UNIVERSIDAD COMPLUTENSE MADRID

ESP-IDF. Events (Event Loop)

ESP-IDF (not only FreeRTOS)

- FreeRTOS provides *EventGroup* to block tasks waiting for events
  - So several tasks may wake-up when an event arrives
  - An event is anything that wakes-up a blocked task: writting a queue, post a semaphore...
- ESP-IDF imeplements *EventGroup* and includes *Event Loop*
  - Works with *callbacks:* we may register events and any task may define a *handler* to that event (code that will be executed when the event is issued)
  - Example: WiFI task sends an event when a connection to network happens. We may declare *handlers (functions)* that will be executed when the event raises.

Events y *event loops*

- Event → something important happened
  - Example. Blueetooth pairing, button pressed...
  - WE (programmers) define what is an event
- Event Loop → ESP-IDF mechanism to send/receive events
  - Se suele crear una tarea específica para un *event loop*
- Steps
  1. Create *event loop:* `esp_event_loop_create()`
  2. Define one ore more *handlers:* `esp_event_handler_t`
  3. Register those *handlers:* `esp_event_handler_regsiter_with()`
  4. Send events: `esp_event_post_to()`

Events: Base - ID

- ESP-IDF event has two parts
  - BASE → *event familiy*
  - ID → event identifier (in its familiy)
- Define families (BASE)
  - `ESP_EVENT_DECLARE_BASE(`*`nombre-base`*`)` → to include in header files
  - `ESP_EVENT_DEFINE_BASE(`*`nombre-base`*`)` → *to source files*
  - `ESP_EVENT_ANY_BASE`
- ID definitions: use enum

```
enum {
    EVENT_ID_1,
    EVENT_ID_2,
    EVENT_ID_3,
    ... }
```

  - `ESP_EVENT_ANY_ID`

Naming the events

- Thera are many eventis defined in ESP-IDF
- The follow a naming rule that you shoulf follow
- BASE → <name>_EVENT
  - WIFI_EVENT
  - ETHERNET_EVENT
- ID → enum In general: <base>_<id>
  - WIFI_EVENT_WIFI_READY, WIFI_EVENT_SCAN_DONE, WIFI_EVENT_STA_START...
  - ETHERNET_EVENT_START, ETHERNET_EVENT_STOP...
- All events in the docs:
  - Eth -> https://docs.espressif.com/projects/esp-idf/en/stable/api-reference/network/esp_eth.html
  - Wifi -> https://docs.espressif.com/projects/esp-idf/en/stable/api-reference/network/esp_wifi.html

loop event

- Init struct:

```
esp_event_loop_handle_t    loop_with_task;
esp_event_loop_args_t loop_with_task_args = {
        .queue_size = 5,
        .task_name = "loop_task", // task will be created
        .task_priority = uxTaskPriorityGet(NULL),
        .task_stack_size = 2048,
        .task_core_id = tskNO_AFFINITY
    };
```

- Creat *loop event*
  - If *.task_name* is not NULL, a new task is created
  - Otherwise we should create one and periodically call invoque periódicamente a `esp_event_loop_run()`

```
esp_event_loop_create(&loop_with_task_args, &loop_with_task)
```

Exameple

https://github.com/espressif/esp-idf/tree/v4.1/examples/system/esp_event/user_event_loops/main

Header file (eventos.h)

```
ESP_EVENT_DECLARE_BASE(TASK_EVENT);

enum {
    TASK_EVENT_ITERATION
};
```

Example

Source file (main.c)

```
esp_event_loop_handle_t loop_with_task;
ESP_EVENT_DEFINE_BASE(TASK_EVENT);

void app_main(void)
{
    esp_event_loop_args_t loop_with_task_args = {
        .queue_size = 5,
        .task_name = "loop_task", // task will be created
        .task_priority = uxTaskPriorityGet(NULL),
        .task_stack_size = 2048,
        .task_core_id = tskNO_AFFINITY
    };

esp_event_loop_create(&loop_with_task_args, &loop_with_task);

esp_event_handler_register_with(loop_with_task, TASK_EVENT, TASK_EVENT_ITERATION,
                                task_iteration_handler, loop_with_task));
```

Example (cont)

New task to send events

```
static void task_event_source(void* args)
{
    for (int iteration = 1; iteration <= TASK_ITERATIONS_COUNT; iteration++) {

        esp_event_post_to(loop_with_task, TASK_EVENT, TASK_EVENT_ITERATION,
                                          &iteration, sizeof(iteration), portMAX_DELAY));

        vTaskDelay(pdMS_TO_TICKS(TASK_PERIOD));
    }

    vTaskDelay(pdMS_TO_TICKS(TASK_PERIOD));
    vTaskDelete(NULL);
}
```

- System creates a copy of the data sent (in this case, *iteration)*

Example (cont)

Handler function

```
static void task_iteration_handler(void* handler_args, esp_event_base_t base, int32_t id, void* event_data)
{
        int iteration = *((int*) event_data);

        if (handler_args == loop_with_task) {

            printf("OK. Value %d\n", iteration);
        }
        else {
            printf("Weird!!!\n");
        }

}
```

Default Event Loop

- There exists a default *event loop* for system events
  - API similar to generic event loop
- Example

https://github.com/espressif/esp-idf/tree/v4.1/examples/system/esp_event/default_event_loop/main