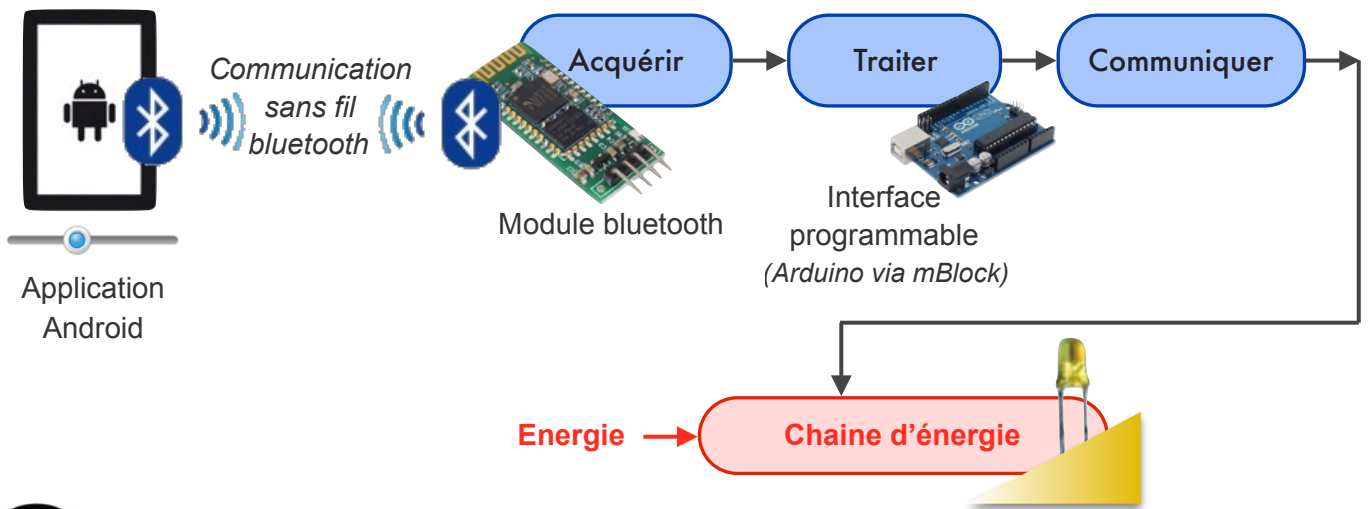


# APP INVENTOR



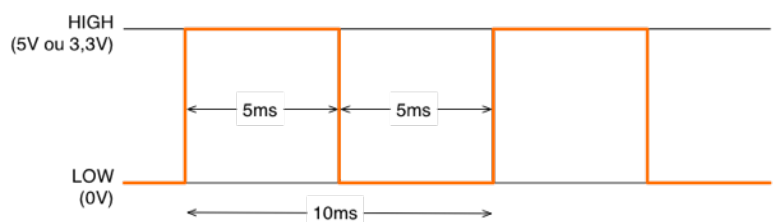
Dans cet exemple il s'agit, de piloter la puissance d'éclairage une del (variation de lumière) depuis le smartphone (application Android)

Pour cela nous allons utiliser les sorties « analogiques » (PWM) des microcontrolleurs (Picaxe ou Arduino) pour faire varier la puissance lumineuse de la del. Voir autre ressource pour davantage de précisions.

Côté application, nous allons utiliser un curseur qui permet de communiquer une valeur entre 0 et 255 (soit les 256 possibilités en 8 bits) en bluetooth.

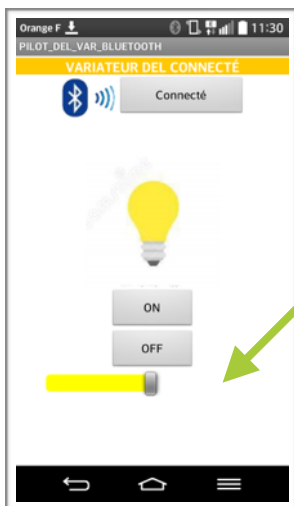
### Exemple avec PWM à 50%

La fréquence est de 100Hz, le rapport cyclique de 50%

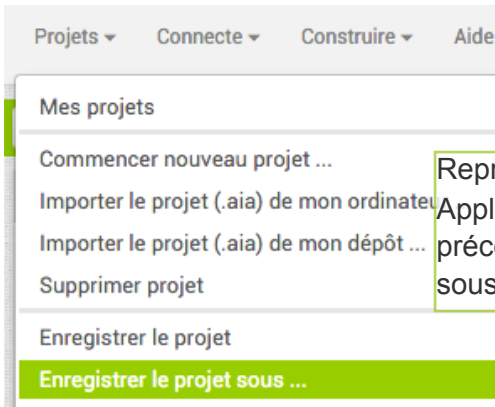


Une sortie PWM sur un microcontrolleur est une sortie Numérique dont les signaux ont toujours une valeur LOW (0 logique) ou HIGH (1 logique). Mais le principe est de construire un signal qui est alternativement LOW et HIGH et de répéter très vite cette alternance en faisant varier la fréquence du signal.

Dans le cas d'une DEL, elle est alternativement allumée et éteinte mais le cycle est tellement rapide que la persistance rétinienne nous donne l'illusion d'une DEL allumée en permanence. Prenons par exemple une période de 10ms, soit une fréquence de 100Hz. Si la DEL est allumée pendant 5ms et éteinte pendant 5ms, comme sur la figure ci-contre, l'impression sera une luminosité de 50% de la luminosité maximum.



Curseur de 0 à 255 et initialement à 128 (valeur milieu)



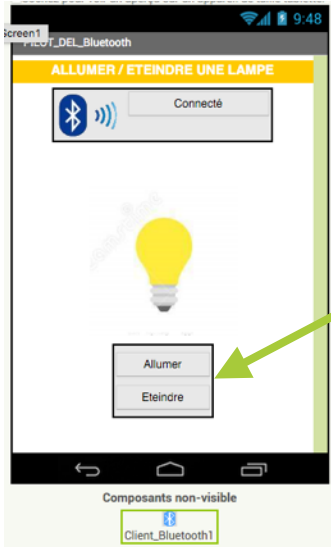
1  
Reprendre l'application Appli\_Lampe\_Bluetooth vue précédemment et l'enregistrer sous un nouveau nom de projet

Ajouter les images que l'on va utiliser par la suite :

2  
Image variation et nouveau logo de l'application



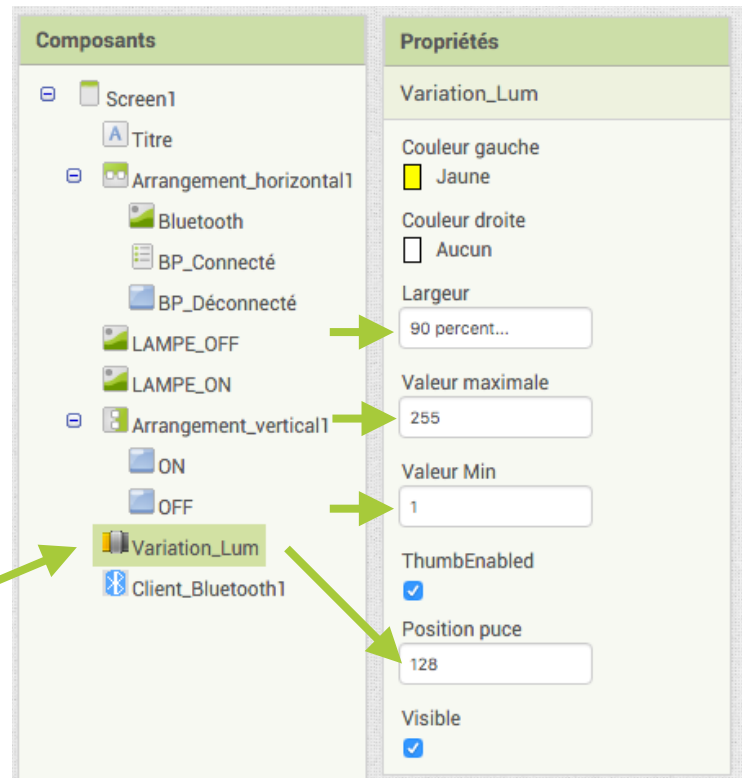
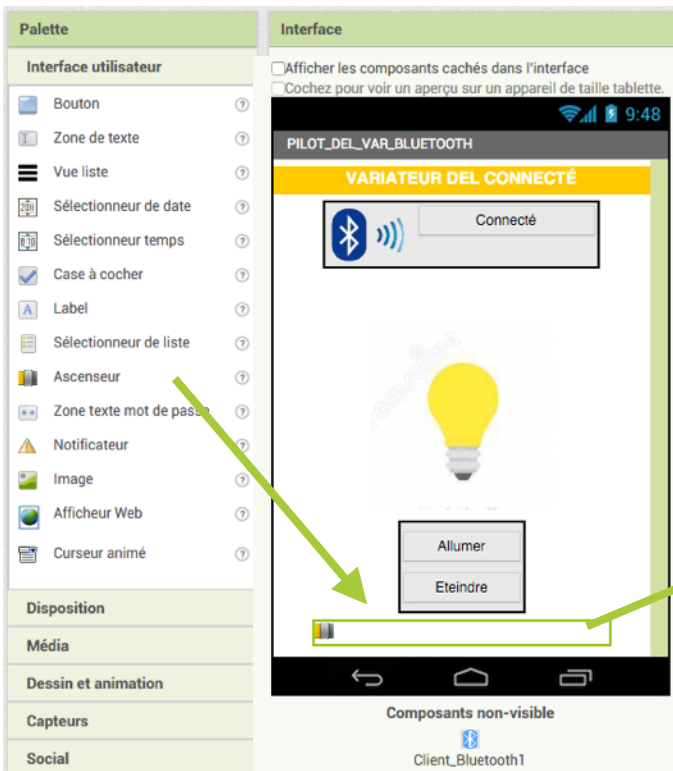
3  
Changez les propriétés de l'application : Logo et nomme l'application



4  
Côté interface design de l'application, il suffit d'ajouter un curseur (renommé « Variation\_Lum »): Ascenseur en dessous des 2 boutons « Allumer » « Eteindre ».



L'ascenseur doit avoir pour valeur max et min respectivement 255 et 1 (car nous sommes en 8 bits, voir tableau en bas de page). Egalement il peut être initialisé en position milieu (Position puce) soit à 128.



Rapport Cyclique %	0	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
Valeur sur 8 bits	0	13	26	38	51	64	77	89	102	115	128	140	153	166	179	191	204	217	230	242	255

Côté programmation ... Quelques ajouts et modifications sont à réaliser pour communiquer les valeurs de l'ascenseur via le bluetooth

```

quand BP_Connecte .Avant prise
faire
mettre BP_Connecte . Eléments à Client_Bluetooth1 . Adresses et noms

quand BP_Connecte .Après prise
faire
mettre BP_Connecte . Activé à appeler Client_Bluetooth1 .Se connecter
adresse BP_Connecte . Sélection
mettre BP_Connecte . Visible à faux
mettre BP_Deconnecte . Visible à vrai

quand BP_Deconnecte .Clic
faire
appeler Client_Bluetooth1 .Déconnecter
mettre BP_Connecte . Visible à vrai
mettre BP_Deconnecte . Visible à faux
  
```



La partie de code correspondant à la fonction bluetooth ne change pas

```
initialise global var_lum à 128
```



Initialiser une variable **var\_lum** à 128 (valeur milieu du curseur par défaut)

```

quand ON .Clic
faire
mettre LAMPE_ON . Visible à vrai
mettre LAMPE_OFF . Visible à faux
appeler Client_Bluetooth1 .Envoyer texte
texte joint " BP: "
obtenir global var_lum
  
```



Le bouton « ON » envoie donc maintenant le contenu de la variable **var\_lum** sous la forme : « BP : var\_lum »

```

quand Variation_Lum .Position changée
Position pouce
faire
mettre global var_lum à arrondi Variation_Lum . Position pouce
appeler Client_Bluetooth1 .Envoyer texte
texte joint " BP: "
obtenir global var_lum
  
```



Quand le curseur de l'ascenseur change de position :  
Le Client\_Bluetooth envoie la position actuelle du curseur :  
Soit une valeur **entière** entre 1 et 255, d'où l'utilisation du bloc **arrondi**

```

quand OFF .Clic
faire
mettre LAMPE_OFF . Visible à vrai
mettre LAMPE_ON . Visible à faux
appeler Client_Bluetooth1 .Envoyer texte
texte joint " BP: "
" 0 "
  
```



Côté bouton « OFF » rien ne change : il communique 0 via le client bluetooth sous la forme : « BP : 0 »

6

L'application est terminée, vous pouvez la tester et l'installer sur l'appareil nomade Android

MaTereApplication Progress Bar  
20%  
Compiling part 1

Lien code à barre pour MaTereApplication

OK

Note: ce code à barre n'est valide que 2 heures. Regardez la FAQ pour voir des informations sur comment partager votre application avec les autres.

Construire ▾ Aide ▾ Mes Projets

App ( Donnez le code QR pour fichier .apk )

App ( enregistrer .apk sur mon ordinateur )



Il reste maintenant à réaliser un montage électronique qui permet de recevoir en bluetooth les valeurs de 1 à 255 générés par l'application.

La solution la plus simple étant d'utiliser une interface programmable Arduino et une sortie PWM associée.



UNO et Grove - générer le code

répéter indéfiniment

si BT: données disponibles sur le port D8 = 1 alors

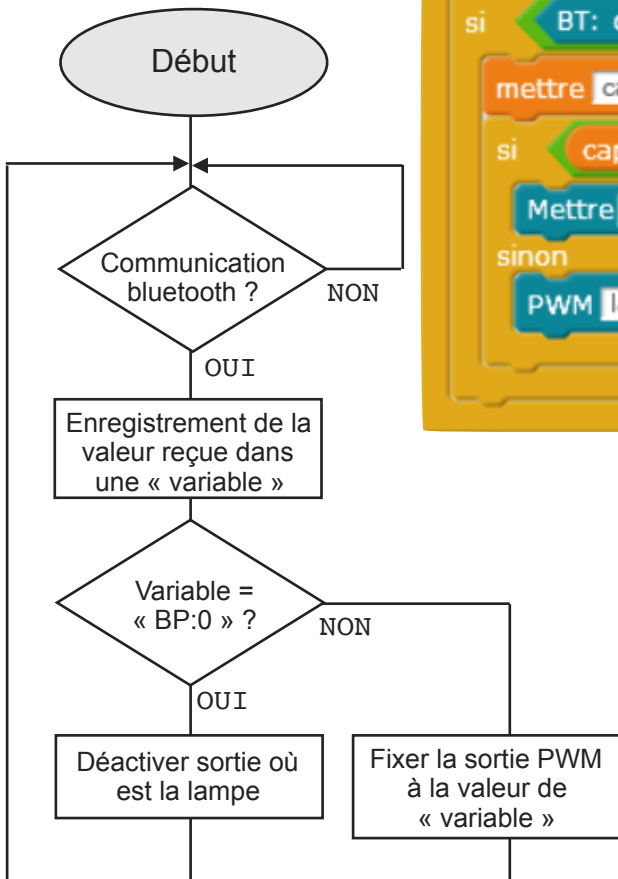
mettre capt\_lum à BT: recevoir la valeur de BP sur le port D8

si capt\_lum = 0 alors

Mettre la led verte sur la broche D3 à bas

sinon

PWM la led verte sur la broche D3 à capt\_lum



7

Programmer l'interface Arduino avec mBlock et la librairie « UNO et Grove » afin de piloter la Del (ici sur la broche D3) en fonction de la donnée reçue via le bluetooth (sur la broche D8)

UNO et Grove - générer le code



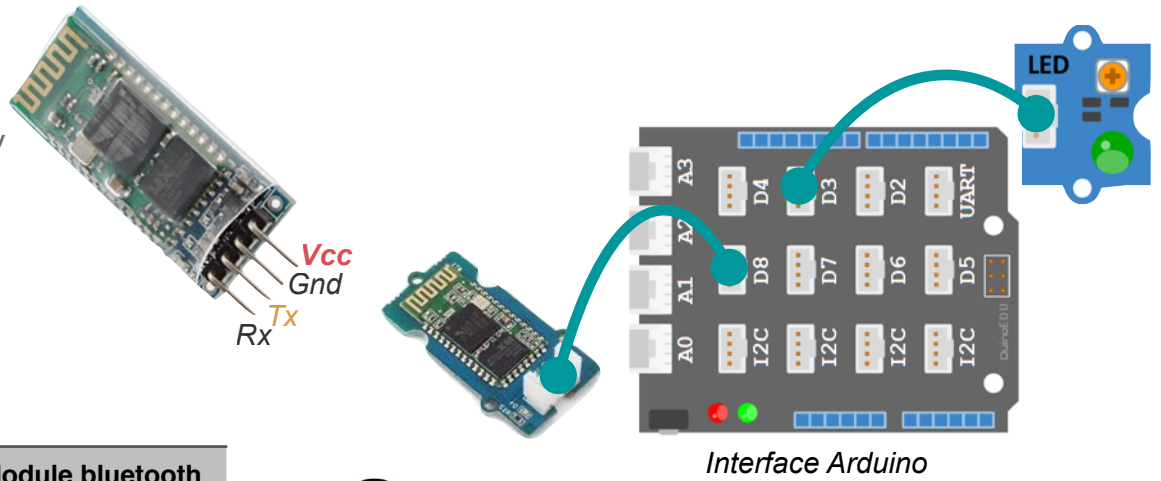
Vcc : alimentation 5V

Gnd : 0V

Tx : Port 8

Rx : Port 9

Communication à 115200 Bauds



Prise	Module bluetooth	
Port D8	Noir	Gnd -
	Rouge	Vcc +
	Blanc	Rx
	Jaune	Tx



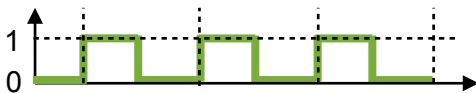
*Il n'est pas possible de téléverser un programme dans l'interface Arduino si un module bluetooth est connecté (alimenté).  
Il faut donc téléverser le programme sans brancher le module bluetooth.*



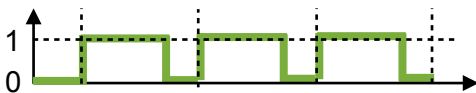
**Attention** ici on utilise la broche **D3** car seuls les ports **3, 5, 6, 9, 10 et 11** peuvent fournir une sortie analogique (PWM). Ils sont repérés par le symbole : ~

8

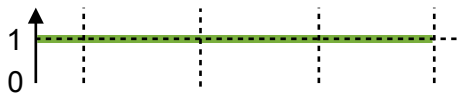
Réaliser le câblage sur l'interface Arduino et tester le bon fonctionnement de l'ensemble



Rapport Cvclique : 50% - Valeur sur 8 bits : 128 soit 2.5V



Rapport Cvclique : 75% - Valeur sur 8 bits : 192 soit 3,75V



Rapport Cvclique : 100% - Valeur sur 8 bits : 255 soit 5V