h>
#include "router.h"
#include "utils.h"

int route_request(struct MHD_Connection *connection, const char *url, const char
*method) {
    const char *response_msg;

    if (strcmp(url, "/hello") == 0 && strcmp(method, "GET") == 0) {
        response_msg = "{\"message\": \"Hello from REST API C\"}";
        return send_response(connection, response_msg, MHD_HTTP_OK);
    }
    else if (strcmp(url, "/status") == 0 && strcmp(method, "GET") == 0) {
        response_msg = "{\"status\": \"OK\", \"service\": \"running\"}";
        return send_response(connection, response_msg, MHD_HTTP_OK);
    }
    else {
        response_msg = "{\"error\": \"Not Found\"}";
```

```
        return send_response(connection, response_msg, MHD_HTTP_NOT_FOUND);
    }
}
```

? 3. include/router.h

```
#ifndef ROUTER_H
#define ROUTER_H

#include <microhttpd.h>

int route_request(struct MHD_Connection *connection, const char *url, const char
*method);

#endif
```

? 4. src/utils.c — Funzioni per risposte HTTP

```
#include <microhttpd.h>
#include <string.h>
#include "utils.h"

int send_response(struct MHD_Connection *connection, const char *msg, int
status_code) {
    struct MHD_Response *response = MHD_create_response_from_buffer(
        strlen(msg), (void*) msg, MHD_RESPMEM_PERSISTENT);

    MHD_add_response_header(response, "Content-Type", "application/json");

    int ret = MHD_queue_response(connection, status_code, response);
    MHD_destroy_response(response);
    return ret;
}
```

? 5. include/utils.h

```
#ifndef UTILS_H
#define UTILS_H

#include <microhttpd.h>

int send_response(struct MHD_Connection *connection, const char *msg, int
status_code);
```

```
#endif
```

? 6. Makefile

```
CC = gcc
CFLAGS = -Wall -Iinclude
LIBS = -lmicrohttpd

SRC = src/api_server.c src/router.c src/utils.c
OBJ = $(SRC:.c=.o)

TARGET = rest_api

all: $(TARGET)

$(TARGET): $(OBJ)
$(CC) $(OBJ) -o $(TARGET) $(LIBS)

clean:
rm -f $(OBJ) $(TARGET)
```

? **Test API**

Compila ed esegui:

```
make
./rest_api
```

Test via curl:

```
curl http://localhost:8080/hello
curl http://localhost:8080/status
```

Output:

```
{"message": "Hello from REST API C"}
{"status": "OK", "service": "running"}
```

Tutorials:

https://www.gnu.org/software/libmicrohttpd/tutorial.html#Hello-browser-example

https://docs.oracle.com/cd/E88353_01/html/E37842/libmicrohttpd-3lib.html

---

Revisione #3
Creato 18 novembre 2025 18:42:59 da Giuseppe Marangi
Aggiornato 19 novembre 2025 05:40:08 da Giuseppe Marangi