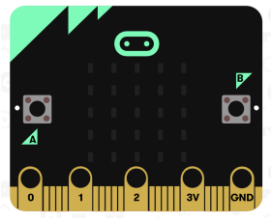
Être efficient	5e et 6e année

Document de programmation

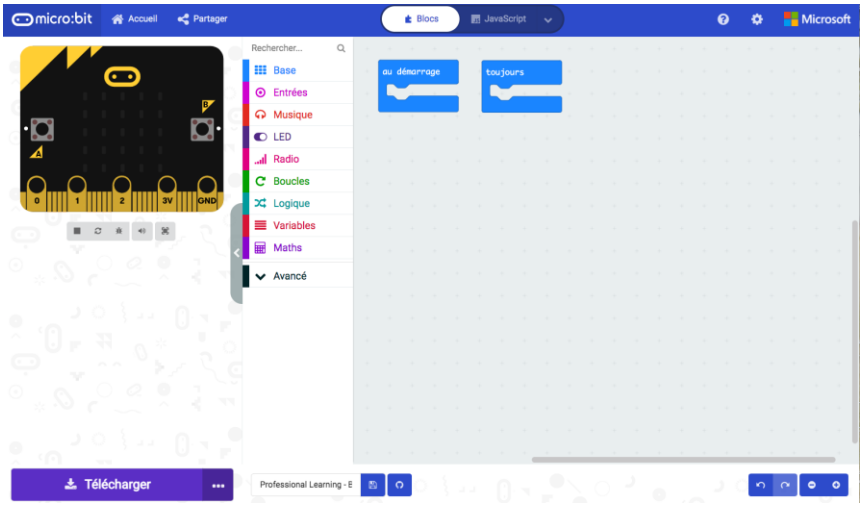
Ce guide étape par étape abordera la façon de faire l'activité avec le simulateur à l'écran. Les instructions sur la façon d'utiliser le matériel si vous y avez accès seront présentées à la fin du document.



Voici à quoi ressemble le simulateur micro:bit. Il y a deux boutons, A et B, une matrice de DEL (diode électroluminescente) et des connexions pour brancher les fils. Dans certains micro:bits plus récents, le logo sert aussi de bouton. Nous ne pouvons pas voir les capteurs sur le simulateur dès le départ, mais lorsque nous composons du code qui les utilise, ils apparaîtront le coin supérieur gauche du microcontrôleur. Le matériel micro:bit comprendra un câble USB pour le brancher à un ordinateur et un connecteur à batterie pour l'alimentation.

Activité 1 : Utiliser un capteur pour programmer une lumière plus efficiente

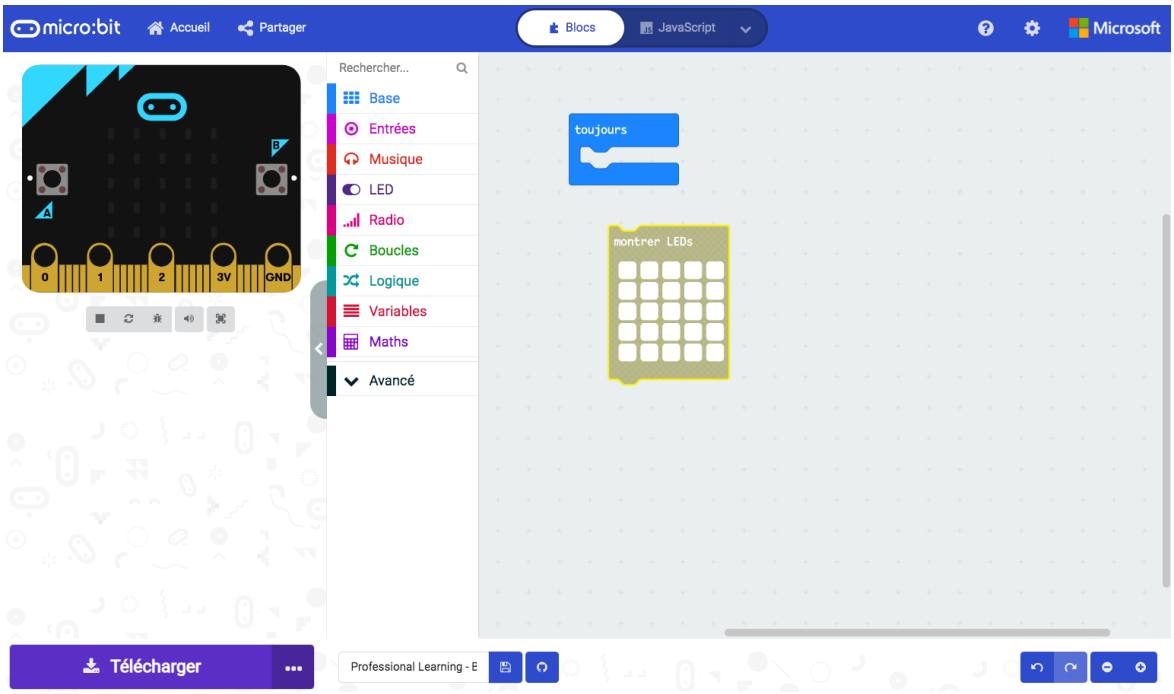
Prenons, par exemple, une lumière qui est toujours allumée. En utilisant un capteur de lumière dans le micro:bit, nous pouvons le programmer pour qu'il s'active seulement lorsqu'il fait noir, comme une veilleuse la nuit. Nous verrons le code pour ces deux cas.



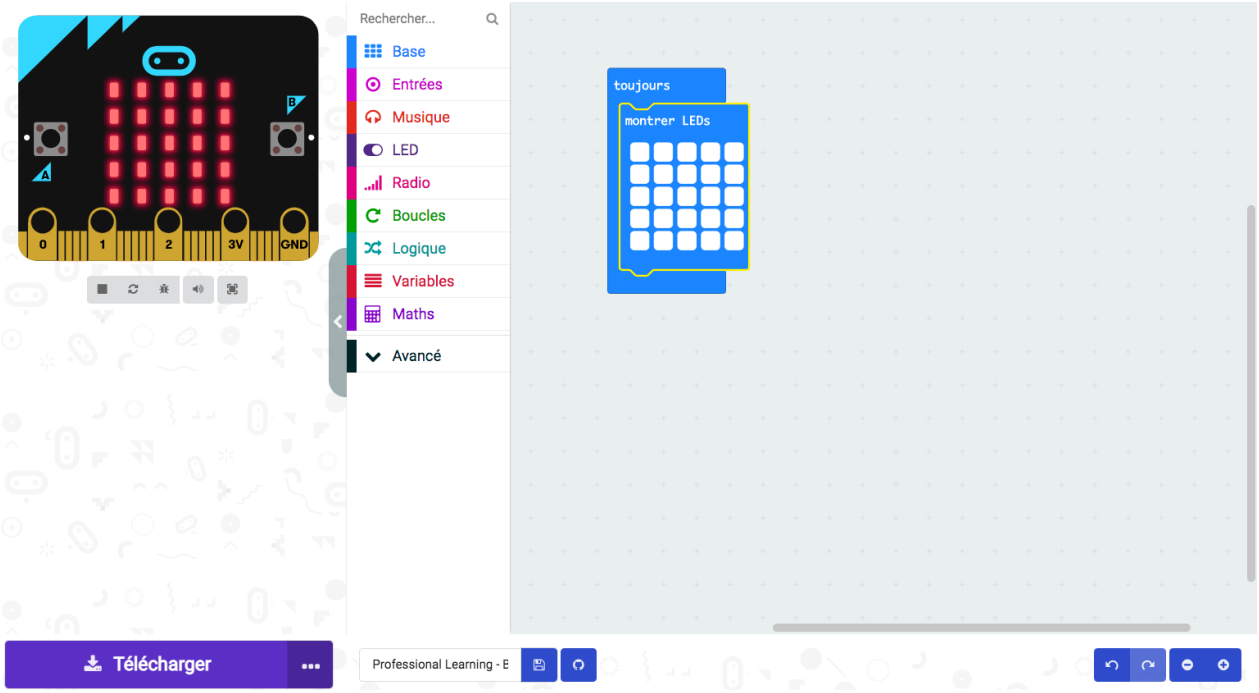
Lorsque nous commençons dans l'éditeur MakeCode, voici ce que nous voyons. À la gauche, il y a le simulateur. Ensuite, nous avons les menus des blocs de code où les différents blocs sont triés et colorés. À la droite, il y a la zone de code. C'est là que nous placerons les blocs de code que nous voulons exécuter.

Si nous voulons programmer une lumière qui est toujours allumée, nous utiliserons la boucle *toujours*. Dans le menu de blocs de code *Base*, nous pouvons en sortir le bloc *montrer LEDs*. Nous pouvons

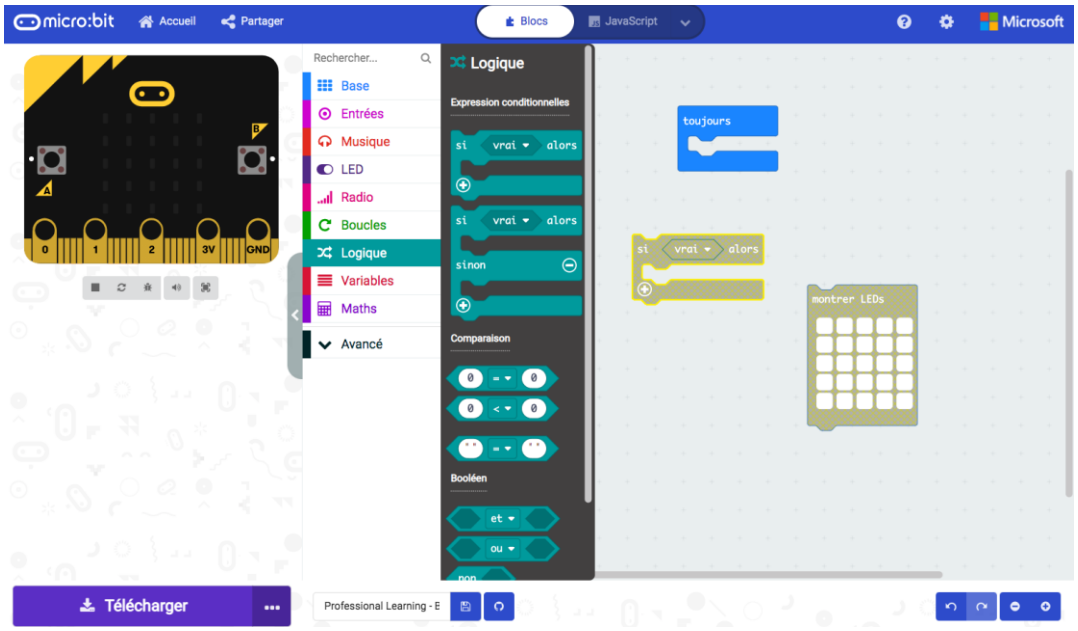
maintenant choisir les DEL dans la matrice qui seront allumées. Pour l'instant, nous les allumerons toutes.



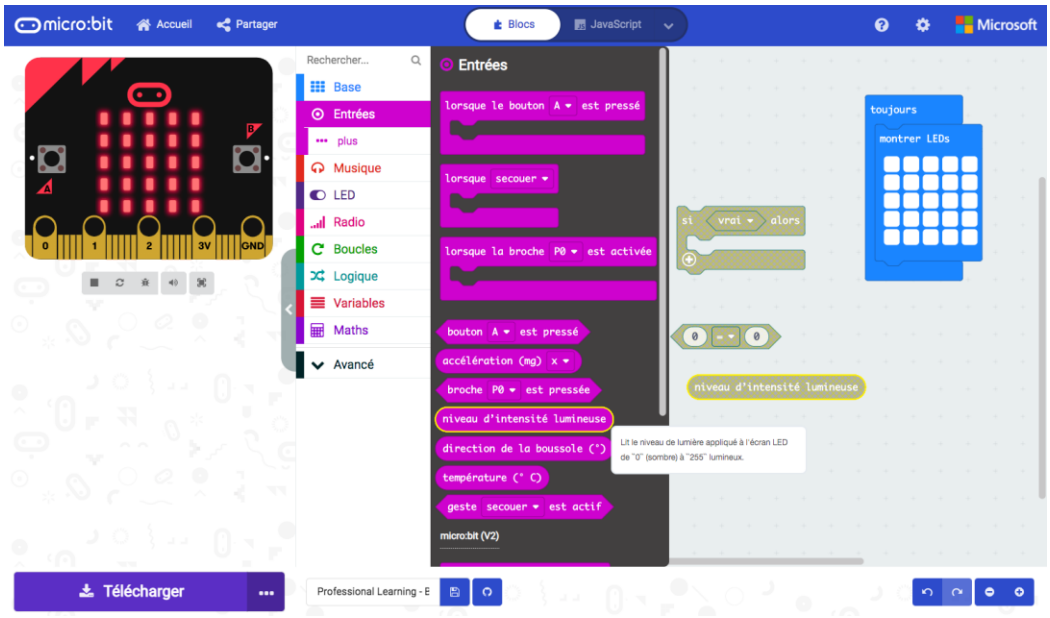
Une fois que nous avons glissé le bloc *montrer LEDs* dans le bloc *toujours*, le simulateur commencera à exécuter le code.



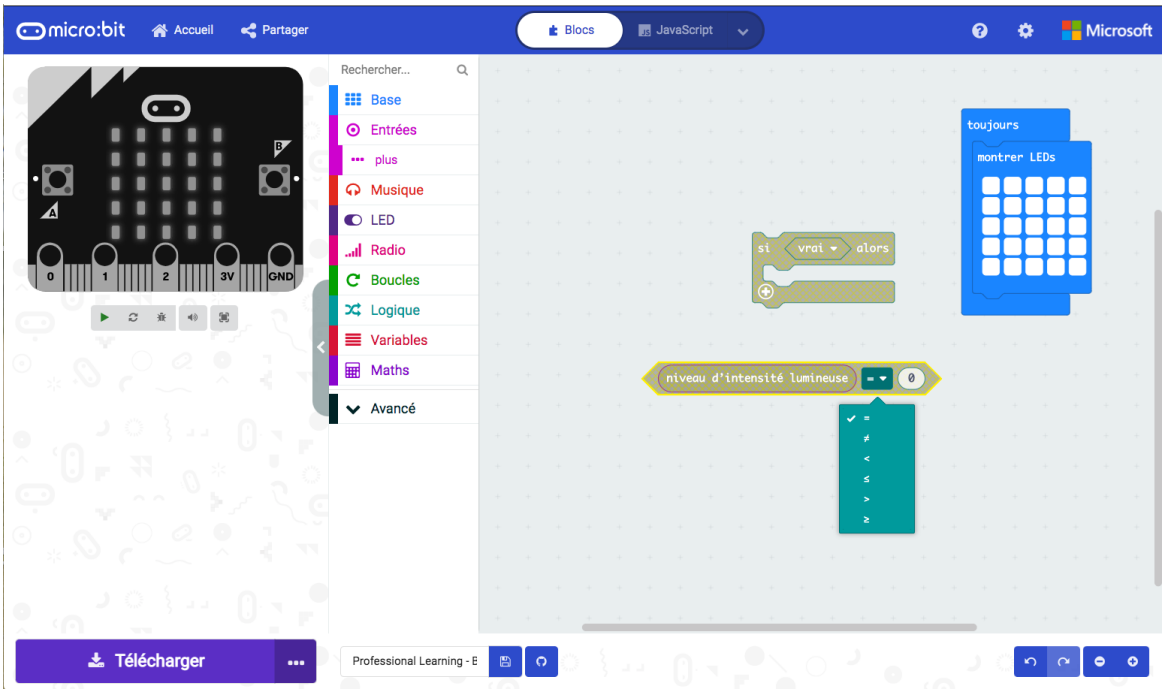
Si nous voulons changer la façon dont les DEL se comportent, nous devons changer leur code. Nous utiliserons le capteur de lumière intégré pour que les lumières s'allument seulement lorsqu'il fait noir. Nous pouvons faire cela au moyen d'une instruction « si-alors ». S'il fait noir, alors allumez les lumières. Nous pouvons choisir un bloc *si-alors* dans le menu *Logique*.



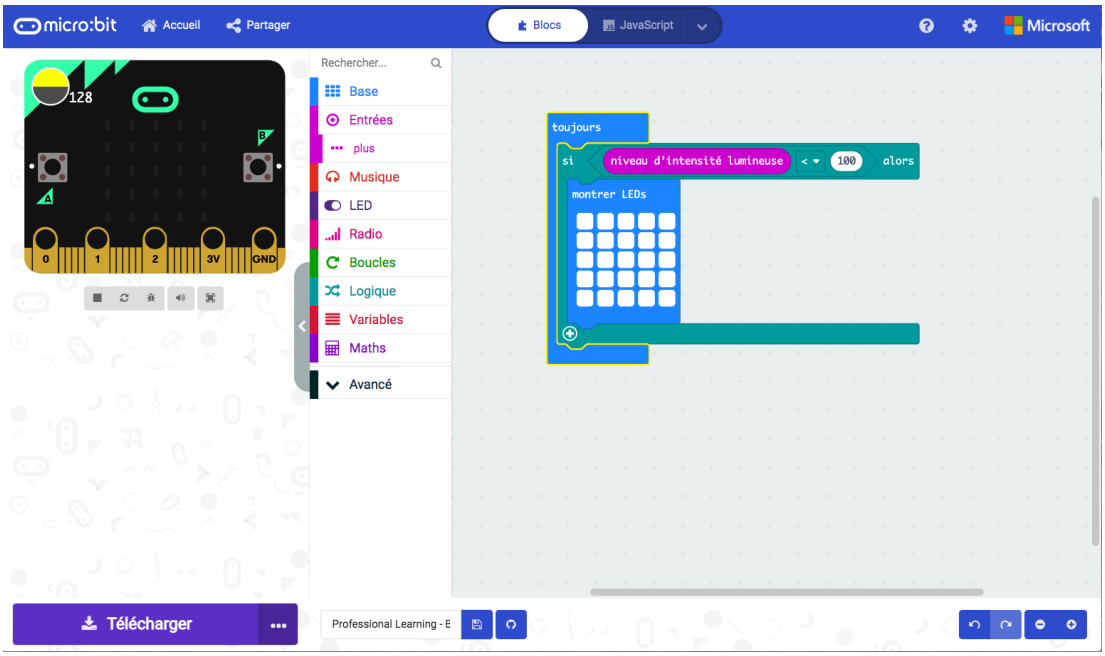
Ensuite, nous devrons configurer le code pour compléter l'instruction si-alors afin que le micro:bit sache qu'il fait « noir ». Le capteur de lumière intégré nous donnera les données dont nous avons besoin pour être en mesure de contrôler les DEL de cette façon. Le capteur de lumière prend l'information lumineuse, puis lui accorde un nombre entre 0 et 255. Lorsqu'il indique 0, il fait le plus noir. Lorsqu'il indique 255, il fait le plus clair. Nous devrons choisir un seuil pour ce que nous considérons être « faire noir » pour l'appliquer au code. Commençons par un seuil de 100 afin de garder les choses simples. Le bloc *niveau d'intensité lumineuse* se trouve dans le menu *Entrées.* Nous aurons également besoin d'un bloc *Comparaison* dans le menu *Logique*.



Ce sont tous les blocs de code dont nous aurons besoin. Nous avons simplement à les connecter ensemble de manière appropriée. Le bloc *niveau d'intensité lumineuse* devra s'insérer dans le bloc *Comparaison.* C'est là que nous ajouterons la valeur de notre seuil de lumière pour indiquer à quel moment les DEL s'allumeront. Si 0 est noir et nous voulons allumer nos lumières lorsqu'il fait plus noir que 100, nous devrons ajuster notre symbole de comparaison pour en tenir compte.



Une fois notre comparaison fixée, nous pouvons la placer dans l'instruction Si. Nous placerons également notre bloc *montre LEDs* dans la partie « alors » du bloc. Ensuite, nous plaçons le bloc *si-alors* dans la boucle *toujours.* Une fois cette étape terminée, le simulateur commencera à exécuter le code.



Nous pouvons utiliser notre curseur pour modifier l'intensité lumineuse sur le capteur de lumière vers le bas ou vers le haut pour voir comment se comporter les DEL lorsque nous changeons l'intensité lumineuse. Qu'arrive-t-il lorsque nous traversons le seuil? Dans ce cas-ci, avec la façon dont le code est écrit, une fois que nous arrivons en dessous de la ligne de 100, les DEL s'allumeront et resteront allumées, peu importe ce qui arrive à l'intensité lumineuse par après. Nous n'avons donné aucune instruction pour les éteindre.

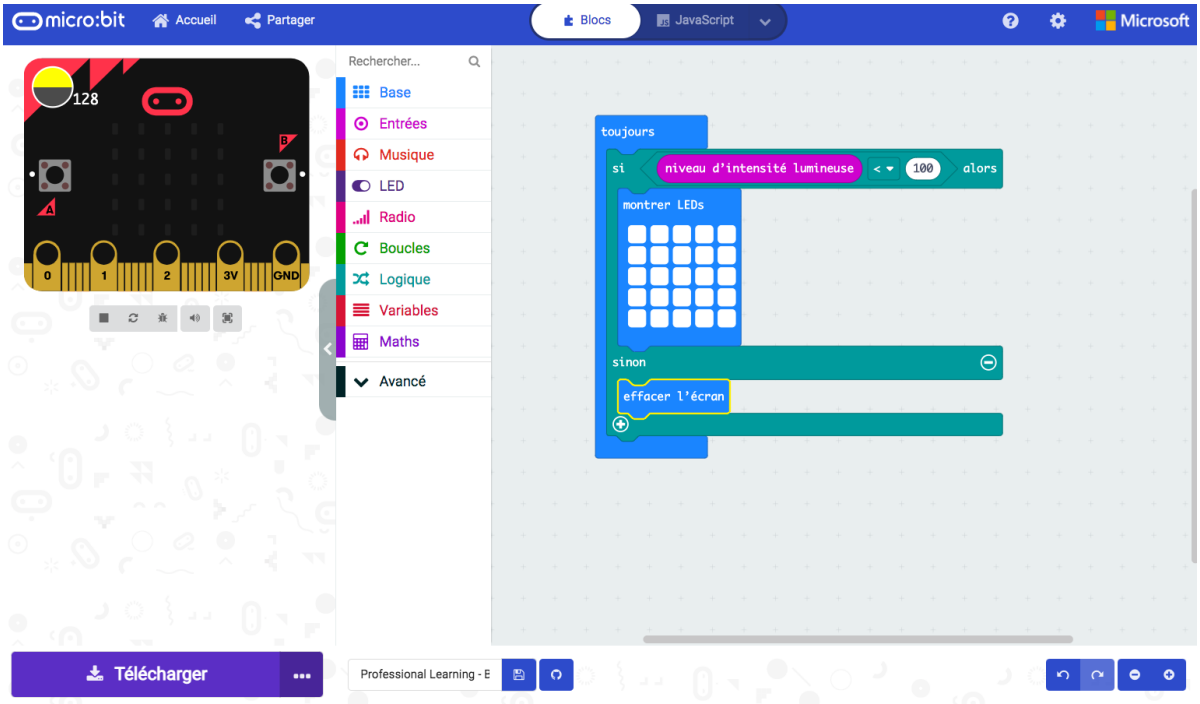
Nous pouvons ajouter un élément *sinon* à notre instruction *si-alors* afin que, si l'intensité lumineuse augmente au-delà de 100, les DEL s'éteignent. Ici nous pouvons revoir l'idée du code efficient puisque nous pourrions composer deux instructions *si-alors* séparées, mais nous pouvons aussi, pour une plus grande efficience, utiliser le *sinon* sur la première instruction à la place. En cliquant sur le symbole plus dans le coin inférieur gauche de l'instruction *si-alors*, nous ajoutons une ligne *sinon.* Nous programmerons les DEL pour qu'elles s'éteignent lorsque l'intensité lumineuse est de 100 ou plus en ajoutant le bloc *effacer l'écran* du menu *Base*.

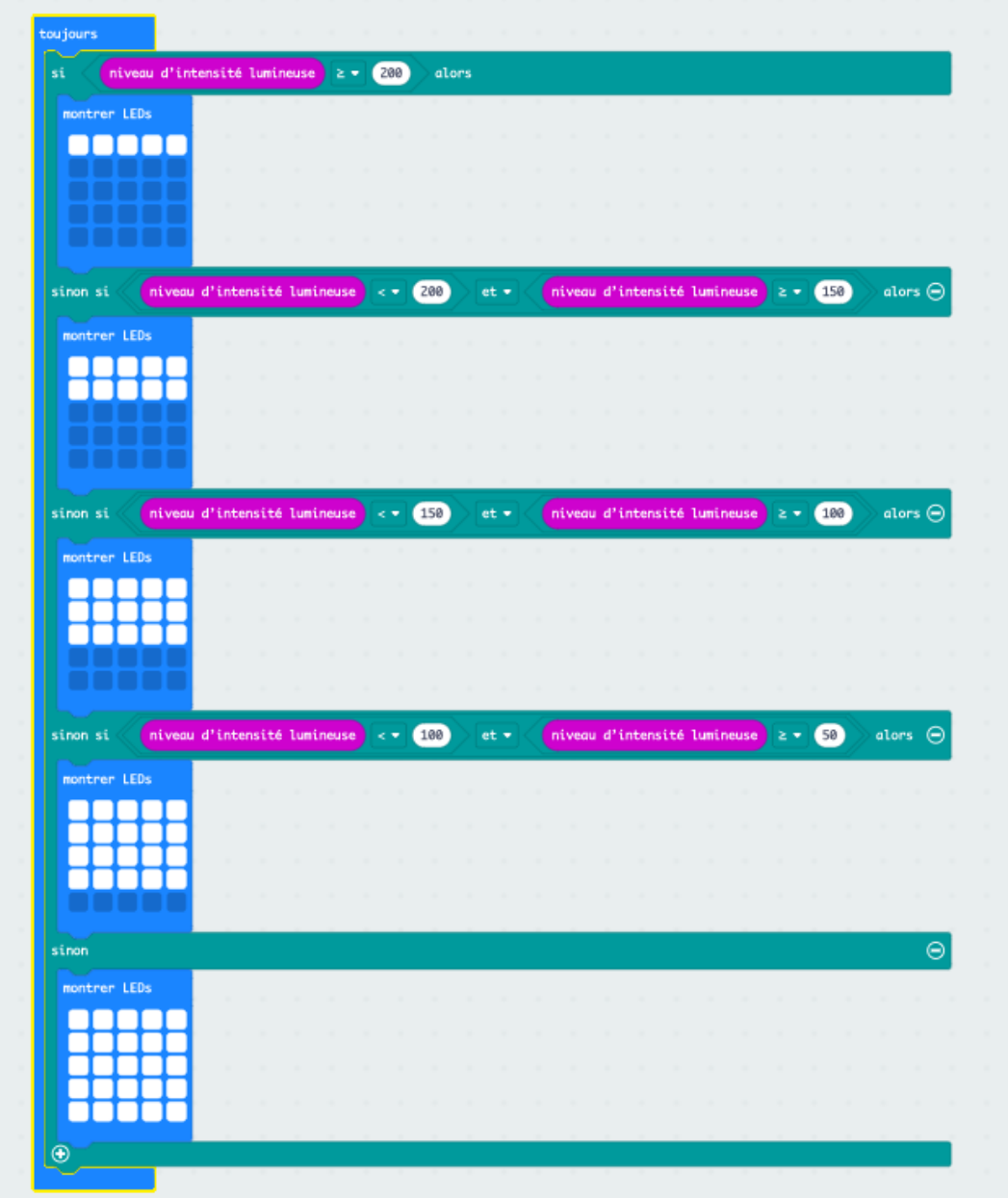
Après avoir ajouté le *sinon,* les DEL s'éteindront lorsque l'intensité lumineuse sera de 100 ou plus.

Mettez au défi vos étudiants de trouver d'autres conditions sous lesquelles leurs lumières pourraient fonctionner afin d'accroître leur efficience. Comment peuvent-ils modifier le code pour satisfaire aux différentes conditions?

Voici des exemples d'autres conditions :
Une lumière qui s'allume seulement lorsqu'un bouton est enfoncé et qu'il fait suffisamment noir.

Une lumière qui devient de plus en plus brillante plus il fait noir (code montré à la page suivante)
Une lumière qui s'allume seulement lorsque l'objet est dans une certaine position.



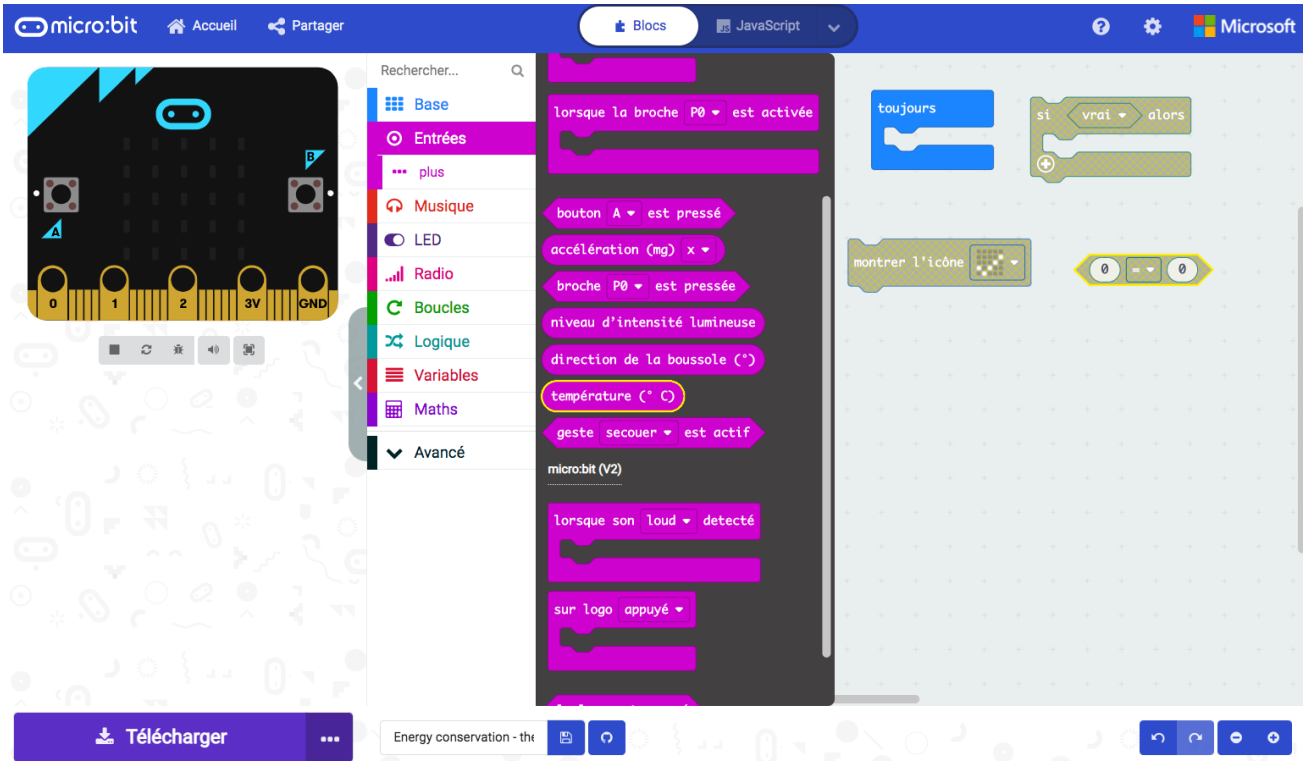


Une autre chose qui peut aider nos maisons à devenir plus efficientes est le thermostat programmable. Le thermostat programmable peut nous aider à fixer un point pour s'allumer et s'éteindre, afin qu'il ne soit pas toujours en fonction.

Puisque le micro:bit ne comprend pas une unité de chauffage, nous choisirons une source de lumière pour remplacer la fonction de chauffage. Plutôt que d'activer le chauffage, nous allumerons quelques lumières dans le panneau de DEL sur le micro:bit.

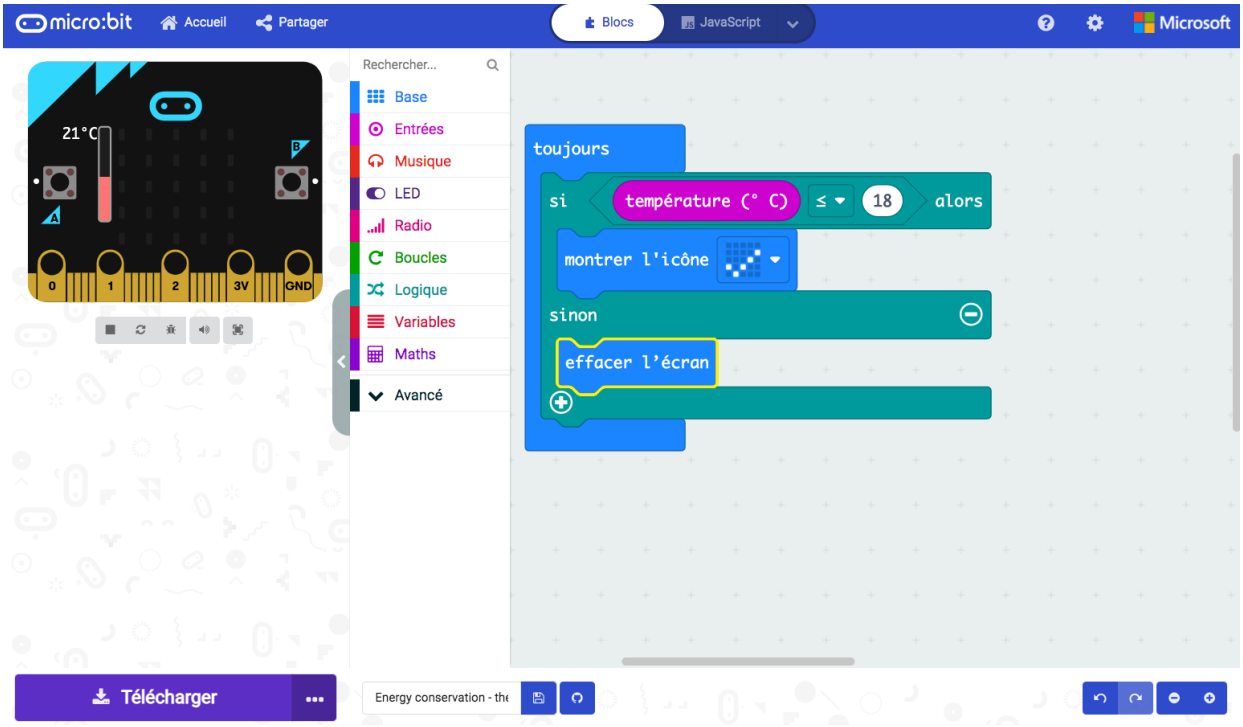
Le premier programme sera un simple thermostat qui contrôle seulement une unité de chauffage. Cela sera très semblable à la lumière qui s'allume seulement lorsqu'il fait noir. Nous utiliserons maintenant un bloc de capteur de température plutôt qu'un capteur de lumière. Le

bloc *température* se trouve dans le menu *Entrée.* Voici ci-dessous les blocs que nous utiliserons pour ce code. Nous avons une boucle *toujours* et des blocs *si-alors*, *comparaison* et *montrer l'icône* et nous sélectionnerons le bloc *température*.



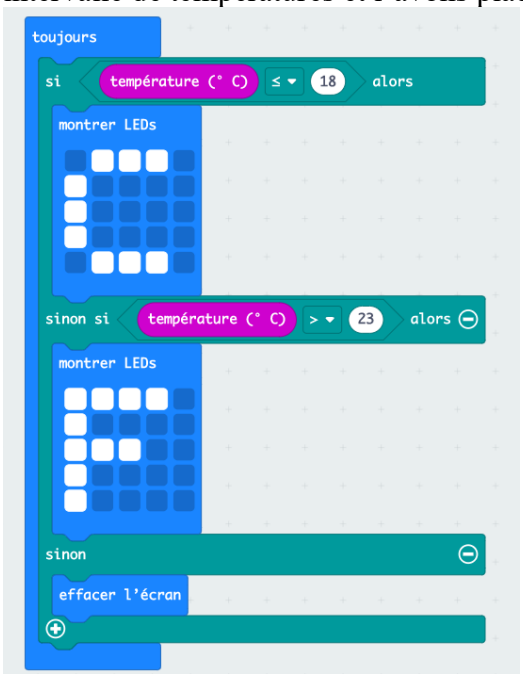
Le bloc *température* indique (°C), donc il mesurera la température en degrés Celsius. Les étudiants devront penser à leur seuil de température. Quelle influence a-t-il sur leur utilisation de l'unité de chauffage? Quelles sont les stratégies qu'ils peuvent utiliser pour être en mesure de maintenir confortablement un seuil inférieur?

Nous avons fixé l'intervalle aux températures inférieures ou égales à 18 °C. Cela signifie que si le capteur de température indique 18 ou moins, l'unité de chauffage s'allumera. Nous devrons aussi inclure un point pour qu'il s'éteigne. Nous utiliserons un *sinon* afin que si notre capteur de température indique au-dessus de 18, donc 19 ou plus, l'unité de chauffage s'éteindra.

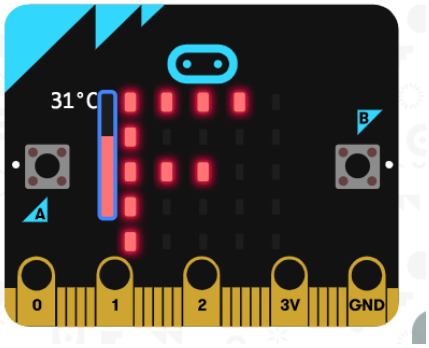


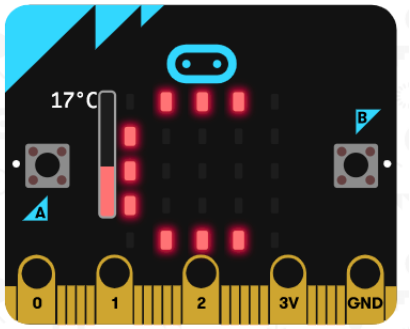
La modification suivante du programme tiendra compte d'une unité de chauffage et d'un conditionneur d'air. Quel intervalle de températures sera confortable?

Nous utiliserons les DEL pour écrire un H et indiquer que l'unité chauffe. Nous utiliserons aussi les DEL pour indiquer C lorsque le conditionneur d'air est en fonction. Nous avons choisi un intervalle de températures et l'avons placé dans le code.



Utilisez le curseur de température pour voir comment votre thermostat contrôle la maison.





Choses à considérer et auxquelles encourager vos étudiants à réfléchir au cours de cette activité : S'agirait-il d'un thermostat efficient? Est-ce que le chauffage et le conditionneur d'air peuvent être en fonction en même temps? Pouvons-nous avoir des codes différents pour différents moments de la journée ou différents moments de l'année? Comment pouvons-nous empêcher ces appareils de faire plus de travail qu'il est nécessaire? Se passer du conditionneur d'air peut être une excellente manière de conserver de l'énergie et ce ne sont pas toutes les maisons qui ont des conditionneurs d'air; y a-t-il d'autres façons de garder nos maisons au frais? Demandez aux étudiants de réfléchir au sujet des conditions ou des capteurs que nous pourrions utiliser pour accroître l'efficience. Si le temps le permet, demandez-leur de programmer ces conditions.

**Remarques sur l'utilisation du matériel micro:bit : Lorsque les étudiants ont terminé d'écrire leur code, ils peuvent cliquer sur le bouton de téléchargement dans le coin inférieur gauche de l'écran. Cela enregistre un fichier .hex sur l'ordinateur. Les étudiants doivent utiliser le connecteur USB pour brancher le micro:bit dans l'ordinateur, puis glisser ce fichier .hex dans le répertoire MICROBIT qui s'affiche. Lorsque le micro:bit sera allumé, il exécutera le code qui a été téléchargé. Cela est excellente façon d'explorer, particulièrement pour l'utilisation avec le code du capteur de lumière. Il est facile de faire varier la quantité de lumière sur un bureau en utilisant des objets ou vos mains pour créer des ombres. Il est plus difficile de faire varier la température de la même façon, mais il s'agit tout de même d'un exercice utile pour interagir avec une interface physique en plus de l'interface virtuelle.