TI-Nspire™ Python Programmierhandbuch

Weitere Informationen zu TI Technology finden Sie in der Online-Hilfe unter education.ti.com/eguide.

Wichtige Informationen

Sofern nicht ausdrücklich in der einem Programm beiliegenden Lizenz angegeben, übernimmt Texas Instruments für die Programme oder das Handbuchmaterial keinerlei Garantie, weder direkt noch indirekt. Dies umfasst auch jegliche indirekte Gewährleistung hinsichtlich der Marktgängigkeit oder der Eignung für einen bestimmten Zweck, ist jedoch nicht hierauf beschränkt und dieses Produkt wird lediglich „so wie es ist" zur Verfügung gestellt. In keinem Fall kann Texas Instruments für Schäden haftbar gemacht werden, die sich entweder in Verbindung mit dem Kauf bzw. Gebrauch dieses Produkts ergeben oder davon verursacht werden. Dies gilt für spezielle, begleitende und versehentliche Schäden sowie für Folgeschäden. Texas Instruments haftet maximal und ausschließlich mit dem in der Lizenz für das Programm genannten Betrag, unabhängig vom jeweiligen Fall. Des Weiteren haftet Texas Instruments nicht für Forderungen, die sich aus dem Gebrauch dieses Produkts durch eine andere Partei ergeben, welcher Art diese Forderungen auch immer sein mögen.

Die tatsächlichen Produkte können geringfügig von den Abbildungen abweichen.

Inhaltsverzeichnis

Erste Schritte in der Python-Programmierung

Bei der Verwendung von Python mit TI-Nspire™-Produkten können Sie:

- Python-Programme zu TNS-Dateien hinzufügen
- Python-Programme unter Verwendung von Vorlagen erstellen
- Interaktion und Datenaustausch mit anderen TI-Nspire™-Anwendungen nutzen
- mit dem TI-Innovator™ Hub und TI-Innovator™ Rover interagieren

Die TI-Nspire™ Python-Implementierung basiert auf MicroPython, einer kleinen Untermenge der Python 3-Standardbibliothek, die für die Ausführung auf Mikrocontrollern entwickelt wurde. Die ursprüngliche MicroPython-Implementierung wurde für die Nutzung durch TI angepasst.

Hinweis: Einige numerische Antworten können aufgrund von Unterschieden in den zugrunde liegenden mathematischen Implementierungen von den Ergebnissen des Rechners abweichen.

Python ist auf diesen TI-Nspire™-Produkten verfügbar:

Handhelds	**Desktop-Software**
TI-Nspire™ CX II	TI-Nspire™ CX Premium Teacher Software
TI-Nspire™ CX II CAS	TI-Nspire™ CX CAS Premium Teacher Software
TI-Nspire™ CX II-T	TI-Nspire™ CX Schülersoftware
TI-Nspire™ CX II-T CAS	TI-Nspire™ CX CAS Schülersoftware
TI-Nspire™ CX II-C	
TI-Nspire™ CX II-C CAS	

Hinweis: In den meisten Fällen ist die Funktionalität zwischen der Handheld- und der Software-Ansicht identisch, es können jedoch einige Unterschiede auftreten. Dieser Leitfaden geht davon aus, dass Sie das Handheld-Gerät oder die Handheld-Ansicht in der Software verwenden.

Python-Module

TI-Nspire™ Python enthält die folgenden Module:

Standard-Module	**TI-Module**
Mathematik (math)	TI PlotLib (ti_plotlib)
Zufallszahl (random)	TI Hub (ti_hub)
Komplexe Mathematik (cmath)	TI Rover (ti_rover)
Zeit (time)	TI System (ti_system)
	TI Draw (ti_draw)
	TI Image (ti_image)

Hinweis: Wenn Sie vorhandene Python-Programme in anderen Python-Entwicklungsumgebungen erstellt haben, müssen Sie diese möglicherweise bearbeiten, damit sie auf der Python-Lösung TI-Nspire™ laufen. Module können im Vergleich zu den TI-Modulen unterschiedliche Methoden, Argumente und Reihenfolge der Methoden in einem Programm verwenden. Achten Sie im Allgemeinen auf Kompatibilität, wenn Sie eine beliebige Version von Python und Python-Modulen verwenden.

Wenn Sie Python-Programme von einer Nicht-TI-Plattform auf eine TI-Plattform übertragen ODER von einem TI-Produkt zum anderen übertragen haben, denken Sie daran:

- Programme, die sprachliche Kernfunktionen und Standard-Libs (Mathematik, Zufallszahl usw.) verwenden, können ohne Änderungen portiert werden.
- Programme, die plattformspezifische Bibliotheken wie Matplotlib für PC oder TI-Module verwenden, müssen bearbeitet werden, bevor sie auf einer anderen Plattform ausgeführt werden können. Dies mag sogar zwischen TI-Plattformen so zutreffen.

Wie bei jeder Version von Python müssen Sie Importe einbinden, um alle Funktionen, Methoden oder Konstanten zu verwenden, die in einem bestimmten Modul enthalten sind. Um z. B. die Funktion cos() aus dem Mathematikmodul auszuführen, verwenden Sie die folgenden Befehle:

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

Eine Liste der Menüs mit ihren Punkten und Beschreibungen finden Sie im Abschnitt Menü-Übersicht.

Installation eines Python-Programms als Modul

So speichern Sie Ihr Python-Programm als Modul:

- Wählen Sie im Editor **Aktionen > Als Python-Modul installieren**.
- Wählen Sie in der Shell **Extras > Als Python-Modul installieren**.

Nach der Auswahl geschieht Folgendes:

- Die Python-Syntax wird überprüft.
- Die Datei wird gespeichert und in den Ordner PyLib verschoben.
- Es erscheint ein Dialog, der bestätigt, dass die Datei als Modul installiert wurde.
- Die Datei wird geschlossen und das Modul ist einsatzbereit.
- Der Modulname wird dem Menü **Weitere Module** als Menüpunkt **aus <module> Import*** hinzugefügt.

Wenn Sie vorhaben, dieses Modul an andere weiterzugeben, wird empfohlen, diese Richtlinien zu befolgen:

- Speichern Sie nur ein Modul pro TNS-Datei.

- Der Modulname stimmt mit dem Namen der TNS-Datei überein (z. B. Modul „my_ program“ ist in der Datei „my_program.tns“).
- Fügen Sie vor dem Python-Editor eine Seite „Notes“ ein, die den Zweck des Moduls, die Version und die Funktionen beschreibt.
- Verwenden Sie die Funktion ver(), um die Versionsnummer des Moduls anzuzeigen.
- (Optional) Fügen Sie eine Hilfefunktion hinzu, um die Liste der Methoden in der Funktion anzuzeigen.

Python-Arbeitsbereiche

Es gibt zwei Arbeitsbereiche für Ihre Python-Programmierung: Den Python-Editor und die Python Shell.

Python-Editor	Python Shell
• Python-Programme erstellen, bearbeiten und speichern • Syntaxhervorhebung und automatische Einrückung • Inline-Eingabeaufforderungen zur Führung mit Funktionsargumenten • Tooltipps zur Anzeige des gültigen Wertebereichs • Die Taste var listet globale Benutzervariablen und Funktionen auf, die im aktuellen Programm definiert sind • Tastenkürzel	• Python-Programme ausführen • Praktisch zum Testen kleiner Code-Fragmente • Interaktion mit der Shell-Historie zur Auswahl früherer Ein- und Ausgaben zur Wiederverwendung • Die Taste var listet globale Benutzervariablen auf, die im letzten ausgeführten Programm im gegebenen Problem definiert wurden

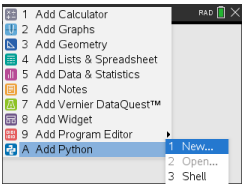
Hinweis: Zu einem Problem können mehrere Python-Programme und Shells hinzugefügt werden.

Python-Editor

Im Python-Editor können Sie Python-Programme erstellen, bearbeiten und speichern.

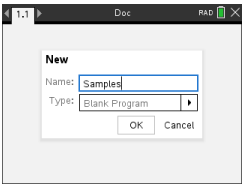
Hinzufügen einer Seite im Python-Editor

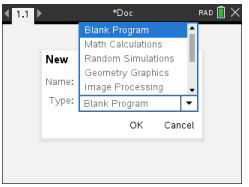
Um eine neue Seite im Python-Editor im aktuellen Problem hinzuzufügen, drücken Sie menu und wählen Sie **Python hinzufügen > Neu**.



Sie können ein leeres Programm anlegen, oder Sie können eine Vorlage auswählen.

Leeres Programm *Vorlage*

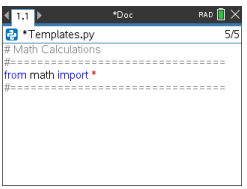




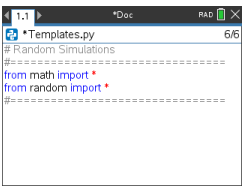
Nach dem Erstellen des Programms wird der Python-Editor angezeigt. Wenn Sie eine Vorlage ausgewählt haben, werden die erforderlichen Import-Anweisungen automatisch hinzugefügt (siehe unten).

Hinweis: Sie können mehrere Programme in einer einzigen TNS-Datei haben, genau wie andere Anwendungen. Wenn das Python-Programm als Modul verwendet werden soll, kann die TNS-Datei im PyLib-Ordner gespeichert werden. Dieses Modul kann dann in anderen Programmen und Dokumenten verwendet werden.

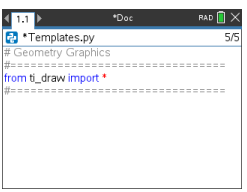
Mathematische Berechnungen



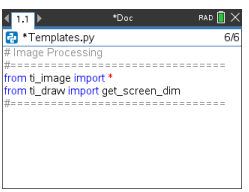
Zufallszahl-Simulationen



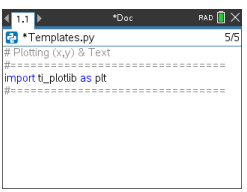
Geometrische Diagramme



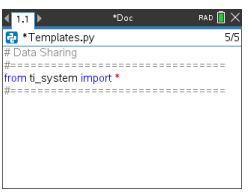
Bildverarbeitung



Grafische Darstellung (x,y) und Text

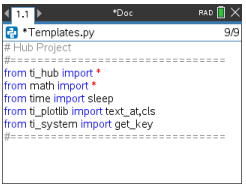


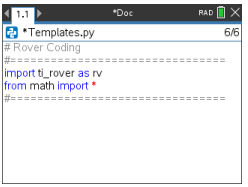
Datenaustausch



TI-Innovator™ Hub-Projekt

TI-Rover-Kodierung





Öffnen eines Python-Programms

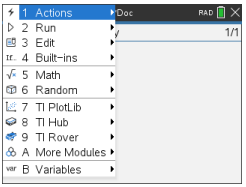
Um ein vorhandenes Python-Programm zu öffnen, drücken Sie doc und wählen dann **Einfügen > Python hinzufügen > Öffnen**. Dadurch wird eine Liste der Programme angezeigt, die in der TNS-Datei gespeichert wurden.

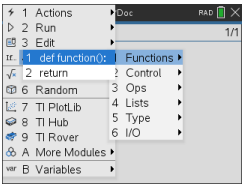
Wenn die zur Erstellung des Programms verwendete Editor-Seite gelöscht wurde, ist das Programm in der TNS-Datei weiterhin verfügbar.

Arbeiten im Python-Editor

Durch Drücken von menu wird das Menü Dokumentwerkzeuge angezeigt. Mit diesen Menüoptionen können Sie Codeblöcke für Ihr Programm hinzufügen, verschieben und kopieren.

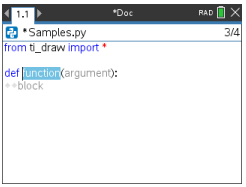
Menü „Dokumentwerkzeuge“





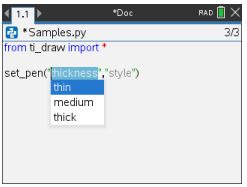
Aus den Modulmenüs ausgewählte Elemente fügen dem Editor automatisch eine Code-Vorlage mit Inline-Eingabeaufforderungen für jeden Teil der Funktion hinzu. Sie können von einem Argument zum nächsten navigieren, indem Sie tab (vorwärts) oder shift+tab (rückwärts) drücken. Wenn verfügbar, erscheinen Tooltipps oder Pop-up-Listen, die Ihnen bei der Auswahl der richtigen Werte helfen.

Inline-Eingabeaufforderungen

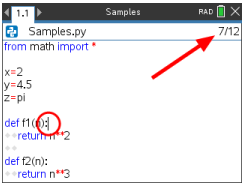


Tooltipps

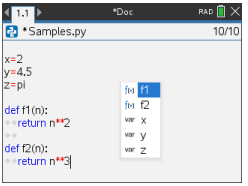
Pop-up-Listen



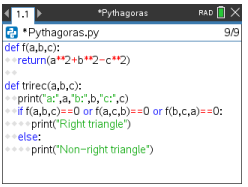
Die Zahlen rechts neben dem Programmnamen geben die aktuelle Zeilennummer des Cursors und die Gesamtanzahl der Zeilen im Programm an.



Globale Funktionen und Variablen, die in den Zeilen über der aktuellen Cursorposition definiert sind, können durch Drücken von var und Auswahl aus der Liste eingefügt werden.

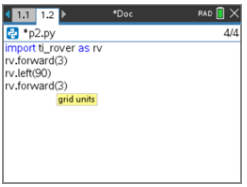


Wenn Sie Ihrem Programm Code hinzufügen, zeigt der Editor Schlüsselwörter, Operatoren, Kommentare, Zeichenfolgen und Einzüge in verschiedenen Farben an, um die Identifizierung der verschiedenen Elemente zu erleichtern.

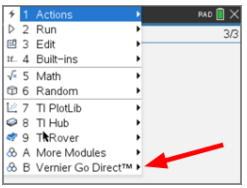


Eingefügte Menüpunkte werden je nach Kontext in der aktuellen Zeile oder in der nächsten Zeile abgelegt.

Vor v6.2.0 *v6.2.0 und höher*

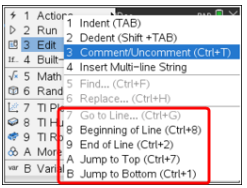


Im Hauptmenü wird das Modulmenü für das bearbeitete Programm angezeigt, um einen schnelleren Zugriff auf diese Optionen zu ermöglichen. Es wird jeweils nur ein Modul angezeigt.

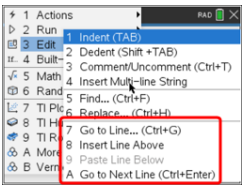


Im Menü „Bearbeiten“ werden häufig verwendete Menüpunkte und Tastenkürzel angezeigt.

Vor v6.2.0



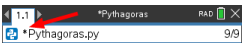
v6.2.0 und höher



Speichern und Ausführen von Programmen

Wenn Sie mit Ihrem Programm fertig sind, drücken Sie menu und wählen Sie **Ausführen > Syntax prüfen & Speichern**. Dadurch wird die Syntax des Python-Programms überprüft und in der TNS-Datei gespeichert.

Hinweis: Wenn Sie ungespeicherte Änderungen in Ihrem Programm vorgenommen haben, wird neben dem Programmnamen ein Sternchen angezeigt.



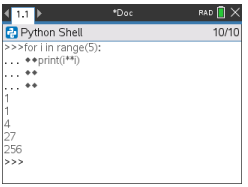
Drücken Sie zum Ausführen des Programms menu und wählen Sie **Ausführen > Ausführen**. Dadurch wird das aktuelle Programm auf der nächsten Seite der Python Shell ausgeführt oder ein neues Programm, wenn die nächste Seite keine Shell ist.

Hinweis: Beim Ausführen des Programms wird die Syntax automatisch geprüft und das Programm gespeichert.

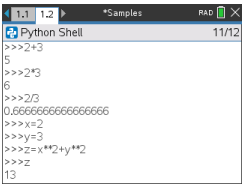
Python Shell

Die Python Shell ist der Interpreter, der Ihre Python-Programme, andere Teile des Python-Codes oder einfache Befehle ausführt.

Python-Code

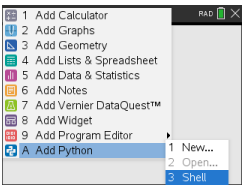


Einfache Befehle

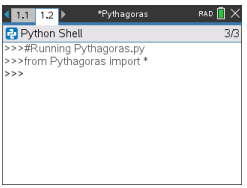


Hinzufügen einer Python Shell-Seite

Um eine neue Python Shell-Seite im aktuellen Problem hinzuzufügen, drücken Sie menu und wählen Sie **Python hinzufügen > Shell**.



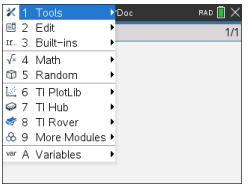
Die Python Shell kann auch aus dem Python-Editor heraus gestartet werden, indem ein Programm durch Drücken von menu und Auswahl von **Ausführen > Ausführen** ausgeführt wird.

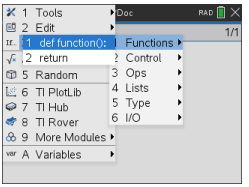


Arbeiten in der Python Shell

Durch Drücken von menu wird das Menü Dokumentwerkzeuge angezeigt. Mit diesen Menüoptionen können Sie Codeblöcke hinzufügen, verschieben und kopieren.

Menü „Dokumentwerkzeuge“

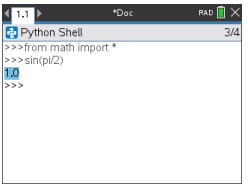




Hinweis: Wenn Sie eine Methode aus einem der verfügbaren Module verwenden, müssen Sie wie in jeder Python-Codierungsumgebung zuerst eine Import-Modul-Anweisung ausführen.

Die Interaktion mit der Shell-Ausgabe ist ähnlich wie bei der Rechneranwendung, wo Sie frühere Ein- und Ausgaben auswählen und kopieren können, um sie an anderer Stelle in der Shell, im Editor oder anderen Anwendungen zu verwenden.

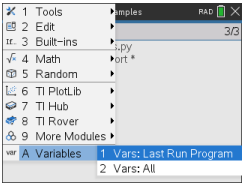
Pfeil nach oben zur Auswahl, dann kopieren und an der gewünschten Stelle einfügen



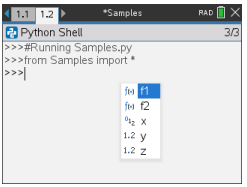
Globale Funktionen und Variablen aus dem zuletzt ausgeführten Programm können durch Drücken von var oder ctrl+L und Auswahl aus der Liste oder durch Drücken von menu und Auswahl von **Variablen > Vars: eingefügt werden Zuletzt ausgeführtes Programm**.

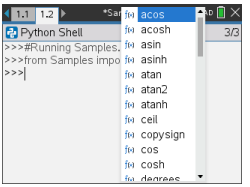
Um aus einer Liste von globalen Funktionen und Variablen sowohl aus dem zuletzt ausgeführten Programm als auch aus allen importierten Modulen auszuwählen, drücken Sie menu und wählen **Variablen > Vars: Alle**.

Menü Variablen



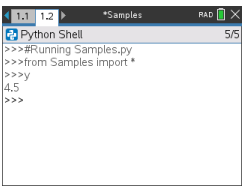
Variablen des zuletzt ausgeführten Programms *Alle Variablen*



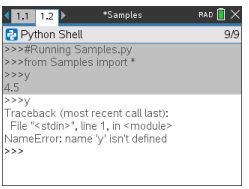


Alle Python Shell-Seiten in demselben Problem haben denselben Status (benutzerdefinierte und importierte Variablendefinitionen). Wenn Sie ein Python-Programm mit diesem Problem speichern oder ausführen oder menu drücken und **Werkzeuge > Shell neu initialisieren** auswählen, wird der Shell-Verlauf dann grau unterlegt, was anzeigt, dass der vorherige Zustand nicht mehr gültig ist.

Vor dem Speichern oder Reinitialisieren



Nach dem Speichern oder Reinitialisieren



Hinweis: Die Option menu **Werkzeuge > Verlauf löschen** löscht den Bildschirm jeder vorherigen Aktivität in der Shell, Variablen sind jedoch weiterhin verfügbar.

Meldungen

Während einer Python-Sitzung können Fehler- und andere Informationsmeldungen angezeigt werden. Wenn bei der Ausführung eines Programms ein Fehler in der Shell angezeigt wird, wird eine Programmzeilennummer angezeigt. Drücken Sie ctrl menu und wählen Sie **Zum Python-Editor gehen**. Drücken Sie im Editor menu und wählen Sie dann **Bearbeiten > Gehe zu Zeile**. Geben Sie die Zeilennummer ein und drücken Sie −. Der Cursor wird auf dem ersten Zeichen der Zeile angezeigt, in der der Fehler aufgetreten ist.

Unterbrechen eines laufenden Programms

Während ein Programm oder eine Funktion ausgeführt wird, wird das Symbol 'Beschäftigt' ⏲ angezeigt.

▶ Um das Programm bzw. die Funktion abzubrechen:

- Windows®: Drücken Sie die Taste **F12**.
- Mac®: Drücken Sie die Taste **F5**.
- Handheld: Drücken Sie die Taste on.

Python-Menü-Übersicht

In diesem Abschnitt werden alle Menüs und Menüpunkte des Python-Editors und der Shell mit einer kurzen Beschreibung aufgeführt.

Hinweis: Mac®-Benutzer sollten für die Menübefehle, die über Tastaturkürzel verfügen, ⌘ (**Cmd**) überall dort verwenden, wo **Ctrl** verwendet wird. Eine vollständige Liste der TI-Nspire™ Handheld- und Software-Verknüpfungen finden Sie im TI-Nspire™ Technology eGuide.

Menü Aktionen

Hinweis: Dies gilt nur für den Editor.

Menüpunkt	Funktion
Neu	Öffnet das Dialogfeld **Neu**, in dem Sie einen Namen eingeben und einen Typ für Ihr neues Programm auswählen.
Öffnen	Öffnet eine Liste der im aktuellen Dokument verfügbaren Programme.

Menüpunkt	Funktion
Kopie erstellen (Create Copy)	Öffnet das Dialogfeld **Kopie erstellen**, in dem Sie das aktuelle Programm unter einem anderen Namen speichern können.
Umbenennen	Öffnet das Dialogfeld **Umbenennen**, in dem Sie das aktuelle Programm umbenennen können.
Schließen	Schließt das aktuelle Programm.
Einstellungen	Öffnet das Dialogfeld **Einstellungen**, in dem Sie die Schriftgröße sowohl für den Editor als auch für die Shell ändern können.
Als Python-Modul installieren	Prüft die Python-Syntax der aktuellen TNS-Datei und verschiebt sie in den PyLib-Ordner.

Menü Ausführen

Hinweis: Dies gilt nur für den Editor.

Menüpunkt	Kürzel	Funktion
Starten	Strg+R	Prüft die Syntax, speichert das Programm und führt in einer Python Shell aus.
Syntax überprüfen und speichern	Strg+B	Prüft die Syntax und speichert das Programm.
Gehe zu Shell	nicht verfügb	Verschiebt den Fokus auf die Shell, die sich auf das aktuelle Programm bezieht, oder öffnet eine neue Shell-Seite neben dem Editor.

Menü Extras

Hinweis: Dies gilt nur für die Shell.

Menüpunkt	Kürzel	Funktion
Wiederholung des letzten Programms	Strg+R	Führt das letzte sich auf die aktuelle Shell beziehende Programm, erneut aus.
Zu Python-Editor wechseln	nicht verfügb	Öffnet die Editor-Seite, die sich auf die aktuelle Shell bezieht.
Starten	nicht verfügb	Öffnet eine Liste der im aktuellen Dokument verfügbaren Programme. Nach der Auswahl wird das gewählte Programm ausgeführt.

Menüpunkt	Kürzel	Funktion
Protokoll löschen	nicht verfügb	Löscht den Verlauf in der aktuellen Shell, aber initialisiert die Shell nicht neu.
Shell erneut initialisieren	nicht verfügb	Setzt den Status aller geöffneten Shell-Seiten im aktuellen Problem zurück. Alle definierten Variablen und importierten Funktionen sind nicht mehr verfügbar.
dir()	nicht verfügb	Zeigt eine Liste der Funktionen im angegebenen Modul an, wenn es nach der Import-Anweisung verwendet wird.
From PROGRAM import *	nicht verfügb	Öffnet eine Liste der im aktuellen Dokument verfügbaren Programme. Nach der Auswahl wird die Import-Anweisung in die Shell eingefügt.
Als Python-Modul installieren	nicht verfügb	Nur für Module im Binärformat aktiviert. Verschiebt die aktuelle TNS-Datei in den Ordner PyLib.

Menü „Bearbeiten“

Hinweis: Mit Strg+A werden alle Code- oder Ausgabezeilen zum Ausschneiden oder Löschen (nur Editor) bzw. Kopieren und Einfügen (Editor und Shell) ausgewählt.

Menüpunkt	Kürzel	Funktion
Einzug	TAB*	Rückt Text in der aktuellen Zeile oder in ausgewählten Zeilen ein. * Gibt es unvollständige Inline-Eingabeaufforderungen, navigieren Sie mit TAB zur nächsten Eingabeaufforderung.
Einzug entfernen	Umschalttaste+TAB**	Rückt Text in der aktuellen Zeile oder in ausgewählten Zeilen aus. * Gibt es unvollständige Inline-Eingabeaufforderungen, navigieren Sie mit TAB zur nächsten Eingabeaufforderung.
Kommentar/Kein Kommentar	Strg+T	Fügt Kommentarzeichen

Menüpunkt	Kürzel	Funktion
		zum Anfang der aktuellen Zeile hinzu oder entfernt es.
Mehrzeilige Zeichenfolge einfügen	nicht verfügbar	(Nur Editor) Fügt eine mehrzeilige Zeichenfolgenvorlage ein.
Suchen	Strg+F	(Nur Editor) Öffnet das Dialogfeld **Suchen** und sucht im aktuellen Programm nach der eingegebenen Zeichenfolge.
Ersetzen	Strg+H	(Nur Editor) Öffnet das Dialogfeld **Ersetzen** und sucht im aktuellen Programm nach der eingegebenen Zeichenfolge.
Gehe zu Zeile	Strg+G	(Nur Editor) Öffnet das Dialogfeld **Gehe zu Zeile** und springt zur angegebenen Zeile im aktuellen Programm.
Anfang der Zeile	Strg+8	Bewegt den Cursor an den Anfang der aktuellen Zeile.
Ende der Zeile	Strg+2	Bewegt den Cursor an das Ende der aktuellen Zeile.
Zum Anfang springen	Strg+7	Bewegt den Cursor an den Anfang der ersten Zeile im Programm.
Zum Ende springen	Strg+1	Bewegt den Cursor an das Ende der letzten Zeile im Programm.
Zeile löschen	Strg+Entf	Löscht die Zeile, in der sich der Cursor befindet.

Menü der Integrationen

Funktionen

Menüpunkt	Funktion
def function():	Definiert eine Funktion in Abhängigkeit von angegebenen Variablen.
return	Definiert den von einer Funktion erzeugten Wert.

Steuerung (Control)

Menüpunkt	Funktion
if..	Bedingte Anweisung.
if..else..	Bedingte Anweisung.
if..elif..else..	Bedingte Anweisung.
for index in range(size):	Iteriert über einen Bereich.
for index in range(start,stop):	Iteriert über einen Bereich.
for index in range(start,stop,step):	Iteriert über einen Bereich.
for index in list:	Iteriert über Listenelemente.
while..	Führt Anweisungen in einem Codeblock aus, bis eine Bedingung als False ausgewertet wird.
elif:	Bedingte Anweisung.
else:	Bedingte Anweisung.

Ops

Menüpunkt	Funktion
x=y	Legt den Variablenwert fest.
x==y	Einfügen von gleich (==) dem Vergleichsoperator.
x!=y	Einfügen von ungleich (!=) dem Vergleichsoperator.
x>y	Einfügen von größer als (>) der Vergleichsoperator.
x>=y	Einfügen von größer als der oder gleich dem (>=) Vergleichsoperator.
x<y	Einfügen von kleiner als (<) der Vergleichsoperator.

Menüpunkt	Funktion
x<=y	Einfügen von kleiner als der oder gleich dem (<=) Vergleichsoperator.
und	Einfügen von und (and) als logischer Operator.
— oder —	Einfügen von oder (or) als logischer Operator.
nicht, not	Einfügen von nicht (not) als logischer Operator.
Wahr	Fügt True Boolean-Wert ein.
Falsch	Fügt False Boolean-Wert ein.

Listen

Menüpunkt	Funktion
[]	Fügt Klammern ([]) ein.
list() (Listendifferenz)	Konvertiert die Sequenz in den Typ „Liste“.
len()	Gibt die Anzahl der Elemente der Liste zurück.
max() (Maximum)	Gibt den Maximalwert in der Liste zurück.
min() (Minimum)	Gibt den Minimalwert in der Liste zurück.
.append()	Die Methode hängt ein Element an eine Liste an.
.remove()	Die Methode entfernt die erste Instanz eines Elements aus einer Liste.
range(start,stop,step)	Gibt eine Reihe von Zahlen zurück.
for index in range(start,stop,step)	Dient zur Iteration über einen Bereich.
.insert()	Die Methode fügt ein Element an der angegebenen Position hinzu.
.split()	Die Methode gibt eine Liste mit Elementen zurück, die durch das angegebene Trennzeichen getrennt sind.
sum() (Summe)	Gibt die Summe der Elemente einer Liste zurück.
sorted()	Gibt eine sortierte Liste zurück.
.sort()	Die Methode sortiert eine vorhandene Liste.

Art

Menüpunkt	Funktion
int()	Gibt einen ganzzahligen Teil zurück.
float()	Gibt eine Gleitkommazahl zurück.
round(x,ndigits)	Gibt eine Gleitkommazahl zurück, die auf die angegebene Anzahl von Stellen gerundet wird.
str()	Gibt eine Zeichenkette zurück.
complex()	Gibt eine komplexe Zahl zurück.
type()	Gibt den Typ des Objekts zurück.

E/A

Menüpunkt	Funktion
print()	Zeigt Argument als Zeichenfolge an.
input()	Fordert den Benutzer zur Eingabe auf.
eval()	Wertet einen als Zeichenfolge dargestellten Ausdruck aus.
.format()	Die Methode formatiert die angegebene Zeichenfolge.

Math-Menü

Hinweis: Bei der Erstellung eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Mathematische Berechnungen** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
from math import *	Importiert alle Methoden (Funktionen) aus dem math-Modul.
fabs()	Gibt den absoluten Wert einer reellen Zahl zurück.
sqrt() (Quadratwurzel)	Gibt die Quadratwurzel einer reellen Zahl zurück.
exp() (e hoch x)	Gibt e**x zurück.
pow(x,y)	Gibt x hochgesetzt in die Potenz y zurück.
log(x,base)	Gibt $\log_{base}(x)$ zurück. log(x) ohne Basis gibt den natürlichen Logarithmus x zurück.
fmod(x,y)	Gibt den Modulwert von x und y zurück. Verwenden Sie diese Option, wenn x und y Fließkommazahlen sind.
ceil()	Gibt die kleinste Ganzzahl zurück, die größer oder gleich einer

Menüpunkt	Funktion
	reellen Zahl ist.
floor() (Untergrenze)	Gibt die größte Ganzzahl zurück, die kleiner oder gleich einer reellen Zahl ist.
trunc()	Beschneidet eine reelle Zahl zu einer ganzen Zahl.
frexp()	Gibt ein Paar (y,n) zurück, wobei x == y * 2**n.

Konst.

Menüpunkt	Funktion
e	Returns value for the constant e.
pi	Returns value for the constant pi.

Trigonometrie

Menüpunkt	Funktion
radians()	Wandelt Winkel in Grad in das Bogenmaß um.
degrees()	Wandelt Winkel im Bogenmaß in Grad um.
sin() (Sinus)	Gibt den Sinus des Arguments im Bogenmaß zurück.
cos() (Kosinus)	Gibt den Kosinus des Arguments im Bogenmaß zurück.
tan() (Tangens)	Gibt den Sinus des Arguments im Bogenmaß zurück.
asin()	Gibt den Arkussinus des Arguments im Bogenmaß zurück.
acos()	Gibt den Arkuskosinus des Arguments im Bogenmaß zurück.
atan()	Gibt den Arkustangens des Arguments im Bogenmaß zurück.
atan2(y,x)	Gibt den Arkustangens von y/x im Bogenmaß zurück.

Menü Zufallszahl

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Zufallssimulationen** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
from random import *	Importiert alle Methoden aus dem Zufallsmodul.
random()	Gibt eine Gleitkommazahl von 0 bis 1,0 zurück.

Menüpunkt	Funktion
uniform(min,max)	Gibt eine Zufallszahl x (float) zurück, so dass min <= x <= max.
randint(min,max)	Gibt eine zufällige ganze Zahl zwischen min und max zurück.
choice(sequence)	Gibt ein Zufallselement aus einer nicht leeren Sequenz zurück.
randrange(start,stop,step)	Gibt schrittweise eine Zufallszahl vom Start bis zum Stopp zurück.
seed()	Initialisiert den Zufallszahlengenerator.

TI PlotLib-Menü

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Plotting (x,y) & Text** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
import ti_plotlib as plt	Importiert alle Methoden (Funktionen) aus dem ti_plotlib-Modul im Namensraum „plt“. Daher wird allen aus den Menüs eingefügten Funktionsnamen „plt.“ vorangestellt.

EinSt

Menüpunkt	Funktion
cls()	Löscht die Plotting-Fläche.
grid(x-scale,y-scale,"style")	Zeigt ein Gitter mit angegebenem Maßstab für die x- und y-Achse an.
window(xmin,xmax,ymin,ymax)	Definiert das Plotting-Fenster, indem es das angegebene horizontale Intervall (xmin, xmax) und das vertikale Intervall (ymin, ymax) auf den zugeteilten Plotting-Bereich (Pixel) abbildet.
auto_window(x-list,y-list)	Automatische Skalierung des Plotting-Fensters zur Anpassung an die Datenbereiche innerhalb der x-Liste und y-Liste, die im Programm vor auto_ window() angegeben wurden.
axes("mode")	Zeigt Achsen im angegebenen Fenster im Plott-Bereich an.
labels("x-label","y-label",x,y)	Zeigt „x-label“- und „y-label“-Beschriftungen auf den Plott-Achsen an den Zeilenpositionen x und y

Menüpunkt	Funktion
	an.
title("title")	Zeigt den „Titel“ zentriert in der obersten Zeile des Fensters an.
show_plot()	Zeigt die gepufferte Zeichnungsausgabe an. Die Funktionen use_buffer() und show_plot() sind nützlich in Fällen, in denen die Anzeige mehrerer Objekte auf dem Bildschirm zu Verzögerungen führen könnte (in den meisten Fällen nicht notwendig).
use_buffer()	Aktiviert einen Off-Screen-Puffer, um das Zeichnen zu beschleunigen.

Sonderzeichen

Menüpunkt	Funktion
color(red,green,blue)	Legt die Farbe für alle folgenden Grafiken/Plotts fest.
cls()	Löscht die Plotting-Fläche.
show_plot()	Führt die Anzeige des Plots wie im Programm eingerichtet aus.
scatter(x-list,y-list,"mark")	Stellt eine Sequenz von geordneten Paaren aus (x-list, y-list) im angegebenen Markierungsstil dar.
plot(x-list,y-list,"mark")	Stellt eine Linie mit geordneten Paaren aus der angegebenen x-Liste und y-Liste dar.
plot(x,y,"mark")	Stellt einen Punkt unter Verwendung der Koordinaten x und y im angegebenen Markierungsstil dar.
line(x1,y1,x2,y2,"mode")	Stellt ein Liniensegment von (x1,y1) bis (x2,y2) dar.
lin_reg(x-list,y-list,"display")	Berechnet und zeichnet das lineare Regressionsmodell, ax+b, der x-list, y-list.
pen("size","style")	Legt das Aussehen aller folgenden Zeilen fest, bis der nächste pen() ausgeführt wird.
text_at(row,"text","align")	Zeigt „text“ im Plott-Bereich bei angegebenem „align“ an.

Eigenschaften

Menüpunkt	Funktion
xmin	Angegebene Variable für Fensterargumente, die als plt.xmin definiert sind.
xmax	Angegebene Variable für Fensterargumente, die als plt.xmax definiert sind.
ymin	Angegebene Variable für Fensterargumente, die als plt.ymin definiert sind.
ymax	Angegebene Variable für Fensterargumente, die als plt.ymax definiert sind.
m	Nachdem plt.linreg() in einem Programm ausgeführt wurde, werden die berechneten Werte von slope, m, und intercept, b, in plt.m und plt.b gespeichert.
b	Nachdem plt.linreg() in einem Programm ausgeführt wurde, werden die berechneten Werte von slope, a, und intercept, b, in plt.a und plt.b gespeichert.

Menü TI Hub

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Hub-Projekt** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
from ti_hub import *	Importiert alle Methoden aus dem ti_hub-Modul.

Im Hub eingebaute Geräte > Farbausgabe

Menüpunkt	Funktion
rgb(red,green,blue)	Legt die Farbe für die RGB-LED fest.
blink(frequency,time)	Legt die Blinkfrequenz und -dauer für die ausgewählte Farbe fest.
off()	Schaltet die RGB-LED aus.

Im Hub eingebaute Geräte > Lichtausgabe

Menüpunkt	Funktion
on()	Schaltet die LED ein.
off()	Schaltet die LED aus.
blink(frequency,time)	Legt die Blinkfrequenz und -dauer für die LED fest.

Im Hub eingebaute Geräte > Tonausgabe

Menüpunkt	Funktion
tone(frequency,time)	Spielt einen Ton in der angegebenen Frequenz für die angegebene Zeitdauer ab.
note("note",time)	Spielt die angegebene Notiz für die angegebene Zeit ab. Die Notiz wird unter Verwendung ihres Namens und einer Oktave festgelegt. Beispiel: A4, C5. Die Namen der Notizen sind C, CS, D, DS, E, F, FS, G, GS, A, AS und B. Die Oktavzahlen reichen von 1 bis 9 (einschließlich).
tone(frequency,time,tempo)	Spielt einen Ton in der angegebenen Frequenz für die angegebene Zeitdauer und im angegebenen Tempo ab. Das Tempo definiert die Anzahl der Pieptöne pro Sekunde im Bereich von 0 bis (einschließlich) 10.
note("note",time,tempo)	Spielt die angegebene Note für die angegebene Zeitdauer und im angegebenen Tempo ab. Die Notiz wird unter Verwendung ihres Namens und einer Oktave festgelegt. Beispiel: A4, C5. Die Namen der Notizen sind C, CS, D, DS, E, F, FS, G, GS, A, AS und B. Die Oktavzahlen reichen von 1 bis 9 (einschließlich). Die Tempozahlen reichen von 0 bis (einschließlich) 10.

Im Hub eingebaute Geräte > Helligkeitseingabe

Menüpunkt	Funktion
measurement()	Liest den eingebauten HELLIGKEITSSENSOR (Lichtniveau) aus und

Menüpunkt	Funktion
	gibt einen Messwert zurück. Der Standardbereich liegt zwischen 0 und 100. Dies kann mit der Funktion range() geändert werden.
range(min,max)	Legt den Bereich für die Abbildung der Messwerte vom Lichtniveausensor fest. Wenn beide fehlen oder auf den Wert „Keiner“ gesetzt sind, wird der Standardhelligkeitsbereich von 0 bis 100 eingestellt.

Eingabegerät hinzufügen

Dieses Menü enthält eine Liste der vom ti_hub-Modul unterstützten Sensoren (Eingabegeräte). Alle Menüpunkte fügen den Namen des Objekts ein und erwarten eine Variable und einen Port, die mit dem Sensor verwendet werden. Jeder Sensor verfügt über ein Messverfahren(), das den Wert des Sensors zurückgibt.

Menüpunkt	Funktion
DHT (Feuchtigkeits- und Temperatursensor)	Gibt eine Liste mit der aktuellen Temperatur und Feuchtigkeit sowie dem Sensortyp und dem letzten zwischengespeicherten Ablesestatus zurück.
Ranger	Gibt die aktuelle Abstandsmessung vom angegebenen Ultraschall Ranger zurück. • **measurement_time()** – Gibt die Zeit an, die das Ultraschallsignal benötigt, um das Objekt zu erreichen (die „Flugzeit“).
Lichtstufe	Gibt das Helligkeitsniveau vom Sensor für das externe Licht (Helligkeit) zurück.
Temperatur	Gibt den Temperaturmesswert vom externen Temperatursensor zurück. In der Standardkonfiguration wird der Seeed-Temperatursensor in den Anschlüssen IN 1, IN 2 oder IN 3 unterstützt. Um den Temperatursensor TI LM19 aus dem TI-Innovator™ Hub-Breadboard-Paket zu verwenden, bearbeiten Sie den Anschluss an den verwendeten BB-Pin und verwenden

Menüpunkt	Funktion
	Sie ein optionales Argument „TIANALOG“. Beispiel: mylm19 = temperature(„BB 5“, „TIANALOG“)
Feuchtigkeit	Gibt den Messwert des Feuchtigkeitssensors zurück.
Magnetisch	Erkennt das Vorhandensein eines Magnetfeldes. Der Schwellenwert zur Bestimmung des Vorhandenseins des Feldes wird durch die Funktion trigger() festgelegt. Der Standardwert des Schwellenwerts ist 150.
Vernier	Liest den Wert aus dem im Befehl angegebenen analogen Vernier-Sensor. Der Befehl unterstützt die folgenden Vernier-Sensoren: • **temperature** – Edelstahl-Temperatursensor. • **lightlevel** – TI Lichtniveausensor. • **pressure** – Original-Gasdrucksensor • **pressure** – Neuerer Gasdrucksensor. • **pH** – pH-Sensor. • **force10** – ±10 N Einstellung, Dualer Kraftsensor. • **force50** – ±50 N Einstellung, Dualer Kraftsensor. • **accelerometer** – Low-G Beschleunigungsmesser. • **generic** – Ermöglicht die Einstellung anderer Sensoren, die oben nicht direkt unterstützt werden, und die Verwendung der oben genannten API calibrate() zur Einstellung der Gleichungskoeffizienten.
Analog In	Unterstützt die Verwendung von generischen Geräten mit analogen Eingaben.

Menüpunkt	Funktion
Digital In	Gibt den aktuellen Status des digitalen Pins zurück, der mit dem DIGITALEN Objekt verbunden ist, oder den zwischengespeicherten Status des digitalen Ausgangswerts, der zuletzt für das Objekt EINGESTELLT wurde.
Potentiometer	Unterstützt einen Potentiometer-Sensor. Der Bereich des Sensors kann mit der Funktion range() geändert werden.
Thermistor	Liest Thermistor-Sensoren aus. Die voreingestellten Koeffizienten sind so ausgelegt, dass sie zu dem im Breadboard-Pack des TI-Innovator™ Hub enthaltenen Thermistor passen, wenn er mit einem 10 kΩ Festwiderstand verwendet wird. Ein neuer Satz von Kalibrierkoeffizienten und Referenzwiderstand für den Thermistor kann mit der Funktion calibrate() konfiguriert werden.
Lautstärke	Unterstützt Tonlautstärkesensoren.
Farbeingabe	Bietet Schnittstellen zu einem I2C-angeschlossenen Farbeingabesensor. Der Pin bb_port wird zusätzlich zum I2C-Port verwendet, um die LED auf dem Farbsensor zu steuern. • **color_number():** Gibt einen Wert von 1 bis 9 zurück, der die vom Sensor erkannte Farbe repräsentiert. Die Zahlen stellen die Farben gemäß der folgenden Abbildung dar: 1: Rot 2: Grün 3: Blau 4: Cyan 5: Magenta

Menüpunkt	Funktion
	6: Gelb 7: Schwarz 8: Weiß 9: Grau • **red():** Gibt einen Wert von 0 bis 255 zurück, der die Intensität des erkannten ROTEN Farbpegels darstellt. • **green():** Gibt einen Wert von 0 bis 255 zurück, der die Intensität des erkannten GRÜNEN Farbpegels darstellt. • **blue():** Gibt einen Wert von 0 bis 255 zurück, der die Intensität des erkannten BLAUEN Farbpegels darstellt. • **gray():** Gibt einen Wert von 0 bis 255 zurück, der die erkannte Graustufe darstellt, wobei 0 schwarz und 255 weiß ist.
BB-Port	Unterstützt die Verwendung aller 10 BB-Port-Pins als kombinierter digitaler Ein-/Ausgabe-Port. Die Initialisierungsfunktionen haben einen optionalen Parameter „Maske“, der die Verwendung der Teilmenge der 10 Pins erlaubt. • **read_port():** Liest die aktuellen Werte an den Eingangs-Pins des BB-Ports. • **write_port(value):** Setzt die Werte der Ausgangs-Pins auf den angegebenen Wert, wobei der Wert zwischen 0 und 1023 liegt. Beachten Sie, dass der Wert auch gegen den Maskenwert in der var=bbport(mask)-Operation angepasst wird, wenn eine Maske bereitgestellt wurde.
Hub-Zeit	Bietet Zugriff auf den internen Millisekunden-Timer.

Menüpunkt	Funktion
TI-RGB Array	Bietet Funktionen zur Programmierung des TI-RGB-Arrays. Die Initialisierungsfunktion akzeptiert einen optionalen „LAMP“-Parameter, um einen hochhellen Modus für das TI-RGB-Array zu aktivieren, der eine externe Stromversorgung erfordert. • **set(led_position, r,g,b):** Setzt eine bestimmte led_position (0–15) auf den angegebenen r,g,b-Wert, wobei r,g,b Werte von 0 bis 255 sind. • **set(led_list,red,green,blue):** Stellt die in der „led_list“ definierten LEDs auf die Farbe ein, die durch „rot“, „grün“ oder „blau“ angegeben ist. Die „led_list“ ist eine Python-Liste, die Indexe der LEDs von 0 bis 15 enthält. Zum Beispiel stellt die Einstellung ([0,2,4,6,15], 0, 0, 255) die LEDs 0, 2, 4, 6 und 15 auf blau ein. • **set_all(r,g,b):** Setzt alle RGB-LEDs im Array auf den gleichen r,g,b-Wert. • **all_off():** Schaltet alle RGBs im Array aus. • **measurement():** Gibt die ungefähre Stromaufnahme, die das RGB-Array vom TI-Innovator™ verwendet, in milliAmps zurück. • **pattern(pattern):** Wenn der Wert des Arguments als Binärwert im Bereich von 0 bis 65535 verwendet wird, werden Pixel eingeschaltet, in denen ein Wert von 1 in der Darstellung stehen würde. Die LEDs werden als ROT mit einem pwm-Wert von 255 eingeschaltet. • **pattern(value,red,green,blue):** Legt die durch das „Muster“ definierten LEDs auf die Farbe ein, die durch „rot“, „grün“ oder „blau“ angegeben ist.

Ausgabegerät hinzufügen

Dieses Menü enthält eine Liste der vom ti_hub-Modul unterstützten Ausgabegeräte. Alle Menüpunkte fügen den Namen des Objekts ein und erwarten eine Variable und einen Port, die mit dem Gerät verwendet werden.

Menüpunkt	Funktion
LED	Funktionen zur Steuerung extern angeschlossener LEDs.
RGB	Unterstützung für die Steuerung externer RGB-LEDs.
TI-RGB Array	Bietet Funktionen zur Programmierung des TI-RGB-Arrays.
Lautsprecher	Funktionen zur Unterstützung eines externen Lautsprechers mit dem TI-Innovator™ Hub. Die Funktionen sind die gleichen wie die für „Ton“ oben.
Leistung	Funktionen zur Steuerung der externen Stromversorgung mit dem TI-Innovator™ Hub. • **set(value):** Setzt die Leistungsstufe auf den angegebenen Wert zwischen 0 und 100. • **on():** Legt die Leistungsstufe auf 100 fest. • **off():** Legt die Leistungsstufe auf 0 fest.
Kontinuierlicher Servo	Funktionen zur Steuerung von kontinuierlichen Servomotoren. • **set_cw(speed,time):** Der Servo dreht sich im Uhrzeigersinn mit der angegebenen Geschwindigkeit (0–255) und für die spezifische Dauer in Sekunden. • **set_ccw(speed,time):** Der Servo dreht sich im Gegenuhrzeigersinn mit der angegebenen Geschwindigkeit (0–255) und für die spezifische Dauer in Sekunden. • **stop():** Stoppt den kontinuierlichen Servo.
Analog Out	Funktionen für die Verwendung von generischen Geräten mit analogen Eingängen.
Vibrationsmotor	Funktionen zur Steuerung von Vibrationsmotoren. • **set(val):** Setzt die Intensität des Vibrationsmotors auf „val“ (0–255) ein. • **off():** Schaltet den Vibrationsmotor aus. • **on():** Schaltet den Vibrationsmotor auf der höchsten Stufe ein.
Relais	Steuert Schnittstellen zur Steuerung von Relais. • **on():** Setzt das Relais in den EIN-Zustand. • **off():** Setzt das Relais in den AUS-Zustand.
Servo	Funktionen zur Steuerung von Servomotoren.

Menüpunkt	Funktion
	• **set_position(pos):** Stellt die Position des Sweep-Servos in einem Bereich von -90 bis +90 ein. • **zero():** Setzt den Sweep-Servo auf die Nullposition.
Squarewave	Funktionen zum Erzeugen einer Rechteckwelle. • **set(frequency,duty,time):** Stellt die Ausgangs-Rechteckwelle mit einem Standard-Tastverhältnis von 50 % (wenn kein Tastverhältnis angegeben ist) und einer durch die „Frequenz“ festgelegten Ausgangsfrequenz ein. Die Frequenz kann zwischen 1 und 500 Hz liegen. Der Arbeitszyklus, falls angegeben, kann zwischen 0 und 100 % betragen. • **off():** Schaltet die Rechteckwelle aus.
Digital Out	Schnittstellen zur Steuerung eines digitalen Ausgangs. • **set(val):** Legt den digitalen Ausgang auf den Wert fest, der durch „val“ (0 oder 1) festgelegt wurde. • **on():** Setzt den Status des digitalen Ausgangs auf hoch (1). • **off():** Setzt den Status des digitalen Ausgangs auf niedrig (0).
BB-Port	Bietet Funktionen zur Programmierung des TI-RGB-Arrays. Siehe die Einzelheiten oben.

Befehle

Menüpunkt	Funktion
sleep(seconds)	Unterbricht das Programm für die angegebene Anzahl von Sekunden. Importiert aus dem Modul „Zeit“.
text_at(row,"text","align")	Zeigt den angegebenen „Text“ im Plott-Bereich bei angegebenem „Align“ an. Teil des ti_plotlib-Moduls.
cls()	Löscht den Shell-Bildschirm zum Plotten. Teil des ti_plotlib-Moduls.
while get_key() != "esc":	Führt die Befehle in der „while“-Schleife aus, bis die „Esc“-Taste gedrückt wird.
get_key()	Gibt eine Zeichenfolge zurück, die die gedrückte Taste darstellt. Die Taste „1“ gibt „1“ zurück, „Esc“ gibt „esc“ zurück, und so weiter.

Menüpunkt	Funktion
	Wenn get_key() ohne Parameter aufgerufen wird, kehrt es sofort zurück. Wenn es mit einem Parameter – get_key(1) – aufgerufen wird, wartet es, bis eine Taste gedrückt wird. Teil des Moduls ti_system.

Ports

Dies sind die auf dem TI-Innovator™ Hub verfügbaren Ein- und Ausgabeports.

Menüpunkt
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

Menü TI Rover

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Rover Coding** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
import ti_rover as rv	Importiert alle Methoden (Funktionen) aus dem ti_rover-Modul im Namensraum „rv“. Folglich wird allen Funktionsnamen, die aus den Menüs eingefügt werden, ein „rv“ vorangestellt.

Antrieb

Menüpunkt	Funktion
forward(distance)	Bewegt den Rover um die angegebene Entfernung in Gittereinheiten vorwärts.
backward(distance)	Bewegt den Rover um die angegebene Entfernung in Gittereinheiten rückwärts.
left(angle_degrees)	Dreht den Rover um den angegebenen Winkel in Grad nach links.
right(angle_degrees)	Dreht den Rover um den angegebenen Winkel in Grad nach rechts.
stop()	Stoppt jede aktuelle Bewegung sofort.
stop_clear()	Stoppt jede aktuelle Bewegung sofort und löscht alle anstehenden Befehle.
resume()	Nimmt die Verarbeitung von Befehlen wieder auf.
stay(time)	Der Rover bleibt für die angegebene Zeitspanne in Sekunden an Ort und Stelle (optional). Wenn keine Zeit angegeben wird, bleibt der Rover 30 Sekunden lang stehen.
to_xy(x,y)	Bewegt den Rover zur Koordinatenposition (x,y) auf dem virtuellen Gitter.
to_polar(r,theta_degrees)	Bewegt den Rover zur Position der Polarkoordinaten (r, theta) auf dem virtuellen Gitter. Der Winkel wird in Grad angegeben.
to_angle(angle,"unit")	Dreht den Rover auf den angegebenen Winkel im virtuellen Gitter. Der Winkel ist relativ zu einem Nullwinkel, der im virtuellen Gitter auf der x-Achse nach unten zeigt.

Fahren > Fahren mit Optionen

Menüpunkt	Funktion
forward_time(time)	Bewegt den Rover für die angegebene Zeit vorwärts.
backward_time(time)	Bewegt den Rover für die angegebene Zeit rückwärts.
forward(distance,"unit")	Bewegt den Rover mit der Standardgeschwindigkeit für die angegebene Entfernung vorwärts. Der Abstand kann in Gittereinheiten, Metern oder Radumdrehungen angegeben werden.
backward(distance,"unit")	Bewegt den Rover mit der Standardgeschwindigkeit für die angegebene Entfernung rückwärts. Der Abstand kann in Gittereinheiten, Metern oder Radumdrehungen angegeben werden.
left(angle,"unit")	Dreht den Rover um den angegebenen Winkel nach links. Der Winkel kann in Grad, Bogenmaß oder Gradienten angegeben werden.
right(angle,"unit")	Dreht den Rover um den angegebenen Winkel in Grad nach rechts. Der Winkel kann in Grad, Bogenmaß oder Gradienten angegeben werden.
forward_time(time,speed,"rate")	Bewegt den Rover für die angegebene Zeit mit der angegebenen Geschwindigkeit vorwärts. Die Geschwindigkeit kann in Gittereinheiten/s, Meter/s oder Radumdrehungen/s angegeben werden.
backward_time(time,speed,"rate")	Bewegt den Rover für die angegebene Zeit mit der angegebenen Geschwindigkeit rückwärts. Die Geschwindigkeit kann in Gittereinheiten/s, Meter/s oder Radumdrehungen/s angegeben werden.
forward(distance,"unit",speed,"rate")	Bewegt den Rover um die angegebene Entfernung mit der angegebenen Geschwindigkeit vorwärts. Der Abstand kann in Gittereinheiten,

Menüpunkt	Funktion
	Metern oder Radumdrehungen angegeben werden. Die Geschwindigkeit kann in Gittereinheiten/s, Meter/s oder Radumdrehungen/s angegeben werden.
backward(distance,"unit",speed,"rate")	Bewegt den Rover um die angegebene Entfernung mit der angegebenen Geschwindigkeit rückwärts. Der Abstand kann in Gittereinheiten, Metern oder Radumdrehungen angegeben werden. Die Geschwindigkeit kann in Gittereinheiten/s, Meter/s oder Radumdrehungen/s angegeben werden.

Eingaben

Menüpunkt	Funktion
ranger_measurement()	Liest den Ultraschall-Abstandssensor an der Vorderseite des Rovers aus und gibt den aktuellen Abstand in Metern zurück.
color_measurement()	Gibt einen Wert von 1 bis 9 zurück, der die vorherrschende Farbe anzeigt, die vom Rover-Farbsensor „gesehen“ wird. 1 = Rot 2 = Grün 3 = Blau 4 = Zyan 5 = Magenta 6 = Gelb 7 = Schwarz 8 = Grau 9 = Weiß
red_measurement()	Gibt einen Wert zwischen 0 und 255 zurück, der den wahrgenommenen Rotwert angibt, der vom Farbsensor gesehen wird.
green_measurement()	Gibt einen Wert zwischen 0 und 255 zurück, der den wahrgenommenen Grünwert angibt, der vom Farbsensor gesehen wird.
blue_measurement()	Gibt einen Wert zwischen 0 und 255 zurück, der

Menüpunkt	Funktion
	den wahrgenommenen Blauwert angibt, der vom Farbsensor gesehen wird.
gray_measurement()	Gibt einen Wert zwischen 0 und 255 zurück, der den wahrgenommenen Grauwert angibt, der vom Farbsensor gesehen wird.
encoders_gyro_measurement()	Gibt eine Werteliste zurück, die die Zählungen vom linken und rechten Radsensor sowie den aktuellen Gyro-Kurs enthält.
gyro_measurement()	Gibt einen Wert zurück, der den aktuellen Gyro-Messwert, einschließlich Drift, in Grad angibt.
ranger_time()	Gibt die Zeit an, die das Ultraschallsignal des TI-Rover Rangers benötigt, um das Objekt zu erreichen (die „Flugzeit“).

Ausgaben

Menüpunkt	Funktion
color_rgb(r,g,b)	Stellt die Farbe der Rover RGB-LED auf die spezifischen Rot-, Grün- und Blauwerte ein.
color_blink(frequency,time)	Legt die Blinkfrequenz und -dauer für die ausgewählte Farbe fest.
color_off()	Schaltet die RGB-LED des Rovers aus.
motor_left(speed,time)	Setzt die linke Motorleistung für die angegebene Dauer auf den angegebenen Wert. Die Geschwindigkeit liegt im Bereich von -255 bis 255, mit 0 als Stopp. Positive Geschwindigkeitswerte bedeuten eine Drehung gegen den Uhrzeigersinn und negative Geschwindigkeitswerte im Uhrzeigersinn. Der optionale Zeitparameter, falls angegeben, hat einen gültigen Bereich von 0,05 bis 655,35 Sekunden. Wenn nicht angegeben, wird eine Voreinstellung von 5 Sekunden verwendet.
motor_right(speed,time)	Setzt die linke Motorleistung für die

Menüpunkt	Funktion
	angegebene Dauer auf den angegebenen Wert. Die Geschwindigkeit liegt im Bereich von -255 bis 255, mit 0 als Stopp. Positive Geschwindigkeitswerte bedeuten eine Drehung gegen den Uhrzeigersinn und negative Geschwindigkeitswerte im Uhrzeigersinn. Der optionale Zeitparameter, falls angegeben, hat einen gültigen Bereich von 0,05 bis 655,35 Sekunden. Wenn nicht angegeben, wird eine Voreinstellung von 5 Sekunden verwendet.
motors("ldir",left_val,"rdir",right_val,time)	Stellt das linke und rechte Rad für eine optionale Zeitspanne in Sekunden auf die angegebenen Geschwindigkeitsstufen ein. Die Geschwindigkeitswerte (left_val, right_val) liegen im Bereich von 0 bis 255, wobei 0 ein Stopp ist. Die Parameter ldir und rdir geben die CW- oder CCW-Drehung der jeweiligen Räder an. Der optionale Zeitparameter, falls angegeben, hat einen gültigen Bereich von 0,05 bis 655,35 Sekunden. Wenn nicht angegeben, wird eine Voreinstellung von 5 Sekunden verwendet.

Pfad

Menüpunkt	Funktion
waypoint_xythdrn()	Liest die x-Koordinate, y-Koordinate, Zeit, Kurs, zurückgelegte Entfernung, Anzahl der Radumdrehungen, Befehlsnummer des aktuellen Wegpunktes. Gibt eine Liste mit all diesen Werten als Elementen zurück.
waypoint_prev	Liest die x-Koordinate, y-Koordinate, Zeit, Kurs, zurückgelegte Entfernung, Anzahl der Radumdrehungen, Befehlsnummer des vorherigen Wegpunktes.
waypoint_eta	Gibt die geschätzte Zeit für die Fahrt zu einem Wegpunkt

Menüpunkt	Funktion
	zurück.
path_done()	Gibt einen Wert von 0 oder 1 zurück, je nachdem, ob sich der Rover bewegt (0) oder mit der gesamten Bewegung fertig ist (1).
pathlist_x()	Gibt eine Liste der X-Werte vom Anfang bis einschließlich des aktuellen X-Wertes des Wegpunktes zurück.
pathlist_y()	Gibt eine Liste der Y-Werte vom Anfang bis einschließlich des aktuellen Y-Wertes des Wegpunktes zurück.
pathlist_time()	Gibt eine Liste der Zeit in Sekunden vom Beginn bis einschließlich des aktuellen Wegpunkt-Zeitwerts zurück.
pathlist_heading()	Gibt eine Liste der Überschriften vom Anfang bis einschließlich des aktuellen Wertes der Wegpunkt-Überschrift zurück.
pathlist_distance()	Gibt eine Liste der von Anfang an zurückgelegten Entfernungen bis einschließlich des aktuellen Wegpunkt-Distanzwertes zurück.
pathlist_revs()	Gibt eine Liste der Anzahl der von Beginn an bis einschließlich des aktuellen Wegpunkt-Umdrehungswertes gemachten Umdrehungen zurück.
pathlist_cmdnum()	Gibt eine Liste der Befehlsnummern für den Pfad zurück.
waypoint_x()	Gibt die x-Koordinate des aktuellen Wegpunktes zurück.
waypoint_y()	Gibt die y-Koordinate des aktuellen Wegpunktes zurück.
waypoint_time()	Gibt die Zeit zurück, die auf der Reise vom vorherigen zum aktuellen Wegpunkt verbracht wurde.
waypoint_heading()	Gibt den absoluten Kurs des aktuellen Wegpunktes zurück.
waypoint_distance()	Gibt die zurückgelegte Entfernung zwischen dem vorherigen und dem aktuellen Wegpunkt zurück.
waypoint_revs()	Gibt die Anzahl der Umdrehungen zurück, die erforderlich sind, um zwischen dem vorherigen und dem aktuellen Wegpunkt zu reisen.

Einstellungen

Menüpunkt	Funktion
Einheiten/s	Option für die Geschwindigkeit in Gittereinheiten pro Sekunde.

Menüpunkt	Funktion
m/s	Option für die Geschwindigkeit in Metern pro Sekunde.
1/s	Option für die Geschwindigkeit in Radumdrehungen pro Sekunde.
Einheiten	Option für die Entfernung in Gittereinheiten.
m	Option für die Entfernung in Metern.
revs	Option für die Entfernung in Radumdrehungen.
Grad	Option zur Drehung in Grad.
Radianten	Option zur Drehung im Bogenmaß.
Neugrad	Option zur Drehung in Neugrad.
im Uhrzeigersinn	Option zur Angabe der Radrichtung.
gegen den Uhrzeigersinn	Option zur Angabe der Radrichtung.

Befehle

Diese Befehle sind eine Sammlung von Funktionen aus anderen Modulen sowie aus dem TI-Rover-Modul.

Menüpunkt	Funktion
sleep(seconds)	Unterbricht das Programm für die angegebene Anzahl von Sekunden. Importiert aus dem Modul „Zeit“.
text_at(row,"text","align")	Zeigt „text“ im Plott-Bereich bei angegebenem „align“ an. Importiert aus dem ti_plotlib-Modul.
cls()	Löscht den Shell-Bildschirm zum Plotten. Importiert aus dem ti_plotlib-Modul.
while get_key() != "esc":	Führt die Befehle in der „while“-Schleife aus, bis die „Esc“-Taste gedrückt wird.
wait_until_done()	Hält das Programm an, bis der Rover den aktuellen Befehl beendet hat. Dies ist eine hilfreiche Möglichkeit, Nicht-Rover-Befehle mit der Rover-Bewegung zu synchronisieren.
while not path_done()	Führt die Befehle in der „while“-Schleife aus, bis der Rover mit allen Bewegungen fertig ist. Die Funktion path_done() gibt einen Wert von 0 oder 1 zurück, je nachdem, ob sich der Rover bewegt (0) oder mit der gesamten Bewegung fertig ist (1).

Menüpunkt	Funktion
position(x,y)	Setzt die Rover-Position auf dem virtuellen Gitter auf die angegebene x,y-Koordinate.
position(x,y,heading,"unit")	Setzt die Rover-Position auf dem virtuellen Gitter auf die angegebene x,y-Koordinate, der virtuelle Steuerkurs wird relativ zur virtuellen x-Achse gesetzt, wenn ein Steuerkurs vorgesehen ist (in den angegebenen Einheiten für Winkel). Es wird angenommen, dass positive Winkel von 0 bis 360 entgegen dem Uhrzeigersinn von der positiven x-Achse aus betrachtet werden. Es wird angenommen, dass negative Winkel von 0 bis -360 gemäß dem Uhrzeigersinn von der positiven x-Achse aus betrachtet werden.
grid_origin()	Stellt RV als bei aktuellem Rasterursprungspunkt von (0,0) ein.
grid_m_unit(scale_value)	Setzt den virtuellen Gitterabstand in Metern pro Einheit (m/Einheit) auf den angegebenen Wert. 0,1 ist der Standardwert m/Einheit und entspricht 1 Einheit = 100 mm oder 10 cm oder 1 dm oder 0,1 m. Der Bereich des gültigen Skalenwertes reicht von 0,01 bis 10,0.
path_clear()	Löscht alle bereits existierenden Pfad- oder Wegpunktinformationen.
zero_gyro()	Setzt den Rover-Kreisel auf einen Winkel von 0,0 zurück und löscht die Zählung der linken und rechten Radsensoren.

Menü „Komplexe Mathematik“

Dieses Untermenü befindet sich unter **Weitere Module**.

Menüpunkt	Funktion
from cmath import *	Importiert alle Methoden aus dem cmath-Modul.
complex(real,imag)	Gibt eine komplexe Zahl zurück.
rect(modulus,argument)	Wandelt Polarkoordinaten einer komplexen Zahl in kartesische Koordinaten um.
.real	Gibt den Realteil der komplexen Zahl zurück.
.imag	Gibt den Imaginärteil einer komplexen Zahl zurück.
polar()	Wandelt kartesische Koordinaten einer komplexen Zahl in Polarkoordinaten um.

Menüpunkt	Funktion
phase()	Gibt den Phasenwinkel einer komplexen Zahl zurück.
exp() (e hoch x)	Gibt e**x zurück.
cos() (Kosinus)	Gibt den Kosinus einer komplexen Zahl zurück.
sin() (Sinus)	Gibt den Sinus einer komplexen Zahl zurück.
log() (Logarithmus)	Gibt den natürlichen Logarithmus einer komplexen Zahl zurück.
log10()	Gibt den Logarithmus zur Basis 10 einer komplexen Zahl zurück.
sqrt() (Quadratwurzel)	Gibt die Quadratwurzel einer komplexen Zahl zurück.

Menü Zeit

Dieses Untermenü befindet sich unter **Weitere Module**.

Menüpunkt	Funktion
from time import *	Importiert alle Methoden aus dem Zeitmodul.
sleep(seconds)	Unterbricht das Programm für die angegebene Anzahl von Sekunden.
clock()	Gibt die aktuelle Prozessorzeit ausgedrückt in Sekunden als Gleitkommazahl zurück.
localtime()	Wandelt eine in Sekunden ausgedrückte Zeit seit dem 1. Januar 2000 in ein Neuner-Tupel um, das Jahr, Monat, Monatstag, Stunde, Minute, Sekunde, Wochentag, Jahrestag und Sommerzeit-Indikator (DST) enthält. Wenn das optionale Argument (Sekunden) nicht angegeben wird, dann wird die Echtzeituhr verwendet.
ticks_cpu()	Gibt einen prozessorspezifischen aufsteigenden Millisekundenzähler mit willkürlichem Referenzpunkt zurück. Um Zeit konsistent über verschiedene Systeme hinweg zu messen, verwenden Sie ticks_ms().
ticks_diff()	Misst den Zeitraum zwischen aufeinanderfolgenden Aufrufen von ticks_cpu() oder ticks_ms(). Diese Funktion sollte nicht verwendet werden, um willkürlich lange Zeiträume zu messen.

TI System-Menü

Dieses Untermenü befindet sich unter **Weitere Module**.

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Datenfreigabe** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
from ti_system import *	Importiert alle Methoden (Funktionen) aus dem Modul ti_system.
recall_value("name")	Ruft eine vordefinierte BS-Variable (Wert) namens „name“" auf.
store_value("name",value)	Speichert eine Python-Variable (Wert) in einer BS-Variablen namens „name“.
recall_list("name")	Ruft eine vordefinierte BS-Liste namens „name“ auf.
store_list("name",list)	Speichert eine Python-Liste (Liste) in einer BS-Listenvariablen namens "name".
eval_function("name",value)	Bewertet eine vordefinierte BS-Funktion mit dem angegebenen Wert.
get_platform()	Gibt „hh“ für Handheld und „dt“ für Desktop zurück.
get_key()	Gibt eine Zeichenfolge zurück, die die gedrückte Taste darstellt. Die Taste „1“ gibt „1“ zurück, „Esc“ gibt „esc“ zurück, und so weiter. Wenn get_key() ohne Parameter aufgerufen wird, kehrt es sofort zurück. Wenn es mit einem Parameter – get_key(1) – aufgerufen wird, wartet es, bis eine Taste gedrückt wird.
get_mouse()	Gibt Mauskoordinaten als Tupel mit zwei Elementen zurück, entweder die Pixelposition der Zeichenfläche oder (-1,-1), wenn sie sich außerhalb der Zeichenfläche befindet.
while get_key() != "esc":	Führt die Befehle in der „while“-Schleife aus, bis die „Esc“-Taste gedrückt wird.
clear_history()	Löscht den Shell-Verlauf.
get_time_ms()	Gibt die Zeit in Millisekunden mit einer Genauigkeit im Millisekundenbereich zurück. Diese Funktionalität kann dazu verwendet werden, eine Dauer zu berechnen, anstatt die tatsächliche Uhrzeit zu ermitteln.

Menü TI Draw

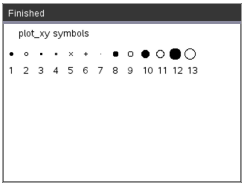
Dieses Untermenü befindet sich unter **Weitere Module**.

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Geometry Geographics** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
from ti_draw import *	Importiert alle Methoden aus dem Modul ti_draw.

Form

Menüpunkt	Funktion
draw_line()	Zeichnet eine Linie von der angegebenen x1,y1-Koordinate bis x2,y2.
draw_rect()	Zeichnet ein Rechteck beginnend an der angegebenen x,y-Koordinate mit der angegebenen Breite und Höhe.
fill_rect()	Zeichnet ein Rechteck, das an der angegebenen x,y-Koordinate mit der angegebenen Breite und Höhe beginnt und mit der angegebenen Farbe gefüllt wird (unter Verwendung von set_color oder schwarz, falls nicht definiert).
draw_circle()	Zeichnet einen Kreis, der an der angegebenen x,y-Mittelpunktkoordinate mit dem angegebenen Radius beginnt.
fill_circle()	Zeichnet einen Kreis, der an der angegebenen x,y-Mittelpunktkoordinate mit dem angegebenen Radius beginnt und mit der angegebenen Farbe ausgefüllt wird (unter Verwendung von set_color oder schwarz, falls nicht definiert).
draw_text()	Zeichnet eine Textzeichenfolge, die an der angegebenen x,y-Koordinate beginnt.
draw_arc()	Zeichnet einen Bogen, der an der angegebenen x,y-Koordinate mit der angegebenen Breite, Höhe und den angegebenen Winkeln beginnt.
fill_arc()	Zeichnet einen Bogen, der an der angegebenen x,y-Koordinate mit der angegebenen Breite, Höhe und den angegebenen Winkeln beginnt und mit der angegebenen Farbe gefüllt ist (unter Verwendung von set_color oder Schwarz, falls nicht definiert).
draw_poly()	Zeichnet ein Polygon unter Verwendung der angegebenen Werte für x-list,y-list.
fill_poly()	Zeichnet ein Polygon unter Verwendung der angegebenen Werte für x-list,y-list, das mit der angegebenen Farbe gefüllt ist (unter Verwendung von set_color oder schwarz, falls nicht definiert).

Menüpunkt	Funktion
plot_xy()	Zeichnet eine Form unter Verwendung der angegebenen x,y-Koordinate und der angegebenen Zahl von 1–13, die verschiedene Formen und Symbole darstellt (siehe unten). 

Steuerung (Control)

Menüpunkt	Funktion
clear()	Löscht den gesamten Bildschirm. Kann mit den Parametern x,y,Breite,Höhe verwendet werden, um ein vorhandenes Rechteck zu löschen.
clear_rect()	Löscht das Rechteck an der angegebenen x,y-Koordinate mit der angegebenen Breite und Höhe.
set_color()	Legt die Farbe der Form(en) fest, die im Programm folgen, bis eine andere Farbe eingestellt wird.
set_pen()	Legt die angegebene Dicke und den Stil des Randes beim Zeichnen von Formen fest (nicht anwendbar bei Verwendung von Füllbefehlen).
set_window()	Legt die Größe des Fensters fest, in dem beliebige Formen gezeichnet werden sollen. Diese Funktion ist nützlich, um die Größe des Fensters an die Daten anzupassen oder um den Ursprung (0,0) der Zeichenfläche zu ändern.
get_screen_dim()	Gibt xmax und ymax der Bildschirmabmessungen zurück.
use_buffer()	Aktiviert einen Off-Screen-Puffer, um das Zeichnen zu beschleunigen.
paint_buffer()	Zeigt die gepufferte Zeichnungsausgabe an. Die Funktionen „use_buffer()“ und „paint_buffer()“ sind in Fällen nützlich, in denen die Anzeige mehrerer Objekte auf dem Bildschirm zu Verzögerungen führen könnte.

Notes

- Die Standardkonfiguration hat (0,0) in der oberen linken Ecke des Bildschirms. Die positive x-Achse zeigt nach rechts und die positive y-Achse zeigt nach unten. Dies kann durch Verwendung der Funktion „set_window()“ geändert werden.
- Die Funktionen im ti_draw-Modul sind nur auf dem Handheld und in der Handheld-Ansicht auf dem Desktop verfügbar.

Menü TI Image

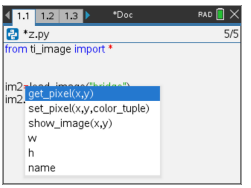
Dieses Untermenü befindet sich unter **Weitere Module**.

Hinweis: Beim Erstellen eines neuen Programms, das dieses Modul verwendet, wird empfohlen, den Programmtyp **Bildverarbeitung** zu verwenden. Dadurch wird sichergestellt, dass alle relevanten Module importiert werden.

Menüpunkt	Funktion
from ti_image import *	Importiert alle Methoden aus dem ti_image-Modul.
new_image(width,height,(r,g,b))	Erstellt ein neues Bild mit der angegebenen Breite und Höhe zur Verwendung im Python-Programm. Die Farbe des neuen Bildes wird durch die (r,g,b)-Werte definiert.
load_image("name")	Lädt das durch „name“ angegebene Bild zur Verwendung im Python-Programm. Das Bild muss Teil des TNS-Dokuments entweder in einer Notes- oder Graphs-Anwendung sein. Die „name“-Eingabeaufforderung zeigt die Bildnamen (falls sie bereits früher genannt wurden) oder eine Zahl, die die Einfügereihenfolge angibt.
copy_image(image)	Erstellt eine Kopie des Bildes, das durch die „image“-Variable festgelegt wurde.

Methoden des Bildobjekts

Zusätzliche Funktionen, die sich auf die Bildobjekte beziehen, sind im Editor und in der Shell verfügbar, indem Sie den Variablennamen gefolgt von einem . (Punkt) eingeben.



- **get_pixel(x,y):** Ruft den (r,g,b)-Wert des Pixels an der durch das (x,y)-Koordinatenpaar definierten Stelle ab.

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

- **set_pixel(x,y,color_tuple):** Setzt das Pixel an der Position (x,y) auf die in „color_ tuple“ angegebene Farbe.

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

  Setzt das Pixel bei (100,100) auf die Farbe (0,0,255).

- **show_image(x,y):** Zeigt das Bild mit der oberen linken Ecke an Position (x,y) an.
- **w, h, name:** Ruft die Parameter Breite, Höhe und Namen des Bildes ab.

Zum Beispiel

```
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)
```

Menü Variablen

Hinweis: Diese Listen enthalten keine Variablen, die in anderen TI-Nspire™-Anwendungen definiert sind.

Menüpunkt	Funktion
Vars: Current Program	(Nur Editor) Zeigt eine Liste der im aktuellen Programm definierten globalen Funktionen und Variablen an
Variable: Zuletzt gelaufenes Programm	(Nur Shell) Zeigt eine Liste der globalen Funktionen und Variablen an, die im zuletzt ausgeführten Programm definiert wurden
Variable: Alles	(Nur Shell) Zeigt eine Liste der globalen Funktionen und Variablen sowohl des zuletzt ausgeführten Programms als auch aller importierten Module an

Anhang

Python-Schlüsselwörter

Die folgenden Schlüsselwörter werden in die TI-Nspire™ Python-Implementierung integriert.

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

Python Tastenzuordnung

Beim Eingeben des Codes im Editor oder in der Shell ist die Tastatur so konzipiert, dass sie die entsprechenden Python-Operationen oder offenen Menüs einfügen, um Funktionen, Schlüsselwörter, Methoden, Operatoren usw. zu öffnen.

Prüfung	Zuordnung
var	Öffnet das Menü Variablen
sto→	Einfügen von = Zeichen
del ←	Löscht das Zeichen links vom Cursor
clear	Keine Aktion
=	Einfügen von = Zeichen
|≠≥▸	Fügt das/die ausgewählten Symbol(e) ein: • >= • <=

Prüfung	Zuordnung
	• != • >= • <= • == • und • — oder — • nicht, not • | • & • ~
trig	Fügt die ausgewählte Funktion ein: • sin • cos • tan • atan2 • asin • acos • atan
?	Zeigt Hinweise an
:=	Einfügen von: =
^	Einfügen von **
n/q	Keine Aktion
x²	Einfügen von **2
√	Einfügen von sqrt()
×	Einfügen von vielen Zeichen (*)
""	Einfügen eines doppelten Apostrophs (")
÷	Einfügen eines Divisions-Zeichens (/)
□/□	Keine Aktion
eˣ	Einfügen von exp()
ln	Einfügen von log()
10ˣ	Einfügen von 10**
log	Einfügen von log(Wert,Basis)
(	Einfügen von (

Prüfung	Zuordnung
)	Einfügen von)
[]	Einfügen von [ ]
{}	Einfügen von { }
(-)	Einfügen eines Subtraktions-Zeichens (-)
≈	Fügt eine neue Zeile nach der aktuellen Zeile ein
EE	Einfügen von E
?!	Fügt das/die ausgewählten Symbol(e) ein: • ? • ! • $ • ° • ' • % • " • : • ; • _ • \ • #
π	Einfügen von „pi“
⚐	Bestehendes Flag-Verhalten
↵	Fügt eine neue Zeile nach der aktuellen Zeile ein

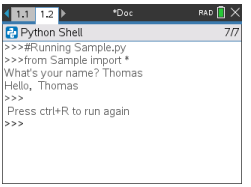
Python-Beispielprogramme

Verwenden Sie die folgenden Beispielprogramme, um sich mit Python-Methoden vertraut zu machen. Sie sind auch in der Datei **Getting Started Python.tns** verfügbar, die sich im Ordner **Beispiele** befindet.

Hinweis: Wenn Sie Beispielcode, der Indikatoren zur Tabulatoreinrückung (• •) enthält, kopieren und in die TI-Nspire™-Software einfügen, müssen Sie diese Indikatoren durch tatsächliche Tabulatoreinrückungen ersetzen.

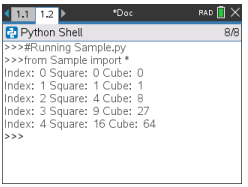
Hallo

```
# 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")
```


1.1
1.2
*Doc
RAD
Python Shell
7/7
>>>#Running Sample.py
>>>from Sample import *
What's your name? Thomas
Hello, Thomas
>>>
Press ctrl+R to run again
>>>

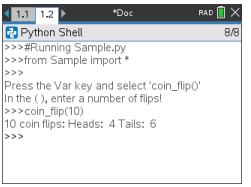
Schleifenbeispiel

```
# 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)
```



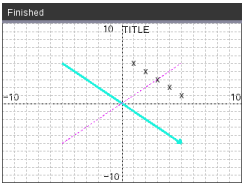
Kopf oder Zahl

```
# 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!")
```



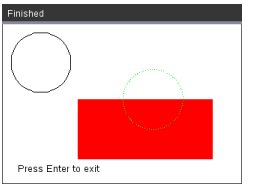
Plotten

```
# 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")
```



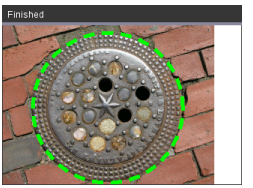
Zeichnen

```
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")
```



Bild

```
# 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

Dieses Programm verwendet Python, um den TI-Innovator™ Hub, einen programmierbaren Mikrocontroller, zu steuern. Wenn Sie das Programm ausführen, ohne einen TI-Innovator™ Hub anzuschließen, wird eine Fehlermeldung angezeigt.

Weitere Informationen über TI-Innovator™ Hub finden Sie unter 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()
```

Allgemeine Informationen

Online-Hilfe

education.ti.com/eguide

Wählen Sie Ihr Land aus, um weitere Produktinformationen zu erhalten.

Kontakt mit TI Support aufnehmen

education.ti.com/ti-cares

Wählen Sie Ihr Land aus, um auf technische und sonstige Support-Ressourcen zuzugreifen.

Service- und Garantieinformationen

education.ti.com/warranty

Wählen Sie für Informationen zur Dauer und den Bedingungen der Garantie bzw. zum Produktservice Ihr Land aus.

Eingeschränkte Garantie. Diese Garantie hat keine Auswirkungen auf Ihre gesetzlichen Rechte.

Texas Instruments Incorporated

12500 TI Blvd.

Dallas, TX 75243