



## Astro Pi : Mission Zero

Se préparer pour Mission Zero



### Étape 1 Introduction

Égaye la routine quotidienne des astronautes de la Station spatiale internationale en leur affichant un message et l'humidité relative à bord, à l'aide du Sense HAT de l'ordinateur Astro Pi.

Le matériel Astro Pi Mark II est si récent que les ordinateurs n'ont même pas encore de nom. Nous donnerons aux ordinateurs Astro Pi le nom de deux scientifiques européens inspirants. Tu peux aider à choisir comment ils s'appelleront en votant pour ton nom préféré dans le cadre de ton programme Mission Zero.

Tu utiliseras l'émulateur en ligne Sense HAT pour créer ton programme, aucun matériel supplémentaire n'est nécessaire - tout se fait dans un navigateur Web. *S'il répond aux critères d'éligibilité*, ton programme terminé sera exécuté dans l'espace sur la Station spatiale internationale (ISS) ! Tu recevras également un certificat spécial qui indique exactement où se trouvait l'ISS lorsque ton programme a été exécuté.

Nous annoncerons les noms choisis pour les nouveaux ordinateurs Astro Pi au printemps 2022.

#### Ce que tu feras

Voici un exemple du type de programme que tu peux créer. Clique sur **Run** (Exécuter) pour le voir en action.





### Ce que tu vas apprendre

Tu découvriras l'Astro Pi et apprendras à l'utiliser et à faire les actions suivantes :

- afficher des messages et ajouter des couleurs
- créer des images
- mesurer l'humidité à l'aide d'un Sense HAT

Ce projet couvre des éléments des sections suivantes du **Raspberry Pi Digital Making Curriculum** (<http://rpf.io/curriculum>) :

- **Utilise des concepts de base de programmation pour créer des programmes simples** (<https://curriculum.raspberrypi.org/programming/creator/>)



### Ce qu'il te faut

#### Matériel informatique

- Un ordinateur avec une connexion Internet

#### Logiciels

- Un navigateur Web (par exemple Google Chrome) pour ouvrir <https://trinket.io/mission-zero> (<https://trinket.io/mission-zero>)



## Notes pour les enseignants et les mentors

Cette activité peut être réalisée en une après-midi. Organisez vos élèves en équipes de deux à quatre personnes et laissez-nous les guider dans l'écriture d'un court programme Python pour afficher un message personnel et l'humidité relative sur l'Astro Pi.

Consultez **les directives officielles** ([https://astro-pi.org/media/mission-zero-guidelines/Astro\\_Pi\\_Mission\\_Zero\\_Guidelines\\_2021\\_22-fr.pdf](https://astro-pi.org/media/mission-zero-guidelines/Astro_Pi_Mission_Zero_Guidelines_2021_22-fr.pdf)) pour Mission Zero.

Vous devrez vous inscrire au défi Mission Zéro pour permettre à votre ou vos équipes de participer.

- Rendez-vous sur la page **de l'émulateur Trinket Mission Zero** (<https://trinket.io/mission-zero>).
- Remplissez le formulaire et cliquez sur **Submit (Soumettre)\***.

\* Veuillez noter que ce formulaire d'inscription est disponible en anglais seulement.

Les champs du formulaire incluent :

Nom de l'enseignant/mentor

Nom d'équipe

Nombre des membres de l'équipe

Noms et âges des membres de l'équipe

Dans quelle langue avez-vous consulté les directives de Mission Zero ?

- Un compte Trinket sera créé pour vous (si vous n'en avez pas déjà un, ou si vous n'êtes pas connecté). Vous pouvez créer un compte par adresse e-mail. Chaque compte a son propre **code de salle de classe**, et vous devez donner ce code à votre ou vos équipes quand elles sont prêtes à soumettre leurs programmes.
- Affichez votre code de salle de classe quelque part où il sera visible, par exemple sur un tableau blanc ou en utilisant un projecteur, et commencez l'activité.

Nous avons créé un **document imprimable de deux pages** (<http://rpf.io/mz-printout>) qui couvre les points clés de Mission Zéro et que les élèves et les jeunes peuvent utiliser avec ce projet en ligne.

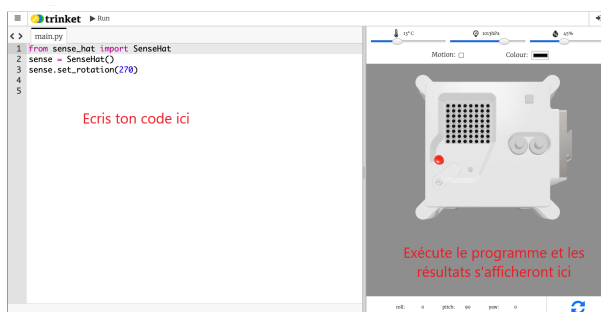
## Étape 2 Qu'est-ce qu'un Astro Pi ?

---

Un Astro Pi est un ordinateur Raspberry Pi intégré dans un boîtier spécialement conçu pour les conditions spatiales. Il possède également une carte supplémentaire appelée Sense HAT, spécialement conçue pour la mission Astro Pi. Le Sense HAT est équipé d'un joystick, d'un écran LED et de capteurs permettant d'enregistrer les conditions d'éclairage, la température, l'humidité, la pression et l'orientation.

Voici une unité originale Mark I Astro Pi sur la Station spatiale internationale, exécutant un code écrit par des élèves. Votre code sera peut-être exécuté sur une nouvelle version des ordinateurs Astro Pi !

Pour cette mission, tu utiliseras l'émulateur Sense HAT. L'émulateur est un logiciel qui simule toutes les fonctions de l'Astro Pi dans ton navigateur Web.



Il existe quelques différences entre le Sense HAT réel et celui qui est émulé :

- Sur l'émulateur, tu peux régler toi-même la couleur de la lumière, la température, la pression et l'humidité à l'aide de curseurs, alors que le véritable Sense HAT de l'Astro Pi utilise des capteurs pour mesurer ces paramètres dans son environnement.
- Tu peux utiliser la souris pour cliquer et faire glisser le Sense HAT émulé afin de le déplacer et de le faire pivoter, simulant ainsi des changements d'orientation. Le véritable Astro Pi (et son Sense HAT) peut se déplacer dans le monde réel, et les capteurs d'orientation du Sense HAT détectent quand et comment il a bougé.

## Étape 3

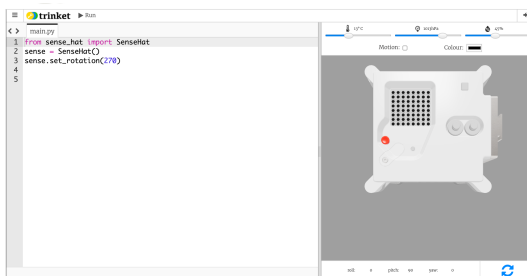
### Afficher un message et choisir un nom pour les nouveaux ordinateurs Astro Pi

Ouvre l'émulateur Sense HAT (<https://trinket.io/mission-zero>) pour le projet Mission Zero.



Tu vas constater que trois lignes de code ont été ajoutées automatiquement :

```
from sense_hat import SenseHat
sense = SenseHat()
sense.set_rotation(270)
```



Ce code se connecte à l'Astro Pi et s'assure que l'écran LED de l'Astro Pi est affiché dans le bon sens. Laisse ce code ici car tu en auras besoin.

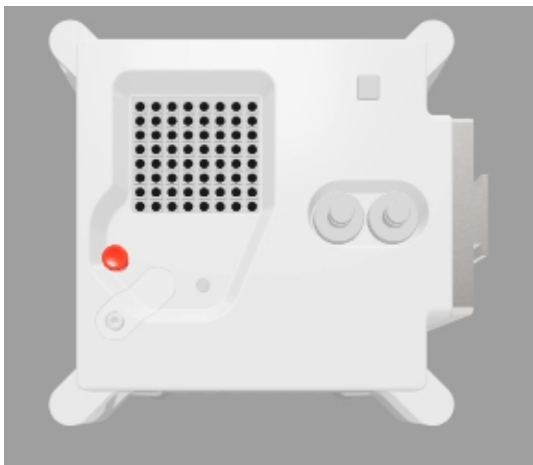
Peut-être pourrais-tu laisser un message de bienvenue aux astronautes de l'ISS qui travaillent près de l'Astro Pi ? Faisons défiler un message sur l'écran.



Ajoute cette ligne en-dessous de l'autre ligne de code :

```
sense.show_message("Astro Pi")
```

Appuie sur le bouton **Run** (Exécuter) et regarde le message **Astro Pi** défiler sur l'écran LED. 




Pour afficher un message différent, tu peux écrire ce que tu veux entre les guillemets ("").

### Quels caractères peuvent être utilisés ?

Le Sense HAT ne peut afficher que le jeu de caractères Latin 1, ce qui signifie que seuls les caractères suivants sont disponibles. Les autres caractères s'afficheront sous la forme d'un ? .

```
+ - * / ! " # $ % & ' ( ) * + , - . : ;  
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z  
a b c d e f g h i j k l m n o p q r s t u v w x y z  
? , ; : | @ % [ & _ ' ] \ ~
```

Tu peux aussi modifier la vitesse de défilement du message sur l'écran. Ajoute un `scroll_speed` (vitesse de défilement) à ta ligne de code, comme ceci : 

```
sense.show_message("Astro Pi", scroll_speed=0.05)
```

La vitesse par défaut du message est `0.1`. En réduisant le nombre, tu fais défiler le message plus rapidement et en augmentant le nombre tu fais défiler le message plus lentement.

**Choisis un nom pour les nouveaux ordinateurs Astro Pi**

Nous donnerons aux ordinateurs Astro Pi le nom de deux scientifiques européens inspirants. Des centaines d'hommes et de femmes ont contribué à la science et à la technologie ; les participants peuvent suggérer leur propre nom ou choisir dans notre liste de suggestions (veillez à utiliser la version anglaise du nom) :



**Ada Lovelace** ([https://en.wikipedia.org/wiki/Ada\\_Lovelace](https://en.wikipedia.org/wiki/Ada_Lovelace))

**Alan Turing** ([https://en.wikipedia.org/wiki/Alan\\_Turing](https://en.wikipedia.org/wiki/Alan_Turing))

**Caroline Herschel** ([https://en.wikipedia.org/wiki/Caroline\\_Herschel](https://en.wikipedia.org/wiki/Caroline_Herschel))

**Edsger Dijkstra** ([https://en.wikipedia.org/wiki/Edsger\\_W.\\_Dijkstra](https://en.wikipedia.org/wiki/Edsger_W._Dijkstra))

**Hedy Lamarr** ([https://en.wikipedia.org/wiki/Hedy\\_Lamarr](https://en.wikipedia.org/wiki/Hedy_Lamarr))

**Hypatia** (<https://en.wikipedia.org/wiki/Hypatia>)

**John Edmonstone** ([https://en.wikipedia.org/wiki/John\\_Edmonstone](https://en.wikipedia.org/wiki/John_Edmonstone))

**Marie Curie** ([https://en.wikipedia.org/wiki/Marie\\_Curie](https://en.wikipedia.org/wiki/Marie_Curie))

**Nikola Tesla** ([https://en.wikipedia.org/wiki/Nikola\\_Tesla](https://en.wikipedia.org/wiki/Nikola_Tesla))

**Tycho Brahe** ([https://en.wikipedia.org/wiki/Tycho\\_Brahe](https://en.wikipedia.org/wiki/Tycho_Brahe))

Tu dois débiter ton message par « My name should be » (en anglais). Par exemple, si tu souhaites voter pour Ada Lovelace, ton code ressemblerait à ceci :

```
sense.show_message("My name should be Ada Lovelace")
```

Si tu souhaites voter, ton message doit commencer par ces mots, sinon nous ne pourrons pas compter ton vote.

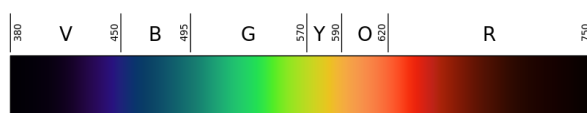
## Étape 4 Ajouter de la couleur

Les LED de l'Astro Pi peuvent également afficher des couleurs. Tu peux spécifier une couleur en créant une variable et en lui attribuant une valeur de couleur RGB.

Tu peux apprendre ici comment créer toutes les couleurs en utilisant différentes valeurs de rouge, de vert et de bleu :

### Représenter des couleurs avec des nombres

La couleur d'un objet dépend de la couleur de la lumière qu'il réfléchit ou émet. La lumière peut avoir différentes longueurs d'onde et la couleur de la lumière dépend de la longueur d'onde. La couleur de la lumière en fonction de sa longueur d'onde peut être vue dans le diagramme ci-dessous. Tu pourrais reconnaître cela comme les couleurs de l'arc-en-ciel.



Les humains voient la couleur grâce à des cellules spéciales dans nos yeux. Ces cellules sont appelées *cônes*. Nous avons trois types de cellules de cône et chaque type détecte soit une lumière rouge, bleue ou verte. Par conséquent, toutes les couleurs que nous voyons ne sont que des mélanges de couleurs rouge, bleu et vert.



Dans le mélange de couleurs additif, trois couleurs (rouge, vert et bleu) sont utilisées pour créer d'autres couleurs. Dans l'image ci-dessus, il y a trois spots de même luminosité, un pour chaque couleur. En l'absence de toute couleur, le résultat est noir. Si les trois couleurs sont mélangées, le résultat est blanc. Lorsque rouge et vert se combinent, le résultat est jaune. Lorsque rouge et bleu se combinent, le résultat est magenta. Lorsque bleu et vert se combinent, le résultat est cyan. Il est possible de faire encore plus de couleurs en faisant varier la luminosité des trois couleurs originales utilisées.

Les ordinateurs stockent tout en tant que 1 et 0. Ces 1 et 0 sont souvent organisés en ensembles de 8, appelés **octets**.

Un seul octet peut représenter n'importe quel nombre de 0 à 255.



Lorsque nous voulons représenter une couleur dans un programme informatique, nous pouvons le faire en définissant les quantités de rouge, de bleu et de vert qui composent cette couleur. Ces quantités sont généralement stockés sous forme d'un seul octet et donc comme un nombre compris entre 0 et 255.

Voici un tableau montrant certaines valeurs de couleur:

### Rouge Vert Bleu Couleur

255	0	0	Rouge
0	255	0	Vert
0	0	255	Bleu
255	255	0	Jaune
255	0	255	Magenta
0	255	255	Cyan

Tu peux trouver un bon **sélecteur de couleurs pour jouer avec** au W3Schools ([https://www.w3schools.com/colors/colors\\_rgb.asp](https://www.w3schools.com/colors/colors_rgb.asp)).

Choisis une couleur et découvre la valeur RGB de cette couleur. Tu peux utiliser un **sélecteur de couleur** ([https://www.w3schools.com/colors/colors\\_rgb.asp](https://www.w3schools.com/colors/colors_rgb.asp)) pour t'aider.



Crée une variable qui servira à stocker la couleur que tu as choisie. Par exemple, si tu as choisi la couleur rouge, tu vas écrire cette ligne de code :

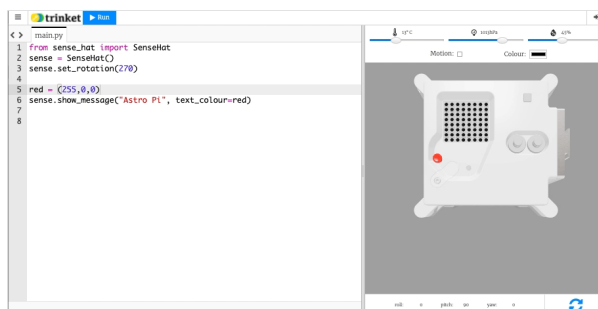


```
red = (255,0,0)
```

Tu peux maintenant afficher ton texte dans la couleur de ton choix ! Pour indiquer au programme d'utiliser la couleur que tu as créée, ajoute le paramètre `text_colour` (couleur du texte) à la ligne de code qui affiche ton texte :



```
red = (255,0,0)
sense.show_message("Astro Pi", text_colour=red)
```



Tu peux également modifier la couleur d'arrière-plan de l'écran. Choisis une autre couleur et crée une autre variable pour cette couleur. Pour indiquer au programme d'utiliser la couleur d'arrière-plan que tu as choisie, ajoute le paramètre `back_colour` (couleur de l'arrière-plan) à ton code :



```
red = (255,0,0)
green = (0,255,0)
sense.show_message("Astro Pi", text_colour=red, back_colour=green)
```

Modifie le texte et la couleur du message de bienvenue - quel message veux-tu envoyer aux astronautes à bord de l'ISS ?



## Étape 5 Afficher une image

Tu peux afficher des images sur la matrice LED de l'Astro Pi. Ton message de bienvenue pour les astronautes pourrait peut-être inclure une image ou un motif avec le texte, ou à la place du texte ?



En bas de ton programme, crée des variables pour les couleurs que tu veux définir pour dessiner une image. Tu peux utiliser autant de couleurs que tu veux mais dans cet exemple, nous nous limiterons à quelques couleurs - blanc (**w**), noir (**b**), et deux nuances de gris (**g** et **s**). Remarque que les nuances sont obtenues en réduisant la quantité de lumière dans les trois canaux tout en gardant les mêmes quantités.



```
w = (255, 255, 255)
b = (0, 0, 0)
g = (50, 50, 50)
s = (200, 255, 200)
r = (255, 0, 0)
```

**Remarque :** Cette fois-ci, il est conseillé de donner aux variables définies pour les couleurs des noms se limitant à une lettre, car cela permettra de gagner du temps à l'étape suivante, quand tu les sairas de nombreuses fois. De plus, en utilisant des noms à une seule lettre tu pourras voir plus facilement l'image que tu vas dessiner.

Sous tes nouvelles variables, crée une liste de 64 éléments. Chaque élément représente un pixel de la matrice du LED et correspond à l'une des variables de couleur que tu as définies. Dessine ton image en mettant une variable à l'endroit où tu veux que la couleur de cette variable apparaisse. Nous avons dessiné un Astro Pi en utilisant les pixels noirs (**b**) pour l'arrière-plan et les pixels gris (**g**) pour dessiner les parties métalliques du boîtier de l'Astro Pi :



```
picture = [  
  g, b, b, b, b, b, b, b, g,  
  b, g, g, g, g, g, g, b,  
  b, g, b, b, g, w, g, g,  
  b, g, b, b, g, g, g, g,  
  b, g, g, g, s, s, g, g,  
  b, g, r, g, g, g, g, g,  
  b, g, g, g, g, g, g, b,  
  g, b, b, b, b, b, b, g  
]
```

Ajoute une ligne de code pour afficher ton image sur l'écran LED.



```
sense.set_pixels(picture)
```

Appuie sur **Run** (Exécuter) pour afficher ton image.



Tu peux ajouter du code pour inclure une courte attente (ou **sleep**) après que l'image soit affichée. Cela donnera aux astronautes le temps de voir ton image avant l'affichage de la partie suivante de ton message. En haut de ton programme, rajoute :



```
from time import sleep
```

Ensuite, sur la ligne qui suit celle qui affiche ton image, ajoute ce code pour créer un temps d'attente de deux secondes :

```
sleep(2)
```

Crée ta propre image ou ton propre motif que tu veux afficher pour les astronautes !



## Étape 6 Mesurer l'humidité

---

Le capteur d'humidité de l'Astro Pi peut mesurer l'humidité de l'air qui l'entoure, c'est une fonction utile pour t'aider à collecter des données sur les conditions dans l'espace.



L'Astro Pi mesure l'humidité dans l'ISS en pourcentage de concentration d'eau dans l'air.

Une partie de ta mission est de contribuer à la vie quotidienne de l'équipage à bord de l'ISS, afin de leur faire savoir que l'humidité à bord de la station spatiale est dans les limites normales, cela les rassurera.

### Qu'est-ce que l'humidité ?

L'humidité est la quantité de vapeur d'eau dans l'air. La vapeur d'eau est l'état gazeux de l'eau.

La quantité de vapeur d'eau en suspension dans l'air dépend de la température :

- Plus la température de l'air est élevée, plus la vapeur d'eau peut être en suspension dans l'air
- Plus la température de l'air est basse, moins la vapeur d'eau peut être en suspension



Lorsque tu sors une canette ou une bouteille froide du réfrigérateur, tu verras rapidement de l'eau y apparaître. Cela se produit parce que la bouteille froide refroidit l'air qui l'entoure, ce qui rend l'air moins apte à suspendre la vapeur d'eau. Cela fait alors que la vapeur d'eau qui ne peut pas être suspendue se transforme en eau liquide. C'est ce qu'on appelle la *condensation*. Donc, compte tenu de ces informations, nous devons ensuite comprendre qu'il existe deux façons de mesurer l'humidité :

- L'humidité **absolue** est la masse totale de vapeur d'eau en suspension dans un volume d'air donné. La température n'est pas prise en considération. Cette mesure est généralement exprimée en grammes d'eau par mètre cube d'air.
- L'humidité **relative**, cependant, est exprimée en pourcentage. Pour une température de l'air donnée, il y a une quantité maximale de vapeur d'eau qui peut être suspendue dans l'air. L'humidité relative est le pourcentage de vapeur d'eau réelle présente, par rapport à la quantité maximale possible.

Cela signifie qu'une quantité connue de vapeur d'eau entraînera des lectures d'humidité relative différentes en fonction de la température de l'air. Par exemple, une température de l'air basse peut entraîner une lecture d'humidité relative élevée, car l'air ne peut pas suspendre beaucoup plus de vapeur d'eau. Augmenter la température de l'air et conserver la même quantité de vapeur d'eau entraînera une baisse de la lecture d'humidité relative, car la quantité maximale de vapeur d'eau qui **pourrait** être suspendue a augmenté.

Ajoute ce code pour effectuer un relevé d'humidité :



```
humid = sense.get_humidity()
```

Cette ligne mesure l'humidité actuelle et stocke la valeur mesurée dans la variable `humid`.

L'humidité est enregistrée de manière très précise, c'est-à-dire que la valeur stockée aura un grand nombre de décimales. Tu peux arrondir la valeur à n'importe quel nombre de décimales. Dans l'exemple, nous avons arrondi à une décimale, mais pour avoir un autre niveau de précision remplace le nombre `1` par le nombre de décimales que tu souhaites.



```
humid = round(sense.get_humidity(), 1)
```

Pour afficher l'humidité actuelle sous la forme d'un message défilant à l'écran, ajoute cette ligne de code :



```
sense.show_message(str(humid))
```

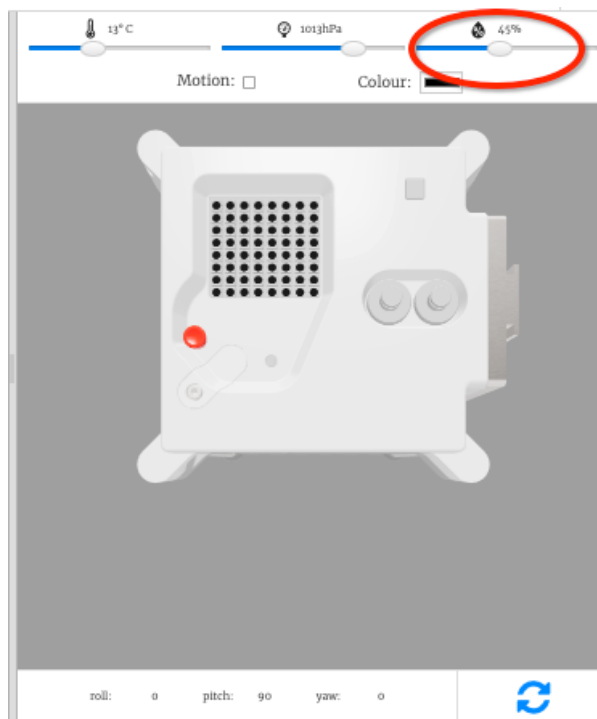
La partie `str()` convertit l'humidité d'un nombre en texte afin que l'Astro Pi puisse l'afficher.

Tu peux également afficher l'humidité dans un autre message en joignant les parties de ton message avec un `+`.



```
sense.show_message( "It is " + str(humid) + " %" )
```

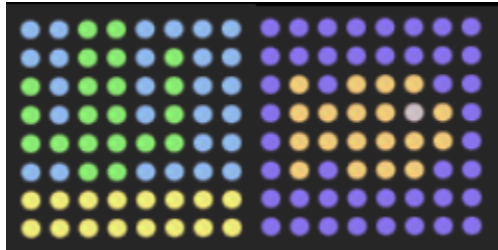
Le vrai Astro Pi mesurera l'humidité autour de lui, mais tu peux déplacer le curseur d'humidité sur l'émulateur Sense HAT pour simuler les changements d'humidité et tester ton code.



**Remarque :** Tu te demandes peut-être pourquoi le curseur d'humidité affiche l'humidité sous forme d'un nombre entier, mais le résultat de la mesure que tu obtiens est en décimales. L'émulateur simule la légère imprécision du capteur réel, de sorte que le résultat de la mesure de l'humidité que tu vois peut être très légèrement supérieure ou inférieure à la valeur que tu as définie avec le curseur.

## Étape 7 Réagir à l'humidité

Tu peux combiner ton relevé d'humidité avec une image pour indiquer également l'humidité d'une manière graphique. Par exemple, tu peux afficher un océan pour une humidité élevée, et un désert pour une faible humidité :



Au bas de ton programme, crée d'autres variables de couleur pour toutes les couleurs que tu souhaites utiliser dans tes images. Tu en as peut-être déjà défini certaines lors d'une étape précédente.



```
o=(255,130,0)
b=(0,0,255)
c=(0,150,255)
e=(80,80,80)
g=(0,255,0)
y=(255,255,0)
```

Comme précédemment, dessine tes images en créant d'abord une liste pour chacune d'entre elles, puis en indiquant la couleur que tu veux donner aux pixels de chaque élément de la liste.



```
wet = [
  b, b, b, b, b, b, b, b,
  b, b, b, b, b, b, b, b,
  b, o, b, o, o, o, b, b,
  b, o, o, o, o, e, o, b,
  b, o, o, o, o, o, o, b,
  b, o, b, o, o, o, b, b,
  b, b, b, b, b, b, b, b,
  b, b, b, b, b, b, b, b
]

dry = [
  c, c, g, g, c, c, c, c,
  c, c, g, g, c, g, c, c,
  g, c, g, g, c, g, c, c,
  g, c, g, g, c, g, c, c,
  g, g, g, g, g, g, c, c,
  c, c, g, g, c, c, c, c,
  y, y, y, y, y, y, y, y,
  y, y, y, y, y, y, y, y
]
```



Ajoute du code pour obtenir l'humidité :



```
humid = sense.get_humidity()
```

Décide maintenant quelle image tu veux afficher. Pour cet exemple, nous afficherons l'image **wet** (humide) si la lecture de l'humidité est de 40 % ou plus, et l'image **dry** (sec) si l'humidité est inférieure à 40 %.



```
humid = sense.get_humidity()
if humid >= 40:
    sense.set_pixels(wet)
else:
    sense.set_pixels(dry)
```

Utilise le curseur d'humidité pour définir une humidité sur l'émulateur. Exécute ton programme et vérifie que l'image que tu as choisie pour cette humidité est correctement affichée.



Modifie ton code pour que ton programme affiche l'humidité pour les astronautes de la manière que tu as choisie.

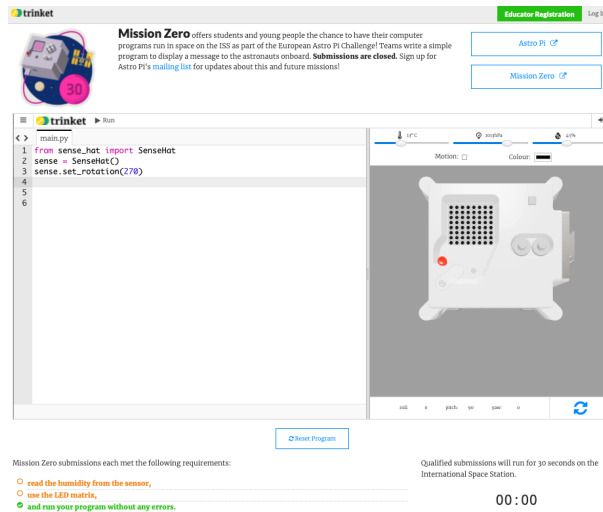


Teste ton code avec quelques réglages d'humidité différents (à l'aide du curseur) pour t'assurer qu'il fonctionne toujours correctement. Si tu as suivi l'exemple ci-dessus, une image est-elle affichée à la fois lorsque l'humidité est réglée à une valeur inférieure à 40 % et aussi quand elle est réglée à plus de 40 % ?



## Étape 8 Soumettre ton programme

Ton code doit respecter quelques règles pour que tu puisses le soumettre et le faire exécuter dans la Station Spatiale Internationale. Si ton code les respecte, les règles situées en bas de l'**émulateur Sense HAT** (<https://trinket.io/mission-zero>) s'allumeront en vert lorsque tu exécuteras le programme.



1. Mesure l'humidité.
2. Allume les LED.
3. Vérifie que ton code s'exécute jusqu'au bout sans erreur. Tu ne dois pas inclure de boucles `while True` dans ton code car cela fera exécuter le code en continu et l'empêchera de se terminer.
4. Teste ton code avec quelques paramètres d'humidité différents (en utilisant le curseur) pour t'assurer qu'il s'exécutera toujours correctement.

Assure-toi également que les critères suivants sont satisfaits :

1. Assure-toi que ton message pour les astronautes ne dure pas plus de 30 secondes, car c'est la durée d'exécution de ton code dans l'ISS
2. Évite d'utiliser des méthodes nécessitant une saisie de donnée
3. Importe uniquement les modules `sense_hat`, `time`, et `random`
4. Assure-toi de ne pas inclure de juron

Une fois que toutes les règles sont passées au vert, tu es prêt à faire ta soumission.

Saisis le code de ta classe dans la case en bas - ton enseignant ou ton mentor te dira quel est ton code.



Les **notes destinées aux enseignants ou aux mentors** se trouvent dans l'étape **Introduction** (<https://projects.raspberrypi.org/fr-FR/projects/astro-pi-mission-zero/1>).

Le nom de ton enseignant va s'afficher. Si c'est le bon nom, clique sur le bouton vert **Continue to form** (Continuer vers le formulaire).



**Classroom Code**

Once your program is ready, enter your classroom code here to continue to the official submission form.

Donne le nom de ton équipe et les noms des membres de l'équipe. Ils seront imprimés sur le certificat une fois que ton code sera exécuté dans l'espace, alors assure-toi de les épeler correctement !



Appuie sur le bouton **Submit** (Soumettre) pour entrer ton code. Ton enseignant ou ton mentor recevra un e-mail de confirmation pour ton inscription.



Si tu le souhaites, tu peux partager le lien vers ton code sur les réseaux sociaux pour dire aux gens que le code que tu as écrit sera exécuté dans l'espace !



## Étape 9 Et ensuite – d'autres projets Astro Pi

---

Maintenant que tu as terminé ta mission, pourquoi ne pas essayer d'autres projets en utilisant les autres capteurs de l'Astro Pi ?

Si tu te sens confiant, tu pourrais prendre part à **Mission Space Lab** (<https://astro-pi.org/missions/space-lab/>) ! Forme une équipe de deux à six personnes pour travailler ensemble comme de vrais scientifiques de l'espace et concevoir ta propre expérience. Reçois gratuitement du matériel informatique pour ta mission et écris le code Python pour mener à bien ton expérience. Si tu atteins le statut de vol, ton code sera envoyé vers la Station Spatiale Internationale et fonctionnera sur Astro Pi pendant trois heures (deux orbites). Toutes les données que ton code collecte dans l'espace - images ou fichiers des données du capteur - seront téléchargées et te seront renvoyées pour analyse.

Ou tu peux aussi essayer l'un des autres projets Astro Pi :

- En savoir **plus sur le Sense HAT** (<https://projects.raspberrypi.org/fr-FR/projects/getting-started-with-the-sense-hat>) et les autres possibilités qu'il offre
- Créer de beaux **scintillements aléatoires** (<https://projects.raspberrypi.org/fr-FR/projects/sense-hat-random-sparkles>) sur l'écran LED du Sense HAT
- Créer un jeu d'**Astronaute Flappy** (<https://projects.raspberrypi.org/fr-FR/projects/flappy-astronaut>)
- Défier tes amis avec un jeu de **labyrinthe de billes** (<https://projects.raspberrypi.org/fr-FR/projects/sense-hat-marble-maze>)
- Recréer le jeu classique **Pong** (<https://projects.raspberrypi.org/fr-FR/projects/sense-hat-pong>)

---

Ce projet a été traduit par des bénévoles:

Jonathan Vannieuwerkerke  
Michel Arnols

Grâce aux bénévoles, nous pouvons donner aux gens du monde entier la chance d'apprendre dans leur propre langue. Vous pouvez nous aider à atteindre plus de personnes en vous portant volontaire pour la traduction - plus d'informations sur [rpf.io/translate](https://rpf.io/translate) (<https://rpf.io/translate>).

---

Publié par **Raspberry Pi Foundation** (<https://www.raspberrypi.org>) sous un **Creative Commons license** (<https://creativecommons.org/licenses/by-sa/4.0/>).

**Voir le projet et la licence sur GitHub** (<https://github.com/RaspberryPiLearning/astro-pi-mission-zero>)