Python TI-Nspire™
Manual de Programação

Saiba mais sobre a tecnologia TI através da ajuda online em education.ti.com/eguide.

Informações importantes

Salvo indicação em contrário constante da Licença que acompanha o programa, a Texas Instruments renuncia a todas as garantias mencionadas, quer sejam expressas ou implícitas, incluindo mas não se limitando a qualquer garantia implícita de comercialização ou adequação a um fim específico, no que respeita aos materiais licenciados são disponibilizados numa base "como estão". A TI não se responsabiliza, em circunstância alguma, por qualquer dano indireto, especial ou acidental, relacionado ou decorrente da utilização destes materiais, e a única e exclusiva responsabilidade da Texas Instruments, independentemente da forma de Acão, não excederá o preço indicado na licença do programa. Além disso, a Texas Instruments não se responsabiliza por qualquer reclamação relacionada com a utilização destes materiais por terceiros.

"Python" e os logótipos Python são marcas comerciais ou marcas comerciais registadas da Python Software Foundation, utilizadas pela Texas Instruments Incorporated com permissão da Fundação.

Os produtos reais podem variar ligeiramente das imagens fornecidas.

Índice

Começar com a Programação em Python

Ao utilizar o Python com os produtos TI-Nspire™, pode:

- adicionar programas Python aos ficheiros TNS
- criar programas Python utilizando modelos
- interagir e partilhar dados com outras aplicações TI-Nspire™
- interagir com o TI-Innovator™ Hub e TI-Innovator™ Rover

A implementação Python do TI-Nspire™ baseia-se no MicroPython, que é um pequeno subconjunto da biblioteca standard Python 3 concebida para funcionar em microcontroladores. A implementação original da MicroPython foi adaptada para utilização pela TI.

Nota: Algumas respostas numéricas podem variar dos resultados da Calculadora devido a diferenças nas implementações matemáticas subjacentes.

O Python está disponível nestes produtos TI-Nspire™:

Unidades portáteis	Software do computador
TI-Nspire™ CX II	Software para Professores TI-Nspire™ CX Premium
TI-Nspire™ CX II CAS	Software para Professores TI-Nspire™ CX CAS Premium
TI-Nspire™ CX II-T	TI-Nspire™ CX Student Software
TI-Nspire™ CX II-T CAS	Software TI-Nspire™ CX CAS Student
TI-Nspire™ CX II-C	
TI-Nspire™ CX II-C CAS	

Nota: Na maioria dos casos, a funcionalidade é idêntica entre as vistas da unidade portátil e as vistas do software, mas pode verificar algumas diferenças. Este manual pressupõe que está a utilizar a unidade portátil ou a vista da unidade portátil no software.

Módulos Python

O Python TI-Nspire™ inclui os seguintes módulos:

Módulos padrão	Módulos TI
Matemática (math)	TI PlotLib (ti_plotlib)
Aleatório (random)	TI Hub (ti_hub)
Matemática complexa (cmath)	TI Rover (ti_rover)
Tempo (time)	Sistema TI (ti_system)
	Desenho TI (ti_draw)
	Imagem TI (ti_imagem)

Nota: Se tiver programas Python criados noutros ambientes de desenvolvimento Python, pode ser necessário editá-los para executar no Python do TI-Nspire™. Os módulos podem utilizar diferentes métodos, argumentos e ordenação de métodos num programa em comparação com os módulos TI. No geral, tenha em mente a compatibilidade quando utilizar qualquer versão do Python e dos módulos Python.

Quando transferir programas Python de uma plataforma não TI para uma plataforma TI OU de um produto TI para outro, lembre-se:

- Os programas que utilizam características de idioma central e bibliotecas padrão (matemática, aleatória, etc.) podem ser apresentados sem alterações.
- Os programas que utilizam bibliotecas específicas da plataforma como matplotlib para os módulos PC ou TI irão necessitar de edições antes de serem executados numa plataforma diferente. Isto pode ser verdade mesmo entre plataformas TI.

Tal como com qualquer versão do Python, terá de incluir importações para utilizar quaisquer funções, métodos ou constantes contidas num determinado módulo. Por exemplo, para executar a função cos() do módulo matemático, utilize os seguintes comandos:

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

Para uma lista de menus com os seus itens e descrições, consulte a secção Mapa do menu.

Instalar um programa Python como um módulo

Para guardar o seu programa Python como um módulo:

- No Editor, selecione **Ações e > Instalar como módulo Python**.
- Na Shell, selecione **Ferramentas e >Instalar como módulo Python**.

Após a seleção, ocorre o seguinte:

- A sintaxe Python é verificada.
- O ficheiro é guardado e movido para a pasta PyLib.
- Uma janela aparece a confirmar que o ficheiro foi instalado como módulo.
- O ficheiro é fechado e o módulo está pronto a ser usado.
- O nome do módulo será adicionado ao menu **Mais Módulos** através de um **item de menu <module> importado ***.

Se planeia partilhar este módulo com outras pessoas, recomenda-se que siga estas diretrizes:

- Armazene apenas um módulo por ficheiro TNS.
- O nome do módulo corresponde ao nome do ficheiro TNS (por exemplo, o módulo "meu_programa" está no ficheiro "meu_programa.tns").

- Adicione uma página de Notas antes do Editor Python que descreva a intenção do módulo, a versão e as funções.
- Utilize a função ver() para apresentar o número da versão do módulo.
- (Opcional) Adicione uma função de ajuda para apresentar a lista de métodos na função.

Área de trabalho Python

Existem duas áreas de trabalho para a sua programação Python: O Editor Python e a Python Shell (Interpretador) ou interpretador.

Python Editor	Python Shell (Interpretador)
• Criar, editar e guardar programas Python • Realce de sintaxe e auto-indentação • Instruções em linha para orientar com argumentos de função • Dicas de ferramentas para mostrar o intervalo de valores válidos • A tecla var lista variáveis de utilizadores globais e funções definidas no programa atual • Atalhos do teclado	• Executar programas Python • Conveniente para testar pequenos fragmentos de código • Interação com o histórico da Shell (Interpretador) para selecionar entradas e saídas anteriores para reutilização • A tecla var lista variáveis de utilizadores globais definidas no último programa executado no respetivo problema

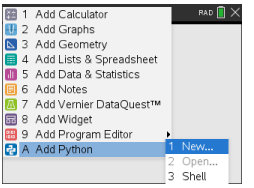
Nota: Vários programas de Python e Shell (Interpretador) podem ser adicionados a um problema.

Python Editor

O Editor Python é onde pode criar, editar e guardar programas Python.

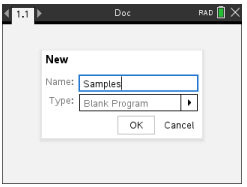
Adicionar uma página do Editor Python

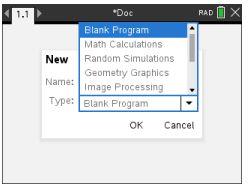
Para adicionar uma nova página do Editor Python no problema atual, prima menu e selecione **Adicionar Python > Novo**.



Pode criar um programa em branco ou pode selecionar um modelo.

Programa em branco *Modelo*

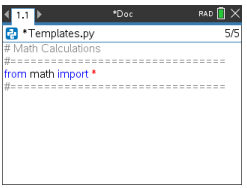




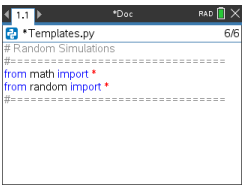
Depois de criar o programa, é apresentado o Python Editor. Se selecionou um modelo, os comandos de importação necessários são adicionados automaticamente (ver abaixo).

Nota: Pode ter vários programas num único ficheiro TNS tal como noutras aplicações. Se o programa Python se destinar a ser utilizado como módulo, o ficheiro TNS pode ser guardado na pasta PyLib. Esse módulo pode então ser utilizado noutros programas e documentos.

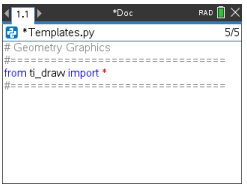
Cálculos matemáticos



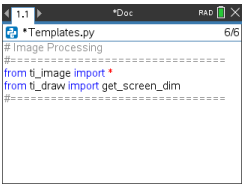
Simulações aleatórias



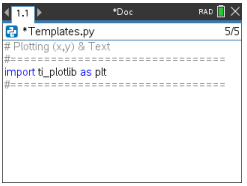
Gráficos da Geometria



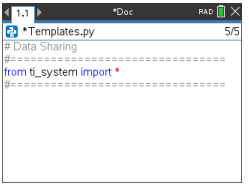
Processamento de imagem



Representar graficamente (x,y) e Texto

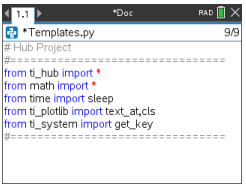


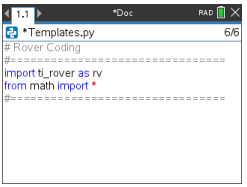
Partilha de dados



Projeto TI-Innovator Hub

Codificação da TI-Rover





Abrir um programa Python

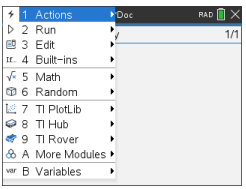
Para abrir um programa Python existente, prima doc e selecione **Inserir > Adicionar Python > Abrir**. Isto irá apresentar uma lista de programas que foram guardados no ficheiro TNS.

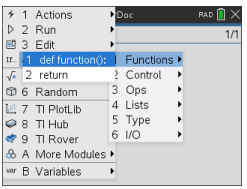
Se a página Editor utilizada para criar o programa tiver sido eliminada, o programa ainda está disponível no ficheiro TNS.

Trabalhar no Editor Python

Pressionar menu exibe o menu Ferramentas do documento. Com estas opções de menu pode adicionar, mover e copiar blocos de código para o seu programa.

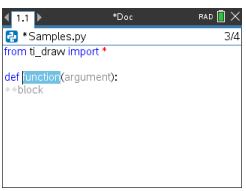
Menu de Ferramentas de Documento





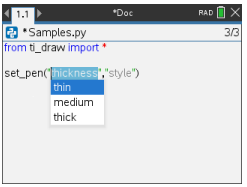
Os itens selecionados nos menus do módulo adicionarão automaticamente um modelo de código ao Editor com pedidos em linha para cada parte da função. Pode navegar de um argumento para o seguinte premindo tab (avançar) ou shift+tab (retroceder). As descrições de ferramentas ou listas de contexto aparecerão quando disponíveis para o ajudar a selecionar os valores adequados.

Pedidos em linha

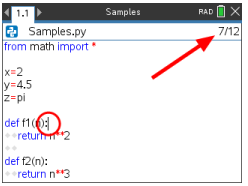


Dicas

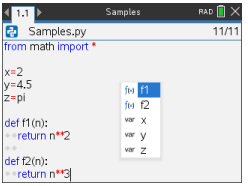
Listas pop-up



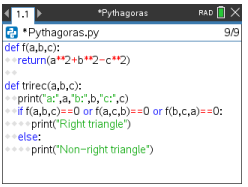
Os números à direita do nome do programa refletem o número da linha atual do cursor e o número total de linhas no programa.



As funções globais e as variáveis definidas nas linhas acima da posição atual do cursor podem ser introduzidas premindo var e selecionando a partir da lista.



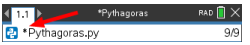
À medida que adiciona código ao seu programa, o Editor apresenta palavras-chave, operadores, comentários, cadeias e indentações em diferentes cores para ajudar a identificar os diferentes elementos.



Guardar e executar programas

Quando terminar o seu programa, prima menu e selecione **Executar > Verificar sintaxe e Guardar**. Isto irá verificar a sintaxe do programa Python e guardá-lo no ficheiro TNS.

Nota: Se tiver alterações não guardadas no seu programa, aparecerá um asterisco junto ao nome do programa.



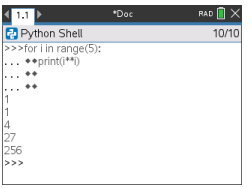
Para executar o programa, prima menu e selecione **Executar > Executar**. Isto irá executar o programa atual na próxima página do Python Shell (Interpretador) ou uma nova se a página seguinte não for uma Shell (Interpretador).

Nota: Executar o programa verifica automaticamente a sintaxe e guarda o programa.

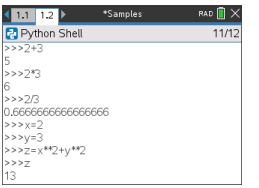
Python Shell (Interpretador)

A Python Shell (Interpretador) é o interpretador que executa os seus programas Python, outras peças do código Python ou comandos simples.

Código Python

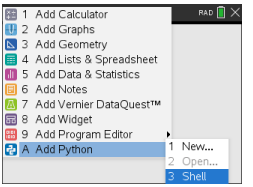


Comandos simples

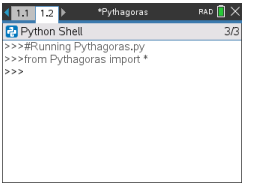


Adicionar uma página Python Shell (Interpretador)

Para adicionar uma nova página Shell (Interpretador) do Python no problema atual, prima menu e selecione **Adicionar Python > Shell**.



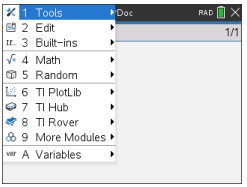
A Python Shell (Interpretador) também pode ser iniciada no Editor Python executando um programa premindo menu e selecionando **Executar > Executar**.

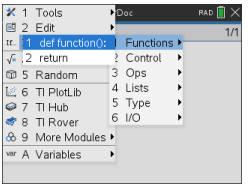


Trabalhar na Python Shell (Interpretador)

Pressionar menu exibe o menu Ferramentas do documento. Com estas opções de menu pode adicionar, mover e copiar blocos de código.

Menu de Ferramentas de Documento

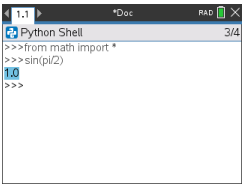




Nota: Se utilizar qualquer método de um dos módulos disponíveis, certifique-se de que executa primeiro uma declaração do módulo de importação como em qualquer ambiente de codificação Python.

A interação com a saída Shell (Interpretador) é semelhante à aplicação Calculadora, onde pode selecionar e copiar entradas e saídas anteriores para utilização noutras aplicações do Shell (Interpretador), Editor ou outras aplicações.

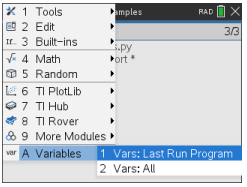
Seta para cima para selecionar e, em seguida, copiar e colar para a localização pretendida



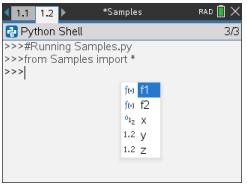
As funções globais e as variáveis do último programa executado podem ser introduzidas premindo var ou ctrl+L e selecionando a partir da lista ou prima menu e selecione **Variáveis > Vars: Último programa executado**.

Para escolher a partir de uma lista de funções e variáveis globais do último programa executado e de quaisquer módulos importados, prima menu e selecione **Variáveis > Vars: Todos**.

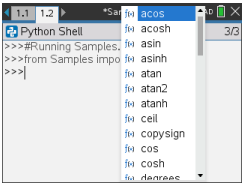
Menu Variáveis



Variáveis último programa executado

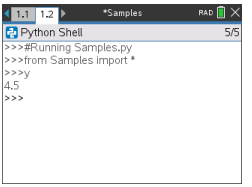


Todas as variáveis

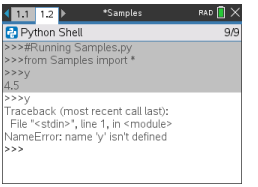


Todas as páginas da Python Shell (Interpretador) no mesmo problema partilham o mesmo estado (definições de variáveis definidas pelo utilizador e importadas). Quando guardar ou executar um programa Python nesse problema, ou prima menu e selecione **Ferramentas > Reiniciar Shell**, o histórico da Shell (Interpretador) terá então um fundo cinzento que indica que o estado anterior já não é válido.

Antes de guardar ou reiniciar



Depois de guardar ou reiniciar



Nota: A opção menu **Ferramentas > Limpar histórico** limpa o ecrã de qualquer atividade anterior na Shell (Interpretador), mas as variáveis ainda estão disponíveis.

Mensagens

Mensagens de erro e outras mensagens informativas podem aparecer enquanto estiver numa sessão Python. Se for apresentado um erro na Shell (Interpretador) quando um programa é executado, será apresentado um número de linha do programa onde o erro ocorreu. Prima ctrl menu e selecione **Ir para o Editor Python**. No Editor, prima menu depois selecione **Editar > Ir para linha**. Introduza o número da linha e pressione enter. O cursor será apresentado no primeiro caractere da linha onde ocorreu o erro.

Interromper um programa em execução

Enquanto uma função ou um programa estiver em execução, aparece o ponteiro de ocupado⏲.

- Para interromper o programa ou a função,
  - Windows®: Prima a tecla **F12**.
  - Mac®: Prima a tecla **F5**.
  - Unidade portátil: Prima a tecla on.

Mapa do menu Python

Esta secção lista todos os menus e itens do menu do Python Editor e Shell (Interpretador) e uma breve descrição para cada um.

Nota: Para os itens do menu que têm atalhos do teclado, os utilizadores Mac® devem substituir ⌘ (**Cmd**) em qualquer lugar onde seja utilizado **Ctrl**. Para obter uma lista completa dos atalhos da unidade portátil e do software TI-Nspire™, consulte o Manual eletrónico do TI-Nspire™.

Menu Ações

Nota: Isto aplica-se apenas ao Editor.

Item	Descrição
Novo	Abre a caixa de diálogo **Novo** onde introduz um nome e seleciona um tipo para o seu novo programa.
Abrir	Abre uma lista de programas disponíveis no documento atual.
Criar cópia…	Abre a caixa de diálogo **Criar cópia** onde pode guardar o programa atual com outro nome.
Renomear (BR)	Abre a caixa de diálogo **Renomear** onde pode mudar o nome do programa atual.
Fechar	Fecha o programa atual.
Definições	Abre a caixa de diálogo **Definições** onde pode alterar o tamanho do tipo de letra tanto para o Editor como para a Shell.
Instalar como módulo Python	Verifica a sintaxe Python do ficheiro TNS atual e move-a para a pasta PyLib.

Menu Executar

Nota: Isto aplica-se apenas ao Editor.

Item	Atalho	Descrição
Executar	Ctrl+R	Verifica sintaxe, guarda o programa e executa na Python Shell (Interpretador).
Verificar sintaxe e Guardar	Ctrl+B	Verifica a sintaxe e guarda o programa.
Ir para a Shell	N/D	Desloca o foco para a Shell (Interpretador) relacionado com o programa atual ou abre uma nova página Shell (Interpretador) junto ao Editor.

Menu Ferramentas

Nota: Isto aplica-se apenas à Shell.

Item	Atalho	Descrição
Voltar a executar o último programa	Ctrl+R	Volta a executar o último programa relacionado com o Shell atual.
Ir para o Editor Python	N/D	Abre a página Editor relacionada com a atual Shell.
Executar	N/D	Abre uma lista de programas disponíveis no documento atual. Após a seleção, o programa escolhido é executado.
Clear History (Apagar histórico)	N/D	Limpa o histórico na Shell atual mas não reinicia a Shell.
Reiniciar Shell	N/D	Repõe o estado de todas as páginas Shell abertas no problema atual. Todas as variáveis definidas e funções importadas já não estão disponíveis.
dir()	N/D	Apresenta a lista de funções no módulo especificado quando utilizado após a comando de importação.
A partir da importação PROGRAM *	N/D	Abre uma lista de programas disponíveis no documento atual. Após a seleção, o comando de importação é colado na Shell.
Instalar como módulo Python	N/D	Ativado apenas para módulos em formato binário. Move o ficheiro TNS atual para a pasta PyLib.

Menu Editar

Nota: Ctrl+A seleciona todas as linhas de código ou saída para cortar ou eliminar (apenas Editor), ou copiar e colar (Editor e Shell (Interpretador)).

Item	Atalho	Descrição
Indentar	TAB*	Indenta o texto na linha atual ou linhas selecionadas. * Se existirem pedidos incompletos em linha, o TAB irá navegar para a próxima linha.
Retirar indentação	Shift+TAB**	Retira a indentação do texto na linha atual ou linhas selecionadas. ** Se existirem pedidos incompletos em linha, o TAB irá navegar para a linha anterior.
Comentar/Retirar comentário	Ctrl+T	Adiciona/remove o símbolo de comentário ao/do início da linha atual.
Inserir cadeia multi-linha	N/D	(Apenas editor) Insere modelo de cadeia multi-linha.
Localizar	Ctrl+F	(Apenas editor) abre a caixa de diálogo **Encontrar** e pesquisa pela cadeia introduzida no programa atual.
Substituir	Ctrl+H	(Apenas editor) abre a caixa de diálogo **Substituir** e pesquisa pela cadeia introduzida no programa atual.
Ir para linha	Ctrl+G	(Apenas editor) abre a caixa de diálogo **Ir para linha** e salta para a linha especificada no programa atual.
Início da linha	Ctrl+8	Move o cursor para o início da linha atual.
Fim da linha	Ctrl+2	Move o cursor para o final da linha atual.
Saltar para o início	Ctrl+7	Move o cursor para o início da primeira linha do programa.
Saltar para o final	Ctrl+1	O cursor move-se para a última linha do programa.

Menu incorporado

funções

Item	Descrição
def function():	Define uma função dependente das variáveis especificadas.
return	Define o valor produzido por uma função.

Control (Controlo)

Item	Descrição
if..	Comando condicional.
if..else..	Comando condicional.
if..elif..else..	Comando condicional.
para o índice no intervalo(size):	Itera num intervalo.
para índice no intervalo(start,stop):	Itera num intervalo.
para índice no intervalo(start,stop,step):	Itera num intervalo.
para índice na lista:	Itera sobre os elementos da lista.
while..	Executa os comandos num bloco de código até que uma condição seja avaliada como Falsa.
elif:	Comando condicional.
else:	Comando condicional.

Ops

Item	Descrição
x=y	Define o valor da variável.
x==y	Cola o operador de comparação igual a (==).
x!=y	Cola o operador de comparação diferente de (!=).
x>y	Cola o operador de comparação maior que (>).
x>=y	Cola o operador de comparação maior ou igual a (>=).
x<y	Cola o operador de comparação menor que (<).

Item	Descrição
x<=y	Cola o operador de comparação menor ou igual a (<=).
e	Cola o operador lógico e (and).
ou	Cola o operador lógico ou (or).
não	Cola o operador lógico não (not).
Verdadeiro	Cola o valor booleano verdadeiro.
Falso	Cola o valor booleano falso.

Listas definidas

Item	Descrição
[]	Cola os parêntesis retos ([]).
list()	Converte a sequência para o tipo "lista".
len()	Devolve o número de elementos da lista.
max ()	Devolve o valor máximo da lista.
min ()	Devolve o valor mínimo da lista.
.append()	O método anexa um elemento a uma lista.
.remove()	O método remove a primeira instância de um elemento de uma lista.
range(start,stop,step)	Devolve um conjunto de números.
para índice no intervalo(start,stop,step)	Utilizado para iterar sobre um intervalo.
.insert()	O método adiciona um elemento na posição especificada.
.split()	O método devolve uma lista com elementos separados por delimitador especificado.
sum ()	Devolve a soma dos elementos de uma lista.
sorted()	Devolve uma lista ordenada.
.sort()	O método ordena uma lista no lugar.

Tipo

Item	Descrição
int ()	Devolve uma parte inteira.

Item	Descrição
float()	Devolve um valor com vírgula flutuante.
round(x,ndigits)	Devolve um número de ponto float arredondado para o número especificado de dígitos.
str()	Devolve uma cadeia.
complex()	Devolve um número complexo.
type()	Devolve o tipo do objeto.

Entrada/Saída

Item	Descrição
print()	Apresenta o argumento como cadeia (string).
input()	Solicita o utilizador para introduzir.
eval()	Avalia uma expressão representada como uma cadeia.
.format()	O método formata a cadeia especificada.

Menu Matemática

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Cálculos matemáticos**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
da importação math *	Importa todos os métodos (funções) do módulo cmath.
fabs()	Devolve o valor absoluto de um número real.
sqrt ()	Devolve a raiz quadrada de um número real.
exp()	Devolve e**x.
pow(x,y)	Devolve x elevado à potência y.
log(x,base)	Devolve $\log_{base}(x)$. log(x) sem base devolve o logaritmo natural x.
fmod(x,y)	Devolve o valor do módulo de x e y, como definido na biblioteca.. Utilizar quando x e y são floats.
ceil()	Devolve o menor número inteiro maior ou igual a um número real.
floor ()	Devolve o maior número inteiro menor ou igual a um número real.
trunc()	Trunca um número real para um número inteiro.
frexp()	Devolve um par (y,n) em que x == y * 2**n.

Const

Item	Descrição
e	Returns value for the constant e.
pi	Returns value for the constant pi.

Trig (Trig)

Item	Descrição
radians()	Converte o ângulo em graus para radianos.
degrees()	Converte o ângulo em radianos para graus.

Item	Descrição
sin ()	Devolve o seno do argumento em radianos.
cos ()	Devolve o cosseno do argumento em radianos.
tan ()	Devolve a tangente do argumento em radianos.
asin()	Devolve o arco seno do argumento, em radianos.
acos()	Devolve o arco cosseno do argumento, em radianos.
atan()	Devolve o arco tangente do argumento, em radianos.
atan2(y,x)	Devolve o arco tangente de x/y, em radianos.

Menu Aleatório

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Simulações aleatórias**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
de importação aleatória *	Importa todos os métodos do módulo aleatório.
random()	Devolve um número de ponto float de 0 a 1.0.
uniform(min,max)	Devolve um número aleatório x (float) de tal modo que o min <= x <= máx.
randint(min,max)	Devolve um número inteiro aleatório entre min e máx.
choice(sequence)	Devolve um elemento aleatório a partir de uma sequência não vazia.
randrange(start,stop,step)	Devolve um número aleatório do início até parar por passo.
seed()	Inicializa o gerador de números aleatórios.

TI Menu PlotLib

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Representação gráfica (x,y) e texto**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
importar ti_plotlib como plt	Importa todos os métodos (funções) do módulo ti_ plotlib no espaço do nome "plt". Como resultado, todos os nomes de funções colados dos menus serão precedidos por "plt.".

Configurar

Item	Descrição
cls()	Limpa o ecrã de representação gráfica.
grid(x-scale,y-scale,"style")	Apresenta uma grelha utilizando a escala especificada para os eixos x e y.
window(xmin,xmax,ymin,ymax)	Define a janela de representação gráfica mapeando o intervalo horizontal especificado (xmin, xmax) e o intervalo vertical (ymin, ymax) para a área de representação gráfica atribuída (píxeis).
auto_window(x-list,y-list)	Define automaticamente a escala da janela de representação gráfica para ajustar os intervalos de dados na x-list e y-list especificados no programa antes de auto_window().
axes("mode")	Apresenta os eixos na janela especificada na área de representação gráfica.
labels("x-label","y-label",x,y)	Apresenta etiquetas "x-label" e "y-label" nos eixos do gráfico nas posições x e y da linha.
title("title")	Apresenta "título" centrado na linha superior da janela.
show_plot()	Apresenta a saída de desenho com buffer. As funções use_buffer() e show_plot() são úteis nos casos em que a apresentação de vários objetos no ecrã pode causar atrasos (não necessariamente na maior parte dos casos).
use_buffer()	Permite um buffer fora do ecrã para acelerar o desenho.

Desenhar

Item	Descrição
color(red,green,blue)	Define a cor para todos os gráficos/ representações gráficas seguintes.
cls()	Limpa o ecrã de representação gráfica.
show_plot()	Executa a apresentação do gráfico conforme configurado no programa.
scatter(x-list,y-list,"mark")	Apresenta graficamente uma sequência de pares ordenados de (x-list,y-list) com o estilo de marcação especificado.
plot(x-list,y-list,"mark")	Desenha uma reta utilizando pares ordenados da lista x especificada e da lista y.
plot(x,y,"mark")	Desenha um ponto utilizando as coordenadas x e y com o estilo de marcação especificado.
line(x1,y1,x2,y2,"mode")	Desenha um segmento de reta de (x1,y1) até (x2,y2).
lin_reg(x-list,y-list,"display")	Calcula e desenha o modelo de regressão linear, ax+b, da x-list,y-list.
pen("size","style")	Define o aspeto de todas as retas seguintes até que a próxima pen() seja executada.
text_at(row,"text","align")	Apresenta o "texto" especificado na área de representação gráfica no "alinhamento" especificado.

Propriedades

Item	Descrição
xmin	Variável especificada para argumentos de janela definidos como plt.xmin.
xmax	Variável especificada para argumentos de janela definidos como plt.xmax.
ymin	Variável especificada para argumentos de janela definidos como plt.ymin.
ymax	Variável especificada para argumentos de janela definidos como plt.ymax.
m	Depois da plt.linreg() ser executada num programa, os valores calculados de declive, m e interceção, b, são armazenados em plt.m e plt.b.
b	Depois da plt.linreg() ser executada num programa, os valores calculados de declive, a e interceção, b, são armazenados em plt.a e plt.b.

Menu do Hub TI

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Projeto do Hub**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
da importação do ti_hub *	Importa todos os métodos do módulo ti_hub.

Dispositivos Hub integrados > Saída de cor

Item	Descrição
rgb(red,green,blue)	Define a cor para o LED RGB.
blink(frequency,time)	Define a frequência de intermitência e a duração da cor selecionada.
off()	Desliga o LED RGB.

Dispositivos Hub integrados > Saída de luz

Item	Descrição
on()	Liga o LED.
off()	Desliga o LED.
blink(frequency,time)	Define a frequência de intermitência e a duração do LED.

Dispositivos Hub integrados > Saída de som

Item	Descrição
tone(frequency,time)	Reproduz um tom da frequência especificada para o tempo especificado.
note("note",time)	Reproduz a nota especificada para o tempo especificado. A nota é especificada utilizando o nome da nota e uma oitava. Por exemplo: A4, C5. Os nomes das notas são C, CS, D, DS, E, F, FS, G, GS, A, AS e B. Os números de oitava variam de 1 a 9 (inclusive).

Item	Descrição
tone(frequency,time,tempo)	Reproduz um tom da frequência especificada para o tempo e ritmo especificados. O ritmo define o número de sinais sonoros por segundo de 0 a 10 (inclusive).
note("note",time,tempo)	Reproduz a nota especificada para o tempo e ritmo especificados. A nota é especificada utilizando o nome da nota e uma oitava. Por exemplo: A4, C5. Os nomes das notas são C, CS, D, DS, E, F, FS, G, GS, A, AS e B. Os números de oitava variam de 1 a 9 (inclusive). Os números de ritmo variam de 0 a 10 (inclusive).

Dispositivos Hub integrados > Entrada de brilho

Item	Descrição
measurement()	Lê o sensor de BRILHO (nível de luz) incorporado e devolve uma leitura. O intervalo predefinido é de 0 a 100. Isto pode ser alterado utilizando a função range().
range(min,max)	Define o intervalo para mapear as leituras do sensor de nível de luz. Se ambos estiverem em falta ou definidos para um valor de Nenhum, então é definido o intervalo de brilho predefinido de 0 a 100.

Adicionar dispositivo de entrada

Este menu tem uma lista dos sensores (dispositivos de entrada) suportados pelo módulo ti_hub. Todos os itens do menu irão colar o nome do objeto e esperar uma variável e uma porta utilizada com o sensor. Cada sensor tem um método measurement() que devolve o valor do sensor.

Item	Descrição
DHT (Humidade Digital e Temp)	Devolve uma lista constituída pela temperatura atual, humidade, tipo de sensor e último estado lido em cache.
Ranger	Devolve a medição de distância atual a partir do sensor ultrassónico. • **measurement_time()** - devolve o tempo que o sinal ultrassónico demora a chegar ao objeto (o

Item	Descrição
	"tempo de voo").
Nível de luz	Devolve o nível de brilho a partir do sensor de nível de luz externo (brilho).
Temperatura	Devolve a leitura de temperatura a partir do sensor de temperatura externo. A configuração predefinida é para suportar o sensor de temperatura Seeed nas portas IN 1, IN 2 ou IN 3. Para utilizar o sensor de temperatura TI LM19 do pacote da placa de ensaio TI-Innovator™ Hub, edite a porta para o pino BB em utilização e utilize um argumento opcional "TIANALOG". Exemplo: mylm19=temperature("BB 5","TIANALOG")
Humidade	Devolve a leitura do sensor de humidade.
Magnético	Deteta a presença de um campo magnético. O valor limiar para determinar a presença do campo é definido através da função trigger(). O valor predefinido do limiar é 150.
Vernier	Lê o valor do sensor analógico Vernier especificado no comando. O comando suporta os seguintes sensores Vernier: • **temperatura** - Sensor de temperatura de aço inoxidável. • **nível de luz** - Sensor de nível de luz TI. • **pressão** - Sensor de pressão de gás original • **pressão** - Sensor de pressão de gás mais recente. • **pH** - Sensor de pH. • **força10** - Definição ±10 N, sensor de força dupla. • **força50** - Definição ±50 N, sensor de força dupla. • **acelerómetro** - Acelerómetro G baixo. • **genérico** - Permite a definição de outros sensores não suportados diretamente acima e a utilização da API calibrate() acima para definir coeficientes de equações.
Analog In	Suporta a utilização de dispositivos genéricos de entrada analógica.

Item	Descrição
Digital In	Devolve o estado atual do pino digital ligado ao objeto DIGITAL ou o estado em cache do valor de saída digital DEFINIDO pela última vez para o objeto.
Potenciómetro	Suporta um sensor do potenciómetro. O intervalo do sensor pode ser alterado pela função range().
Termístor	Lê os sensores do termístor. Os coeficientes predefinidos são concebidos para corresponder ao termístor incluído no pacote da placa de ensaio do TI-Innovator™ Hub, quando utilizado com uma resistência fixa de 10KΩ. Pode configurar um novo conjunto de coeficientes de calibração e resistência de referência para o termístor utilizando a função calibrate().
Intensidade sonora	Suporta sensores de intensidade sonora.
Entrada de cor	Fornece interfaces a um sensor de entrada de cor ligada ao I2C. O pino do bb_port é utilizado para além da porta I2C para controlar o LED no sensor de cor. • **color_number():** Devolve um valor de 1 a 9 que representa a cor que o sensor deteta. Os números representam as cores de acordo com o seguinte mapeamento: 1: Vermelho 2: Verde 3: Azul 4: Ciano 5: Magenta 6: Amarelo 7: Preto 8: Branco 9: Cinzento • **red():** Devolve um valor de 0 a 255 que representa a intensidade do nível de cor VERMELHO a ser detetado. • **green():** Devolve um valor de 0 a 255 que representa a intensidade do nível de cor VERDE a ser detetado. • **blue():** Devolve um valor de 0 a 255 que representa a intensidade do nível de cor AZUL a

Item	Descrição
	ser detetado. • **gray():** Devolve um valor de 0 a 255 que representa o nível cinzento a ser detetado, em que 0 é preto e 255 é branco.
Porta BB	Fornece suporte para utilizar todos os 10 pinos de porta BB como porta de entrada/saída digital combinada. As funções de inicialização têm um parâmetro de "máscara" opcional que permite a utilização do subconjunto dos 10 pinos. • **read_port():** Lê os valores atuais nos pinos de entrada da porta BB. • **write_port(value):** Define os valores dos pinos de saída para o valor especificado, onde o valor está entre 0 e 1023. Tenha em atenção que o valor também é ajustado contra o valor da máscara na operação var=bbport(máscara), se tiver sido fornecida uma máscara.
Tempo do Hub	Fornece acesso ao temporizador de milissegundo interno.
TI-RGB Array	Fornece funções para programar a matriz TI-RGB. A função de inicialização aceita um parâmetro opcional "LAMP" para permitir um modo de alto brilho para a TI-RGB Array que requer uma fonte de alimentação externa. • **set(led_position, r,g,b):** Define uma led_ position específica (0-15) para o valor r,g,b especificado, em que r,g,b são valores de 0 a 255. • **set(led_list,red,green,blue):** Define os LED definidos na "led_list" para a cor especificada por "vermelho", "verde", "azul". A "led_list" é uma lista Python que inclui índices dos LED de 0 a 15. Por exemplo, o conjunto([0,2,4,6,15], 0, 255) irá definir os LED 0, 2, 4, 6 e 15 para azul. • **set_all(r,g,b):** Define todos os LED RGB na matriz com o mesmo valor r,g,b. • **all_off():** Desliga todos os RGBs no conjunto. • **measurement():** Devolve a corrente aproximada que o conjunto RGB está a utilizar a partir do TI-Innovator™ em miliamperes. • **pattern(pattern):** Utilizando o valor do

Item	Descrição
	argumento como um valor binário no intervalo de 0 a 65535, liga os píxeis onde estaria um valor de 1 na representação. Os LED são ligados como VERMELHO com valor de nível pwm de 255. • **pattern(value,red,green,blue):** Define os LEDs definidos pelo "padrão" para a cor especificada por "vermelho", "verde", "azul".

Adicionar dispositivo de saída

Este menu tem uma lista de dispositivos de saída suportados pelo módulo ti_hub. Todos os itens do menu irão colar o nome do objeto e esperar uma variável e uma porta utilizada com o sensor.

Item	Descrição
LED	Funções para controlar os LEDs ligados externamente.
RGB	Apoio para controlar LEDs RGB externos.
TI-RGB Array	Fornece funções para programar a matriz TI-RGB.
Coluna	Funções para apoiar uma coluna externa com o TI-Innovator™ Hub. As funções são as mesmas que para o "som" acima.
Potência	Funções para controlar a alimentação externa com o TI-Innovator™ Hub. • **set(value):** Define o nível de potência para o valor especificado, entre 0 e 100. • **on():** Define o nível de potência para 100. • **off():** Define o nível de potência para 0.
Servomotor contínuo	Funções para controlar servomotores contínuos. • **set_cw(speed,time):** O servomotor irá rodar no sentido dos ponteiros do relógio à velocidade especificada (0-255) e pela duração específica em segundos. • **set_ccw(speed,time):** O servomotor irá rodar no sentido dos ponteiros do relógio à velocidade especificada (0-255) e pela duração específica em segundos. • **stop():** Para o servomotor contínuo.
Saída analógica	Funções para utilização de dispositivos genéricos de entrada analógica.
Motor de vibração	Funções para controlar os motores de vibração. • **set(val):** Define a intensidade do motor de vibração para

Item	Descrição
	"val" (0-255). • **off():** Desliga o motor de vibração. • **on():** Liga o motor de vibração no nível mais alto.
Relé	Controla interfaces para relés de controlo. • **on():** Define o relé para o estado ON (Ligado). • **off():** Define o relé para o estado OFF (Desligado).
Servomotor	Funções para controlar servomotores contínuos. • **set_position(pos):** Define a posição do servomotor de varrimento num intervalo de -90 a +90. • **zero():** Define o servomotor de varrimento para a posição zero.
Squarewave	Funções para gerar uma onda quadrada. • **set(frequency,duty,time):** Define a onda quadrada de saída com um ciclo de serviço predefinido de 50% (se o ciclo não for especificado) e uma frequência de saída especificada por "frequência". A frequência pode ser de 1 a 500 Hz. O ciclo de trabalho, se especificado, pode ser de 0 a 100%. • **off():** Desliga a onda quadrada.
Saída digital	Interfaces para controlar uma saída digital. • **set(val):** Define a saída digital para o valor especificado por "val" (0 ou 1). • **on():** Define o estado da saída digital para alto (1). • **off():** Define o estado da saída digital para baixo (0).
Porta BB	Fornece funções para programar a matriz TI-RGB. Consulte os detalhes acima.

Comandos

Item	Descrição
sleep(seconds)	Pausa o programa durante um número especificado de segundos. Importado do módulo "time".
text_at(row,"text","align")	Apresenta o "texto" especificado na área de representação gráfica no "alinhamento" especificado. Parte do módulo ti_plotlib.
cls()	Limpa o ecrã Shell para representar graficamente. Parte do módulo ti_plotlib.

Item	Descrição
enquanto get_key() != "esc":	Executa os comandos no ciclo "while" até que a tecla "esc" seja premida.
get_key()	Devolve uma cadeia que representa a tecla premida. A tecla "1" devolve "1", "esc" devolve "esc" e por aí adiante. Quando chamada sem quaisquer parâmetros - get_key() - devolve imediatamente. Quando é chamado com um parâmetro - get_key(1) - aguarda até que seja premida uma tecla. Parte do módulo ti_system.

Portas

Estas são as portas de entrada e saída disponíveis no TI-Innovator™ Hub.

Item
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
I2C

Menu TI Rover

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Codificação Rover**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
importar ti_rover como rv	Importa todos os métodos (funções) do módulo ti_rover no espaço do nome "rv". Como resultado, todos os nomes de funções colados dos menus serão precedidos por "plt.".

Condução

Item	Descrição
forward(distance)	Move o Rover para a frente na distância especificada em unidades de grelha.
backward(distance)	Move o Rover para trás na distância especificada em unidades de grelha.
left(angle_degrees)	Vira o Rover para a esquerda no ângulo especificado em graus.
right(angle_degrees)	Vira o Rover para a direita no ângulo especificado em graus.
stop()	Para qualquer atual movimento imediatamente.
stop_clear()	Pára imediatamente qualquer movimento atual e apaga todos os comandos pendentes.
resume()	Retoma o processamento de comandos.
stay(time)	O rover permanece no lugar por um período de tempo especificado em segundos (opcional). Se não for especificado qualquer tempo, o Rover permanece durante 30 segundos.
to_xy(x,y)	Move o Rover para a posição de coordenadas (x,y) na grelha virtual.
to_polar(r,theta_degrees)	Move o Rover para a posição de coordenadas polares (x,y) na grelha virtual. O ângulo é especificado em graus.
to_angle(angle,"unit")	Gira o rover para o ângulo especificado na grelha virtual. O ângulo é relativo a um ângulo zero que aponta para o eixo x na grelha virtual.

Condução > Condução com opções

Item	Descrição
forward_time(time)	Move o Rover para a frente durante o tempo especificado.
backward_time(time)	Move o Rover para trás durante o tempo especificado.
forward(distance,"unit")	Move o Rover para a frente à velocidade predefinida para a distância especificada. A distância pode ser especificada em unidades de grelha, metros ou rotações da roda.
backward(distance,"unit")	Move o Rover para trás à velocidade predefinida para a distância especificada. A distância pode ser especificada em unidades de grelha, metros ou rotações da roda.
left(angle,"unit")	Vira o Rover para a esquerda segundo o ângulo especificado. O ângulo pode ser em graus, radianos ou gradianos.
right(angle,"unit")	Vira o Rover para a direita segundo o ângulo especificado. O ângulo pode ser em graus, radianos ou gradianos.
forward_time(time,speed,"rate")	Move o Rover para a frente durante o tempo especificado à velocidade especificada. A velocidade pode ser especificada em unidades de grelha/s, metros/s ou rotações da roda/s.
backward_time(time,speed,"rate")	Move o Rover para trás durante o tempo especificado à velocidade especificada. A velocidade pode ser especificada em unidades de grelha/s, metros/s ou rotações da roda/s.
forward(distance,"unit",speed,"rate")	Move o Rover para a frente na distância especificada à velocidade especificada. A distância pode ser especificada em unidades de grelha, metros ou rotações da roda. A velocidade pode ser especificada em

Item	Descrição
	unidades de grelha/s, metros/s ou rotações da roda/s.
backward(distance,"unit",speed,"rate")	Move o Rover para trás na distância especificada à velocidade especificada. A distância pode ser especificada em unidades de grelha, metros ou rotações da roda. A velocidade pode ser especificada em unidades de grelha/s, metros/s ou rotações da roda/s.

Entradas

Item	Descrição
ranger_measurement()	Lê o sensor de distância ultrassónica na parte frontal do Rover, devolvendo a distância atual em metros.
color_measurement()	Devolve um valor de 1 a 9, indicando a cor predominante a ser "visualizada" pelo sensor de entrada de cor do Rover. 1 = vermelho 2 = verde 3 = azul 4 = ciano 5 = magenta 6 = amarelo 7 = preto 8 = cinzento 9 = branco
red_measurement()	Devolve um valor entre 0 e 255 que indica o nível vermelho captado pelo sensor de entrada de cor.
green_measurement()	Devolve um valor entre 0 e 255 que indica o nível verde captado pelo sensor de entrada de cor.
blue_measurement()	Devolve um valor entre 0 e 255 que indica o nível azul captado pelo sensor de entrada de cor.
gray_measurement()	Devolve um valor entre 0 e 255 que indica o nível cinzento captado pelo sensor de entrada de cor.
encoders_gyro_measurement()	Devolve uma lista de valores que contêm contagens do codificador da roda esquerda e direita, bem como a direção atual do giroscópio.

Item	Descrição
gyro_measurement()	Devolve um valor que representa a leitura atual do giroscópio, incluindo o desvio, nos graus.
ranger_time()	Devolve o tempo que o sinal ultrassónico do sensor TI-Rover demora a chegar ao objeto (o "tempo de voo").

Saídas

Item	Descrição
cor_rgb(r,g,b)	Define a cor do LED RGB do Rover para os valores específicos de vermelho, verde e azul.
color_blink(frequency,time)	Define a frequência de intermitência e a duração da cor selecionada.
cor_off()	Desliga o LED RGB do Rover.
motor_left(speed,time)	Define a potência do motor esquerdo para o valor especificado para a duração especificada. A velocidade está no intervalo -255 a 255 com 0 a representar paragem. Os valores de velocidade positivos são a rotação no sentido anti-horário e os valores da velocidade negativa no sentido horário. O parâmetro de tempo opcional, se especificado, tem um intervalo válido de 0,05 a 655,35 segundos. Se não for especificado, é utilizada uma predefinição de 5 segundos.
motor_right(speed,time)	Define a potência do motor esquerdo para o valor especificado para a duração especificada. A velocidade está no intervalo -255 a 255 com 0 a representar paragem. Os valores de velocidade positivos são a rotação no sentido anti-horário e os valores da velocidade negativa no sentido horário. O parâmetro de tempo opcional, se especificado, tem um intervalo válido de 0,05 a 655,35 segundos. Se não for especificado, é utilizada uma

Item	Descrição
	predefinição de 5 segundos.
motors("ldir",left_val,"rdir",right_val,time)	Define a roda esquerda e direita para os níveis de velocidade especificados, durante um período de tempo opcional em segundos. Os valores da velocidade (left_val, right_val) estão no intervalo de 0 a 255 com 0 a representar paragem. Os parâmetros ldir e rdir especificam a rotação CW (sentido horário) ou CCW (sentido anti-horário) das respetivas rodas. O parâmetro de tempo opcional, se especificado, tem um intervalo válido de 0,05 a 655,35 segundos. Se não for especificado, é utilizada uma predefinição de 5 segundos.

Caminho

Item	Descrição
waypoint_xythdrn()	Lê a coordenada x, coordenada y, tempo, direção, distância percorrida, número de rotações da roda, número de comando da localização atual. Retorna uma lista com todos estes valores como elementos.
waypoint_prev	Lê a coordenada x, coordenada y, tempo, direção, distância percorrida, número de rotações da roda, número de comando da localização anterior.
waypoint_eta	Devolve o tempo estimado para conduzir até uma localização.
path_done()	Devolve um valor de 0 ou 1, dependendo se o Rover está a mover-se (0) ou terminou todos os movimentos (1).
pathlist_x()	Devolve uma lista de valores X desde o início até e incluindo o valor X da localização atual.
pathlist_y()	Devolve uma lista de valores Y desde o início até e incluindo o valor Y da localização atual.
pathlist_time()	Devolve uma lista do tempo em segundos desde o início até e incluindo o valor atual de tempo da localização.
pathlist_heading()	Devolve uma lista das direções desde o início até e incluindo o valor atual de direção da localização.
pathlist_distance()	Devolve uma lista das distâncias percorridas desde o início até e incluindo o valor atual da distância da localização.

Item	Descrição
pathlist_revs()	Devolve uma lista do número de rotações percorridas desde o início até e incluindo o valor atual de rotações da localização.
pathlist_cmdnum()	Devolve uma lista de números de comando para o caminho.
waypoint_x()	Devolve a coordenada x da localização atual.
waypoint_y()	Devolve a coordenada y da localização atual.
waypoint_time()	Devolve o tempo gasto a viajar da localização anterior à atual.
waypoint_heading()	Devolve a direção absoluta da localização atual.
waypoint_distance()	Devolve a distância percorrida entre a localização anterior e a atual.
waypoint_revs()	Devolve o número de rotações necessárias para viajar entre a localização anterior e a atual.

Definições

Item	Descrição
units/s	Opção para velocidade em unidades de grelha por segundo.
m/s	Opção para velocidade em metros por segundo.
revs/s	Opção para velocidade nas rotações das rodas por segundo.
Unidades	Opção de distância em unidades de grelha.
m	Opção para distância em metros.
revs	Opção para distância em rotações da roda.
graus	Opção de viragem em graus.
radianos	Opção de viragem em radianos.
gradians	Opção de viragem em grados.
no sentido horário	Opção para especificar a direção da roda.
no sentido anti-horário	Opção para especificar a direção da roda.

Comandos

Estes comandos são uma recolha de funções de outros módulos, bem como do módulo TI Rover.

Item	Descrição
sleep(seconds)	Pausa o programa durante um número especificado de segundos. Importado do módulo time.
text_at(row,"text","align")	Apresenta o "texto" especificado na área de representação gráfica no "alinhamento" especificado. Importado do módulo ti_plotlib.
cls()	Limpa o ecrã Shell para representar graficamente. Importado do módulo ti_plotlib.
enquanto get_key() != "esc":	Executa os comandos no ciclo "while" até que a tecla "esc" seja premida.
wait_until_done()	Pausa o programa até que o Rover termine o comando atual. Esta é uma forma útil de sincronizar os comandos não Rover com o movimento do Rover.
while not path_done()	Executa os comandos no ciclo "while" até que o Rover termine todo o movimento. Devolve um valor de 0 ou 1, dependendo se o Rover está a mover-se (0) ou terminou todos os movimentos (1).
position(x,y)	Define a posição do Rover na grelha virtual para a coordenada x,y especificada.
position(x,y,heading,"unit")	Define a posição do Rover na grelha virtual para a coordenada x,y especificada e a direção virtual, relativamente ao eixo x virtual, é definida se for fornecido uma direção (nas unidades para ângulos especificados). Presume-se que ângulos positivos de 0 a 360 sejam no sentido anti-horário a partir do eixo x positivo. Presume-se que ângulos negativos de 0 a -360 sejam no sentido horário a partir do eixo x positivo.
grid_origin()	Define RV como estando no ponto de origem da referencial atual, (0,0).
grid_m_unit(scale_value)	Define o espaçamento da grelha virtual em metros por unidade (m/unidade) para o valor especificado. 0,1 é a m/unidade predefinida e traduz-se para 1 unidade = 100 mm ou 10 cm ou 1 dm ou 0,1 m. O intervalo de scale_value válido é de 0,01 a 10,0.
path_clear()	Limpa qualquer informação de caminho ou localização pré-existente.

Item	Descrição
zero_gyro()	Repõe o ângulo do giroscópio do Rover para 0,0 e limpa as contagens do codificador da roda esquerda e direita.

Menu matemática complexa

Este submenu está localizado em **Mais módulos**.

Item	Descrição
da importação cmath *	Importa todos os métodos do módulo cmath.
complex(real,imag)	Devolve um número complexo.
rect(modulus,argument)	Converte coordenadas polares para a forma retangular de um número complexo.
.real	Devolve a parte real do número complexo.
.imag	Devolve a parte imaginária de um número complexo.
polar()	Converte coordenadas retangulares para coordenadas polares de um número complexo.
phase()	Devolve a fase de um número complexo.
exp()	Devolve e**x.
cos ()	Devolve o cosseno de um número complexo.
sin ()	Devolve o seno de um número complexo.
log ()	Devolve o logaritmo natural de um número complexo.
log10()	Devolve o logaritmo de base 10 de um número complexo.
sqrt ()	Devolve a raiz quadrada de um número complexo.

Menu de tempo

Este submenu está localizado em **Mais módulos**.

Item	Descrição
a partir de time import *	Importa todos os métodos do módulo time.
sleep(seconds)	Pausa o programa durante um número especificado de segundos.
clock()	Devolve o tempo do processador atual como um número float expresso em segundos.
localtime()	Converte um tempo expresso em segundos desde 1 de janeiro de 2000 num indicador de nove tuplos contendo uma etiqueta de ano, mês, dia do mês, hora, minutos, segundos, dia da semana, dia do ano e Hora de verão (DST). Se o argumento opcional (segundos) não for fornecido, então é utilizado o relógio em tempo real.
ticks_cpu()	Devolve um contador de milissegundos crescente específico do processador com ponto de referência arbitrário. Para medir o tempo de forma consistente em diferentes sistemas, utilize ticks_ms().
ticks_diff()	Mede o período entre chamadas consecutivas para ticks_cpu() ou ticks_ms(). Esta função não deve ser utilizada para medir períodos de tempo arbitrariamente longos.

Menu do sistema TI

Este submenu está localizado em **Mais módulos**.

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Partilha de dados**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
da importação do ti_system *	Importa todos os métodos (funções) do módulo ti_ system.
recall_value("name")	Chama novamente uma variável do SO predefinida (value) com o nome "name".
store_value("name",value)	Armazena uma variável Python (value) para uma variável do SO com o nome "nome".
recall_list("name")	Chama novamente uma lista de SO predefinida com o nome "name".
store_list("name",list)	Armazena uma lista de Python (lista) para uma variável de lista do SO designada por "name".
eval_function("name",value)	Avalia uma função do SO predefinida no valor especificado.
get_platform()	Devolve "hh" para unidade portátil e "dt" para ambiente de trabalho.
get_key()	Devolve uma cadeia que representa a tecla premida. A tecla "1" devolve "1", "esc" devolve "esc" e por aí adiante. Quando chamada sem quaisquer parâmetros - get_key() - devolve imediatamente. Quando é chamado com um parâmetro - get_key(1) - aguarda até que seja premida uma tecla.
get_mouse()	Devolve as coordenadas do rato como um a lista ordenada (tuple) de dois elementos, seja a posição do píxel do ecrã ou (-1,-1) se estiver fora do ecrã.
enquanto get_key() != "esc":	Executa os comandos no ciclo "while" até que a tecla "esc" seja premida.
clear_history()	Limpa o histórico da Shell (Interpretador).
get_time_ms()	Devolve o tempo em milissegundos com uma precisão de milissegundos. Esta funcionalidade pode ser utilizada para calcular uma duração em vez de determinar o tempo real do relógio.

Menu Desenho TI

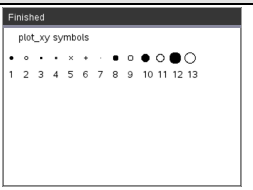
Este submenu está localizado em **Mais módulos**.

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Gráficos de Geometria**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
de ti_draw import *	Importa todos os métodos do módulo ti_draw.

Forma

Item	Descrição
draw_line()	Desenha uma linha a partir da coordenada x1,y1 especificada para x2,y2.
draw_rect()	Desenha um retângulo começando na coordenada x,y especificada com a largura e altura especificadas.
fill_rect()	Desenha um retângulo começando na coordenada x,y especificada com a largura e altura especificadas e preenchidas com a cor especificada (utilizando set_color ou preto, se não definido).
draw_circle()	Desenha um círculo começando na coordenada x,y do centro especificado com o raio especificado.
fill_circle()	Desenha um círculo começando na coordenada x,y especificada com o raio especificado e preenchido com a cor especificada (utilizando set_color ou preto, se não definido).
draw_text()	Desenha uma cadeia de texto começando na coordenada x,y especificada.
draw_arc()	Desenha um arco começando na coordenada x,y especificada com a largura e altura e ângulo especificados.
fill_arc()	Desenha um arco começando na coordenada x,y especificada com a largura e altura e ângulo especificados e preenchido com a cor especificada (utilizando set_color ou preto, se não definido).
draw_poly()	Desenha um polígono utilizando os valores x-list,y-list especificados.
fill_poly()	Desenha um polígono utilizando os valores x-list,y-lista especificados preenchidos com a cor especificada (utilizando set_color ou preto se não definido).
plot_xy()	Desenha uma forma utilizando a coordenada x,y especificada e o número especificado de 1 a 13 representando diferentes formas e símbolos (ver abaixo).

Item	Descrição
	

Control (Controlo)

Item	Descrição
clear()	Limpa todo o ecrã. Pode ser utilizado com parâmetros x,y,largura,altura, para limpar um retângulo existente.
clear_rect()	Limpa o retângulo na coordenada x,y especificada com a largura e altura especificadas.
set_color()	Define a cor da(s) forma(s) que se seguem no programa até que outra cor seja definida.
set_pen()	Define a espessura e o estilo especificados da margem quando desenhar formas (não aplicável ao utilizar comandos de preenchimento).
set_window()	Define o tamanho da janela em que serão desenhadas formas. Esta função é útil para redimensionar a janela para corresponder aos dados ou para alterar a origem (0,0) da tela de desenho.
get_screen_dim()	Devolve o xmax e o ymax das dimensões do ecrã.
use_buffer()	Permite um buffer fora do ecrã para acelerar o desenho.
paint_buffer()	Apresenta a saída de desenho com buffer. As funções use_buffer() e paint_buffer() são úteis nos casos em que a apresentação de vários objetos no ecrã pode causar atrasos.

Notas

- A configuração predefinida tem (0,0) no canto superior esquerdo do ecrã. O eixo x positivo aponta para a direita e o eixo y positivo aponta para o fundo. Isto pode ser modificado utilizando a função set_window().
- As funções no módulo ti_draw estão disponíveis apenas na unidade portátil e na vista portátil do ambiente de trabalho.

Menu de imagem TI

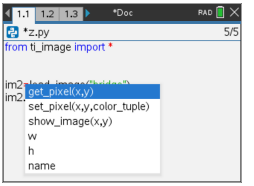
Este submenu está localizado em **Mais módulos**.

Nota: Ao criar um novo programa que utilize este módulo, recomenda-se a utilização do tipo de programa **Processamento de imagem**. Isto irá garantir que todos os módulos relevantes são importados.

Item	Descrição
a partir de ti_image import *	Importa todos os métodos do módulo ti_image.
new_image(width,height,(r,g,b))	Cria uma nova imagem com a largura e altura especificadas para utilização no programa Python. A cor da nova imagem é definida pelos valores (r,g,b).
load_image("name")	Carrega a imagem especificada pelo "nome" para utilização no programa Python. A imagem tem de fazer parte do documento TNS numa aplicação Notas ou Gráficos. O pedido "nome" apresentará os nomes das imagens (se tiverem sido nomeadas anteriormente) ou um número indicando a sua ordem de inserção.
copy_image(image)	Cria uma cópia da imagem especificada pela variável "imagem".

Métodos do objeto da imagem

As funções adicionais relacionadas com os objetos da imagem estão disponíveis no Editor e na Shell (Interpretador), escrevendo o nome da variável seguido por um . (ponto).



- **get_pixel(x,y):** Obtém o valor (r,g,b) do píxel no local definido pelo par de coordenadas (x,y).

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

- **set_pixel(x,y,color_tuple):** Define o píxel no local (x,y) para a cor especificada na color_tuple.

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

Define o píxel em (100,100) para a cor (0,0,255).

- **show_image(x,y):** Apresenta a imagem com o canto superior esquerdo no local (x,y).
- **w, h, name:** Obtém os parâmetros de largura, altura e nome da imagem.

Por exemplo

```
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 Variáveis

Nota: Estas listas não incluem variáveis definidas noutras aplicações TI-Nspire™.

Item	Descrição
Variáveis: Programa atual	(Apenas editor) Apresenta uma lista de funções e variáveis globais definidas no programa atual
Vars: Último programa executado	(Apenas Shell (Interpretador)) Apresenta uma lista de funções e variáveis globais definidas no último programa executado
Vars: Todos	(Apenas Shell (Interpretador)) Apresenta uma lista de funções e variáveis globais definidas no último programa executado e quaisquer módulos importados

Anexo

Palavras-chave de Python

As seguintes palavras-chave estão integradas na implementação Python do 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

Mapeamento chave de Python

Ao introduzir o código no Editor ou na Shell (Interpretador), o teclado foi concebido para colar as operações apropriadas Python ou menus abertos para fácil entrada de funções, palavras-chave, métodos, operadores, etc.

Tecla	Mapeamento
var	Abre o menu Variáveis
sto→	Colar = sinal
del ←	Apaga o caractere à esquerda do cursor
clear	Nenhuma ação
=	Colar = sinal
|≠≥▸	Colar o(s) símbolo(s) selecionado(s): • > • < • != • >= • <= • == • e • ou • não • | • & • ~
trig	Colar a função selecionada: • sin • cos • tan • atan2 • asin • acos • atan
ⓘ	Apresentar sugestões
:=	Colar :=
^	Colar **
$\sqrt[n]{x}$	Nenhuma ação
x^2	Colar **2

Tecla	Mapeamento
√	Colar sqrt()
×	Colar sinal de multiplicação (*)
"□"	Colar um sinal de aspas duplas (")
÷	Colar sinal da divisão (/)
a/b	Nenhuma ação
e^x	Colar exp()
ln	Colar log()
10^x	Colar 10**
log	Colar log(valor,base)
(	Colar (
)	Colar)
[]	Colar [ ]
{}	Colar { }
−	Colar sinal de subtrair (-)
≈	Adicionar uma nova linha após a linha atual
EE	Colar E
?!•	Colar o(s) símbolo(s) selecionado(s): • ? • ! • $ • ° • ' • % • " • : • ; • _ • \ • #
π•	Colar "pi"
⇆	Comportamento assinalado existente
↵	Adicionar uma nova linha após a linha atual

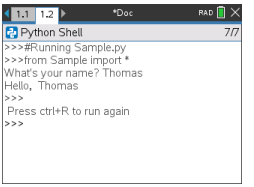
Exemplos de programas em Python

Utilize os seguintes exemplos para se familiarizar com os métodos Python. Estão também disponíveis no ficheiro **Getting Started Python.tns** localizado na pasta **Exemplos**.

Nota: Se copiar e colar qualquer código de amostra que contenha indicadores de indentação do separador (••) para o software TI-Nspire™, terá de substituir essas instâncias com as indentações atuais do separador.

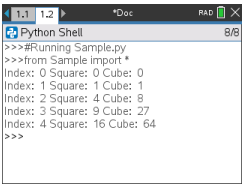
Olá

```
# 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")
```



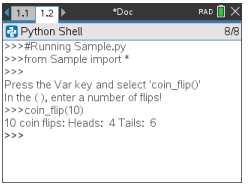
Exemplo de ciclo (loop)

```
# 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)
```



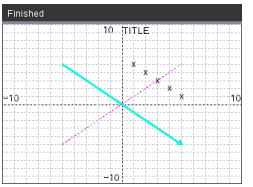
Cara ou coroa

```
# 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!")
```



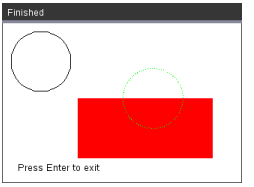
Representar graficamente

```
# 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")
```



Desenhar

```
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")
```



Imagem

```
# 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

Este programa utiliza o Python para controlar o TI-Innovator™ Hub, um microcontrolador programável. Executar o programa sem conectar a um TI-Innovator™ Hub irá originar uma mensagem de erro.

Para mais informações sobre o TI-Innovator™ Hub, visite 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()
```

Informações gerais

Ajuda online

education.ti.com/eguide

Selecione o seu país para obter mais informação sobre o produto.

Contacte a assistência técnica da TI

education.ti.com/ti-cares

Selecione o seu país para obter recursos técnicos ou assistência.

Informações da Assistência e Garantia

education.ti.com/warranty

Selecione o seu país para obter informações sobre a duração e os termos da garantia ou sobre a assistência ao produto.

Garantia Limitada. Esta garantia não afeta os seus direitos legais.

Texas Instruments Incorporated

12500 TI Blvd.

Dallas, TX 75243