# TP-3 : FAIRE CLIGNOTER UNE LED À L'AIDE DE BLOCKLY

## **Objectifs**

Partie 1: Examiner un premier programme Blockly

Partie 2: Contrôler une LED RGB avec Blockly

Partie 3 : Ajouter un bouton pour contrôler une LED RGB

Partie 4 : Exercice non guidé (Python au lieu de Blockly)

#### Contexte

Blockly est un langage de programmation graphique qui permet aux utilisateurs de créer des programmes en connectant des blocs (représentant différentes structures de langage logiques) plutôt qu'en écrivant le code. Blockly, qui s'exécute dans un navigateur web, peut traduire le programme créé graphiquement en langage JavaScript, PHP ou Python. Au cours de ces travaux pratiques, vous utiliserez Blockly pour examiner la programmation Blockly et contrôler une LED.

#### Scénario

Utiliser la programmation Blockly pour contrôler la LED d'un objet IoT. Au cours de ces travaux pratiques, nous utilisons Cisco Packet Tracer, car il assure la prise en charge de Blockly avec les objets IoT.

### **Ressources requises**

• Cisco Packet Tracer 7.1.1 et version supérieure est installé et disponible.

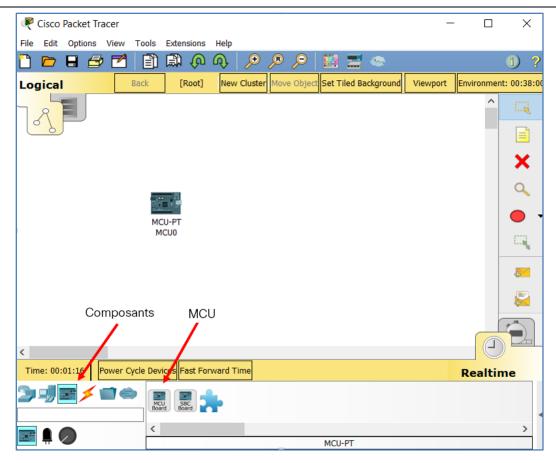
### Partie 1 : Lancer Cisco Packet Tracer (PT) et utiliser Blockly

Au cours de la première partie, vous accéderez au programme Cisco Packet Tracer et examinerez le contrôle de voyants à l'aide de la programmation Blockly.

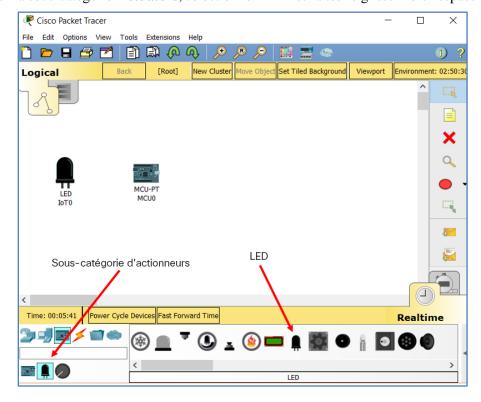
## **Étape 1 : Lancez Packet Tracer.**

a. Cliquez sur la catégorie Components, puis sur MCU Board et faites-la glisser vers l'espace de travail.

m.baslam@usms.ma Page 1 sur 16

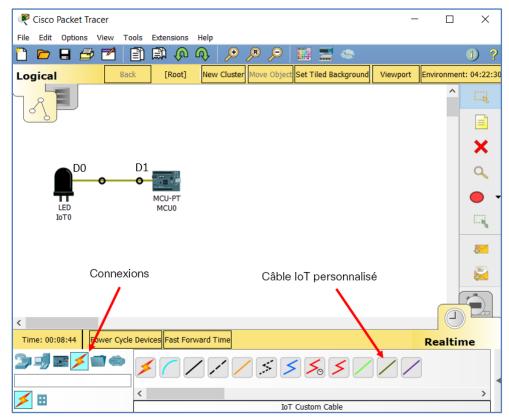


b. Cliquez sur la sous-catégorie **Actuators**, sélectionnez **LED** et faites-le glisser vers l'espace de travail.



m.baslam@usms.ma Page 2 sur 16

c. Cliquez sur la catégorie **Connections** et sélectionnez **IoT Custom Cable** pour relier le MCU au port **D1** et le voyant au port **D0**.



d. Double-cliquez sur le MCU. Sa fenêtre de configuration s'affiche.



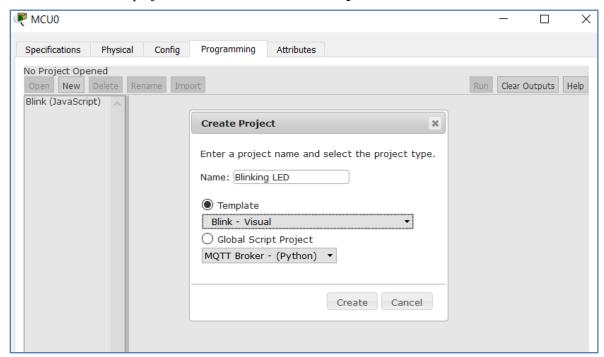
e. Cliquez sur l'onglet **Programming**. (Si l'onglet Programming n'est pas affiché, cliquez sur le bouton **Advanced** en bas à droite.)



m.baslam@usms.ma Page 3 sur 16

## Étape 2: Examiner un programme Blockly prégénéré

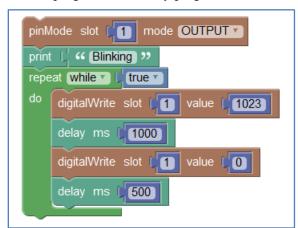
a. Sous la note **No Project Opened**, cliquez sur **New**. Dans la fenêtre **Create Project**, saisissez **Blinking LED** comme nom du projet. Dans le menu déroulant **Template**, sélectionnez **Blink – Visual**.



b. Cliquez sur Créer.



c. Double-cliquez sur **main.visual**. Le programme Blockly prégénéré s'affiche.



m.baslam@usms.ma Page 4 sur 16

d. Cliquez sur **Exécuter**. La LED clignote-t-elle ?

OUI

e. Cliquez sur **Stop** et saisissez 1 à la place de 1023 pour le champ **Value** du premier bloc **digitalWrite**.

```
print ("Blinking")
repeat While T true T

do digitalWrite slot (1 value (1)
delay ms (1000)
digitalWrite slot (1 value (0)
delay ms (500)
```

f. Cliquez sur **Exécuter**, la LED clignote-t-elle ?

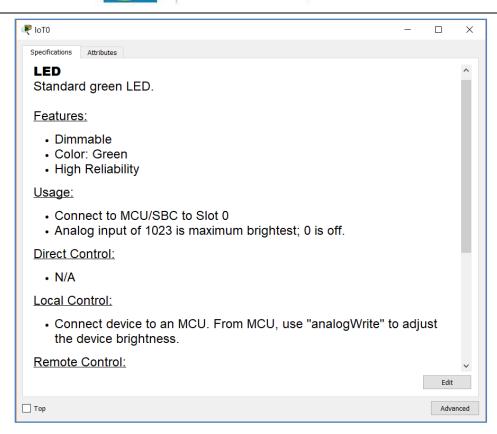
NON

Pourquoi ne clignotait-elle pas lorsque la valeur n'était pas définie sur 1023 ?

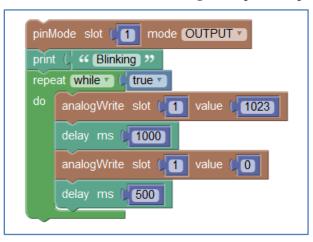
La sortie numérique considère la valeur 1 023 comme étant « élevée » et allume le courant de sortie.

- g. Modifier les valeurs des deux blocs **delay ms** (par exemple au lieu de 1000 et (resp. 500) mettez 500 (resp. 250) pour augmenter la fréquence de clignotement de LED),
- h. Cliquez sur la LED et étudiez ses caractéristiques.

m.baslam@usms.ma Page 5 sur 16



i. Elle indique que nous pouvons utiliser « analogWrite » pour régler la luminosité de l'appareil. Développez le groupe **Pin Access** et utilisez le bloc **analogWrite** pour remplacer le bloc **digitalWrite**.



j. À présent, modifiez les valeurs de premier (1023) et deuxième blocs (0) **analogWrite** et observez les différents niveaux de luminosité du voyant.

m.baslam@usms.ma Page 6 sur 16

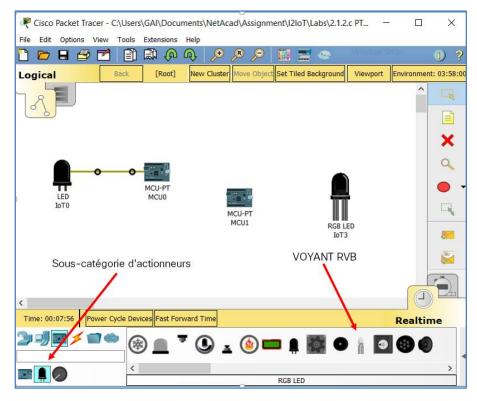
## Partie 2: Contrôler une LED RGB avec Blockly

Dans la deuxième partie, vous avez utilisé Blockly pour contrôler une LED RGB. Un voyant RGB peut afficher différentes couleurs combinant le rouge, le vert et le bleu.

## Étape 1: Ajoutez un MCU et une LED RGB.

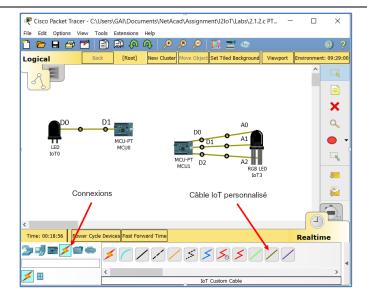
L'étape 1 consiste à ajouter une autre carte MCU et une LED RGB à l'espace de travail.

a. Cliquez sur la sous-catégorie **Actuators**, sélectionnez **RGB LED** et faites-le glisser vers l'espace de travail. Ajoutez une autre carte MCU.

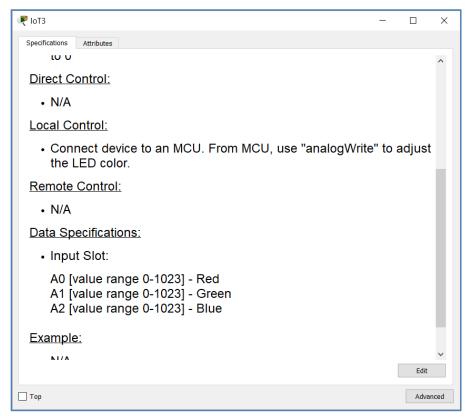


b. Cliquez sur la catégorie **Connections**, sélectionnez trois **IoT Custom Cables** pour lier le MCU et la LED RGB.

m.baslam@usms.ma Page **7** sur **16** 

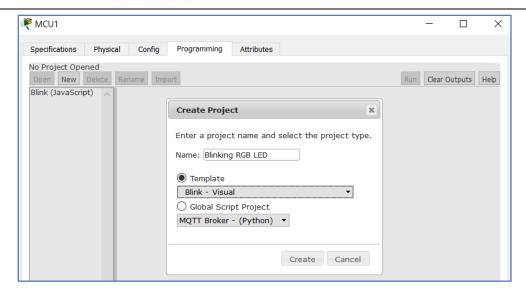


c. Cliquez sur **RGB LED** et vérifiez ses caractéristiques. Notez que les entrées de broche différentes représentent différentes couleurs.

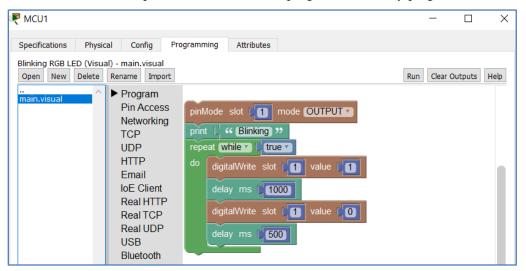


d. Ouvrez le programme Blockly prégénéré. Cliquez sur MCU -> Programming. Sous la note **No Project Opened**, cliquez sur **New**. Dans la fenêtre **Create Project**, saisissez **Blinking RGB LED** comme nom du projet. Dans le menu déroulant Template, sélectionnez **Blink – Visual**.

m.baslam@usms.ma Page 8 sur 16

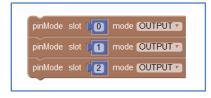


e. Cliquez sur Créer. Double-cliquez sur main.visual. Le programme Blockly prégénéré s'affiche.



# Étape 2: Modifier le programme Blockly

a. Développez le groupe **Pin Access**, et ajoutez deux autres blocs **pinMode** pour défini trois logements comme **OUTPUT** (depuis MCB, envoi d'un signal vers la LED RGB).



b. Dans le **Pin Access group**, sélectionnez des blocs **analogWrite** pour remplacer les blocs **digitalWrite**. Ajoutez également quelques blocs **print**.

m.baslam@usms.ma Page 9 sur 16

```
print ("Blinking Red")
analogWrite slot (0) value (1023)
delay ms (1000)
analogWrite slot (0) value (0)
delay ms (1500)
print ("Blinking Green")
analogWrite slot (1) value (1023)
delay ms (1500)
print ("Blinking Blue")
analogWrite slot (2) value (1023)
delay ms (1000)
analogWrite slot (2) value (1023)
delay ms (1000)
analogWrite slot (2) value (1023)
```

c. Le programme final se présente comme suit :

m.baslam@usms.ma Page 10 sur 16

```
pinMode slot (0) mode OUTPUT
pinMode slot (
            1 mode OUTPUT ▼
pinMode slot (2) mode OUTPUT ▼
repeat while
              true 🔻
          " (Blinking Red) "
   analogWrite slot (10) value (1023)
    delay ms (1000)
   analogWrite slot (10) value (1
              1500
    print ( Blinking Green >>
    analogWrite slot (1) value (1000)
    delay ms
              1000
    analogWrite slot (11)
    delay ms (1500)
    analogWrite slot (2) value (1023)
    delay ms (1000)
    analogWrite slot (2) value (
              1500
```

d. Exécutez le programme. Le voyant doit s'afficher en ROUGE, VERT et BLEU dans l'ordre.

# Challenge

e. Amuser vous en modifiant le programme pour afficher le clignotement des couleurs combinées à partir des trois entrées.

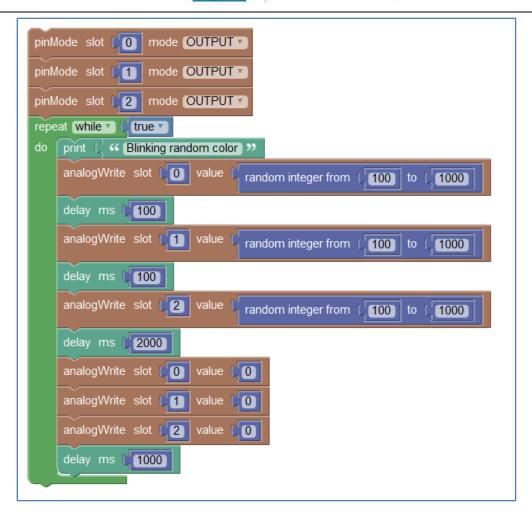
m.baslam@usms.ma Page 11 sur 16

```
pinMode slot
              0
                   mode OUTPUT -
                   mode OUTPUT ▼
pinMode slot
                   mode OUTPUT -
              2
pinMode slot
print (
       " Blinking "
repeat while *
                true
    print (
            66 Couleur Rouge+Bleu >>
    analogWrite slot (10)
                           value 🌘
                                  1023
    analogWrite slot [1] 2
                           value (
                                  1023
    delay ms
                1000
    digitalWrite slot
                          value
    analogWrite
                slot 🙀 2
                           value
    delay ms
                500
            " Couleur Rouge+Verte+Bleu >>>
    analogWrite slot
                           value
                                  1023
    analogWrite
                slot
                           value
                                  1023
    analogWrite
                slot 📜 2
                           value (
                                  1023
    delay ms
                1000
    digitalWrite slot
                     0
                          value
    analogWrite
    analogWrite
                slot
                           value
    delay ms
                500
```

f. Faites la même chose que la dernière tache, mais avec des valeurs différentes générées aléatoirement pour chaque logement

## Voici un exemple:

m.baslam@usms.ma Page 12 sur 16



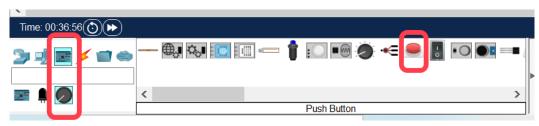
# Partie 3: Ajouter un bouton pour contrôler une LED RGB

Dans cette partie nous allons ajouter un objet IoT de type bouton pour contrôler la LED RGB que vous avez créé dans la partie précidente.

# **Étape 1 : Ajouter un bouton IoT**

L'étape 1 consiste à créer un nouveau circuit en ajoutant un bouton et le connecter à la carte MCU

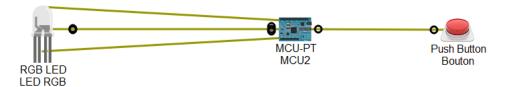
- a. Ajouter un nouveau MCU, une nouvelle LED RGB.
- b. Cliquez sur la catégorie **Components** puis **Sensors** puis **Push Button** comme il est illustré dans la figure suivante :



 c. Cliquez sur la catégorie Connections, sélectionnez un IoT Custom Cables pour lier le connecteur D3 de MCU avec le connecteur D0 du Push Button.

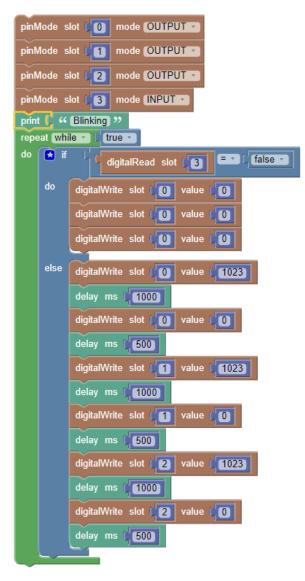
m.baslam@usms.ma Page 13 sur 16

d. Cliquez sur la catégorie **Connections**, sélectionnez un **IoT Custom Cables** pour lier les connecteurs D0, D1, et D2 de **MCU** respectivement avec les connecteur D0, D1 et D2 du **LED RGB.** 



# Étape 2: Créer le programme Blockly

- a. Cliquez sur MCU -> Programming. Sous la note No Project Opened, cliquez sur New. Dans la fenêtre Create Project, saisissez Blinking Buton LED comme nom du projet. Dans le menu déroulant Template, sélectionnez Blink - Visual.
- b. Développez le programme ci-dessous



c. Exécuter le programme et aller sur le circuit et utiliser **ALT+clic** pour allumer le bouton.

m.baslam@usms.ma Page 14 sur 16

- d. Pour découvrir différents boutons, remplacer le **Push Button** par **Toggle Push Button** en le connectant à D3 de **MCU.**
- e. Utiliser toujours **ALT+clic** pour allumer le nouveau bouton.
- f. Remplacer le **Toggle Push Button** par **Rocker Switch** en le connectant à D3 de **MCU.** Et utiliser toujours **ALT+clic** pour allumer le nouveau bouton.



g. Remarquer la différence entre les trois boutons

## Partie 4 : Exercice non guidé (Python au lieu de Blockly)

- a. Vous allez créer un nouveau programme portant le nom 'Blink Python', en choisissant le Template Blink-Python.
- b. Ecrire un programme en python qui fait les mêmes fonctionnalités du programme dans la Partie4→Etape2→b

```
1
    from gpio import *
 2
    from time import *
 3
 4 - def main():
 5
        pinMode(0, OUT)
 6
        pinMode(1, OUT)
 7
        pinMode(2, OUT)
 8
         pinMode(3, IN)
 9
         print("Blinking")
10 -
         while True:
11 -
             if (digitalRead(3) == False):
12
                 digitalWrite(0, LOW);
13
                 digitalWrite(1, LOW);
14
                 digitalWrite(2, LOW);
15 -
             else:
16
                 digitalWrite(0, HIGH);
17
                 sleep(1)
18
                 digitalWrite(0, LOW);
19
                 sleep(0.5)
20
                 digitalWrite(1, HIGH);
21
                 sleep(1)
22
                 digitalWrite(1, LOW);
23
                 sleep(0.5)
24
                 digitalWrite(2, HIGH);
25
                 sleep(1)
26
                 digitalWrite(2, LOW);
27
                 sleep(0.5)
28
29 - if
                      main
          name
30
        main()
```

m.baslam@usms.ma Page 15 sur 16

- c. Modifier ce programme pour pouvoir arrêter le clignotement immédiatement après avoir arrêté le bouton.
- d. Pensez à d'autres scénarios .....

m.baslam@usms.ma Page **16** sur **16**