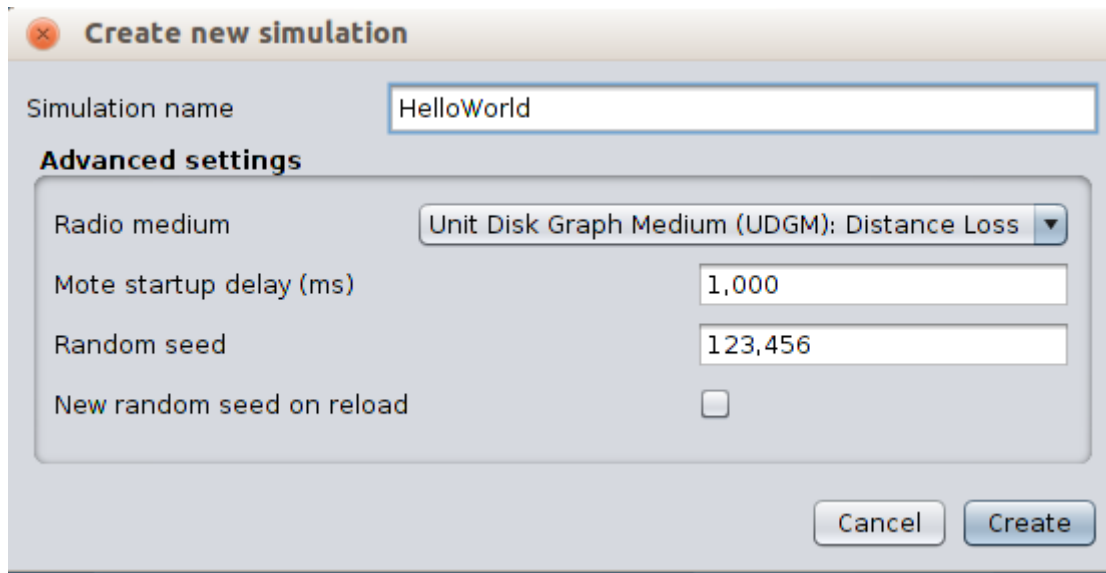


Fiche TP N°3 : Technologies Sans Fils

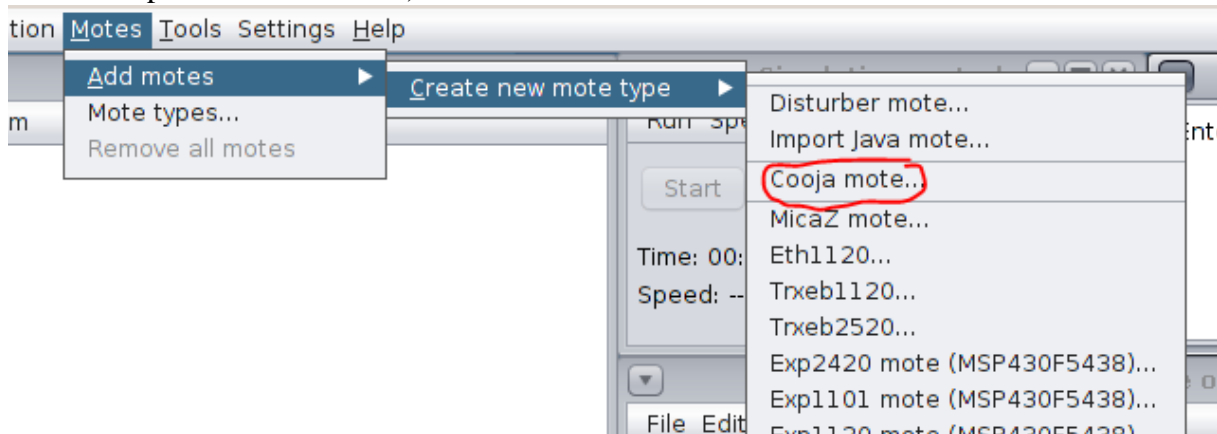
Solution :

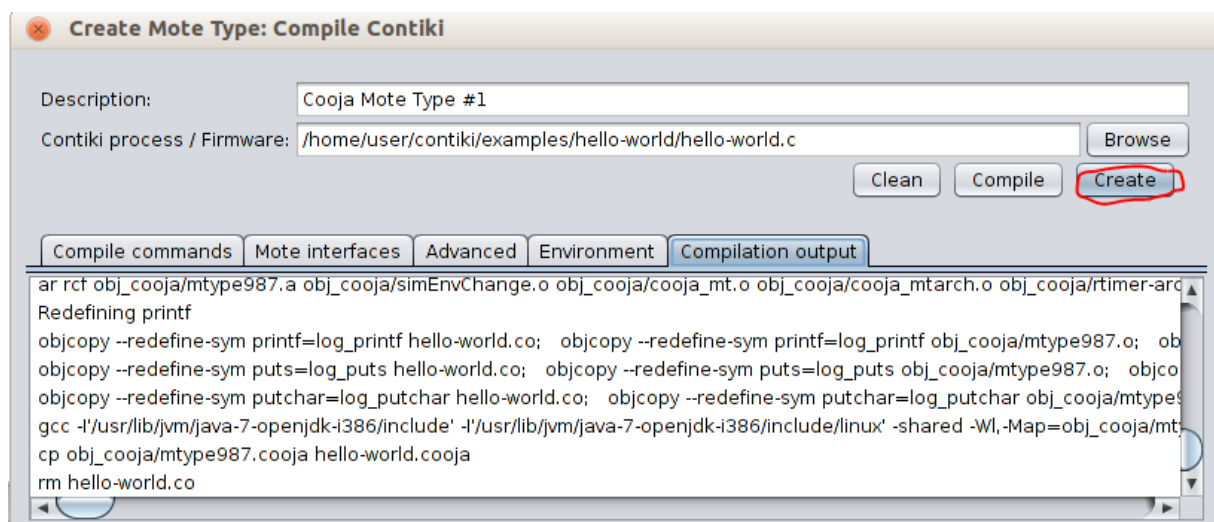
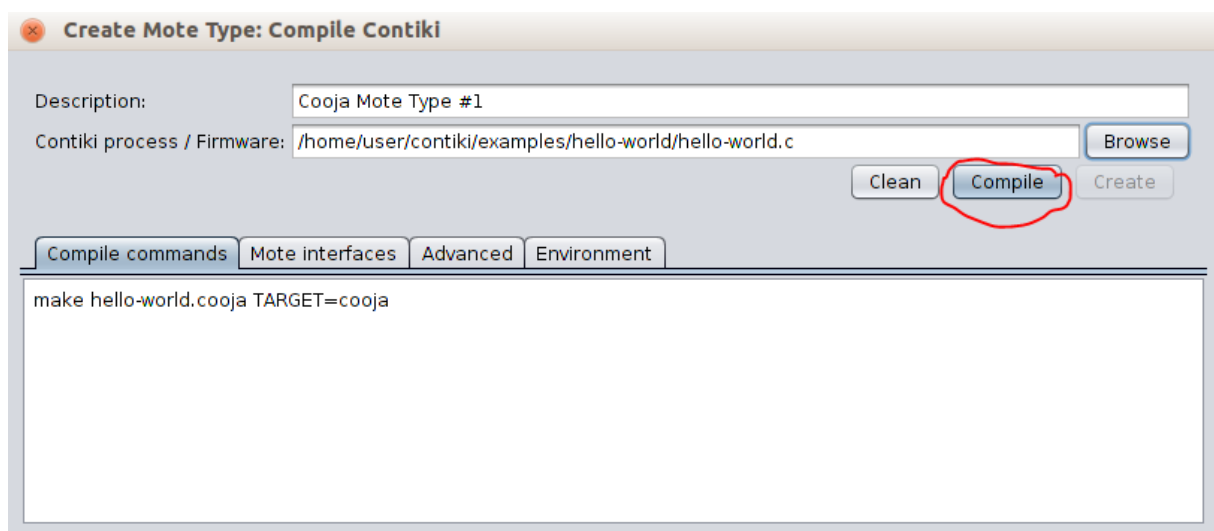
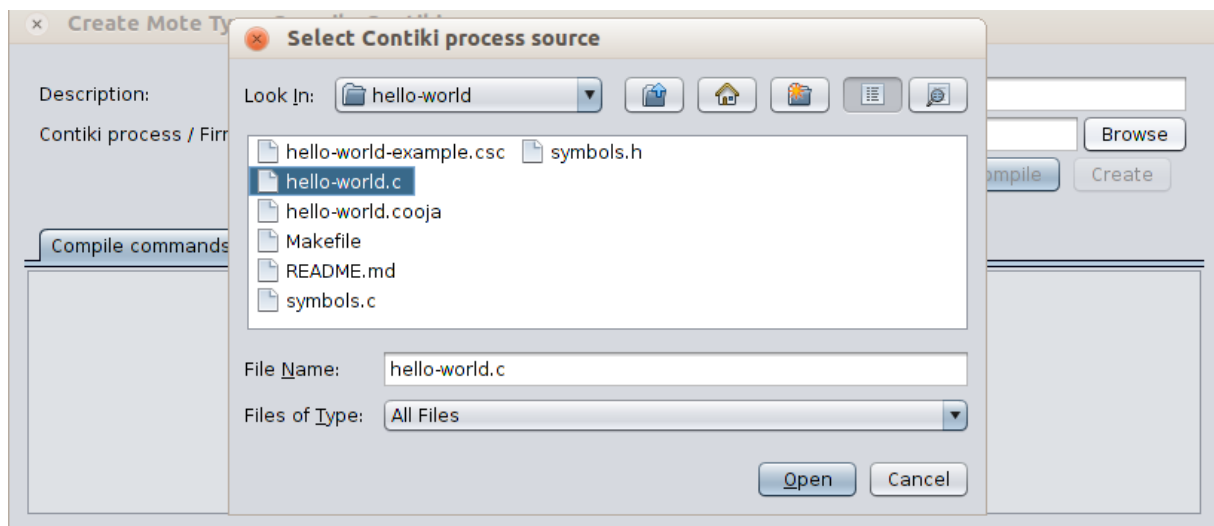
1. Premier Exemple :

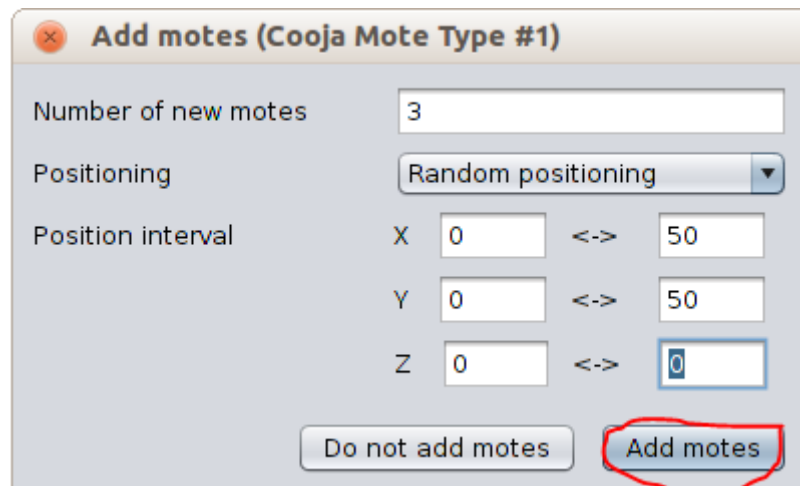
1. D'abord, créer une nouvelle simulation appelée : **HelloWorld**



2. Ensuite il faut ajouter des capteurs (*motes*) dans votre environnement (dans notre cas ça sera des capteurs de simulation) :

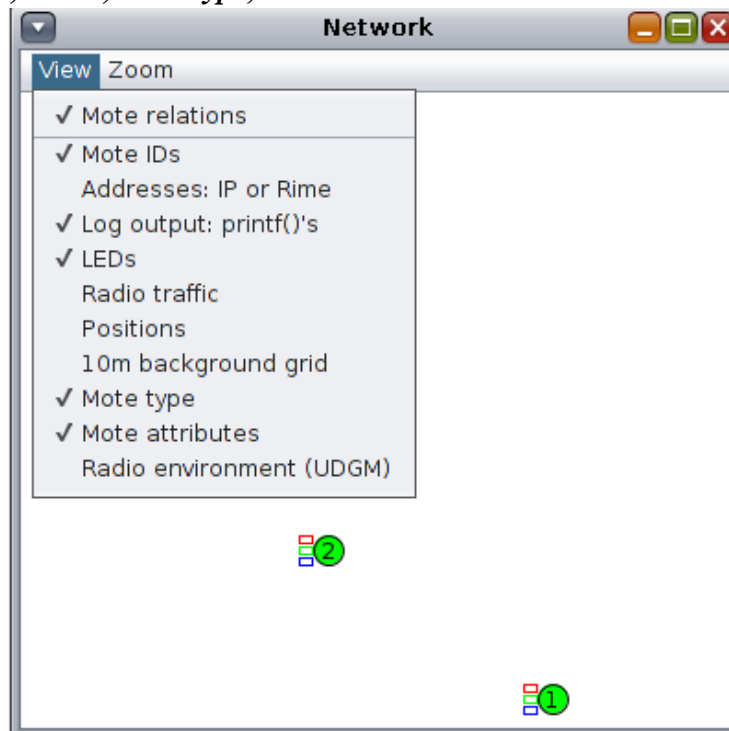




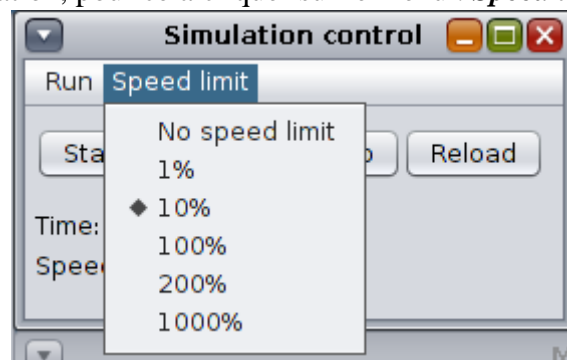


3. Avant de lancer la simulation, il faut configurer quelques paramètres :

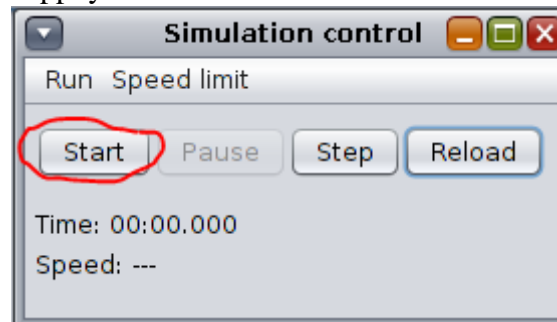
- Dans la fenêtre Network, cliquer sur **View**, et cocher les options suivantes : **Mots IDs**, **Log output**, **LEDs**, **Mote type**, **Mote attributes** :



- Dans la fenêtre **Simulation Control** réduire le vitesse du temps afin de pouvoir visualiser la simulation, pour cela cliquer sur le menu : **Speed limit** -> **10%**.

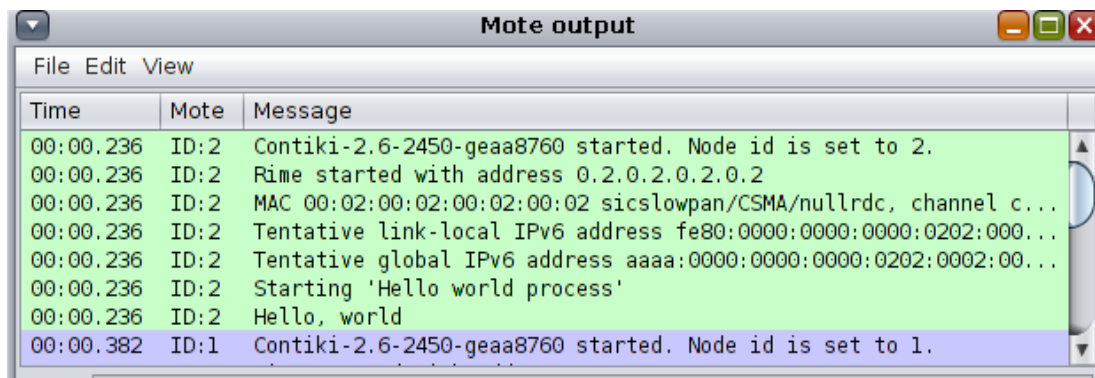


4. Lancer la simulation en appuyant sur le bouton start de la fenêtre **Simulation Control**.



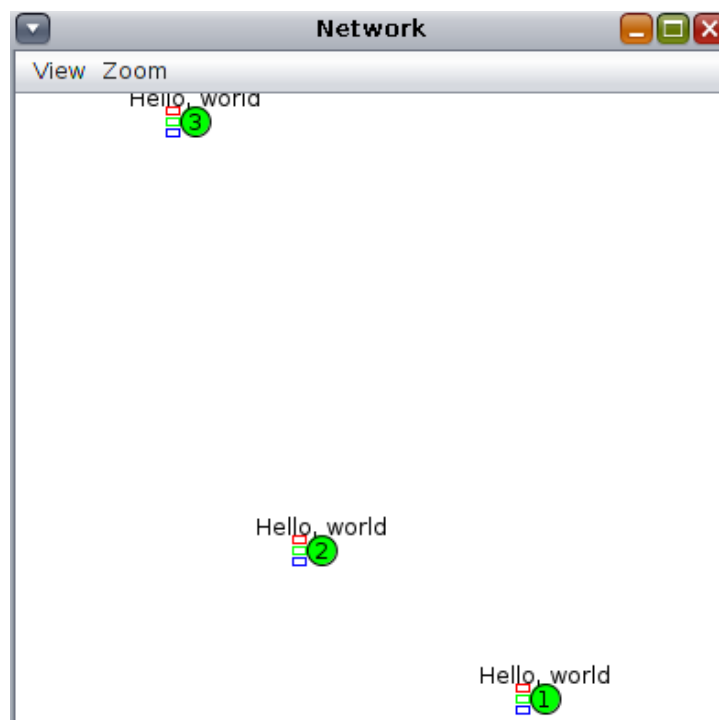
5. Expliquer ce qui affiche dans la fenêtre **Mote output** et **network**.

Dans la fenêtre Mote Output :



Cette fenêtre affiche le déroulement de la simulation et les différentes sorties et affichage des capteurs (motes). Dans cet exemple, Contiki va d'abord créer le capteur avec l'ID2, ensuite il lui affecte une adresse RIME, une adresse MAC et une adresse IPv6, enfin il lance le processus « hello world » du fichier .c qui a été affecté lors de la compilation. Ce processus va afficher le texte : **Hello, world**.

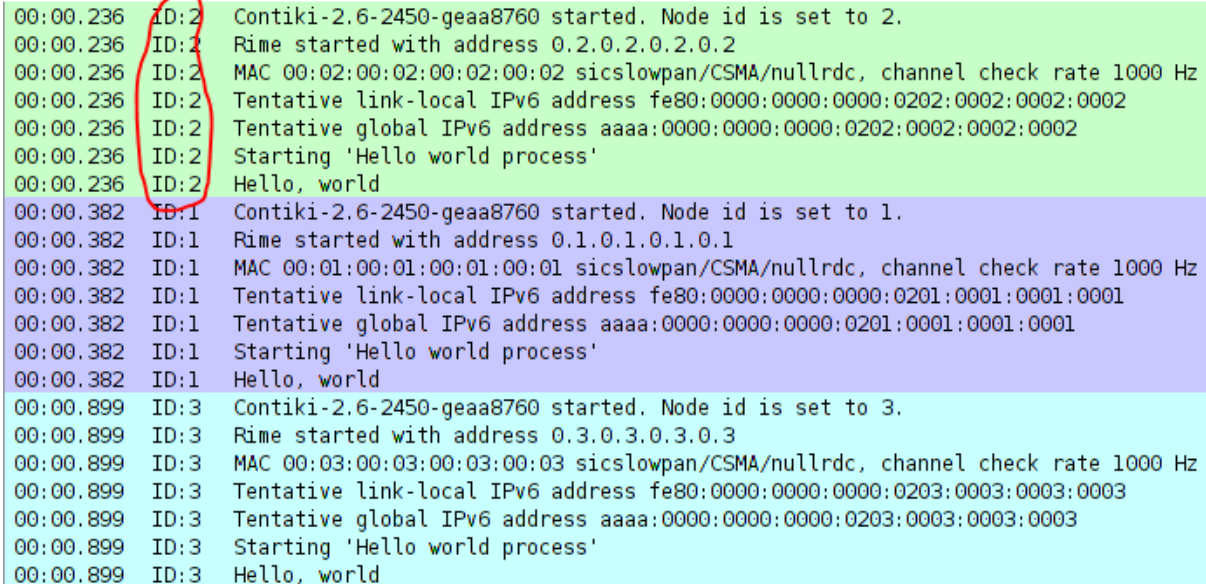
Dans la fenêtre Network :



Cette fenêtre affichera la localisation des capteurs. Dans cette exemple chaque capteur va afficher le texte **Hello, world**.

6. A quoi correspond les différentes couleurs dans la fenêtre **Mote output** ?

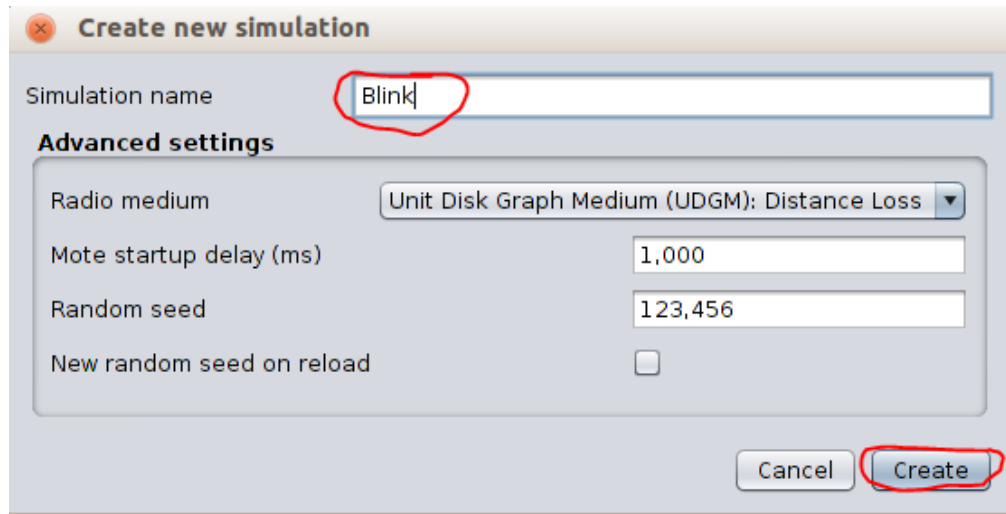
Chaque couleur correspond à un capteur différent.



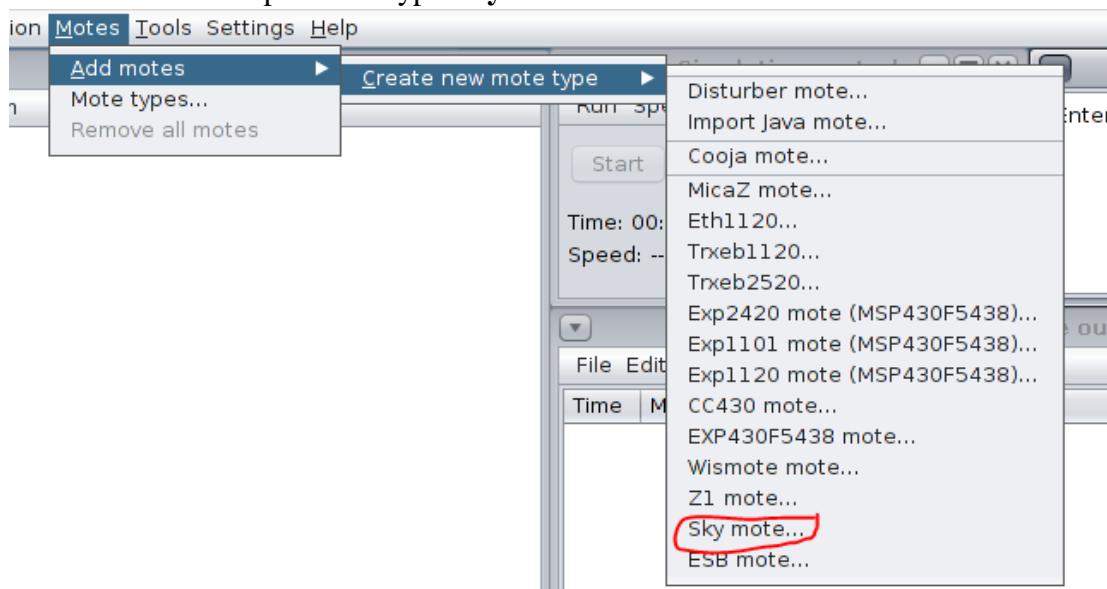
```
00:00.236 ID:2 Contiki-2.6-2450-geaa8760 started. Node id is set to 2.
00:00.236 ID:2 Rime started with address 0.2.0.2.0.2.0.2
00:00.236 ID:2 MAC 00:02:00:02:00:02:00:02 sicslowpan/CSMA/nullrdc, channel check rate 1000 Hz
00:00.236 ID:2 Tentative link-local IPv6 address fe80:0000:0000:0000:0202:0002:0002:0002
00:00.236 ID:2 Tentative global IPv6 address aaaa:0000:0000:0000:0202:0002:0002:0002
00:00.236 ID:2 Starting 'Hello world process'
00:00.236 ID:2 Hello, world
00:00.382 ID:1 Contiki-2.6-2450-geaa8760 started. Node id is set to 1.
00:00.382 ID:1 Rime started with address 0.1.0.1.0.1.0.1
00:00.382 ID:1 MAC 00:01:00:01:00:01:00:01 sicslowpan/CSMA/nullrdc, channel check rate 1000 Hz
00:00.382 ID:1 Tentative link-local IPv6 address fe80:0000:0000:0000:0201:0001:0001:0001
00:00.382 ID:1 Tentative global IPv6 address aaaa:0000:0000:0000:0201:0001:0001:0001
00:00.382 ID:1 Starting 'Hello world process'
00:00.382 ID:1 Hello, world
00:00.899 ID:3 Contiki-2.6-2450-geaa8760 started. Node id is set to 3.
00:00.899 ID:3 Rime started with address 0.3.0.3.0.3.0.3
00:00.899 ID:3 MAC 00:03:00:03:00:03:00:03 sicslowpan/CSMA/nullrdc, channel check rate 1000 Hz
00:00.899 ID:3 Tentative link-local IPv6 address fe80:0000:0000:0000:0203:0003:0003:0003
00:00.899 ID:3 Tentative global IPv6 address aaaa:0000:0000:0000:0203:0003:0003:0003
00:00.899 ID:3 Starting 'Hello world process'
00:00.899 ID:3 Hello, world
```

2. Deuxième exemple :

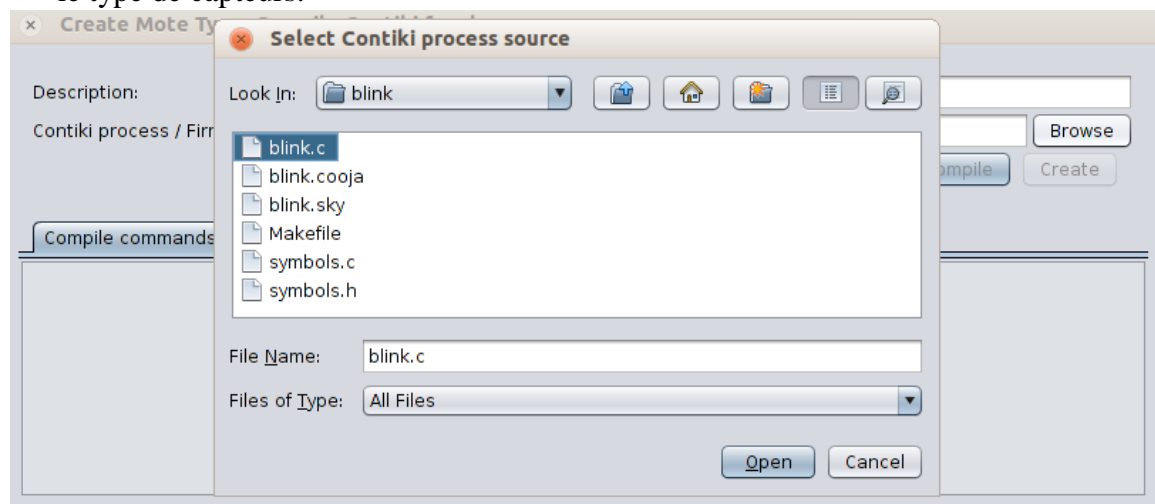
- Créer une deuxième simulation avec le nom Blink

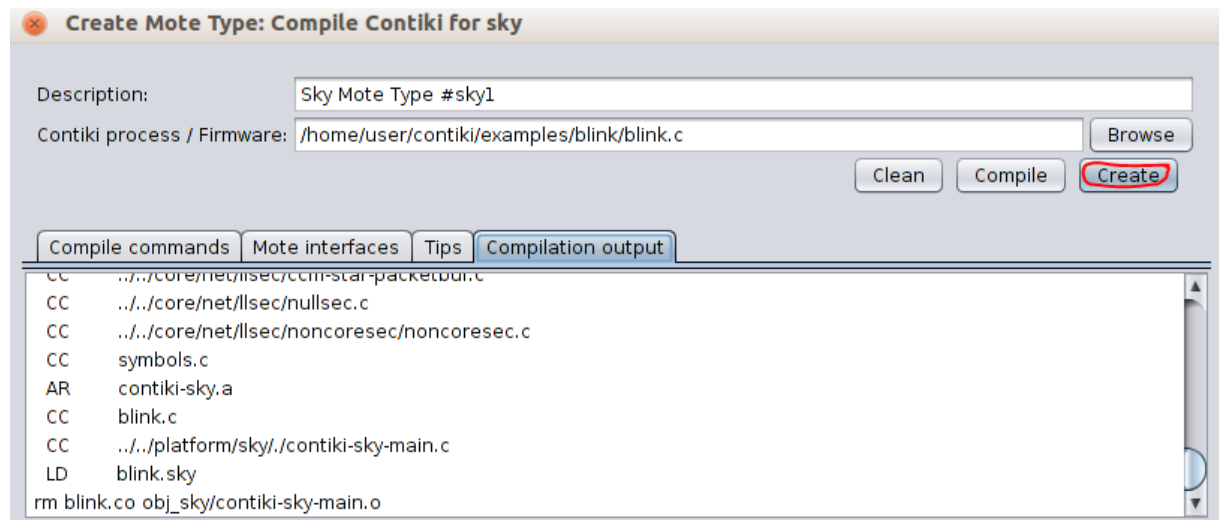


- Créer des capteurs de type Sky mote...

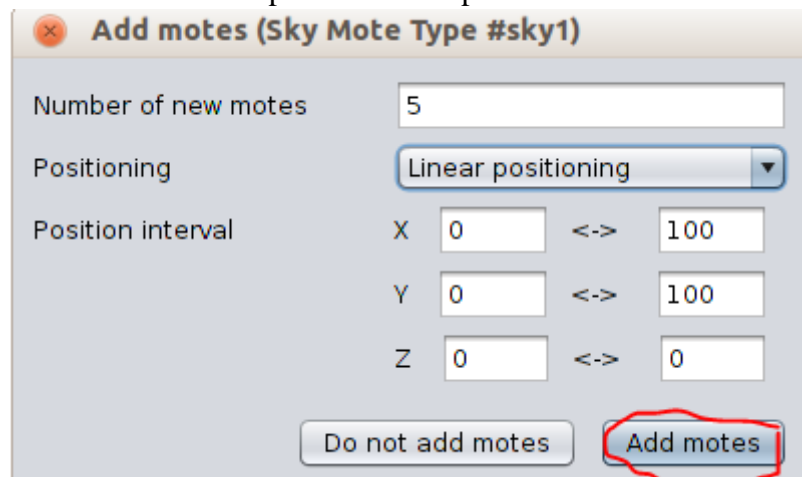


- Préciser le fichier `/home/user/contiki/examples/blink/blink.c` comme programme pour le type de capteurs.

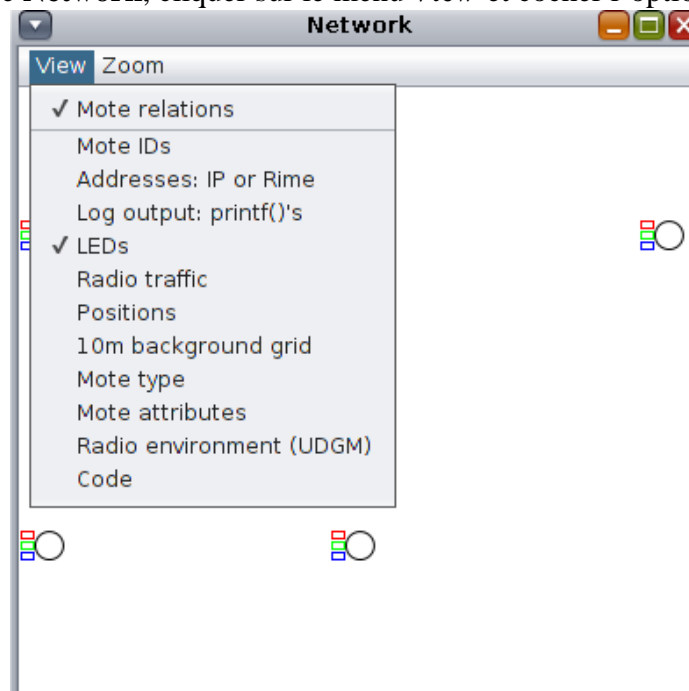




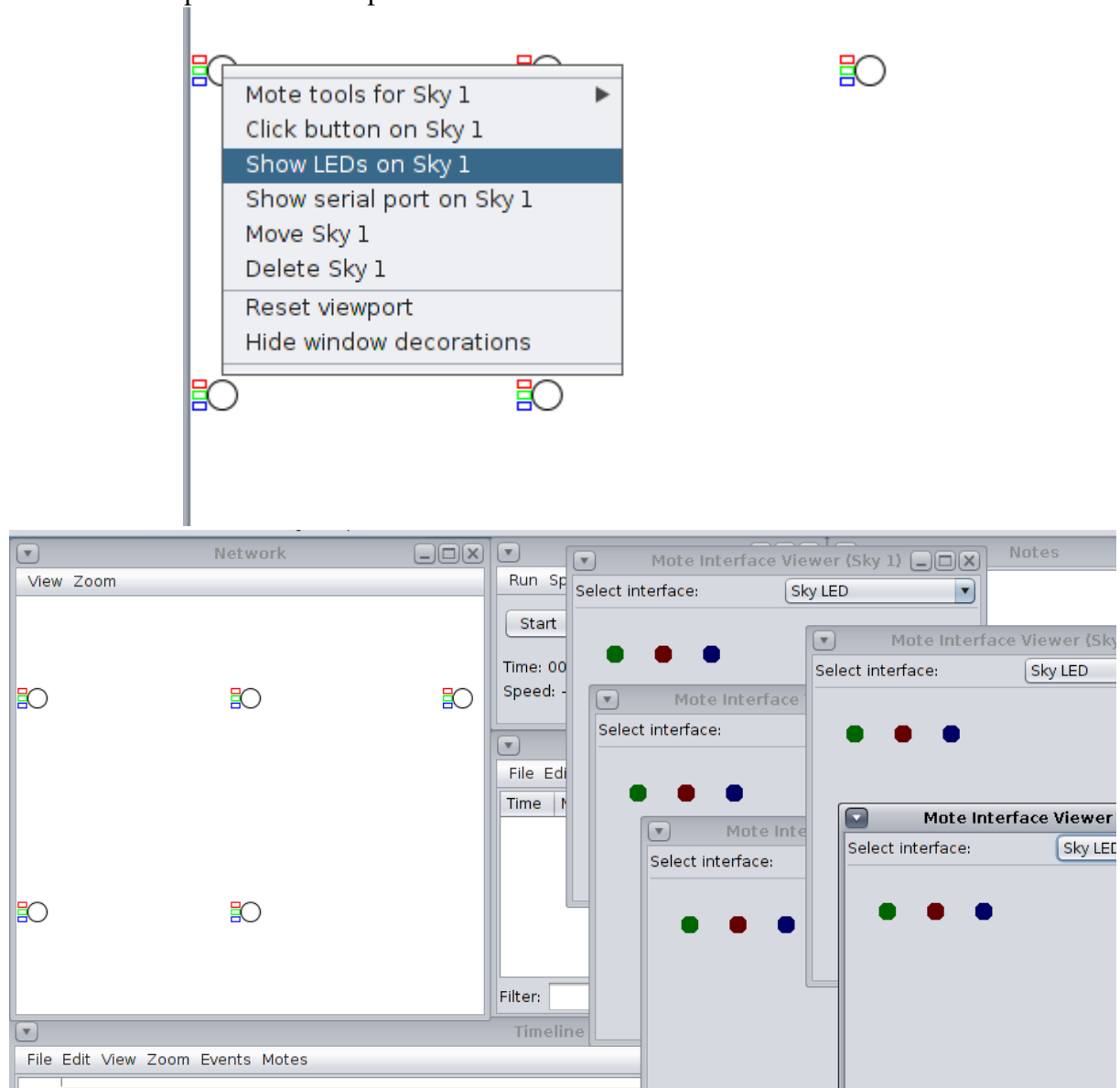
- Ajouter à cette simulation 5 capteurs avec un positionnement linéaire entre eux.



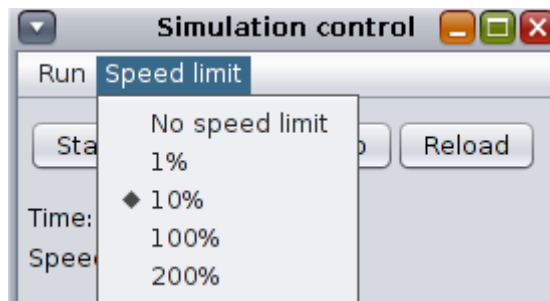
- Dans la fenêtre **Network**, cliquer sur le menu **View** et cocher l'option **LEDs**.

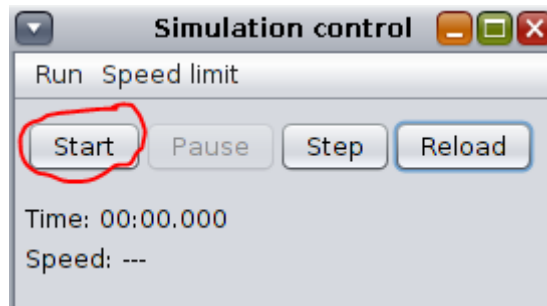


- Toujours dans la fenêtre Network, faites un clic droit sur un capteur -> **Show LEDs**.
Faites ceci pour tous les capteurs.



- Régler la vitesse de la simulation à 10% et lancer la simulation.





- Expliquer le fonctionnement de la simulation.

Les LEDs des capteurs vont s'allumer et s'éteindre au fur et à mesure de la simulation comme le montre le programme .c de simulation.

D'abord le processus va faire se mettre en pause ensuite il va faire appel à la fonction `leds_toggle()` pour tous les LEDs qui va inverser l'état des LEDs.

- Modifier le fichier `blink.c` de façon à allumer seulement les LEDs rouge et non tous les LEDs.

```
blink.c (~/.contiki/examples/blink) - gedit
File Edit View Search Tools Documents Help
Open Save Undo Redo
blink.c x
#include "contiki.h"
#include "dev/leds.h"

/*-----*/
PROCESS(blink_process, "Blink process");
AUTOSTART_PROCESSES(&blink_process);
/*-----*/
PROCESS_THREAD(blink_process, ev, data)
{
    PROCESS_BEGIN();

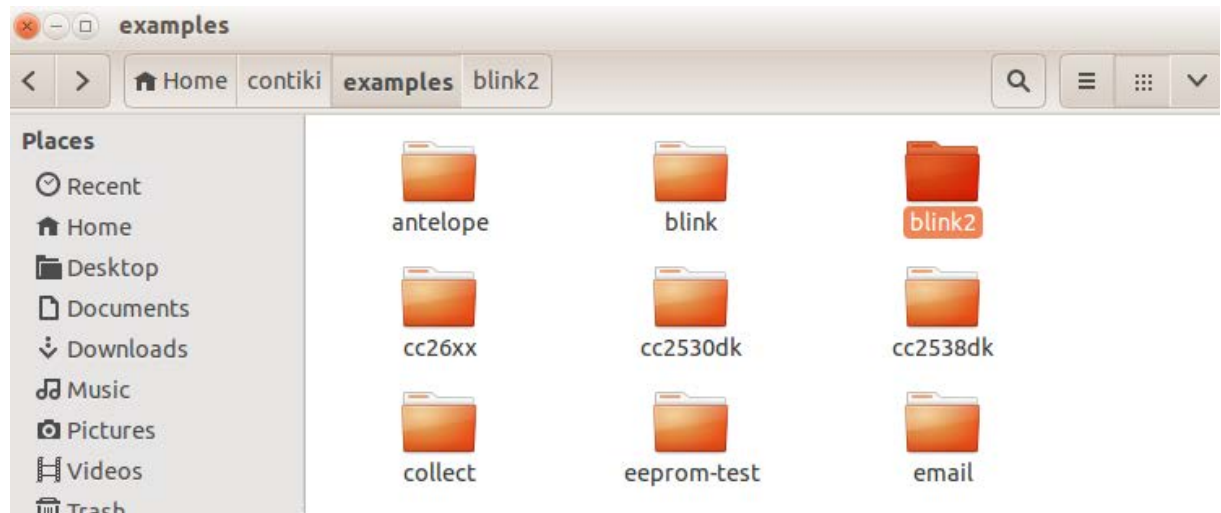
    while(1){
        static struct etimer et;
        etimer_set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_toggle(LED_RED);
    }

    PROCESS_END();
}
/*-----*/
```

3. Travail à faire

1. Créer un programme blink2.c qui permet de faire d'allumer les LEDS dans cette ordre (1 seul LED à la fois) : rouge, jaune, vert ensuite de les faire cligner.

On créer un nouveau dossier **blink2** dans le dossier **examples** :



On créer à l'intérieur de ce dossier le fichier **blink2.c** avec le code suivant :

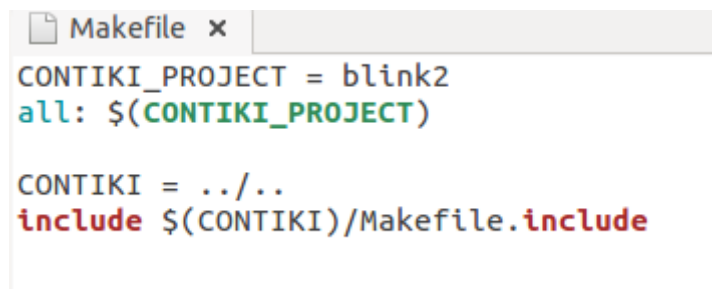
```
#include "contiki.h"
#include "dev/leds.h"

/*-----*/
PROCESS(blink2_process, "Blink2 process");
AUTOSTART_PROCESSES(&blink2_process);
/*-----*/
PROCESS_THREAD(blink2_process, ev, data)
{
    PROCESS_BEGIN();

    while(1){
        static struct etimer et;
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_on(LEDS_RED);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_off(LEDS_RED);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_on(LEDS_YELLOW);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_off(LEDS_YELLOW);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_on(LEDS_GREEN);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_off(LEDS_GREEN);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_toggle(LEDS_ALL);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_toggle(LEDS_ALL);
    }

    PROCESS_END();
}
/*-----*/
```

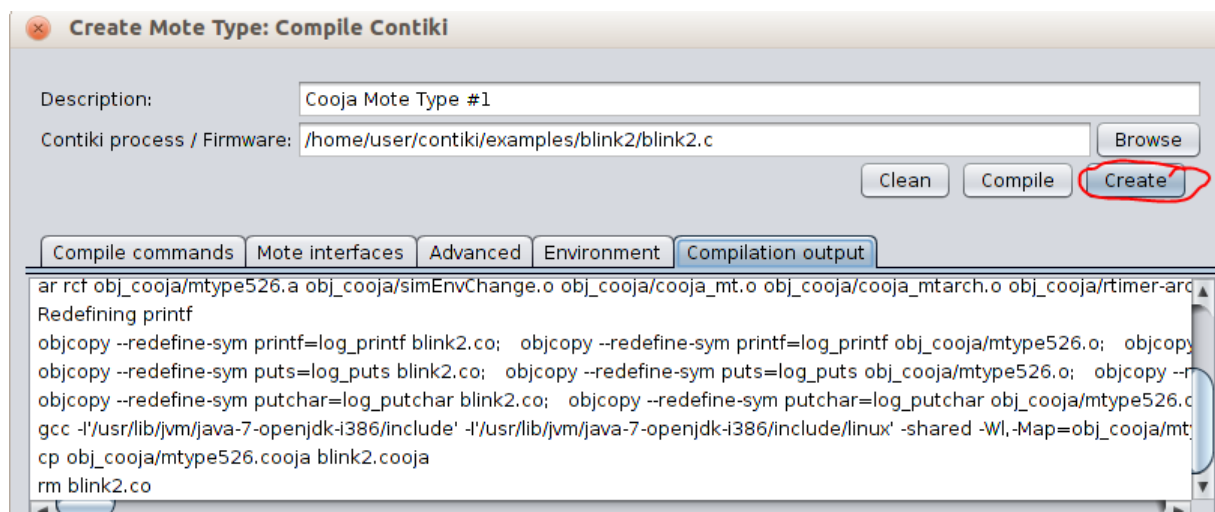
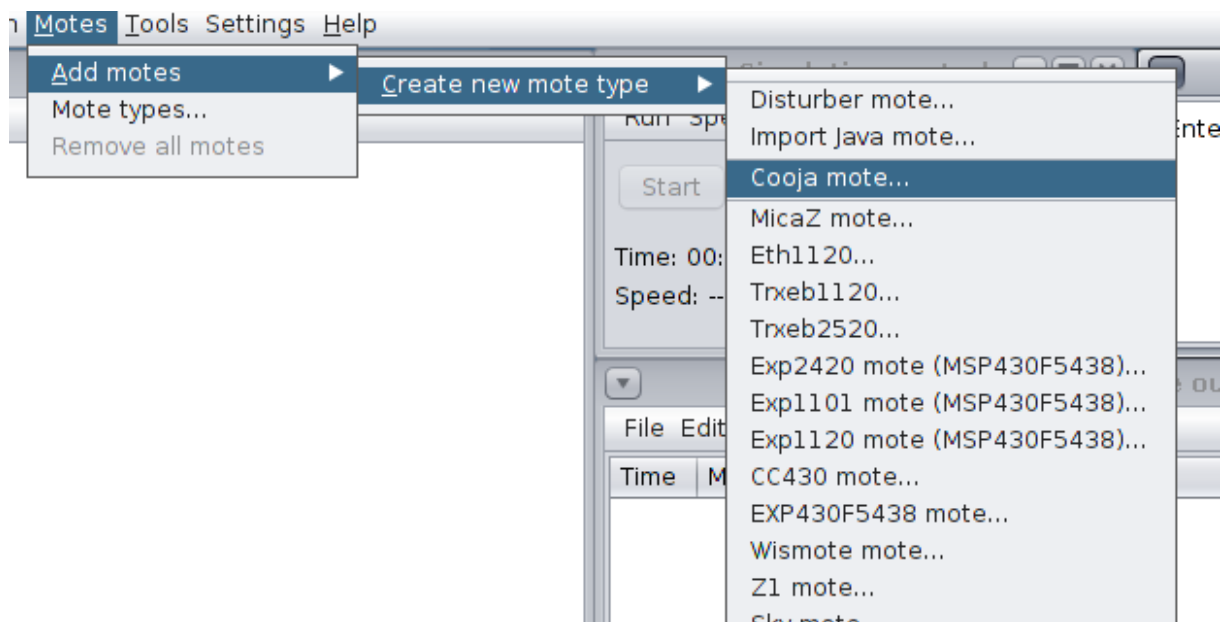
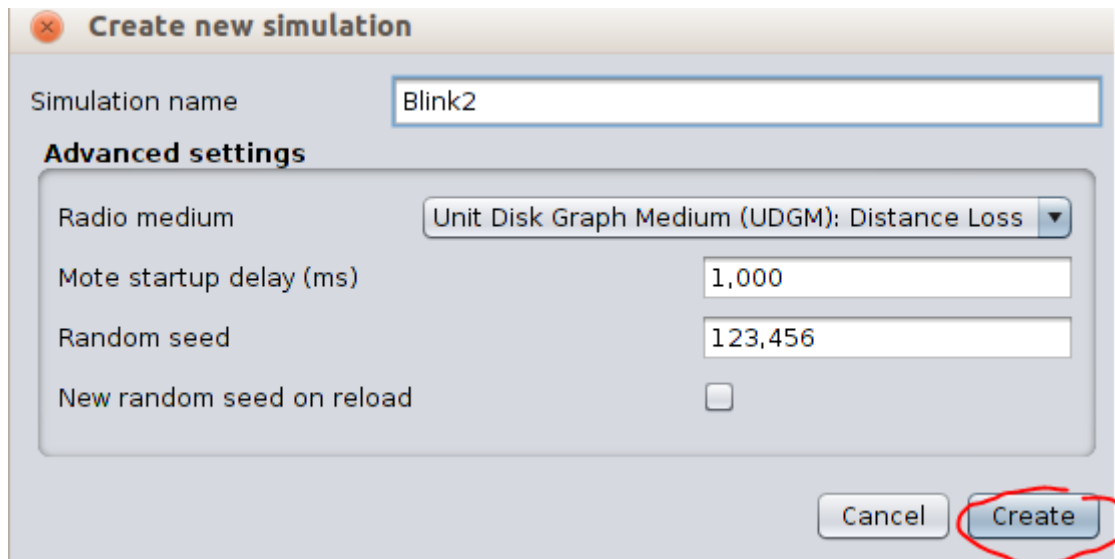
On créer aussi le fichier Makefile avec le code suivant :

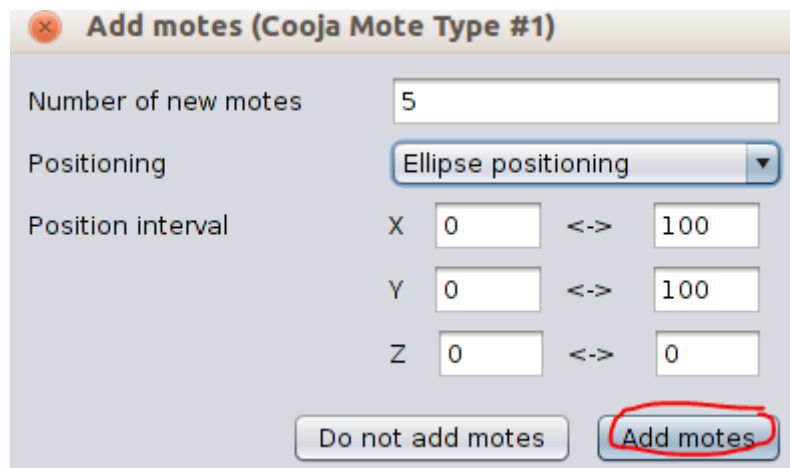


```
CONTIKI_PROJECT = blink2
all: $(CONTIKI_PROJECT)

CONTIKI = ../..
include $(CONTIKI)/Makefile.include
```

2. Créer une simulation avec 5 capteurs de type Cooja mote distribués de façon elliptique.⁵





3. Modifier le programme blink2.c de façon à afficher un texte avec la couleur du LED allumé à chaque fois.

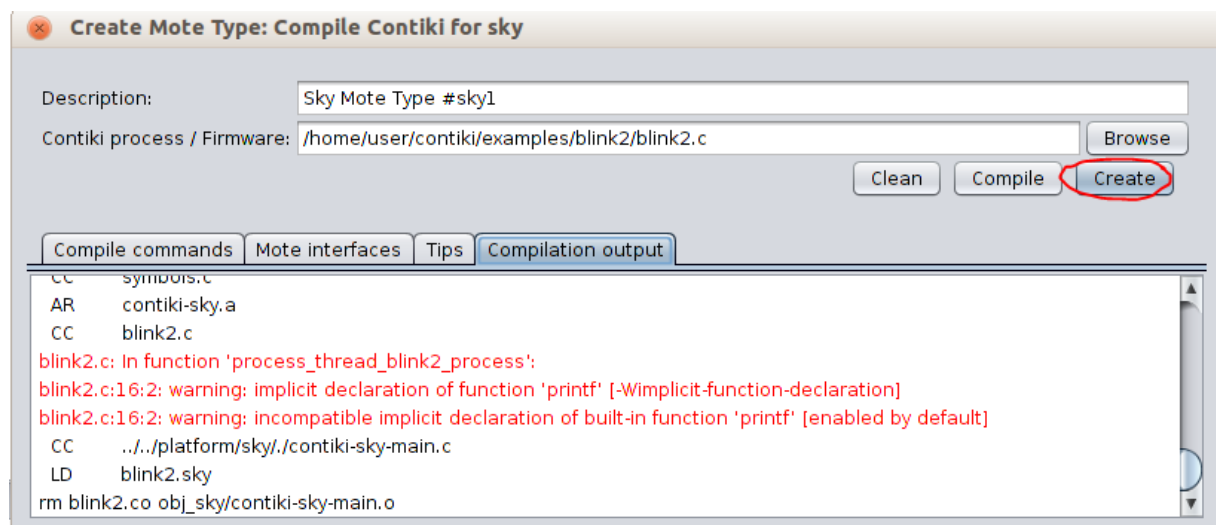
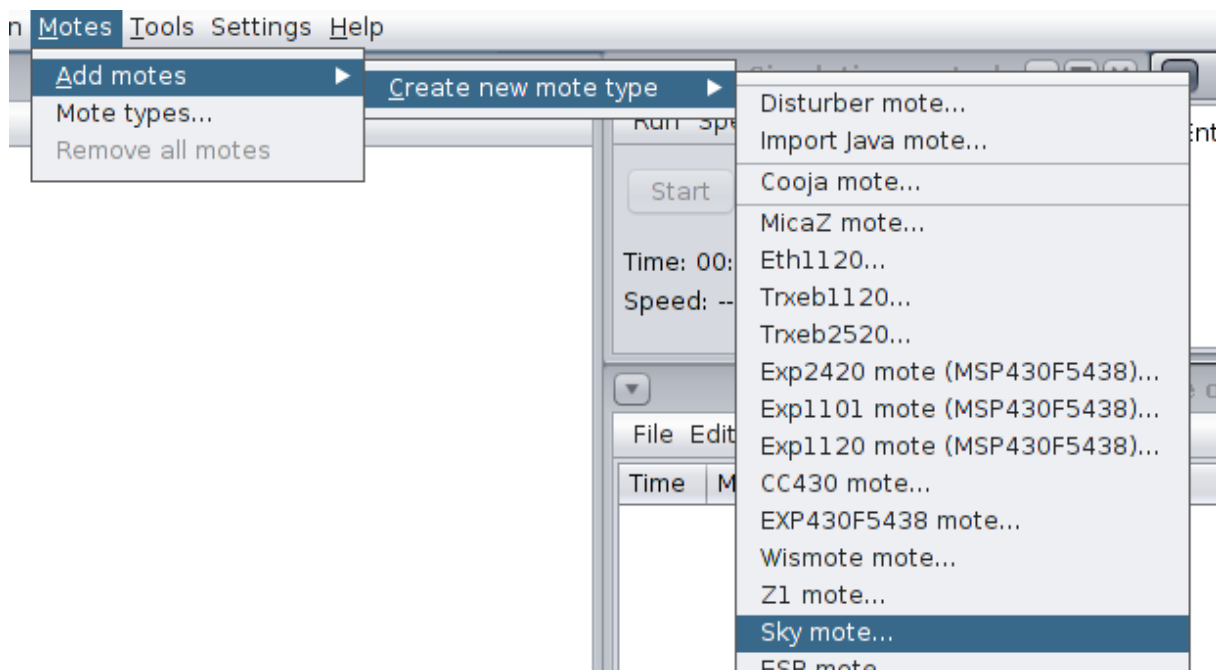
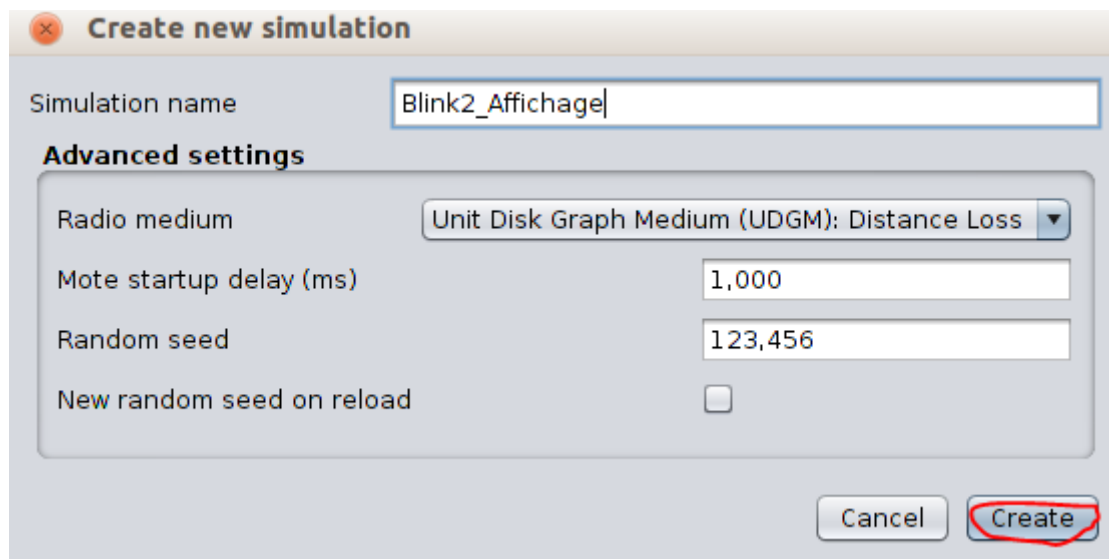
```
#include "contiki.h"
#include "dev/leds.h"

/*-----*/
PROCESS(blink2_process, "Blink2 process");
AUTOSTART_PROCESSES(&blink2_process);
/*-----*/
PROCESS_THREAD(blink2_process, ev, data)
{
    PROCESS_BEGIN();

    while(1){
        static struct etimer et;
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_on(LEDS_RED);
        printf("Rouge\n");
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_off(LEDS_RED);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_on(LEDS_YELLOW);
        printf("Jaune\n");
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_off(LEDS_YELLOW);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_on(LEDS_GREEN);
        printf("Vert\n");
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_off(LEDS_GREEN);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_toggle(LEDS_ALL);
        etimer set(&et, CLOCK_SECOND);
        PROCESS_WAIT_EVENT_UNTIL(etimer_expired(&et));
        leds_toggle(LEDS_ALL);
    }

    PROCESS_END();
}
/*-----*/
```

4. Créer une simulation avec 4 capteurs de type Sky mote distribués de façon aléatoire.



Add notes (Sky Mote Type #sky1)

Number of new notes:

Positioning:

Position interval:

| | | | |
|---|--------------------------------|-----|----------------------------------|
| X | <input type="text" value="0"/> | <-> | <input type="text" value="100"/> |
| Y | <input type="text" value="0"/> | <-> | <input type="text" value="100"/> |
| Z | <input type="text" value="0"/> | <-> | <input type="text" value="0"/> |

