Python dans l'environnement TI-Nspire™
Guide de programmation

Pour en savoir plus sur la technologie TI, consultez l'aide en ligne à l'adresse education.ti.com/eguide.

Informations importantes

Sauf disposition contraire expressément formulée dans la licence qui accompagne un programme, Texas Instruments n'émet aucune garantie expresse ou implicite, y compris sans s'y limiter, toute garantie implicite de valeur marchande et d'adéquation à un usage particulier, concernant les programmes ou la documentation, ceux-ci étant fournis « tels quels » sans autre recours. En aucun cas, Texas Instruments ne saurait être tenue responsable de dommages spéciaux, collatéraux, fortuits ou indirects en relation avec, ou imputables à l'achat ou à l'utilisation de ce matériel. La seule responsabilité exclusive de Texas Instruments, indépendamment de la forme d'action, ne saurait dépasser le prix fixé dans la licence pour ce programme. Par ailleurs, la responsabilité de Texas Instruments ne saurait être engagée pour quelque réclamation que ce soit en rapport avec l'utilisation desdits matériels par toute autre tierce partie.

« Python » et les logos Python sont des marques commerciales ou des marques déposées de la Python Software Foundation, utilisées par Texas Instruments Incorporated avec la permission de ladite fondation.

Les produits peuvent varier légèrement des images fournies.

Table des matières

Démarrer avec la programmation en Python

En utilisant Python avec les produits TI-Nspire™, vous pouvez :

- ajouter des programmes Python aux fichiers TNS,
- créer des programmes Python en utilisant des modèles,
- communiquer avec d'autres applications TI-Nspire™ afin de partager des données,
- communiquer avec TI-Innovator™ Hub et TI-Innovator™ Rover.

L'implémentation de Python dans l'environnement TI-Nspire™ est basée sur MicroPython, un petit sous-ensemble de la bibliothèque standard Python 3 conçue pour les microcontrôleurs. L'implémentation originale de MicroPython a été adaptée pour être utilisée par TI.

Remarque : Certaines réponses numériques peuvent varier par rapport aux résultats des calculatrices en raison de différences dans les implémentations mathématiques sous-jacentes.

Python est disponible sur les produits TI-Nspire™ suivants :

Unités	Logiciel pour ordinateur
TI-Nspire™ CX II	Logiciel enseignant TI-Nspire™ CX Premium
TI-Nspire™ CX II CAS	Logiciel enseignant TI-Nspire™ CX CAS Premium
TI-Nspire™ CX II-T	Logiciel TI-Nspire™ CX version Élève
TI-Nspire™ CX II-T CAS	TI-Nspire™ CX CAS Student Software
TI-Nspire™ CX II-C	
TI-Nspire™ CX II-C CAS	

Remarque : Dans la plupart des cas, les fonctionnalités sont identiques entre la calculatrice et les vues logicielles, mais des différences peuvent apparaître. Ce guide suppose que vous utilisez la calculatrice ou la vue calculatrice dans le logiciel.

Modules Python

Python dans l'environnement TI-Nspire™ comprend les modules suivants :

Modules standard	Modules TI
Mathématiques (math)	TI PlotLib (ti_plotlib)
Nombre aléatoire (random)	TI Hub (ti_hub)
Nombres complexes (cmath)	TI Rover (ti_rover)
Temps (time)	TI System (ti_system)
	TI Draw (ti_draw)
	TI Image (ti_image)

Remarque : Si vous possédez des programmes Python créés dans d'autres environnements de développement, vous devrez peut-être les modifier afin de les exécuter sur la solution Python dans l'environnement TI-Nspire™. Les modules peuvent utiliser des méthodes, des arguments et un ordonnancement des méthodes différents dans un programme par rapport aux modules TI. Il convient en général d'être attentif à la compatibilité lors de l'utilisation de n'importe quelle version de Python et des modules Python.

Lors du transfert de programmes Python d'une plate-forme non-TI à une plate-forme TI OU d'un produit TI à un autre, n'oubliez pas que :

- les programmes qui utilisent les fonctionnalités du langage de base et les librairies standard (math, random, etc.) peuvent être portés sans modifications ;
- les programmes qui utilisent des bibliothèques spécifiques à une plate-forme, telles que matplotlib pour PC ou modules TI, devront être modifiés avant d'être exécutés sur une autre plate-forme. Cela peut être vrai même entre les plates-formes TI.

Comme pour toute version de Python, vous devrez inclure des importations pour utiliser toutes les fonctions, méthodes ou constantes contenues dans un module donné. Par exemple, pour exécuter la fonction cos() du module mathématique, utilisez les commandes suivantes :

```
>>>from math import *
>>>cos(0)
1.0
```

Pour une liste des menus avec leurs options et leurs descriptions, veuillez consulter la section Guide des menus.

Installation d'un programme Python en tant que module

Pour enregistrer votre programme Python en tant que module :

- Dans l'éditeur, sélectionnez **Actions > Installer en tant que module Python**.
- Dans le Shell, sélectionnez **Outils > Installer en tant que module Python**.

Après la sélection, les événements suivants se produisent :

- La syntaxe Python est vérifiée.
- Le fichier est enregistré et déplacé dans le classeur PyLib.
- Une boîte de dialogue s'affiche pour confirmer que le fichier a été installé en tant que module.
- Le fichier est fermé et le module est prêt à l'emploi.
- Le nom du module sera ajouté au menu **Plus de modules** avec un élément de menu **from <module> import *** .

Si vous prévoyez de partager ce module avec d'autres personnes, il est recommandé de respecter les directives suivantes :

- Ne stockez qu'un seul module par fichier TNS.

- Le nom du module correspond au nom du fichier TNS (par exemple, le module "my_program" est dans le fichier "my_program.tns").
- Ajoutez une page Notes avant l'éditeur Python pour décrire l'objectif du module, la version et les fonctions.
- Utilisez la fonction ver() pour afficher le numéro de version du module.
- (Facultatif) Ajoutez une fonction d'aide pour afficher la liste des méthodes dans la fonction.

Espaces de travail Python

Il existe deux espaces de travail pour votre programmation en Python : l'éditeur Python et la console Python (Shell).

Éditeur Python	Console Python (Shell)
• Créer, modifier et sauvegarder des programmes Python • Mise en évidence syntaxique et auto-indentation • Invite de commandes en ligne pour vous guider avec les arguments de fonction • Info-bulles pour indiquer la plage de valeurs possibles • La touche var énumère les variables et les fonctions utilisateur globales définies dans le programme en cours • Raccourcis clavier	• Exécution des programmes Python • Utile pour tester de petits fragments de codes • Interaction avec l'historique de la console pour sélectionner les entrées et sorties précédentes en vue de leur réutilisation • La touche var énumère les variables utilisateur globales définies dans le dernier programme exécuté pour une activité donnée

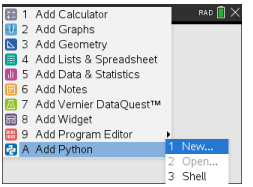
Remarque : Plusieurs programmes et consoles Python peuvent être ajoutés à une activité.

Éditeur Python

L'éditeur Python est l'endroit où vous pouvez créer, modifier et sauvegarder des programmes Python.

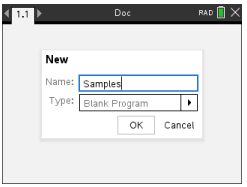
Ajout d'une page de l'éditeur Python

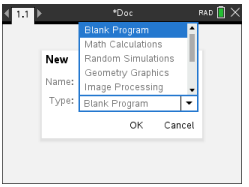
Pour ajouter une nouvelle page de l'éditeur Python dans l'activité en cours, appuyez sur menu et sélectionnez **Ajouter Python (Add Python) > Nouveau (New)**.



Vous pouvez créer un programme vierge, ou vous pouvez sélectionner un modèle.

Programme vierge *Modèle*

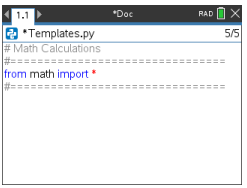




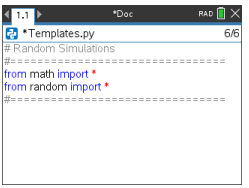
Après la création du programme, l'éditeur Python apparaît. Si vous avez sélectionné un modèle, les instructions d'importation nécessaires sont automatiquement ajoutées (voir ci-dessous).

Remarque : Vous pouvez avoir plusieurs programmes dans un seul fichier TNS, tout comme pour d'autres applications. Si le programme Python est destiné à être utilisé comme module, le fichier TNS peut être enregistré dans le dossier PyLib. Ce module peut ensuite être utilisé dans d'autres programmes et classeurs.

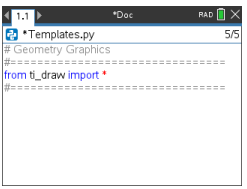
Calculs mathématiques



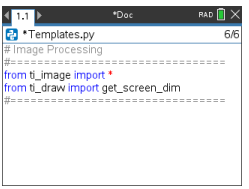
Simulations aléatoires



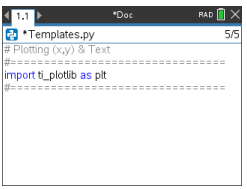
Graphiques géométriques



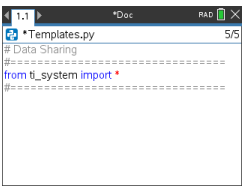
Traitement d'image



Tracé (x,y) &et texte

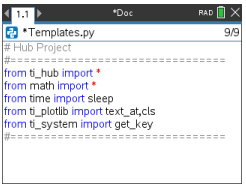


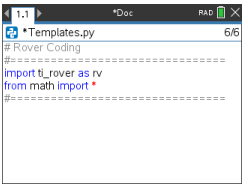
Partage de données



Projet TI-Innovator Hub

Codage TI-Rover





Ouverture d'un programme Python

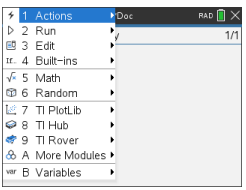
Pour ouvrir un programme Python existant, appuyez sur doc et sélectionnez **Insérer (Insert) > Ajouter Python (Add Python) > Ouvrir (Open)**. Une liste des programmes enregistrés dans le fichier TNS s'affiche.

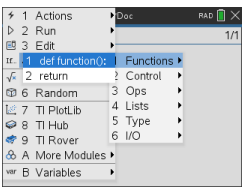
Si la page de l'éditeur utilisée pour créer le programme a été supprimée, celui-ci est toujours disponible dans le fichier TNS.

Utilisation de l'éditeur Python

En appuyant sur menu le menu Outils Classeur (Document Tools) s'affiche. Grâce à ces options de menu, vous pouvez ajouter, déplacer et copier des blocs de code pour votre programme.

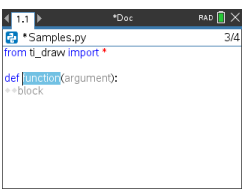
Menu Outils du Classeur

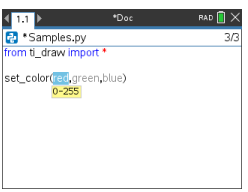




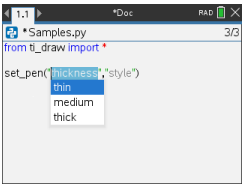
Les éléments sélectionnés dans les menus du module ajouteront automatiquement un modèle de code à l'éditeur avec des invites de commandes en ligne pour chaque partie de la fonction. Vous pouvez naviguer d'un argument à l'autre en appuyant sur tab (avancer) ou shift+tab (reculer). Des info-bulles ou des listes contextuelles s'afficheront lorsqu'elles seront disponibles pour vous aider à sélectionner les bonnes valeurs.

Invites de commandes en ligne *Info-bulles*

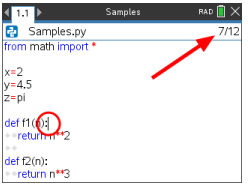




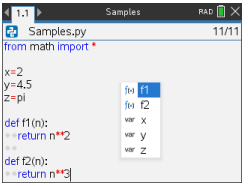
Listes contextuelles



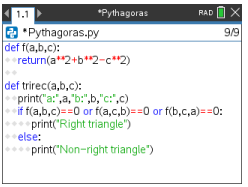
Les numéros à droite du nom du programme indiquent le numéro de ligne actuel du curseur et le nombre total de lignes du programme.



Les fonctions et variables globales définies dans les lignes au-dessus de la position actuelle du curseur peuvent être insérées en appuyant sur var et en les sélectionnant dans la liste.



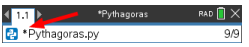
Lorsque vous ajoutez du code à votre programme, l'éditeur affiche des mots-clés, des opérateurs, des commentaires, des chaînes de caractères et des indentations de différentes couleurs pour vous aider à identifier les différents éléments.



Sauvegarde et exécution des programmes

Lorsque vous avez terminé votre programme, appuyez sur menu et sélectionnez **Exécuter (Run) > Vérifier la syntaxe et sauvegarder (Check Syntax & Save)**. Cela permettra de vérifier la syntaxe du programme Python et de le sauvegarder dans le fichier TNS.

Remarque : Si des modifications de votre programme n'ont pas été sauvegardées, un astérisque s'affiche à côté du nom du programme.



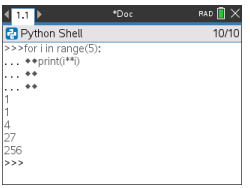
Pour exécuter le programme, appuyez sur menu et sélectionnez **Exécuter (Run) > Exécuter (Run)**. Cela exécutera le programme courant dans la page de console Python suivante ou dans une nouvelle si la suivante n'est pas une page Shell.

Remarque : L'exécution du programme vérifie automatiquement sa syntaxe, puis le sauvegarde.

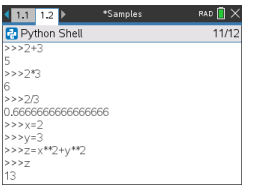
Console Python (Shell)

La console Python (Shell) est l'interpréteur qui exécute vos programmes Python, d'autres fragments de code Python ou de simples commandes.

Code Python

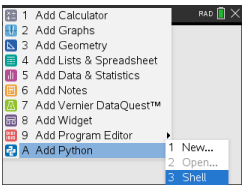


Commandes simples

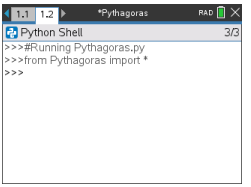


Ajout d'une page de console Python

Pour ajouter une nouvelle page de console Python dans l'activité en cours, appuyez sur menu et sélectionnez **Ajouter Python (Add Python) > Shell**.



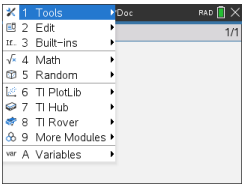
La console Python peut également être lancée à partir de l'éditeur Python en exécutant un programme via la touche menu et en sélectionnant **Exécuter (Run) > Exécuter (Run)**.

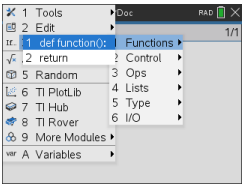


Utilisation de la console Python (Shell)

En appuyant sur menu le menu Outils Classeur (Document Tools) s'affiche. Grâce à ces options de menu, vous pouvez ajouter, déplacer et copier des blocs de code.

Menu Outils du Classeur

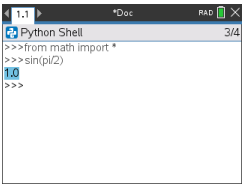




Remarque : Si vous utilisez une méthode de l'un des modules disponibles, veillez à exécuter d'abord une instruction d'importation de module comme dans tout environnement de codage Python.

L'interaction avec les sorties de la console est similaire à l'application Calculs où vous pouvez sélectionner et copier les entrées et sorties précédentes pour les utiliser ailleurs dans la console, l'éditeur ou d'autres applications.

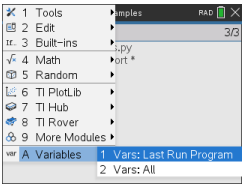
Flèche vers le haut pour sélectionner, puis copier et coller à l'endroit souhaité



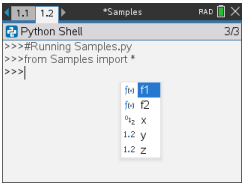
Les fonctions et variables globales du dernier programme exécuté peuvent être insérées en appuyant sur var ou ctrl+L et en sélectionnant dans la liste, ou appuyez sur menu et sélectionnez **Variables > Vars : Dernier programme exécuté**.

Pour choisir parmi une liste de fonctions et de variables globales provenant à la fois du dernier programme exécuté et des modules importés, appuyez sur menu et sélectionnez **Variables > Vars : Tout**.

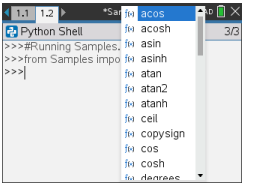
Menu Variables



Les variables du dernier programme exécuté

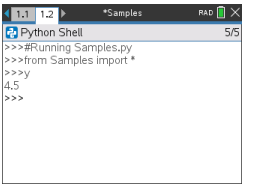


Toutes les variables

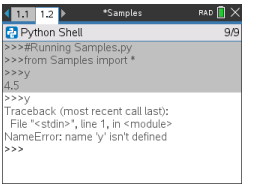


Toutes les pages de console Python de la même activité partagent le même état (les variables définies par l'utilisateur et importées). Lorsque vous enregistrez ou exécutez un programme Python dans cette activité ou lorsque vous appuyez sur menu et sélectionnez **Outils (Tools) > Réinitialiser la console (Reinitialize Shell)**, l'historique de la console aura alors un fond gris indiquant que l'état précédent n'est plus valide.

Avant la sauvegarde ou la réinitialisation



Après la sauvegarde ou la réinitialisation



Remarque : L'option menu **Outils (Tools) > Effacer l'historique (Clear History)** efface l'écran de toute activité antérieure dans la console, mais les variables sont toujours disponibles.

Messages

Des messages d'erreur et d'autres informations peuvent s'afficher lorsque vous vous trouvez dans une session Python. Si une erreur s'affiche dans la console lorsqu'un programme s'exécute, un numéro de ligne de programme s'affiche. Appuyez sur ctrl menu et sélectionnez **Accéder à l'éditeur Python (Go to Python Editor)**. Dans l'éditeur, appuyez sur menu puis sélectionnez **Modifier (Edit) > Aller à la ligne (Go to Line)**. Saisissez le numéro de la ligne et appuyez sur enter. Le curseur s'affichera sur le premier caractère de la ligne où l'erreur s'est produite.

Interruption de l'exécution d'un programme

Lors de l'exécution d'une fonction ou d'un programme, le pointeur en forme d'horloge ⏲ s'affiche.

- Pour arrêter la fonction ou le programme,
  - Windows® : Appuyez sur la touche **F12**.
  - Sur Mac® : Appuyez sur la touche **F5**.
  - Unité : Appuyez sur la touche on.

Guide des menus Python

Cette section énumère tous les menus et les options de menu de l'éditeur et de la console Python. Elle fournit une brève description de chacun élément.

Remarque : Pour les options de menu dotées de raccourcis clavier, les utilisateurs de Mac® doivent remplacer **Ctrl** par ⌘ (**Cmd**) partout où il est utilisé. Pour obtenir une liste complète des raccourcis pour le logiciel et les unités TI-Nspire™, consultez le Guide numérique de la technologie TI-Nspire™.

Menu Actions

Remarque : Cela s'applique uniquement à l'éditeur.

Élément	descriptions
Nouveau (New)	Ouvre la boîte de dialogue **Nouveau (New)** : elle permet de saisir le nom du nouveau programme et d'en sélectionner le type.
Ouvrir	Ouvre une liste des programmes disponibles dans le classeur courant.
Créer une copie (Create Copy)	Ouvre la boîte de dialogue **Créer une copie (Create Copy)** : elle permet d'enregistrer le programme courant sous un autre nom.
Renommer	Ouvre la boîte de dialogue **Renommer (Rename)** : elle permet de renommer le programme courant.
Fermer	Ferme le programme courant.
Paramètres	Ouvre la boîte de dialogue **Réglages (Settings)** : elle permet de modifier la taille de la police pour l'éditeur et la console (Shell).
Installer en tant que module Python	Vérifie la syntaxe Python du fichier TNS actuel et le déplace vers le dossier PyLib.

Menu Exécuter (Run)

Remarque : Cela s'applique uniquement à l'éditeur.

Élément	Raccourcis clavier	descriptions
Exécuter	Ctrl+R	Vérifie la syntaxe du programme, l'enregistre puis l'exécute dans une console Python.
Vérifier la syntaxe et enregistrer (Check Syntax & Save)	Ctrl+B	Vérifie la syntaxe et enregistre le programme.
Accéder au Shell (Go to Shell)	N/D	Active la console (Shell) liée au programme en cours ou ouvre une nouvelle page du Shell à côté de l'éditeur.

Menu Outils

Remarque : Cela s'applique uniquement à la console (Shell).

Élément	Raccourcis clavier	descriptions
Réexécuter le dernier programme	Ctrl+R	Réexécute le dernier programme associé à la console en cours.
Accéder à l'éditeur Python	N/D	Ouvre la page de l'éditeur relative à la console en cours.
Exécuter	N/D	Ouvre une liste des programmes disponibles dans le classeur courant. Après la sélection, le programme choisi est exécuté.
Effacer historique	N/D	Efface l'historique de la console en cours, mais ne réinitialise pas la console.
Réinitialiser la console	N/D	Réinitialise l'état de toutes les pages ouvertes de la console pour l'activité en cours. Toutes les variables définies et les fonctions importées ne sont plus disponibles.
dir()	N/D	Affiche la liste des fonctions dans le module spécifié lorsqu'il est utilisé après l'instruction d'importation.
From PROGRAM import *	N/D	Ouvre une liste des programmes disponibles dans le classeur courant. Après la sélection, l'instruction d'importation est collée dans la console.
Installer en tant que module Python	N/D	Activé uniquement pour les modules au format binaire. Déplace le fichier TNS actuel vers le dossier PyLib.

Menu Édition (Edit)

Remarque : Ctrl+A sélectionne toutes les lignes de code ou les données de sortie afin de les couper ou de les supprimer (éditeur uniquement), ou de les copier-coller (éditeur et console).

Élément	Raccourcis clavier	descriptions
Indentation	TAB*	Indente le texte sur la ligne en cours ou sur les lignes sélectionnées. * Si des invites de commandes en ligne sont incomplètes, TAB passera à l'invite suivante.
Désindentation	Maj+TAB**	Désindente le texte sur la ligne en cours ou sur les lignes sélectionnées. * Si des invites de commandes en ligne sont incomplètes, Maj+TAB passera à l'invite précédente.
Commenter/Décommenter	Ctrl+T	Ajoute/supprime le symbole commentaire au/du début de la ligne en cours.
Insérer une chaîne de caractères de plusieurs lignes	N/D	(éditeur uniquement) Insère un modèle de chaîne de caractères de plusieurs lignes.
Rechercher	Ctrl+F	(éditeur uniquement) Ouvre la boîte de dialogue **Rechercher (Find)** et recherche la chaîne saisie dans le programme courant.
Remplacer	Ctrl+H	(éditeur uniquement) Ouvre la boîte de dialogue **Remplacer (Replace)** et recherche la chaîne saisie dans le programme courant.
Aller à la ligne	Ctrl+G	(éditeur uniquement) Ouvre la boîte de dialogue **Aller à la ligne (Go to Line)** et saute à la ligne spécifiée dans le programme courant.
Début de ligne	Ctrl+8	Déplace le curseur au début de la ligne courante.
Fin de ligne	Ctrl+2	Déplace le curseur à la fin de la ligne courante.
Aller au début (Jump to Top)	Ctrl+7	Déplace le curseur au début de la première ligne du programme.
Aller à la fin (Jump to	Ctrl+1	Déplace le curseur à la fin de la dernière

Élément	Raccourcis clavier	descriptions
Bottom)		ligne du programme.

Menu Commandes natives (Built-ins)

Fonctions

Élément	descriptions
def function():	Définit une fonction qui dépend de variables spécifiées.
return	Définit la valeur générée par une fonction.

Contrôle (Control)

Élément	descriptions
if..	Instruction conditionnelle.
if..else..	Instruction conditionnelle.
if..elif..else..	Instruction conditionnelle.
for index in range(size):	Itère sur une plage.
for index in range(start,stop):	Itère sur une plage.
for index in range(start,stop,step):	Itère sur une plage.
for index in list:	Itère sur les éléments d'une liste.
while..	Exécute les instructions dans un bloc de code jusqu'à ce qu'une condition soit évaluée à Faux.
elif:	Instruction conditionnelle.
else:	Instruction conditionnelle.

Ops

Élément	descriptions
x=y	Définit la valeur de la variable.
x==y	Colle l'opérateur de comparaison « égal à » (==).
x!=y	Colle l'opérateur de comparaison « différent de » (!=).
x>y	Colle l'opérateur de comparaison « strictement supérieur à » (>).
x>=y	Colle l'opérateur de comparaison « supérieur ou égal à » (>=).
x<y	Colle l'opérateur de comparaison « strictement inférieur à » (<).
x<=y	Colle l'opérateur de comparaison « inférieur ou égal à » (<=).

Élément	descriptions
et	Colle l'opérateur logique « et » (and).
—ou—	Colle l'opérateur logique « ou » (or).
non	Colle l'opérateur logique « non » (not).
Vrai	Colle la valeur booléenne Vraie (True).
Faux	Colle la valeur booléenne Faux (False).

liste

Élément	descriptions
[]	Colle les crochets ([]).
list()	Convertit une séquence en type « liste ».
len()	Renvoie le nombre d'éléments de la liste.
max()	Renvoie la valeur maximale de la liste.
min()	Renvoie la valeur minimale de la liste.
.append()	La méthode rajoute un élément à une liste.
.remove()	La méthode supprime la première instance d'un élément d'une liste.
range(start,stop,step)	Renvoie un ensemble de nombres.
for index in range(start,stop,step)	Utilisé pour itérer sur une plage.
.insert()	La méthode ajoute un élément à la position spécifiée.
.split()	La méthode renvoie une liste dont les éléments sont séparés par le délimiteur spécifié.
sum()	Renvoie la somme des éléments d'une liste.
sorted()	Renvoie une liste triée.
.sort()	La méthode trie une liste en place.

Type

Élément	descriptions
int()	Renvoie une partie entière.
float()	Renvoie un nombre à virgule flottante.

Élément	descriptions
round(x,ndigits)	Renvoie une valeur à virgule flottante arrondie au nombre de chiffres spécifié.
str()	Renvoie une chaîne de caractères.
complex()	Donne un nombre complexe.
type()	Renvoie le type de l'objet.

E/S

Élément	descriptions
print()	Affiche l'argument sous forme d'une chaîne de caractères.
input()	Invite l'utilisateur à saisir des données.
eval()	Évalue une expression représentée sous forme d'une chaîne de caractères.
.format()	La méthode formate la chaîne de caractères spécifiée.

Menu Mathématiques (Math)

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Calculs mathématiques (Math Calculations)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
from math import *	Importe toutes les méthodes (fonctions) du module math.
fabs()	Renvoie la valeur absolue d'un nombre réel.
sqrt()	Renvoie la racine carrée d'un nombre réel.
exp()	Renvoie e**x.
pow(x,y)	Renvoie x élevé à la puissance y.
log(x,base)	Retourne le $\log_{base}$ (x). log(x) sans base donne le logarithme népérien de x.
fmod(x,y)	Renvoie la valeur de x modulo y. À utiliser lorsque x et y sont des nombres en virgule flottante.
ceil()	Renvoie le plus petit entier supérieur ou égal à un nombre réel.
floor()	Renvoie le plus grand nombre entier inférieur ou égal à un nombre réel.
trunc()	Troncature d'un nombre réel, donne un nombre entier.
frexp()	Retourne la mantisse et l'exposant d'un nombre x, comme paire (y,n) telle que x == y * 2**n.

Const

Élément	descriptions
C :	Returns value for the constant e.
pi	Returns value for the constant pi.

Zoom trigo

Élément	descriptions
radians()	Convertit un angle de degrés en radians.
degrees()	Convertit un angle de radians en degrés.

Élément	descriptions
sin()	Renvoie le sinus de l’argument exprimé en radians.
cos()	Renvoie le cosinus de l’argument exprimé en radians.
tan()	Renvoie la tangente de l’argument exprimé en radians.
asin()	Renvoie l’Arc sinus de l'argument exprimé en radians.
acos()	Renvoie l’Arc cosinus de l’argument exprimé en radians.
atan()	Renvoie l’Arc tangente de l’argument exprimé en radians.
atan2(y,x)	Renvoie l’Arc tangente de y/x exprimé en radians.

Menu Nombre aléatoire (Random)

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Simulations aléatoires (Random Simulations)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
from random import *	Importe toutes les méthodes du module nombres aléatoires.
random()	Renvoie un nombre à virgule flottante compris entre 0 et 1,0.
uniform(min,max)	Renvoie un nombre aléatoire x (à virgule flottante) tel que min <= x <= max.
randint(min,max)	Renvoie un nombre aléatoire entier compris entre min et max.
choice(sequence)	Renvoie un élément aléatoire d'une séquence non vide.
randrange(start,stop,step)	Renvoie un nombre aléatoire d'une plage comprise entre « start » et « stop » avec un pas « step ».
seed()	Initialise le générateur de nombres aléatoires.

Menu TI PlotLib

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Tracé (x,y) et texte (Plotting (x,y) & Text)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
import ti_plotlib as plt	Importe toutes les méthodes (fonctions) du module ti_plotlib dans l'espace de nom « plt ». Par conséquent, tous les noms de fonctions collés à partir des menus seront précédés de « plt. ».

Config

Élément	descriptions
cls()	Efface la zone de représentation graphique.
grid(x-scale,y-scale,"style")	Affiche une grille basée sur les échelles spécifiées pour les axes des x (x-scale) et des y (y-scale).
window(xmin,xmax,ymin,ymax)	Définit la fenêtre de représentation graphique en faisant correspondre l'intervalle horizontal (xmin, xmax) et l'intervalle vertical (ymin, ymax) spécifiés à la zone graphique allouée (pixels).
auto_window(x-list,y-list)	Met automatiquement à l'échelle la fenêtre graphique pour qu'elle corresponde à la plage de données comprises dans les listes x (x-list) et y (y-list) spécifiées dans le programme, préalablement à la fonction auto_window().
axes("mode")	Affiche les axes sur la fenêtre spécifiée dans la zone graphique.
labels("x-label","y-label",x,y)	Affiche les noms des axes x (x-label) et y (y-label) sur la représentation graphique aux positions définies respectivement par les arguments x et y.
title("title")	Affiche le titre (title) centré sur la ligne supérieure de la fenêtre.
show_plot()	Affiche le tracé en mémoire tampon. Les fonctions use_buffer() et show_plot() sont utiles lorsque l'affichage de plusieurs objets à l'écran risque de provoquer des problèmes des ralentissements (dans certains cas).
use_buffer()	Active une mémoire tampon hors écran pour accélérer le tracé.

Ponctuation

Élément	descriptions
color(red,green,blue)	Définit la couleur pour tous les graphiques/graphes suivants.
cls()	Efface la zone de représentation graphique.
show_plot()	Exécute l'affichage de la représentation graphique tel qu'elle est configurée dans le programme.
scatter(x-list,y-list,"mark")	Trace une séquence de paires ordonnées à partir des listes de coordonnées x (x-list) et y (y-list) avec le style de marquage (mark) spécifié.
plot(x-list,y-list,"mark")	Trace une ligne en utilisant des paires ordonnées des listes de coordonnées x (x-list) et y (y-list) spécifiées.
plot(x,y,"mark")	Trace un point en utilisant les coordonnées x et y avec le style de marquage (mark) spécifié.
line(x1,y1,x2,y2,"mode")	Trace un segment de droite allant de (x1,y1) à (x2,y2).
lin_reg(x-list,y-list,"display")	Calcule et dessine le modèle de régression linéaire, ax+b, des listes x (x-list) et y (y-list).
pen("size","style")	Définit l'apparence de toutes les courbes suivantes jusqu'à ce que la fonction pen() suivante soit exécutée.
text_at(row,"text","align")	Affiche le texte (text) dans la zone graphique avec l'alignement (align) spécifié.

Propriétés

Élément	descriptions
xmin	Variable spécifiée pour les arguments de fenêtre définis comme plt.xmin.
xmax	Variable spécifiée pour les arguments de fenêtre définis comme plt.xmax.
ymin	Variable spécifiée pour les arguments de fenêtre définis comme plt.ymin.
ymax	Variable spécifiée pour les arguments de fenêtre définis comme plt.ymax.
m	Après l'exécution de plt.linreg() dans un programme, les valeurs calculées de la pente m et de l'ordonnée à l'origine b, sont stockées dans plt.m et plt.b.
b	Après l'exécution de plt.linreg() dans un programme, les valeurs calculées de la pente a et de l'ordonnée à l'origine b, sont stockées dans plt.a et plt.b.

Menu TI Hub

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Projet hub (Hub Project)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
from ti_hub import *	Importe toutes les méthodes du module ti_hub.

Dispositifs intégrés du hub > Sortie couleur

Élément	descriptions
rgb(red,green,blue)	Définit la couleur de la DEL RGB.
blink(frequency,time)	Définit la fréquence (frequency) et la durée (time) de clignotement pour la couleur sélectionnée.
off()	Éteint la DEL RGB.

Dispositifs intégrés au Hub (Hub Built-in Devices) > Sortie lumière (Light Output)

Élément	descriptions
on()	Allume la DEL.
off()	Éteint la DEL.
blink(frequency,time)	Définit la fréquence (frequency) et la durée (time) de clignotement de la DEL.

Dispositifs intégrés au Hub (Hub Built-in Devices) > Sortie audio (Sound Output)

Élément	descriptions
tone(frequency,time)	Joue une tonalité à la fréquence spécifiée pendant le temps spécifié.
note("note",time)	Joue la note spécifiée pendant la durée (time) spécifiée. La note est spécifiée au moyen de son nom (en anglais) et d'une octave. Par exemple : A4, C5. Les noms des notes (en anglais) sont C, CS, D, DS, E, F, FS, G, GS, A, AS et B.

Élément	descriptions
	Les octaves sont comprises entre 1 et 9 (inclus).
tone(frequency,time,tempo)	Joue une tonalité à la fréquence spécifiée pendant le temps et au tempo spécifiés. Le tempo définit le nombre de bips par seconde entre 0 et 10 (inclus).
note("note",time,tempo)	Joue la note spécifiée pendant le temps et au tempo spécifiés. La note est spécifiée au moyen de son nom (en anglais) et d'une octave. Par exemple : A4, C5. Les noms des notes (en anglais) sont C, CS, D, DS, E, F, FS, G, GS, A, AS et B. Les octaves sont comprises entre 1 et 9 (inclus). Les valeurs de tempo sont comprises entre 0 et 10 (inclus).

Dispositifs intégrés au Hub (Hub Built-in Devices) > Entrée luminosité (Brightness Input)

Élément	descriptions
measurement()	Lit le capteur de LUMINOSITÉ (niveau de lumière) intégré et retourne la valeur relevée. La plage par défaut est de 0 à 100. Cela peut être modifié via la fonction range().
range(min,max)	Définit la plage pour la configuration des relevés du capteur de niveau de lumière. Si les deux valeurs sont manquantes, ou si elles sont définies sur une valeur évaluée à None, alors la plage de luminosité par défaut (de 0 à 100) est utilisée.

Ajouter un dispositif d'entrée (Add Input Device)

Ce menu contient une liste des capteurs (dispositifs d'entrée) pris en charge par le module ti_hub. Toutes les options du menu collent le nom de l'objet et attendent une variable et un port utilisés avec le capteur. Chaque capteur possède une méthode measurement() qui retourne la valeur mesurée.

Élément	descriptions
DHT (Capteur de température et d'humidité) (DHT, Digital Humidity & Temp)	Ajoute une liste comportant la

Élément	descriptions
	température, l'humidité, le type de capteur actuels et le dernier état mis en mémoire.
Ranger (Capteur de distance)	Renvoie la mesure de la distance courante issue du capteur de distance à ultrasons spécifié. • **measurement_ time()** : renvoie le temps pris par le signal ultrasonique pour atteindre l'objet (le « temps de vol »).
Niveau de lumière	Renvoie le niveau de luminosité issu du capteur de lumière externe (luminosité).
température	Renvoie la lecture de la température issue du capteur de température externe. La configuration par défaut prend en charge le capteur de température Seeed sur les ports IN 1, IN 2 ou IN 3. Pour utiliser le capteur de température TI LM19 du pack de platines d'essais TI-Innovator™ Hub, modifiez le port sur la broche BB en service et utilisez un argument optionnel « TIANALOG ». Exemple : mylm19=temperatur

Élément	descriptions
	e("BB 5","TIANALOG")
Humidité	Renvoie la lecture du capteur d'humidité.
Champ magnétique	Détecte la présence d'un champ magnétique. La valeur seuil pour déterminer la présence du champ est fixée par la fonction trigger(). La valeur par défaut du seuil est de 150.
App Vernier	Lit la valeur du capteur analogique Vernier spécifié dans la commande. La commande prend en charge les capteurs Vernier suivants : • **température** : capteur de température en acier inoxydable. • **niveau de lumière** : capteur de niveau de lumière TI. • **pression** : capteur de pression de gaz d'origine. • **pression** : capteur de pression de gaz plus récent. • **pH** : capteur de pH. • **force10** : réglage ±10 N, capteur de force double

Élément	descriptions
	échelle. • **force50** : réglage ±50 N, capteur de force double échelle. • **accéléromètre** : accéléromètre Low-G. • **générique** : permet le réglage d'autres capteurs non pris en charge directement comme ci-dessus et l'utilisation de l'API calibrate() ci-dessus pour définir des coefficients d'équation.
Entrée analogique	Prend en charge l'utilisation de dispositifs génériques d'entrée analogique.
Entrée numérique	Renvoie l'état actuel de la broche numérique connectée à l'objet DIGITAL ou l'état de mise en cache de la valeur de sortie numérique DÉFINIE en dernier lieu sur l'objet.
Potentiomètre	Prend en charge un potentiomètre. La plage du capteur peut être modifiée par la fonction range ().
Thermistor	Lit le thermistor. Les coefficients par

Élément	descriptions
	défaut sont adaptés au thermistor inclus dans le pack de platine d'essais TI-Innovator™ Hub, lorsqu'il est utilisé avec un résistor fixe de 10 kΩ. Un nouveau jeu de coefficients d'étalonnage et une résistance de référence pour le thermistor peuvent être configurés à l'aide de la fonction calibrate().
Intensité sonore	Prend en charge les capteurs de niveau sonore.
Entrée couleur	Fournit des interfaces à un capteur colorimétrique d'entrée connecté au port I2C. La broche bb_port est utilisée en plus du port I2C pour contrôler la DEL du capteur de couleur. • **color_number():** Retourne une valeur comprise entre 1 et 9 qui représente la couleur détectée par le capteur. Les chiffres représentent les couleurs selon les correspondances suivantes : 1: Rouge 2: Vert

Élément	descriptions
	3: Bleu 4: Cyan 5: Magenta 6: Jaune 7: Noir 8 : Blanc 9: Gris • **red():** Retourne une valeur de 0 à 255 qui représente l'intensité du niveau de couleur ROUGE détecté. • **green():** Retourne une valeur de 0 à 255 qui représente l'intensité du niveau de couleur VERTE détecté. • **blue():** Retourne une valeur de 0 à 255 qui représente l'intensité du niveau de couleur BLEUE détecté. • **gray():** Retourne une valeur de 0 à 255 qui représente le niveau de gris détecté, où 0 correspond au noir et 255 au blanc.
Port BB	Permet d'utiliser les 10 broches du port BB comme un port d'entrée/sortie

Élément	descriptions
	numérique combiné. Les fonctions d'initialisation disposent d'un paramètre « masque (mask) » optionnel qui permet d'utiliser le sous-ensemble de 10 broches. • **read_port():** Lit les valeurs courantes sur les broches d'entrée du port BB. • **write_port (value):** Règle la broche de sortie à la valeur spécifiée (la valeur est comprise entre 0 et 1023). Notez que la valeur est également ajustée par rapport à la valeur du masque dans l'opération var=bbport(mask), si un masque a été fourni.
Hub Time	Permet d'accéder au timer interne (en millisecondes).
TI-RGB Array	Fournit des fonctions pour la programmation de TI-RGB Array. La fonction d'initialisation accepte un paramètre optionnel « LAMP » pour activer le mode

Élément	descriptions
	haute luminosité de TI-RGB Array. Il requiert une alimentation électrique externe. • **set(led_position, r,g,b):** Règle la valeur r,g,b spécifiée pour la DEL située à la position (led_ position) spécifiée (0-15). r, g et b sont des valeurs comprises entre 0 et 255. • **set(led_ list,red,green,blue):** Définit les DEL de la « led_ list » sur la couleur spécifiée « rouge », « vert » ou « bleu ». La « led_list » est une liste Python qui inclut les index des DEL de 0 à 15. Par exemple, set ([0,2,4,6,15], 0, 0, 255) définira les DEL 0, 2, 4, 6 et 15 sur le bleu. • **set_all(r,g,b):** Règle la même valeur r,g,b pour toutes les DEL RGB de la matrice. • **all_off():** Éteint toutes les DEL RGB de la matrice.

Élément	descriptions
	• **measurement():** Retourne la valeur approximative du courant consommé (en milliampères) sur TI-Innovator™ par la matrice RGB. • **pattern(pattern):** L'argument « pattern » est une valeur binaire de 16 bits (de 0 à 65 535), où la valeur « 1 » active le pixel correspondant. Les DEL sont allumées en ROUGE avec une valeur de niveau MLI de 255. • **pattern (value,red,green, blue):** Définit les DEL caractérisées par le « modèle » sur la couleur spécifiée « rouge », « vert » ou « bleu ».

Ajouter un dispositif de sortie

Ce menu contient une liste des dispositifs de sortie pris en charge par le module ti_ hub. Toutes les options du menu collent le nom de l'objet et attendent une variable et un port utilisés avec le dispositif.

Élément	descriptions
VOYANT	Fonctions de contrôle des DEL connectées externes.
RVB	Prise en charge du contrôle des DEL RGB externes.

Élément	descriptions
TI-RGB Array	Fournit des fonctions pour la programmation de TI-RGB Array.
Haut-parleur	Fonctions de prise en charge d'un haut-parleur externe avec le TI-Innovator™ Hub. Les fonctions sont les mêmes que celles relatives au « son » décrites plus haut.
Puissance	Fonctions de contrôle des alimentations externes avec le TI-Innovator™ Hub. • **set(value):** Règle le niveau de puissance à la valeur (value) spécifiée, entre 0 et 100. • **on():** Règle le niveau de puissance à 100. • **off():** Règle le niveau de puissance à 0.
Servomoteur continu	Fonctions de contrôle des servomoteurs continus. • **set_cw(speed,time):** Le servomoteur tournera dans le sens des aiguilles d'une montre à la vitesse (speed) spécifiée (0-255) et pendant la durée (time) spécifiée (en secondes). • **set_ccw(speed,time):** Le servomoteur tournera dans le sens inverse des aiguilles d'une montre à la vitesse (speed) spécifiée (0-255) et pendant la durée (time) spécifiée (en secondes). • **stop():** Arrête le servomoteur continu.
Sortie analogique	Fonctions pour l'utilisation de dispositifs génériques d'entrée analogique.
Moteur vibrant	Fonctions de contrôle des moteurs vibrants. • **set(val):** Règle l'intensité du moteur vibrant au niveau « val » (0-255). • **off():** Éteint le moteur vibrant. • **on():** Met en marche le moteur vibrant au niveau le plus élevé.
Relais	Interfaces de contrôle des relais. • **on():** Règle le relais sur l'état ON. • **off():** Règle le relais sur l'état OFF.
Servo	Fonctions de contrôle des servomoteurs. • **set_position(pos):** Règle la position angulaire (pos) du servomoteur entre -90° et +90°. • **zero():** Règle le servomoteur angulaire sur la position zéro.
Squarewave	Fonctions de génération d'un signal carré. • **set(frequency,duty,time):** Règle le signal carré de sortie avec un rapport cyclique (duty) par défaut de 50 % (s'il

Élément	descriptions
	n'est pas spécifié) et une fréquence de sortie spécifiée par « frequency ». La fréquence peut être comprise entre 1 et 500 Hz. Le rapport cyclique, s'il est spécifié, peut être compris entre 0 et 100 %. • **off():** Désactive le signal carré.
Sortie numérique	Interfaces de contrôle d'une sortie numérique. • **set(val):** Règle la sortie numérique à la valeur spécifiée par « val » (0 ou 1). • **on():** Règle la sortie numérique à un niveau haut (1). • **off():** Règle la sortie numérique à un niveau bas (0).
Port BB	Fournit des fonctions pour la programmation de TI-RGB Array. Voir les détails ci-dessus.

Commandes

Élément	descriptions
sleep(seconds)	Interrompt le programme pendant le nombre spécifié de secondes. Importé depuis le module « temps ».
text_at(row,"text","align")	Affiche, dans la zone de graphe, le texte (text) spécifié avec l'alignement (align) spécifié. Fait partie du module ti_plotlib.
cls()	Efface l'écran de la console (Shell) pour une représentation graphique. Fait partie du module ti_plotlib.
while get_key() != "esc":	Exécute les commandes dans la boucle « while » jusqu'à ce que la touche d'échappement « esc » soit enfoncée.
get_key()	Retourne une chaîne représentant la touche enfoncée. La touche « 1 » retourne « 1 », la touche d'échappement retourne « esc », etc. Lorsque la fonction est appelée sans paramètres (get_key ()), elle retourne immédiatement la chaîne de caractères vide (''). Lorsque la fonction est appelée avec un paramètre (get_ key(1)), elle attend qu'une touche soit enfoncée. Fait partie du module ti_system.

Ports

Ce sont les ports d'entrée et de sortie disponibles sur le TI-Innovator™ Hub.

Élément
OUT 1
OUT 2
OUT 3
IN 1
IN 2
IN 3
BB 1
BB 2
BB 3
BB 4
BB 5
BB 6
BB 7
BB 8
BB 9
BB 10
Port I2C

Menu TI Rover

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Codage du Rover (Rover Coding)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
import ti_rover as rv	Importe toutes les méthodes (fonctions) du module ti_rover dans l'espace de nom « rv ». Par conséquent, tous les noms de fonctions collés à partir des menus seront précédés de « rv. ».

Déplacement

Élément	descriptions
forward(distance)	Fait avancer le Rover de la distance spécifiée en unités de grille.
backward(distance)	Fait reculer le Rover de la distance spécifiée en unités de grille.
left(angle_degrees)	Fait tourner le Rover à gauche de l'angle spécifié en degrés.
right(angle_degrees)	Fait tourner le Rover à droite de l'angle spécifié en degrés.
stop()	Arrête immédiatement tout mouvement en cours.
stop_clear()	Arrête immédiatement tout mouvement en cours et efface toutes les commandes en attente.
resume()	Reprend le traitement des commandes.
stay(time)	Le Rover est immobilisé pendant la durée spécifiée en secondes (facultatif). Si aucune durée n'est spécifiée, le Rover est immobilisé pendant 30 secondes.
to_xy(x,y)	Déplace le Rover à la position de coordonnées (x,y) sur la grille virtuelle.
to_polar(r,theta_degrees)	Déplace le Rover à la position en coordonnées polaires (r, thêta) sur la grille virtuelle. L'angle est spécifié en degrés.
to_angle(angle,"unit")	Fait tourner le Rover de l'angle spécifié dans la grille virtuelle. L'angle est défini par rapport à un angle nul qui pointe vers le bas de l'axe des x dans la grille virtuelle.

Déplacement (Drive) > Options de déplacement (Drive with Options)

Élément	descriptions
forward_time(time)	Fait avancer le Rover pendant la durée (time) spécifiée.
backward_time(time)	Fait reculer le Rover pendant la durée (time) spécifiée.
forward(distance,"unit")	Fait avancer le Rover à la vitesse par défaut sur la distance spécifiée. La distance peut être spécifiée en unités de grille, en mètres ou en tours de roue (unit).
backward(distance,"unit")	Fait reculer le Rover à la vitesse par défaut sur la distance spécifiée. La distance peut être spécifiée en unités de grille, en mètres ou en tours de roue (unit).
left(angle,"unit")	Fait tourner le Rover à gauche de l'angle spécifié. L'angle peut être en degrés, en radians ou en grades.
right(angle,"unit")	Fait tourner le Rover à droite de l'angle spécifié. L'angle peut être en degrés, en radians ou en grades.
forward_time(time,speed,"rate")	Fait avancer le Rover pendant la durée (time) spécifiée et à la vitesse (speed) spécifiée. La vitesse peut être spécifiée en unités de grille/s, en mètres/s ou en tours de roue/s.
backward_time(time,speed,"rate")	Fait reculer le Rover pendant la durée (time) spécifiée et à la vitesse (speed) spécifiée. La vitesse peut être spécifiée en unités de grille/s, en mètres/s ou en tours de roue/s.
forward(distance,"unit",speed,"rate")	Fait avancer le Rover sur la distance spécifiée et à la vitesse (speed) spécifiée. La distance peut être spécifiée en unités de grille, en mètres ou en tours de roue (unit). La vitesse peut être spécifiée en unités de grille/s, en mètres/s ou en tours de roue/s.
backward(distance,"unit",speed,"rate")	Fait reculer le Rover sur la distance spécifiée et à la vitesse (speed) spécifiée. La distance peut être spécifiée en unités de

Élément	descriptions
	grille, en mètres ou en tours de roue (unit). La vitesse peut être spécifiée en unités de grille/s, en mètres/s ou en tours de roue/s.

Entrées

Élément	descriptions
ranger_measurement()	Lit le capteur de distance à ultrasons situé à l'avant du Rover et renvoie la distance courante en mètres.
color_measurement()	Retourne une valeur comprise entre 1 et 9, indiquant la couleur prédominante « vue » par le capteur colorimétrique d'entrée du Rover. 1 = rouge 2 = vert 3 = bleu 4 = cyan 5 = magenta 6 = jaune 7 = noir 8 = gris 9 = blanc
red_measurement()	Renvoie une valeur comprise entre 0 et 255 qui indique le niveau de rouge perçu par le capteur colorimétrique d'entrée.
green_measurement()	Renvoie une valeur comprise entre 0 et 255 qui indique le niveau de vert perçu par le capteur colorimétrique d'entrée.
blue_measurement()	Renvoie une valeur comprise entre 0 et 255 qui indique le niveau de bleu perçu par le capteur colorimétrique d'entrée.
gray_measurement()	Renvoie une valeur comprise entre 0 et 255 qui indique le niveau de gris perçu par le capteur colorimétrique d'entrée.
encoders_gyro_measurement()	Renvoie une liste de valeurs qui contient le comptage des encodeurs des roues gauche et droite ainsi que le cap gyroscopique courant.
gyro_measurement()	Renvoie une valeur qui représente la lecture courante du gyroscope, y compris la dérive, en degrés.

Élément	descriptions
ranger_time()	Renvoie le temps pris par le signal ultrasonique du capteur de distance TI-Rover pour atteindre l'objet (le « temps de vol »).

Sorties

Élément	descriptions
color_rgb(r,g,b)	Règle la couleur de la DEL RGB du Rover sur les valeurs spécifiées de rouge, vert et bleu (r,g,b).
color_blink(frequency,time)	Définit la fréquence (frequency) et la durée (time) de clignotement pour la couleur sélectionnée.
color_off()	Éteint la DEL RGB du Rover.
motor_left(speed,time)	Règle la puissance du moteur gauche à la valeur (speed) spécifiée pour la durée (time) spécifiée. La vitesse est comprise entre -255 et 255, 0 correspondant à l'arrêt. Les valeurs de vitesse positives correspondent à une rotation dans le sens inverse des aiguilles d'une montre et les valeurs de vitesse négatives à une rotation dans le sens des aiguilles d'une montre. Le paramètre optionnel de durée (time), s'il est spécifié, a une plage valide de 0,05 à 655,35 secondes. S'il n'est pas spécifié, une valeur par défaut de 5 secondes est utilisée.
motor_right(speed,time)	Règle la puissance du moteur gauche à la valeur (speed) spécifiée pour la durée (time) spécifiée. La vitesse est comprise entre -255 et 255, 0 correspondant à l'arrêt. Les valeurs de vitesse positives correspondent à une rotation dans le sens inverse des aiguilles d'une montre et les valeurs de vitesse négatives à une rotation dans le sens des aiguilles d'une montre. Le paramètre optionnel de durée (time),

Élément	descriptions
	s'il est spécifié, a une plage valide de 0,05 à 655,35 secondes. S'il n'est pas spécifié, une valeur par défaut de 5 secondes est utilisée.
motors("ldir",left_val,"rdir",right_val,time)	Règle la roue gauche et la roue droite sur les niveaux de vitesse spécifiés, pour une durée (time) en secondes optionnelle. Les valeurs de vitesse (left_val, right_val) sont comprises entre 0 et 255, 0 correspondant à l'arrêt. Les paramètres ldir et rdir spécifient la rotation en sens horaire ou en sens antihoraire des roues respectives. Le paramètre optionnel de durée (time), s'il est spécifié, a une plage valide de 0,05 à 655,35 secondes. S'il n'est pas spécifié, une valeur par défaut de 5 secondes est utilisée.

Chemin

Élément	descriptions
waypoint_xythdrn()	Lit l'abscisse x, l'ordonnée y, la durée, la direction, la distance parcourue, le nombre de tours de roue, le numéro de commande au point de cheminement courant. Renvoie une liste avec toutes ces valeurs en tant qu'éléments.
waypoint_prev	Lit l'abscisse x, l'ordonnée y, la durée, la direction, la distance parcourue, le nombre de tours de roue, le numéro de commande au point de cheminement précédent.
waypoint_eta	Renvoie le temps estimé pour se rendre à un point de cheminement.
path_done()	Renvoie une valeur 0 ou 1 selon que le Rover est en mouvement (0) ou qu'il est complètement immobilisé (1).
pathlist_x()	Renvoie une liste des valeurs de x depuis le début jusqu'à la valeur de x au point de cheminement courant comprise.
pathlist_y()	Renvoie une liste des valeurs de y depuis le début jusqu'à la valeur de y au point de cheminement courant comprise.
pathlist_time()	Renvoie une liste des temps en secondes depuis le début jusqu'au temps au point de cheminement courant compris.

Élément	descriptions
pathlist_heading()	Renvoie une liste des directions depuis le début jusqu'à la valeur de la direction au point de cheminement courant comprise.
pathlist_distance()	Renvoie une liste des distances parcourues depuis le début jusqu'à la valeur de la distance au point de cheminement courant comprise.
pathlist_revs()	Renvoie une liste du nombre de tours de roue effectués depuis le début jusqu'à la valeur du nombre de tours de roue au point de cheminement courant compris.
pathlist_cmdnum()	Renvoie une liste de numéros de commande pour le chemin.
waypoint_x()	Renvoie l'abscisse x du point de cheminement actuel.
waypoint_y()	Renvoie l'ordonnée y du point de cheminement actuel.
waypoint_time()	Renvoie le temps de parcours entre le point de cheminement précédent et le point de cheminement courant.
waypoint_heading()	Renvoie la direction absolue du point de cheminement courant.
waypoint_distance()	Renvoie la distance parcourue entre le point de cheminement précédent et le point de cheminement courant.
waypoint_revs()	Renvoie le nombre de tours de roue nécessaires pour parcourir la distance entre le point de cheminement précédent et le point de cheminement courant.

Paramètres

Élément	descriptions
unités/s	Option : vitesse en unités de grille par seconde.
m/s	Option : vitesse en mètres par seconde.
tr/s	Option : vitesse en tours de roue par seconde.
unité	Option : distance en unités de grille.
m	Option : distance en mètres.
tr	Option : distance en tours de roue.
Degré	Option : virage en degrés.
radians	Option : virage en radians.
grades	Option : virage en grades.
sens horaire	Option : spécification du sens de rotation de la roue.
sens antihoraire	Option : spécification du sens de rotation de la roue.

Commandes

Ces commandes sont un ensemble de fonctions provenant d'autres modules ainsi que du module TI Rover.

Élément	descriptions
sleep(seconds)	Interrompt le programme pendant le nombre spécifié de secondes. Importées du module time.
text_at(row,"text","align")	Affiche le texte (text) dans la zone graphique avec l'alignement (align) spécifié. Importées du module ti_plotlib.
cls()	Efface l'écran de la console (Shell) pour une représentation graphique. Importées du module ti_plotlib.
while get_key() != "esc":	Exécute les commandes dans la boucle « while » jusqu'à ce que la touche d'échappement « esc » soit enfoncée.
wait_until_done()	Met le programme en pause jusqu'à ce que le Rover termine la commande en cours. Cela permet de synchroniser les commandes non Rover avec les mouvements du Rover.
while not path_done()	Exécute les commandes dans la boucle « while » jusqu'à ce que le Rover ait terminé tous ses mouvements. La fonction path_done() renvoie 0 ou 1 selon que le Rover est en mouvement (0) ou qu'il est complètement immobilisé (1).
position(x,y)	Définit la position du Rover sur la grille virtuelle aux coordonnées x,y spécifiées.
position(x,y,heading,"unit")	Définit la position du Rover sur la grille virtuelle aux coordonnées x,y spécifiées. Si une direction (heading) est fourni (dans l'unité d'angle [unit] spécifiée), le cap virtuel est fixé par rapport à l'axe x virtuel. Les angles positifs de 0 à 360 sont considérés comme étant dans le sens antihoraire par rapport à l'axe x positif. Les angles négatifs de 0 à -360 sont considérés comme étant dans le sens horaire par rapport à l'axe x positif.
grid_origin()	Définit le RV comme étant à l'origine de la grille actuelle, point (0,0).
grid_m_unit(scale_value)	Définit le quadrillage virtuel en mètres par unité (m/unité) à la valeur spécifiée. 0,1 constitue l'échelle

Élément	descriptions
	(en m/unité) par défaut : 1 unité = 100 mm ou 10 cm ou 1 dm ou 0,1 m. La plage des valeurs d'échelle (scale_value) valides est comprise entre 0,01 et 10,0.
path_clear()	Efface toutes les informations de chemin ou points de cheminement préexistants.
zero_gyro()	Réinitialise le gyroscope du Rover à l'angle nul et efface les compteurs des encodeurs des roues gauche et droite.

Menu nombres complexes (Complex Math)

Ce sous-menu est situé sous **Plus de modules (More Modules)**.

Élément	descriptions
from cmath import *	Importe toutes les méthodes du module cmath.
complex(real,imag)	Donne un nombre complexe.
rect(modulus,argument)	Convertit les coordonnées polaires d'un nombre complexe en coordonnées cartésiennes.
.real	Renvoie la partie réelle du nombre complexe.
.imag	Renvoie la partie imaginaire du nombre complexe.
polar()	Convertit les coordonnées cartésiennes d'un nombre complexe en coordonnées polaires.
phase()	Renvoie l'argument d'un nombre complexe.
exp()	Renvoie e**x.
cos()	Renvoie le cosinus d'un nombre complexe.
sin()	Renvoie le sinus d'un nombre complexe.
log()	Renvoie le logarithme népérien d'un nombre complexe.
log10()	Renvoie le logarithme en base 10 d'un nombre complexe.
sqrt()	Renvoie la racine carrée d'un nombre complexe.

Menu Time

Ce sous-menu est situé sous **Plus de modules (More Modules)**.

Élément	descriptions
from time import *	Importe toutes les méthodes du module time.
sleep(seconds)	Interrompt le programme pendant le nombre spécifié de secondes.
clock()	Retourne le temps courant du processeur sous la forme d'un nombre flottant exprimé en secondes.
localtime()	Convertit un temps exprimé en secondes depuis le 1er janvier 2000 en un tuple de neuf éléments contenant : l'année, le mois, le jour du mois, l'heure, la minute, la seconde, le jour de la semaine, le jour de l'année et un indicateur de l'heure d'été (Daylight Savings Time, DST). Si l'argument optionnel (secondes) n'est pas fourni, alors l'horloge temps réel est utilisée.
ticks_cpu()	Retourne un compteur croissant (en millisecondes) spécifique au processeur (cpu) avec un point de référence arbitraire. Pour mesurer le temps de manière cohérente à travers différents systèmes, utilisez ticks_ms().
ticks_diff()	Mesure la période entre deux appels consécutifs à la fonction ticks_cpu() ou ticks_ms(). Cette fonction ne doit pas être utilisée pour mesurer des périodes de temps longues.

Menu TI System

Ce sous-menu est situé sous **Plus de modules (More Modules)**.

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Partage de données (Data Sharing)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
from ti_system import *	Importe toutes les méthodes (fonctions) du module ti_system.
recall_value("name")	Rappelle la valeur (value) d'une variable prédéfinie de l'OS nommée "name".
store_value("name",value)	Stocke la valeur (value) d'une variable Python dans une variable de l'OS nommée "name".
recall_list("name")	Rappelle une liste prédéfinie de l'OS nommée "name".
store_list("name",list)	Stocke une liste Python (list) dans une variable de type liste de l'OS nommée "name".
eval_function("name",value)	Évalue une fonction prédéfinie de l'OS à la valeur spécifiée.
get_platform()	Retourne « hh » pour la calculatrice et « dt » pour l'ordinateur de bureau.
get_key()	Retourne une chaîne représentant la touche enfoncée. La touche « 1 » retourne « 1 », la touche d'échappement retourne « esc », etc. Lorsque la fonction est appelée sans paramètres (get_key()), elle retourne immédiatement la chaîne de caractères vide ('''). Lorsque la fonction est appelée avec un paramètre (get_key(1)), elle attend qu'une touche soit enfoncée.
get_mouse()	Retourne les coordonnées de la souris sous la forme d'un tuple à deux éléments, soit la position du pixel de la zone de tracé, soit (-1,-1) si la position est à l'extérieur de la zone de tracé.
while get_key() != "esc":	Exécutez les commandes dans la boucle « while » jusqu'à ce que la touche d'échappement « esc » soit enfoncée.
clear_history()	Efface l'historique de la console (Shell).
get_time_ms()	Retourne le temps en millisecondes avec une précision de l'ordre de la milliseconde. Cette fonctionnalité peut être utilisée pour calculer une durée plutôt que de déterminer l'heure effective

Élément	descriptions
	de l’horloge.

Menu TI Draw

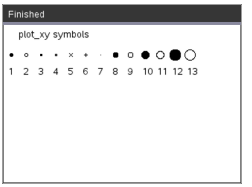
Ce sous-menu est situé sous **Plus de modules (More Modules)**.

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Graphiques Géométrie (Geometry Graphics)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
from ti_draw import *	Importe toutes les méthodes du module ti_draw.

Figure

Élément	descriptions
draw_line()	Trace un segment de droite partant du point de coordonnées spécifiées x1,y1 jusqu'à point de coordonnées x2,y2.
draw_rect()	Trace un rectangle à partir du point de coordonnées x,y spécifiées de largeur et de hauteur spécifiées.
fill_rect()	Trace un rectangle à partir du point de coordonnées x,y spécifiées de largeur et de hauteur spécifiées et avec la couleur de remplissage spécifiée (via le paramètre set_color ou noir si la couleur n'est pas définie).
draw_circle()	Trace un cercle de centre le point de coordonnées x,y spécifiées et de rayon spécifié.
fill_circle()	Trace un cercle de centre le point de coordonnées x,y spécifiées et de rayon spécifié et avec la couleur de remplissage spécifiée (via le paramètre set_color ou noir si la couleur n'est pas définie).
draw_text()	Trace une chaîne de texte à partir du point de coordonnées x,y spécifiées.
draw_arc()	Trace un arc à partir du point de coordonnées x,y spécifiées de largeur, hauteur et d'angle spécifiés.
fill_arc()	Trace un arc à partir du point de coordonnées x,y spécifiées de largeur, hauteur et d'angle spécifiés et avec la couleur de remplissage spécifiée (via le paramètre set_color ou noir si la couleur n'est pas définie).
draw_poly()	Trace un polygone en utilisant les listes d'abscisses x et d'ordonnées y spécifiées.
fill_poly()	Trace un polygone en utilisant les listes d'abscisses x et d'ordonnées y spécifiées et avec la couleur de remplissage spécifiée (via le paramètre set_color ou noir si la couleur n'est pas définie)..
plot_xy()	Trace une forme en utilisant les coordonnées x,y spécifiées et un

Élément	descriptions
	nombre spécifié compris entre 1 et 13 qui représente différentes formes et symboles (voir ci-dessous). 

Contrôle (Control)

Élément	descriptions
clear()	Efface la totalité de l'écran. Peut être utilisé avec les paramètres (x,y,largeur,hauteur) pour effacer une zone rectangulaire donnée.
clear_rect()	Efface le rectangle situé aux coordonnées x,y spécifiées de largeur et hauteur spécifiées.
set_color()	Définit la couleur de la ou des formes qui suivent dans le programme jusqu'à ce qu'une autre couleur soit définie.
set_pen()	Définit l'épaisseur et le style spécifiés de la bordure pour le tracé des formes (non applicable en cas d'utilisation de commandes de remplissage).
set_window()	Définit la taille de la fenêtre dans laquelle toutes les formes seront tracées. Cette fonction est utile pour redimensionner la fenêtre afin qu'elle corresponde aux données ou pour modifier l'origine (0,0) de la zone de dessin.
get_screen_dim()	Retourne les dimensions xmax et ymax de l'écran.
use_buffer()	Active une mémoire tampon hors écran pour accélérer le tracé.
paint_buffer()	Affiche le tracé en mémoire tampon. Les fonctions use_buffer() et paint_buffer() sont utiles lorsque l'affichage de plusieurs objets à l'écran risque de provoquer des problèmes de ralentissements.

Éditeur mathématique

- La configuration par défaut attribue les coordonnées (0,0) au coin supérieur gauche de l'écran. L'axe des x positifs pointe vers la droite et l'axe des y positifs pointe vers le bas. Cela peut être modifié via la fonction set_window().

- Les fonctions du module ti_draw sont uniquement disponibles sur la calculatrice ou sur la vue calculatrice de l'ordinateur de bureau.

Menu TI Image

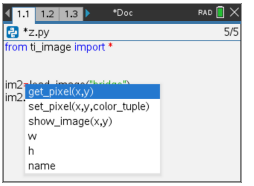
Ce sous-menu est situé sous **Plus de modules (More Modules)**.

Remarque : Lors de la création d'un nouveau programme qui utilise ce module, il est recommandé d'utiliser le type de programme **Traitement d'image (Image Processing)**. Ceci garantit que tous les modules nécessaires sont importés.

Élément	descriptions
from ti_image import *	Importe toutes les méthodes du module ti_image.
new_image(width,height,(r,g,b))	Crée une nouvelle image avec la largeur (width) et la hauteur (height) spécifiées pour l'utiliser dans le programme Python. La couleur de la nouvelle image est définie par les valeurs (r,g,b).
load_image("name")	Charge l'image spécifiée par son nom (name) pour une utilisation dans le programme Python. L'image doit faire partie du classeur TNS, soit dans une application Éditeur mathématique ou Graphiques. L'invite « nom » (name) affiche les noms des images (si elles ont été nommées auparavant) ou un numéro indiquant leur ordre d'insertion.
copy_image(image)	Crée une copie de l'image spécifiée par la variable « image ».

Méthodes de l'objet image

Des fonctions supplémentaires concernant les objets images sont disponibles dans l'éditeur et la console en tapant le nom de la variable suivi d'un . (point).



- **get_pixel(x,y):** Donne la valeur (r,g,b) du pixel à l'emplacement défini par la paire de coordonnées (x,y).

```
px_val = get_pixel(100,100)
print(px_val)
```

- **set_pixel(x,y,color_tuple):** Définit la couleur du pixel spécifiée par un tuple (color_ tuple) à l'emplacement (x,y).

```
set_pixel(100,100,(0,0,255))
```

Règle le pixel de coordonnées (100,100) à la couleur (0,0,255).

- **show_image(x,y):** Affiche l'image avec le coin supérieur gauche à l'emplacement (x,y).
- **w, h, name:** Permet d'obtenir les paramètres largeur (w), hauteur (h) et nom (name) de l'image.

Par exemple

```
from ti_image import *

# An image has been previously inserted into the TNS document in a
Notes application and named "bridge"
im1=load_image("bridge")
px_val = im1.get_pixel(100,100)
print(px_val)

# Set the pixel at 100,100 to blue (0,0,255)
im1.set_pixel(100,100,(0,0,255))
new_px = im1.get_pixel(100,100)
print(new_px)

# Print the width, height and name of the image
print(im1.w, im1.h, im1.name)
```

Menu Variables

Remarque : Ces listes ne comprennent pas les variables définies dans d'autres applications TI-Nspire™.

Élément	descriptions
Vars : Programme en cours	(éditeur uniquement) Affiche une liste de fonctions et de variables globales définies dans le programme en cours
Vars: Dernier programme exécuté	(console uniquement) Affiche une liste de fonctions et de variables globales définies dans le programme en cours
Vars: Tout	(console uniquement) Affiche une liste de fonctions et de variables globales du dernier programme exécuté et de tous les modules importés

Annexe

Mots-clés Python

Les mots-clés suivants sont intégrés à l'implémentation de Python dans TI-Nspire™.

False	elif	lambda
None	else	nonlocal
True	except	not
and	finally	or
as	for	pass
assert	from	raise
break	global	return
class	if	try
continue	import	while
def	in	with
del	is	yield

Mappage des touches dans Python

Lors de la saisie du code dans l'éditeur ou dans la console, le clavier est conçu pour coller les opérations Python appropriées ou ouvrir des menus afin de faciliter la saisie de fonctions, de mots-clés, de méthodes, d'opérateurs, etc.

Touche	Mappage
var	Ouvre le menu Variables
sto→	Colle le signe =
del ←	Supprime le caractère à gauche du curseur
clear	Aucune action
=	Colle le signe =
|≠≥▸	Colle le ou les symboles sélectionnés : • > • < • != • >= • <= • == • et • —ou— • non • | • & • ~
trig	Colle la fonction sélectionnée : • sin • cos • tan • atan2 • asin • acos • atan
ⓘ	Affiche des conseils
:=	Colle :=
^	Colle **
$\sqrt[n]{x}$	Aucune action
x^2	Colle **2

Touche	Mappage
√	Colle sqrt()
×	Colle le signe de multiplication (*)
"□"	Colle un guillemet anglais (")
÷	Colle le signe de division (/)
□/□	Aucune action
e^x	Colle exp()
ln	Colle log()
10^x	Colle 10**
log	Colle log(value,base)
(	Colle (
)	Colle)
[]	Colle [ ]
{}	Colle { }
−	Colle le signe moins (-)
≈	Ajoute une nouvelle ligne après la ligne courante
EE	Colle E
?!•	Colle le ou les symboles sélectionnés : • ? • ! • $ • ° • ' • % • " • : • ; • _ • \ • #
π•	Colle « pi »
⚐	État courant de l'indicateur (flag)
↵	Ajoute une nouvelle ligne après la ligne courante

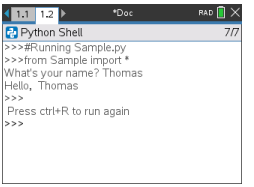
Exemples de programmes Python

Utilisez les exemples de programmes suivants pour vous familiariser avec les méthodes Python. Ils sont également disponibles dans le fichier **Getting Started Python.tns** situé dans le dossier **Exemples (Examples)**.

Remarque : Si vous copiez et collez un exemple de code contenant des indicateurs d'indentation par tabulation (••) dans le logiciel TI-Nspire™, vous devrez remplacer ces instances par de véritables indentations par tabulation.

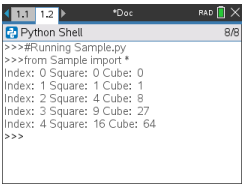
Bonjour

```
# This program asks for your name and uses
# it in an output message.
# Run the program here by typing "Ctrl R"

name=input("What's your name? ")
print("Hello, ", name)
print("\n Press ctrl+R to run again")
```



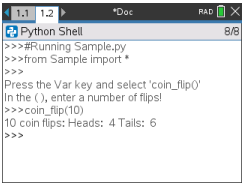
Exemple de boucle

```
# This program uses a "for" loop to calculate
# the squares and cubes of the first 5 numbers
# 0,1,2,3,4
# Note: Python starts counting at 0

for index in range(5):
••square = index**2
••cube = index**3
••print("Index: ", index, "Square: ", square,
••••"Cube: ", cube)
```



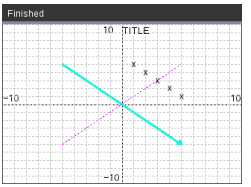
Pile ou face

```
# Use random numbers to simulate a coin flip
# We will count the number of heads and tails
# Run the program here by typing "Ctrl R"

# Import all the functions of the "random" module
from random import *

# n is the number of times the die is rolled
def coin_flip(n):
    heads = tails = 0
    for i in range(n):
# Generate a random integer - 0 or 1
# "0" means head, "1" means tails
        side=randint(0,1)
        if (side == 0):
            heads = heads + 1
        else:
            tails = tails + 1
# Print the total number of heads and tails
    print(n, "coin flips: Heads: ", heads, "Tails: ", tails)

print("\nPress the Var key and select 'coin_flip()'")
print("In the ( ), enter a number of flips!")
```



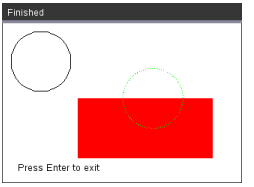
Graphe

```
# Plotting example
import ti_plotlib as plt

# Set up the graph window
plt.window(-10,10,-10,10)
plt.axes("on")
plt.grid(1,1,"dashed")
# Add leading spaces to position the title
plt.title("                TITLE")

# Set the pen style and the graph color
plt.pen("medium","solid")
plt.color(28,242,221)
plt.line(-5,5,5,-5,"arrow")

plt.pen("thin","dashed")
plt.color(224,54,243)
plt.line(-5,-5,5,5,"")

# Scatter plot from 2 lists
plt.color(0,0,0)
xlist=[1,2,3,4,5]
ylist=[5,4,3,2,1]
plt.scatter(xlist,ylist, "x")
```



Tracé

```
from ti_draw import *

# (0,0) is in top left corner of screen
# Let's draw some circles and squares
# Circle with center at (50,50) and radius 40
draw_circle(50,50,40)

# Set color to red (255,0,0) and fill a rectangle of
# of width 180, height 80 with top left corner at
# (100,100)
set_color(255,0,0)
fill_rect(100,100,180,80)

# Set color to green and pen style to "thin"
# and "dotted".
# Then, draw a circle with center at (200,100)
# and radius 40
set_color(0,255,0)
set_pen("thin","dotted")
draw_circle(200,100,40)

set_color(0,0,0)
draw_text(20,200,"Press Enter to exit")
```



Image

```
# Image Processing
#================================
from ti_image import *
from ti_draw import *
#================================

# Load and show the 'manhole_cover' image
# It's in a Notes app
# Draw a circle on top
im1=load_image("manhole_cover")
im1.show_image(0,0)
set_color(0,255,0)
set_pen("thick","dashed")
draw_circle(140,110,100)
```

Hub

Ce programme utilise Python pour contrôler un TI-Innovator™ Hub, un microcontrôleur programmable. Si vous exécutez le programme sans connecter un TI-Innovator™ Hub, un message d'erreur apparaîtra.

Pour plus d'informations sur le TI-Innovator™ Hub, rendez-vous sur education.ti.com.

```
#========== Import Section ==========
from ti_hub import *
from math import *
from random import *
from time import sleep
from ti_plotlib import text_at,cls
from ti_system import get_key
#======== End of Import Section =======

print("Connect the TI-Innovator Hub and hit 'enter'")
input()
print("Blinking the RGB LED for 4 seconds")
# Set the RGB LED on the Hub to purple
color.rgb(255,0,255)

# Blink the LED 2 times a second for 4 seconds
color.blink(2,4)

sleep(5)

print("The brightness sensor reading is: ", brightness.measurement())

# Generate 10 random colors for the RGB LED
# Play a tone on the Hub based on the random
# color
print("Generate 10 random colors on the Hub & play a tone")
for i in range(10):
  r=randint(0,255)
  b=randint(0,255)
  g=randint(0,255)
  color.rgb(r,g,b)
  sound.tone((r+g+b)/3,1)
  sleep(1)

color.off()
```

Informations générales

Aide en ligne

education.ti.com/eguide

Sélectionnez votre pays pour obtenir d'autres informations relatives aux produits.

Contacter l'assistance technique TI

education.ti.com/ti-cares

Sélectionnez votre pays pour obtenir une assistance technique ou d'autres types de support.

Informations Garantie et Assistance

education.ti.com/warranty

Sélectionnez votre pays pour en savoir plus sur la durée et les termes de la garantie et sur l'assistance pour le produit.

Garantie limitée. Cette garantie n'affecte pas vos droits statutaires.

Texas Instruments Incorporated

12500 TI Blvd.

Dallas, TX 75243