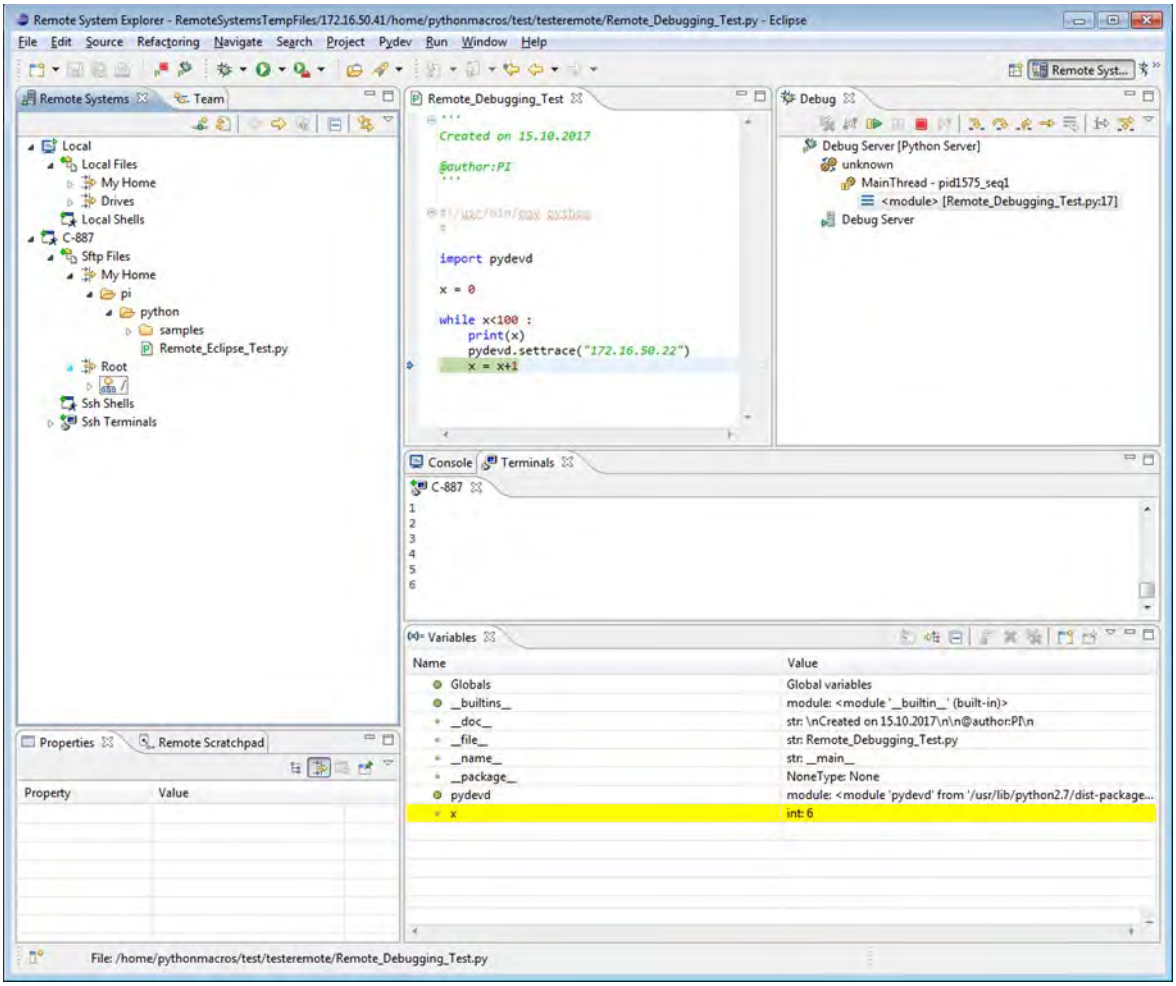
Benutzerhandbuch

SM157D, gilt für PI-Controller mit PIPython
FRIE, ASt, 17/8/2021

PIPython

Programmierung von PI Controllern mit Python



Physik Instrumente (PI) GmbH & Co. KG, Auf der Römerstr. 1, 76228 Karlsruhe, Deutschland
Tel. +49 721 4846-0, Fax +49 721 4846-1019, E-Mail info@pi.de, http://www.pi.de

Benutzerhandbuch

SM157D, gilt für PI-Controller mit PIPython
FRIE, ASt, 17/8/2021

Inhalt

Benutzerhandbuch

SM157D, gilt für PI-Controller mit PIPython
FRIE, ASt, 17/8/2021

Über dieses Dokument

PI bietet Controller an, die mit PIPython ausgestattet sind. Zur automatisierten Ansteuerung solcher Controller können neben den herkömmlichen Makros mit GCS-Befehlen auch Python-Skripte verwendet werden.

Dieses Dokument beschreibt die Arbeit mit Python auf PI-Controllern, die mit PIPython ausgestattet sind. Die enthaltenen Code-Beispiele beziehen sich auf die Python-Version 3.

Symbole und Kennzeichnungen

Folgende Symbole und Kennzeichnungen werden im Dokument verwendet:

Symbol	Bedeutung
1. 2.	Handlung mit mehreren Schritten, deren Reihenfolge eingehalten werden muss
➢	Handlung mit einem Schritt oder mehreren Schritten, deren Reihenfolge nicht relevant ist
▪	Aufzählung
S. 5	Querverweis auf Seite 5
Start > Einstellungen	Menüpfad in der PC-Software (Beispiel: Zum Aufrufen des Menüs muss nacheinander auf die Schaltflächen Start und Einstellungen geklickt werden)
`print`	Befehlszeile oder Befehl in einem Skript
5	Wert, der über die PC-Software eingegeben bzw. ausgewählt werden muss

Abbildungen

Zugunsten eines besseren Verständnisses können Farbgebung, Größenverhältnisse und Detaillierungsgrad in Illustrationen von den tatsächlichen Gegebenheiten abweichen. Auch fotografische Abbildungen können abweichen und stellen keine zugesicherten Eigenschaften dar.

Hinweise zu Copyright und Warenzeichen

In Software-Produkten von PI können Software-Komponenten von Drittanbietern eingebunden sein oder verwendet werden. Weitere Informationen finden Sie in der Third Party Software Note auf unserer Webseite.

Ebenfalls auf unserer Webseite finden Sie das General Software License Agreement.

Haftungsausschluss

Für Inhalte oder Links der in diesem Dokument genannten Webseiten Dritter wird keine Haftung übernommen. PI distanziert sich eindeutig von allen extern verlinkten Webseiten, deren Gestaltung, Daten und Inhalten sowie der damit verbundenen Haftung. Für den Inhalt der verlinkten Webseiten sind ausschließlich deren Betreiber verantwortlich.

PI übernimmt keine Gewähr für die Korrektheit der angegebenen Links zu Webseiten Dritter.

Übersicht PIPython

Bestimmungsgemäße Verwendung

PIPython wird von einigen PI Controllern mit Linux-Betriebssystem zur Verfügung gestellt. PIPython ist dafür vorgesehen, Controller und damit auch die angeschlossenen Positioniersysteme über Skripte automatisiert anzusteuern.

Der Einsatz von Python als Programmiersprache bietet vielfältige Möglichkeiten. PI bietet mit PIPython eine Sammlung an Python-Modulen, um PI Geräte und GCS-Daten komfortabel zu nutzen. Diese Module können mit Python ab Version 3.6 auf Windows, Linux, OS X und über Sockets auch auf anderen Betriebssystemen eingesetzt werden.

Es besteht, vergleichbar zu GCS-Makros, die Möglichkeit, Python-Skripte sowohl auf dem PC als auch direkt auf dem Controller zu verwenden.

PIPython darf nicht für andere als die in diesem Benutzerhandbuch genannten Zwecke verwendet werden.

Verzeichnisstruktur auf dem Controller

Auf dem Controller sind für PIPython folgende Verzeichnisse relevant:

- Home-Verzeichnis: /home/piuser/
- Verzeichnis für Python-Skripte: /home/piuser/python/

Zusätzlich gibt es ein Verzeichnis für Sample-Skripte: /home/piuser/python/samples/

Das Modul "PIPython" befindet sich in folgendem Verzeichnis: /home/piuser/python/pipython/

Zugangsdaten für PIPython

Zur Übertragung und Verwaltung von Dateien (Skripten) für PIPython wird das SFTP-Protokoll verwendet. Dafür werden folgende Zugangsdaten benötigt (Groß-/Kleinschreibung beachten):

- Benutzername: piuser
- Passwort: PI2020user!

Voraussetzungen auf dem PC

Zur Verwendung von PIPython muss auf dem PC eine Python-Installation vorhanden sein (Download z. B. über https://www.python.org/downloads/). Zusätzlich muss PIPython installiert werden.

Installation von PIPython

- Installieren Sie PIPython auf dem PC.
  - Das Modul befindet sich auf der PI Software-CD. Hinweise zur Installation finden Sie in der Datei "readme.rst".
  - Zur Installation von PIPython über den Python Package-Installer geben Sie im Kommandofenster des PC folgenden Befehl ein: `pip install pipython`

Für das PC-basierte Arbeiten wird empfohlen, zusätzlich die PI_GCS2_DLL.dll zu installieren:

- Installieren Sie PI_GCS2_DLL.dll auf dem PC.

  Das entsprechende Setup befindet sich auf der PI Software-CD.

Verwendung einer Entwicklungsumgebung

Für ein komfortables Arbeiten mit PIPython wird empfohlen, eine Entwicklungsumgebung (IDE, Integrated Development Environment = integrierte Entwicklungsumgebung) einzusetzen.

In diesem Dokument wird die Arbeit mit PIPython für zwei verbreitete IDEs beschrieben:

- **Eclipse** (S. 6)
  als Open-Source Lösung
- **PyCharm Professional** (S. 16)
  als kommerzielles Produkt
  (Anmerkung: Der kostenfreien PyCharm Community Edition fehlt die Möglichkeit des Remote Debugging.)

Mit beiden IDEs ist es einfach möglich, eigene Python-Skripte zu entwickeln und zu starten. Für den Einsatz direkt auf dem Controller ist es zudem möglich, die Skripte dorthin zu übertragen und sie vom PC aus zu debuggen.

Verwendung von Eclipse

Installation von Eclipse

Wollen Sie Eclipse für die Arbeit mit PIPython verwenden, führen Sie auf dem PC folgende Installationen durch:

1. IDE: **Eclipse** (Download z. B. über https://eclipse.org/downloads/)

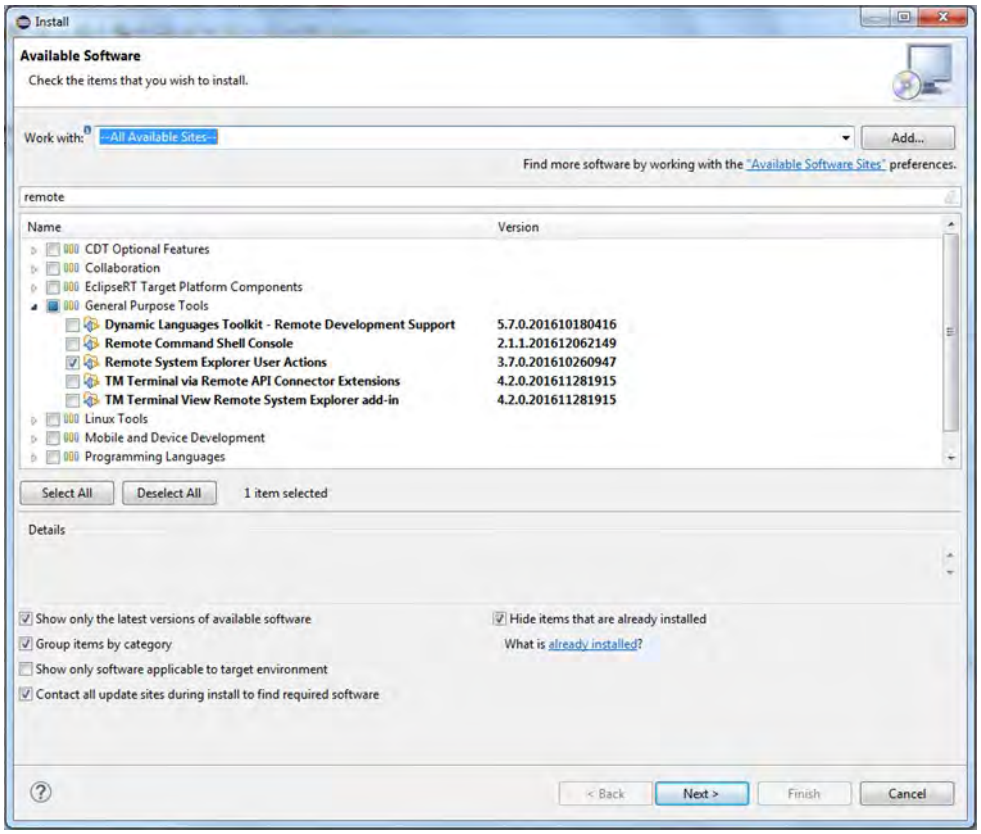
   Alle folgenden Beschreibungen und Abbildungen beziehen sich auf die Version Neon.2 (4.6.2) unter Windows.

2. Plug-In: **PyDev** (Download z. B. über http://www.pydev.org/)

   Eine detaillierte Beschreibung dieses Open Source Plug-Ins sowie Informationen zur Installation finden Sie auf der entsprechenden Webseite (z. B. http://www.pydev.org/manual_101_install.html).

3. Um PIPython auf dem Controller vollständig nutzen zu können, muss zusätzlich der **Remote System Explorer (RSE)** für Eclipse installiert werden.

   Dieses Plug-In ist nicht direkt im "Eclipse marketplace" sondern über die "Available Software Sites" und den Filter "remote" verfügbar (***Help > Install New Software***).



4. Abhängig von der eingesetzten Version von Eclipse wird zusätzlich empfohlen, zwei weitere Plug-Ins zu installieren: **TM Terminal** sowie das **TM Terminal View Remote System Explorer add-in**.

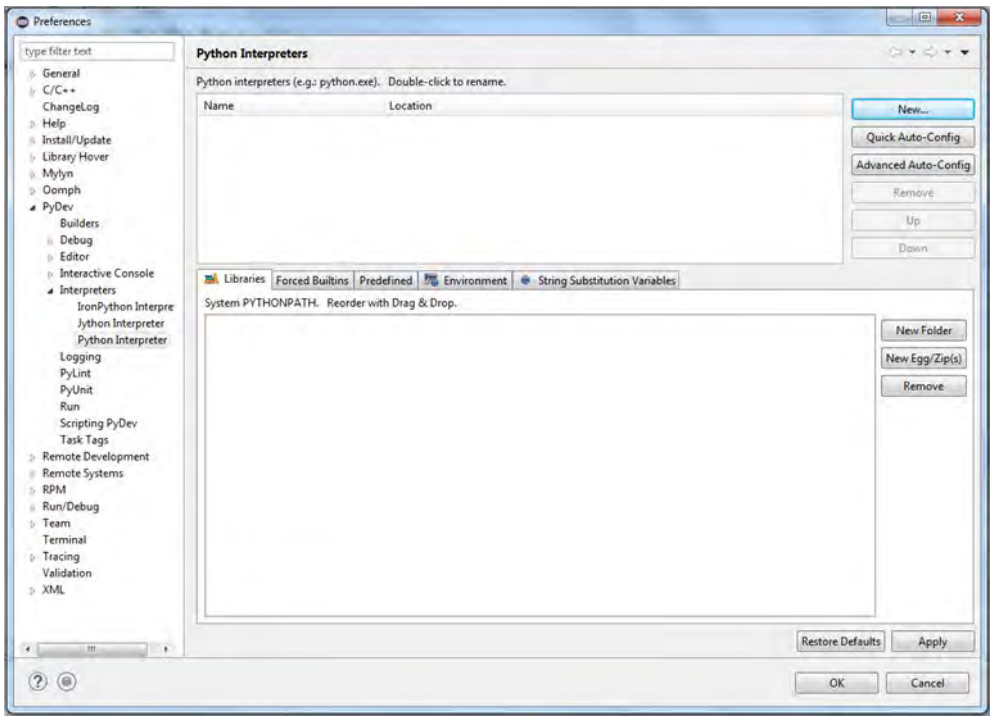
   Beide können ebenfalls direkt aus Eclipse heraus installiert werden.

   Nach der Installation muss Eclipse neu gestartet werden.

Konfiguration von Eclipse

In Eclipse muss zunächst der Pfad zur Python-Installation eingetragen werden:

1. Starten Sie ***Eclipse***.
2. Öffnen Sie über ***Window > Preferences*** den Dialog ***Preferences***.
3. Öffnen Sie im Dialog ***Preferences*** in der Liste im linken Fenster den Eintrag ***PyDev > Interpreters > Python Interpreter***.



4. Klicken Sie die Schaltfäche ***New***.

   Der Dialog ***Select interpreter*** zur Auswahl des Interpreters wird geöffnet.

5. Wählen Sie im Dialog ***Select interpreter*** den auf dem PC installierten Python Interpreter aus, und klicken Sie auf ***OK***.

   Der Interpreter wird geladen.

6. Schließen Sie den Dialog ***Preferences*** durch Klicken auf ***OK***.

   Die Konfiguration von Eclipse ist abgeschlossen. Sie befinden sich wieder im Hauptfenster des Programms.

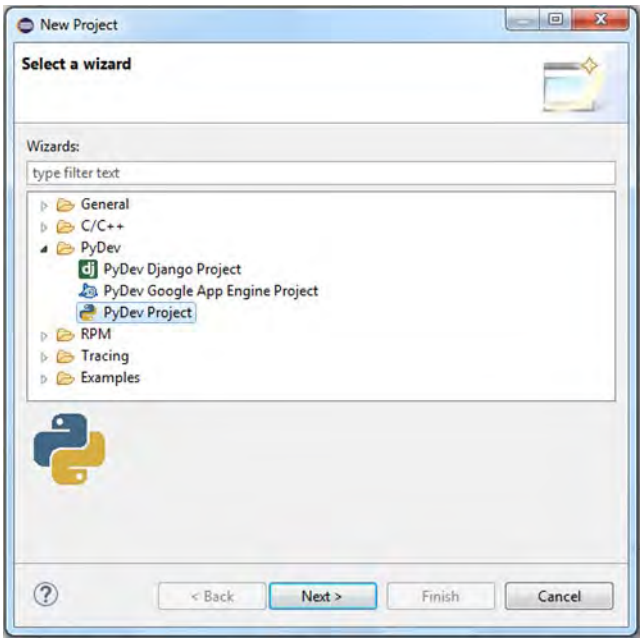
Einsatz von Eclipse und PIPython auf dem PC

Projekt erstellen

Nach der korrekten Konfiguration von Eclipse kann das erste Projekt angelegt werden. Gehen Sie folgendermaßen vor:

1. Wählen Sie im Hauptfenster des Programms die Menüoption ***File > New > Project > PyDev > PyDev Project***.

   Der Dialog ***New Project*** zur Auswahl des Project Wizard wird geöffnet.



2. Markieren Sie den Eintrag ***PyDev > PyDevProject***, und klicken Sie auf ***Next***.

   Der Dialog ***PyDev Project*** zum Anlegen des Projekts wird geöffnet.

3. Geben Sie im Dialog ***PyDev Project*** den Namen des Projekts ein, und wählen Sie den installierten Python Interpreter aus.

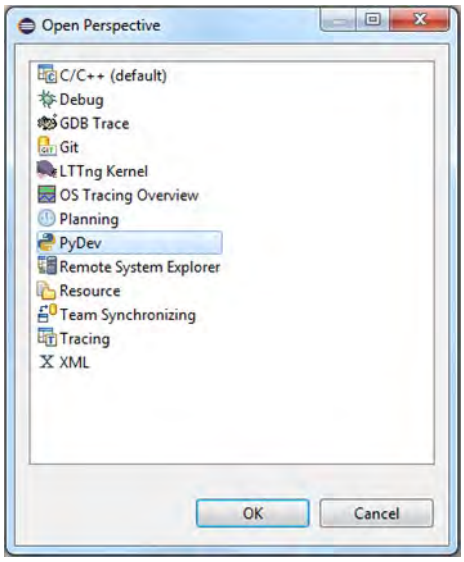
4. Klicken Sie ***Finish***, um das Projekt zu erstellen.

   Der Dialog zur Projekterstellung wird geschlossen, und im Hauptfenster von Eclipse wird das angelegte Projekt in der PyDev-Perspektive geöffnet.

Falls sich die PyDev-Perspektive nicht automatisch öffnet:

a) Öffnen Sie im Hauptfenster von Eclipse über ***Window > Perspective > Open Perspective > Other*** den Dialog ***Open Perspective***.

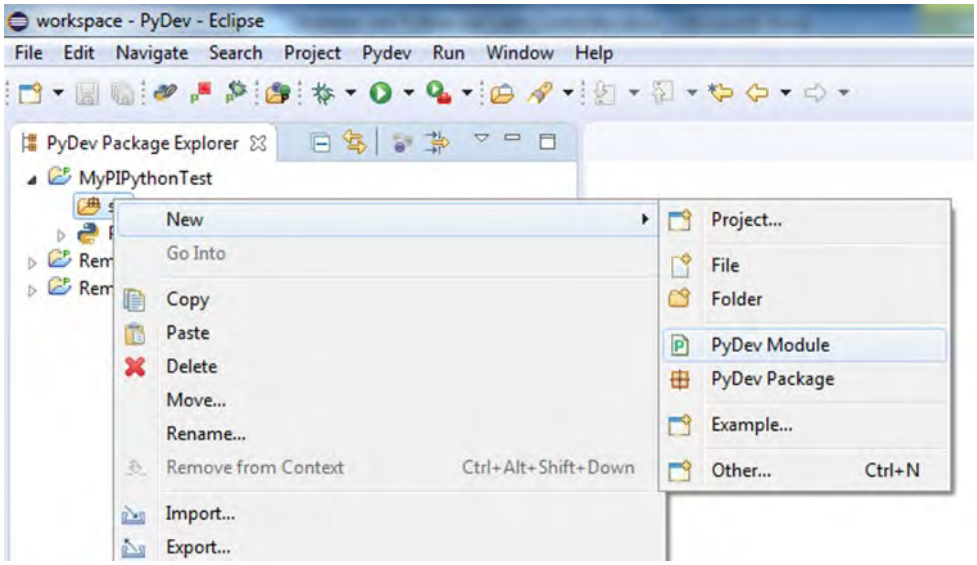
b) Markieren Sie im Dialog ***Open Perspective*** den Eintrag ***PyDev***, und klicken Sie ***OK***.



Skript erstellen und ausführen

Nach der Erstellung des Projekts kann in der geöffneten PyDev-Perspektive in Eclipse ein erstes Skript erstellt werden:

1. Klicken Sie auf dem Ordner ***src*** unterhalb des neu angelegten Projekts die rechte Maustaste, um das Kontextmenü zu öffnen, und rufen Sie die Option ***New > PyDev Module*** auf.

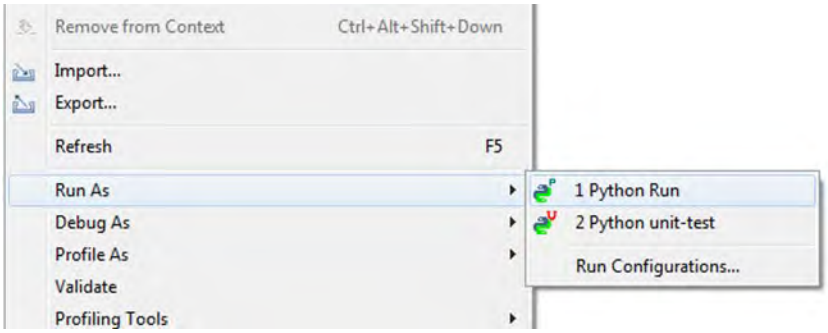


2. Nun können Sie das Skript schreiben.

   Das folgende Beispiel-Skript öffnet einen Verbindungsdialog und fragt den Identifikations-String des Controllers ab. Das Skript nutzt die PI_GCS2_DLL.dll.

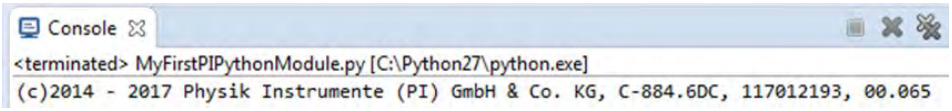
```
from pipython import GCSDevice
gcs = GCSDevice('C-884')
gcs.InterfaceSetupDlg()
print(gcs.qIDN())
gcs.CloseConnection()
```

3. Zum Starten des Moduls klicken Sie auf dem erstellten Skript die rechte Maustaste, um das Kontextmenü zu öffnen, und rufen Sie die Option ***Run As > 1 Python Run*** auf.



   Das Skript wird ausgeführt.

Bei der Ausführung unseres Beispiel-Skripts öffnet sich nach dem Start ein Verbindungsdialog. Nach erfolgreicher Verbindungsherstellung wird der Identifikations-String des Controllers in das Konsolenfenster von Eclipse ausgegeben.



Einsatz von Eclipse und PIPython auf dem Controller

Um PIPython auf dem Controller vollständig nutzen zu können, werden die unter "Installation von Eclipse" (S. 6) und "Konfiguration von Eclipse" (S. 7) beschriebenen Einstellungen vorausgesetzt.

Verbindung zum Controller konfigurieren

1. Um eine Verbindung (über SSH) zum Controller herzustellen, muss in Eclipse die RSE-Perspektive (Remote System Explorer) geöffnet werden:

   a) Öffnen Sie im Hauptfenster von Eclipse über ***Window > Perspective > Open Perspective > Other*** den Dialog ***Open Perspective***.

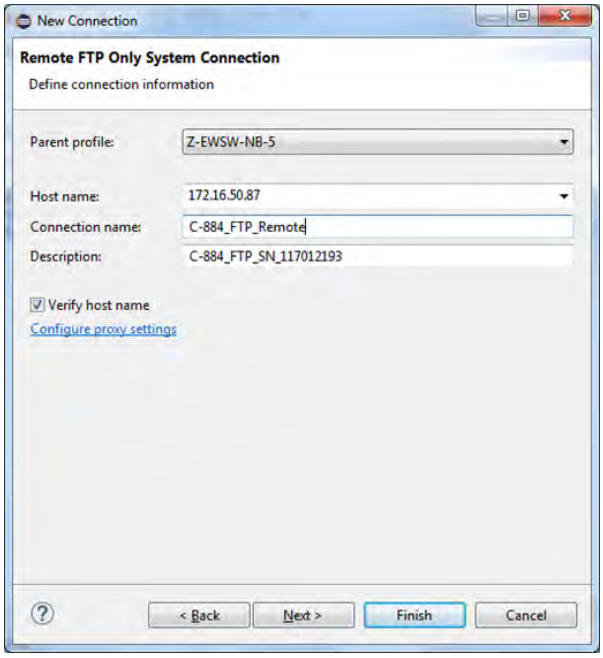
   b) Markieren Sie im Dialog ***Open Perspective*** den Eintrag ***Remote System Explorer***, und klicken Sie ***OK***.

2. Klicken Sie im Fenster der RSE-Perspektive die rechte Maustaste, um das Kontextmenü zu öffnen, und rufen Sie die Option ***New > Connection… > FTP Only*** auf.

   Der Dialog ***New Connection*** zur Konfiguration einer Remote-Verbindung wird geöffnet.

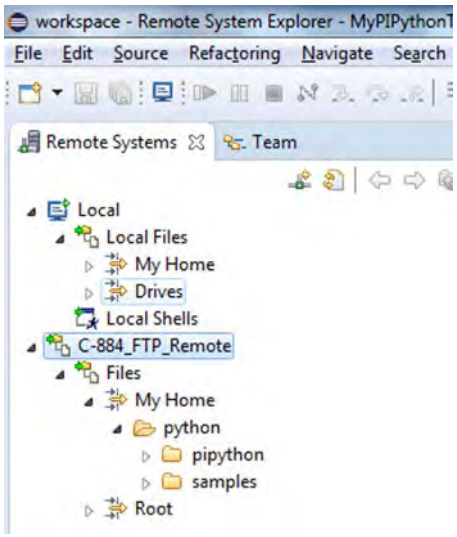
3. Geben Sie im Dialog ***New Connection*** folgende Informationen für die Verbindung an:

   - ***Host name***: IP-Adresse des Controllers, zu dem die Verbindung hergestellt werden soll
   - ***Connection name***: frei wählbarer Name, unter dem die Verbindung in Eclipse gespeichert wird
   - ***Description***: frei wählbare Beschreibung der Verbindung



4. Schließen Sie den Dialog über ***Finish***.

   Die konfigurierte Verbindung wird in der RSE-Perspektive in Eclipse angezeigt:



Verbindung zum Controller aufbauen

- Markieren Sie in der RSE-Perspektive in Eclipse den Eintrag der neu konfigurierten Remote-Verbindung, und klicken Sie die rechte Maustaste, um das Kontextmenü zu öffnen, und rufen Sie die Option ***Connect*** auf, um die Verbindung zu öffnen.

  Benutzername und Passwort für die Anmeldung entnehmen Sie bitte dem Abschnitt "Zugangsdaten für PIPython" (S. 4).

Im Verzeichnisbaum kann nun direkt auf das Dateisystem des Controllers zugegriffen werden. Standardmäßig sollten sich eigene Skripte im Ordner /piuser/python unterhalb des Homeverzeichnisses befinden. Dort befindet sich ebenfalls ein Unterordner, in welchem sich bereits einige Beispielskripte befinden.

Eigene Skripte können Sie direkt auf dem Controller erstellen, indem Sie über Rechtsklick auf dem entsprechenden Ordner das Kontextmenü öffnen und die Option ***New > File*** (+ Dateierweiterung "py") aufrufen. Ebenso besteht die Möglichkeit, Skripte von der lokalen Festplatte auf den Controller zu kopieren.

Im Gegensatz zur Ausführung von Skripten auf dem PC wird bei Skripten, die auf dem Controller ausgeführt werden, keine DLL verwendet. Der Import der benötigten Komponenten erfolgt deshalb, wie im folgenden Beispiel-Skript zu sehen, leicht abweichend. Aus diesem Grund kann die Verbindung auch nicht über einen Dialog aufgebaut werden; sie wird stattdessen über einen Socket etabliert.

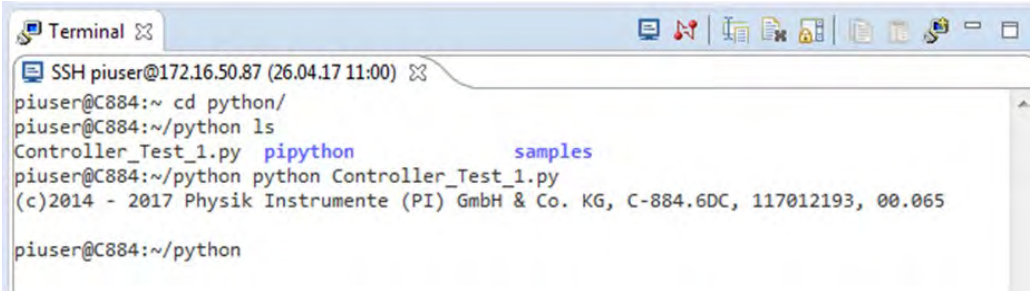
Das nachfolgende Beispiel-Skript baut eine Verbindung zur Controller-Firmware auf, fragt den Identifikations-String des Controller ab und gibt ihn aus.

```
from pipython.gcscommands import GCSCommands
from pipython.gcsmessages import GCSMessages
from pipython.interfaces.pisocket import PISocket
gateway = PISocket(host='127.0.0.1')
messages = GCSMessages(gateway)
gcs = GCSCommands(messages)
print(gcs.qIDN())
gateway.close()
```

Um die Verbindung zum Controller herzustellen, kann in Eclipse über ***Window > Show View*** das Terminalfenster angezeigt werden. Hier kann über das entsprechende Symbol (rechts oben, siehe Abbildung) eine Verbindung aufgebaut werden.

Nach dem Wechsel in das entsprechende Verzeichnis (home/piuser/python) können Python-Skripte durch Aufruf des Kommandos `python <Skriptname>` gestartet werden.

Im Beispiel in der folgenden Abbildung wird das Skript "Controller_Test_1.py" gestartet. Es öffnet einen Verbindungsdialog und fragt den Identifikations-String des Controllers ab. Der Aufruf lautet: `python Controller_Test_1.py`. Die Abbildung zeigt das erfolgreich ausgeführte Skript.



Verwendung des Remote Debuggers in Eclipse

Es besteht die Möglichkeit, Python-Skripte remote zu debuggen. Hierbei können z. B. Variablen während der Ausführung des Skripts auf dem Controller direkt über die Eclipse IDE beobachtet werden.

Debug-Funktion im Skript aktivieren

Um ein Skript remote zu debuggen, muss das zu debuggende Skript selbst Folgendes beinhalten:

1. Importieren Sie im Skript, das Sie debuggen wollen, das Modul "pydevd", indem Sie folgenden Befehl eingeben: `import pydevd`
2. Rufen Sie die pydevd-Funktion "settrace" mit der IP-Adresse des PC auf, z. B.: `pydevd.settrace('172.17.130.84')`

Beispiel für ein einfaches Skript, das für ein Remote-Debugging vorbereitet wurde:

```
import sys
sys.path.append('/home/piuser/python/pysrc')
import pydevd
x = 0
while x < 100 :
    print(x)
    pydevd.settrace('172.17.130.84')
    x = x + 1
```

Remote Debugger konfigurieren

Wenn Sie den Python Remote Debugger in Eclipse nutzen wollen, führen Sie folgende Schritte aus:

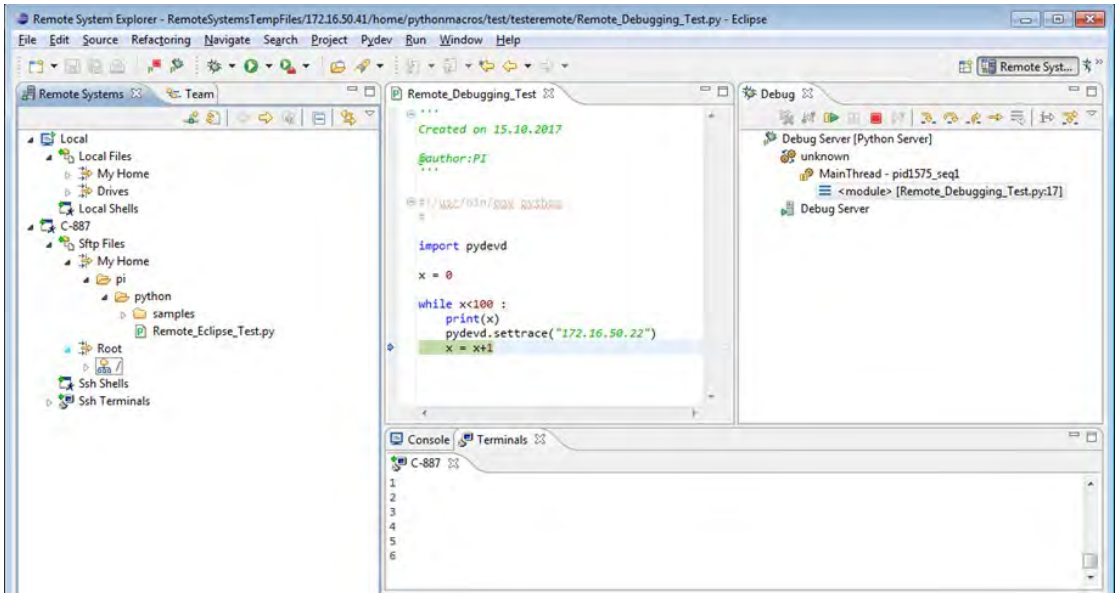
1. Kopieren Sie den vollständigen Inhalt des Verzeichnisses, in dem sich die Datei ***pydevd.py*** befindet, in das folgende Verzeichnis auf dem Controller:
   /home/piuser/python/pysrc
   Die Datei pydevd.py befindet sich je nach Version von Eclipse in unterschiedlichen Verzeichnissen, z. B. unter:
   C:\Program Files (x86)\Eclipse\eclipse\plugins\org.python.pydev_VERSION\pysrc
2. Um das Python Remote-Debugging komfortabel nutzen zu können, sollte nun die aktuelle Perspektive in Eclipse um einige Ansichten erweitert werden:
   a) Fügen Sie der Toolbar die Buttons zum Starten und Stoppen des PyDev-Servers hinzu, indem Sie ***Window > Customize perspective > Command groups availability > PyDev debug*** aufrufen.

b) Erweitern Sie Eclipse um die Ansichten "Debug" und "Variables", indem Sie ***Window > Show View > Other > Debug*** aufrufen.

Debug-Funktion verwenden

In Eclipse kann nun der PyDev-Server über den grünen Button (siehe Abbildung) gestartet werden. Im Anschluss kann das Skript wie bisher über das Terminal gestartet werden.

Mit der Funktionstaste ***F8*** oder dem ***Resume*** Button kann das Skript nun schrittweise ausgeführt werden.



Die bei manchen Versionen erscheinende Warnung des pydev-Debuggers kann ignoriert werden.

Weitere Informationen zur Konfiguration des Remote-Debuggers mit Eclipse/Pydev finden Sie unter http://www.pydev.org/manual_adv_remote_debugger.html.

Verwendung von PyCharm Professional

Installation von PyCharm Professional

Wollen Sie PyCharm Professional für die Arbeit mit PIPython verwenden, führen Sie auf dem PC folgende Installationen durch:

- IDE: PyCharm Professional (Download z. B. über https://www.jetbrains.com/pycharm/)

  Alle folgenden Beschreibungen und Abbildungen beziehen sich auf die Version 2017.1.2 unter Windows.

Konfiguration von PyCharm Professional

Nach der Installation von PyCharm müssen, abgesehen von der Wahl des Python-Interpreters, keine weiteren Einstellungen vorgenommen werden, um Python und PIPython vom PC aus zu verwenden.

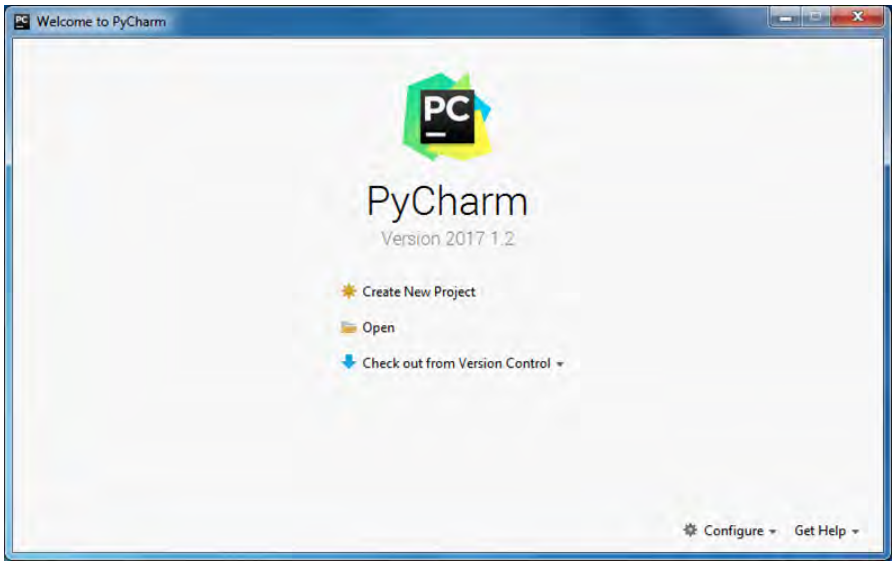
Die Auswahl des Interpreters kann global über ***File > Default Settings > Project Interpreter*** oder projektspezifisch über ***File > Settings > Project: Name > Project Interpreter*** definiert werden.

Einsatz von PyCharm und PIPython auf dem PC

Projekt erstellen

Nach der Installation und Konfiguration von PyCharm kann das erste Projekt angelegt werden. Gehen Sie folgendermaßen vor:

1. Wählen Sie im Hauptfenster des Programms die Option ***Create New Project***.



   Der Dialog ***New Project*** wird geöffnet.

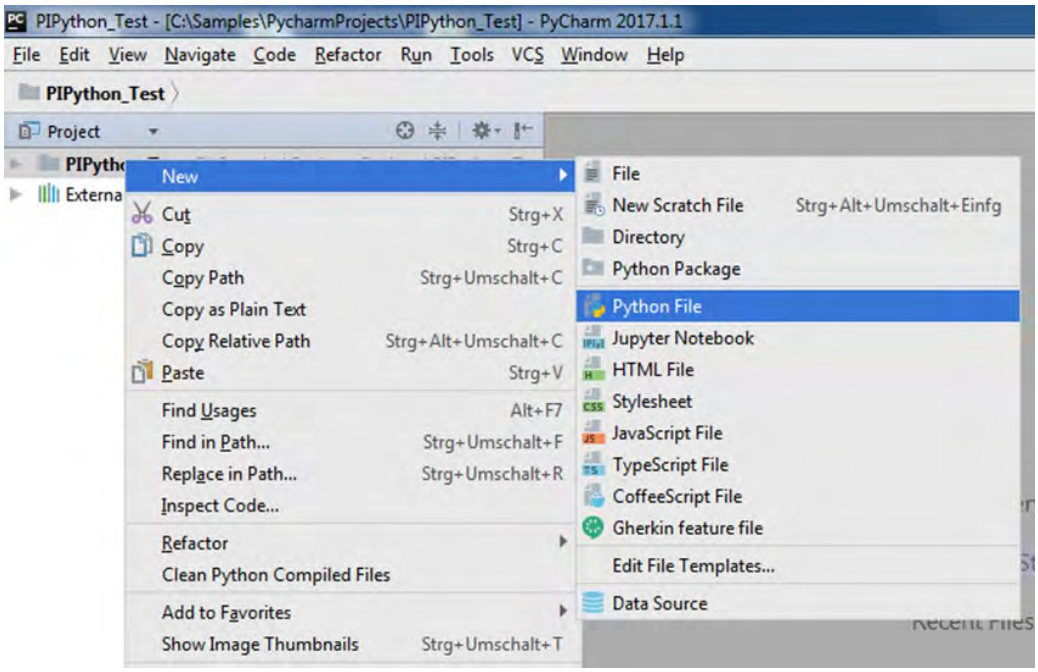
2. Geben Sie im Dialog ***New Project*** im Feld ***Location*** den Namen ein, unter dem das Projekt gespeichert werden soll, und wählen Sie im Feld ***Interpreter*** ggf. den zu verwendenden Python Interpreter aus.
3. Klicken Sie ***Create***, um das Projekt zu erstellen.

   Der Dialog zur Projekterstellung wird geschlossen, und im Hauptfenster von PyCharm wird das angelegte Projekt geöffnet.

Skript erstellen und ausführen

Nach der Erstellung des Projekts kann in PyCharm ein erstes Skript erstellt werden:

1. Klicken Sie auf dem neu angelegten Projekt die rechte Maustaste, um das Kontextmenü zu öffnen, und rufen Sie die Option ***New > Python File*** auf.

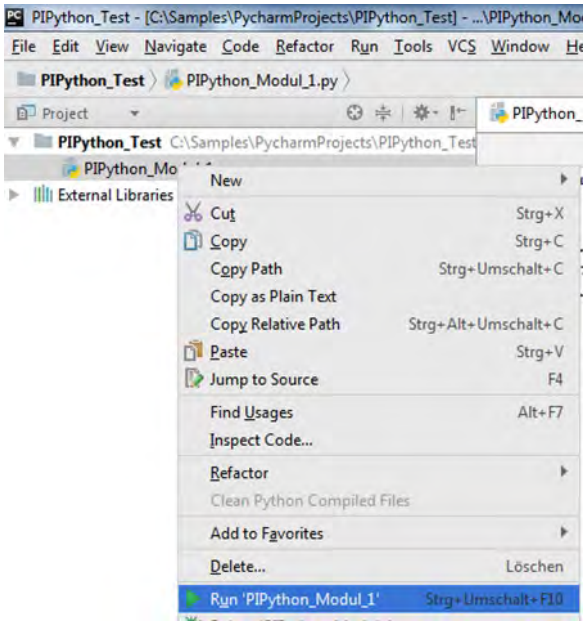


2. Nun können Sie das Skript schreiben.

   Das folgende Beispiel-Skript öffnet einen Verbindungsdialog und fragt den Identifikations-String des Controllers ab. Das Skript nutzt die PI_GCS2_DLL.dll.

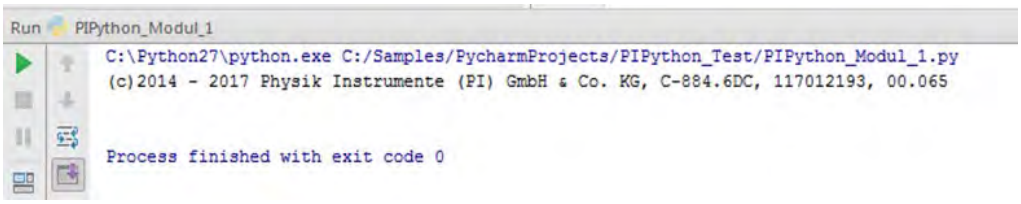
```
from pipython import GCSDevice
gcs = GCSDevice('C-884')
gcs.InterfaceSetupDlg()
print(gcs.qIDN())
gcs.CloseConnection()
```

3. Zum Starten des Skripts klicken Sie auf dem erstellten Skript die rechte Maustaste, um das Kontextmenü zu öffnen, und rufen Sie die Option ***Run <Skriptname>*** auf.



Das Skript wird ausgeführt.

Bei der Ausführung des Beispiel-Skripts öffnet sich nach dem Start ein Verbindungsdialog. Nach erfolgreicher Verbindungsherstellung wird der Identifikations-String des Controllers in das Konsolenfenster von PyCharm ausgegeben.



Einsatz von PyCharm und PIPython auf dem Controller

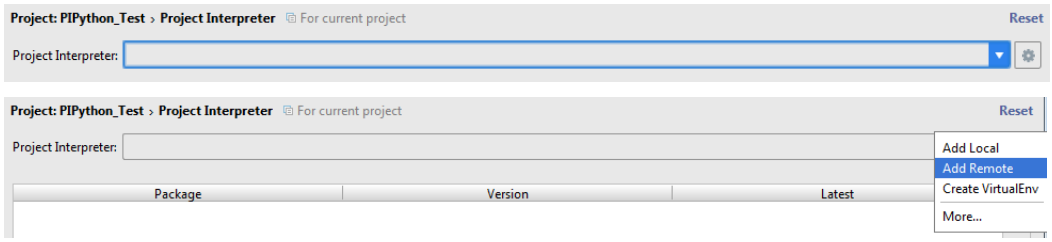
Zum Arbeiten mit dem Python-Interpreter auf dem Controller muss das "SSH Remote Run" Plugin von PyCharm aktiviert sein. Dies kann unter ***File > Settings > Plugins*** überprüft und ggf. aktiviert werden.

Um PIPython auf dem Controller vollständig nutzen zu können, werden die unter "Konfiguration von PyCharm Professional" (S. 16) beschriebenen Einstellungen vorausgesetzt.

Verbindung zum Controller für Remote Interpreter konfigurieren

Führen Sie in PyCharm Professional folgende Schritte aus:

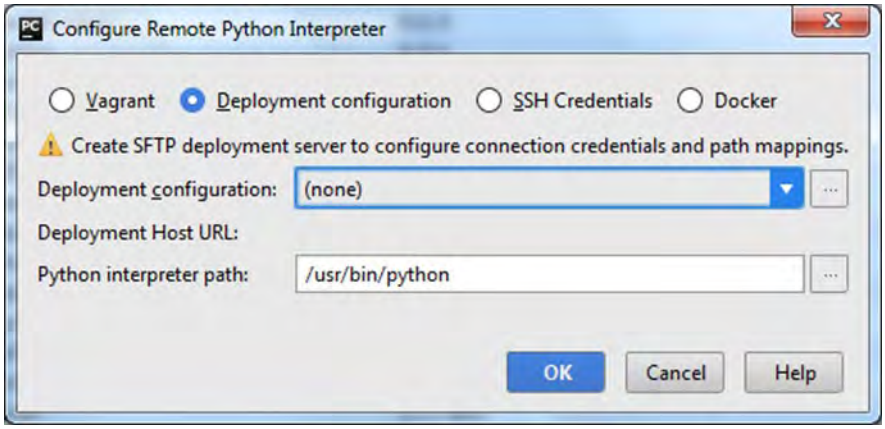
1. Fügen Sie Ihrem Projekt einen Remote Interpreter hinzu, indem Sie ***File > Settings > Project: Name > Project Interpreter*** aufrufen (Name = der Name Ihres Projekts).
2. Klicken Sie mit der rechten Maustaste auf den Button  hinter dem Eingabefeld ***Project Interpreter***, und wählen Sie im Kontextmenü die Option ***Add remote***.



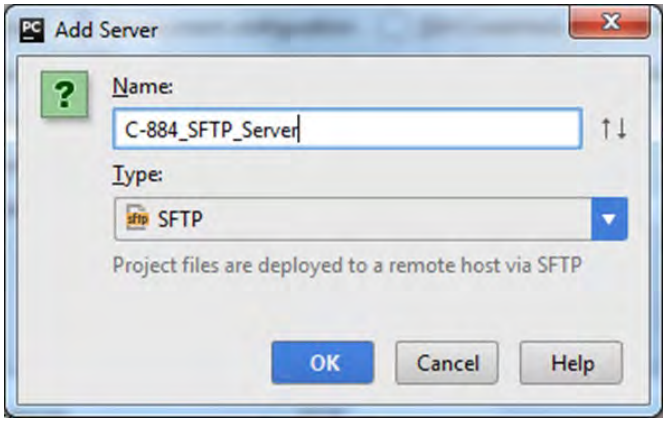
Der Konfigurationsdialog für den Remote Interpreter wird geöffnet.

Anmerkung: Beim Hinzufügen eines Python Remote Interpreters werden automatisch die PyCharm helpers-Dateien auf den Controller kopiert. Diese werden u.a. zum Debuggen benötigt. Die Versionsinformationen der Helper-Dateien werden von PyCharm bei jedem Remote Lauf geprüft und bei Bedarf aktualisiert.

3. Markieren Sie im Konfigurationsdialog die Option ***Deployment configuration***.

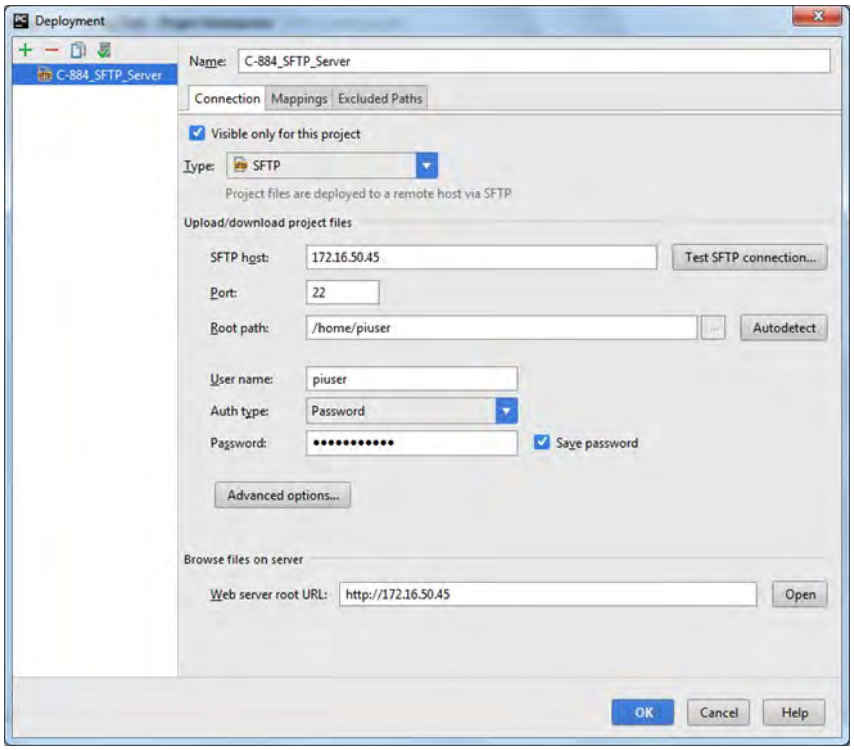


4. Klicken Sie auf den Auswahl-Button (***...***) hinter dem Feld ***Deployment configuration***, um den Dialog ***Add Server*** zur Definition eines Servers zu öffnen.

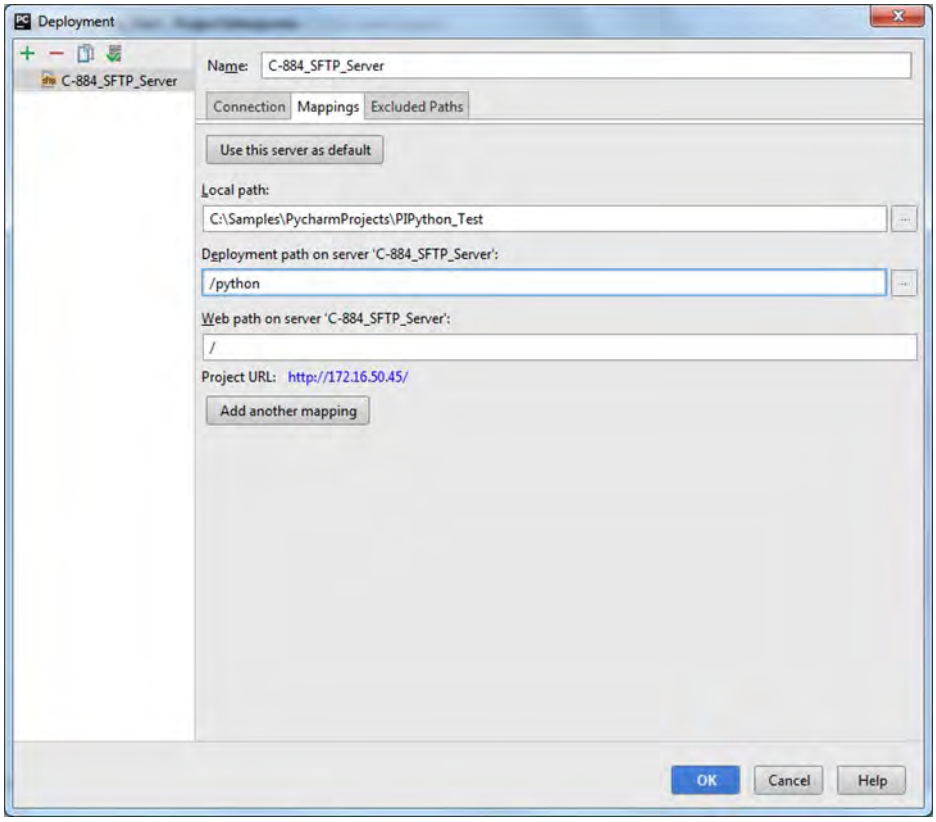


5. Tragen Sie im Dialog ***Add Server*** im Feld ***Name*** einen Namen (frei wählbar) für den Server ein, und wählen Sie im Feld ***Type*** *SFTP* aus.
6. Schließen Sie den Dialog ***Add Server*** mit ***OK***.

   Der Dialog ***Deployment*** wird geöffnet.
7. Konfigurieren Sie hier im Tab ***Connection*** die Verbindungseinstellungen wie in der folgenden Abbildung dargestellt. Benutzername und Passwort für die Anmeldung entnehmen Sie bitte dem Abschnitt "Zugangsdaten für PIPython" (S. 4).



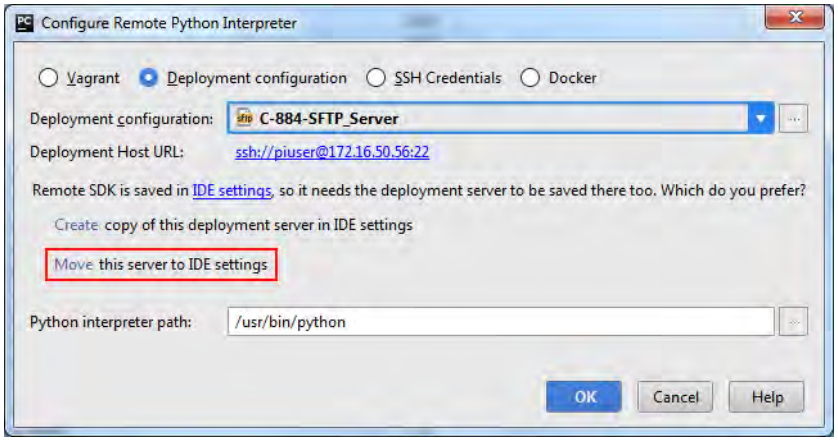
8. Stellen Sie im Tab ***Mappings*** den Deployment-Pfad auf dem Controller ein wie in der folgenden Abbildung dargestellt. An diesen Ort werden die Skripte bei Bedarf übertragen.



9. Schließen Sie die Deployment-Konfiguration mit ***OK*** ab.

   Der Dialog ***Deployment*** wird geschlossen, und Sie befinden sich wieder im Dialog ***Configure Remote Python Interpreter***.

10. Aktivieren Sie die Funktion ***Move*** (in der Abbildung markiert), um die erstellte Konfiguration für alle Projekte zu übernehmen. Im Anschluss werden automatisch die PyCharm helper-Dateien auf den Controller kopiert.



11. Schließen Sie den Dialog ***Configure Remote Python Interpreter*** mit ***OK***.

    Damit ist die Konfiguration der Verbindung zum Controller für den Remote Interpreter abgeschlossen.

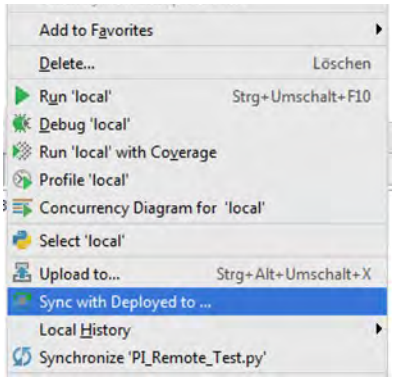
Skripte auf dem Controller ausführen

Im Gegensatz zur Ausführung von Skripten auf dem PC wird bei Skripten, die auf dem Controller ausgeführt werden, keine DLL verwendet. Der Import der benötigten Komponenten erfolgt deshalb, wie im folgenden Beispiel-Skript zu sehen, leicht abweichend. Aus diesem Grund kann die Verbindung auch nicht über einen Dialog aufgebaut werden; sie wird stattdessen über einen Socket etabliert.

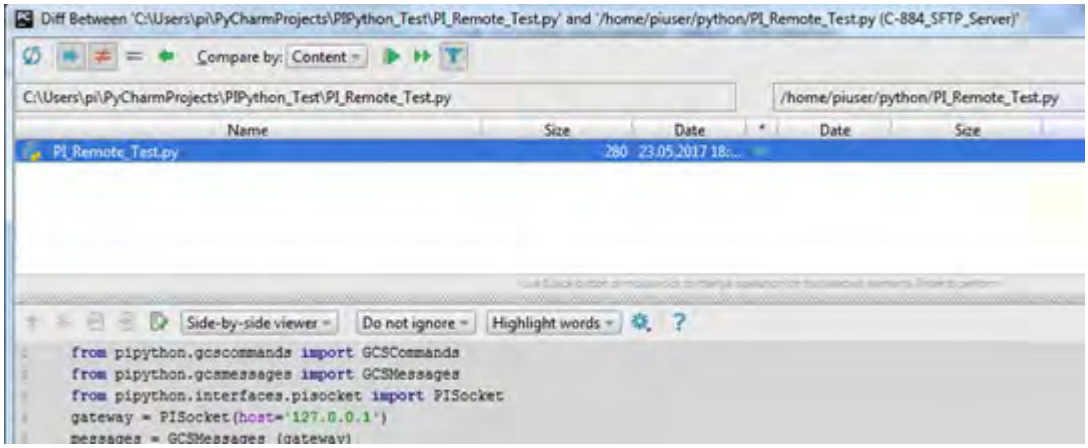
Das nachfolgende Beispiel-Skript baut eine Verbindung zur Controller-Firmware auf, fragt den Identifikations-String des Controller ab und gibt ihn aus.

```
from pipython.gcscommands import GCSCommands
from pipython.gcsmessages import GCSMessages
from pipython.interfaces.pisocket import PISocket
gateway = PISocket(host='127.0.0.1')
messages = GCSMessages(gateway)
gcs = GCSCommands(messages)
print(gcs.qIDN())
gateway.close()
```

Skripte können entweder über die Kontextmenü-Option ***Upload to ...*** direkt in das angegebene Verzeichnis auf dem Controller übertragen oder über die Kontextmenü-Option ***Sync with Deployed to ...*** mit diesem synchronisiert werden.

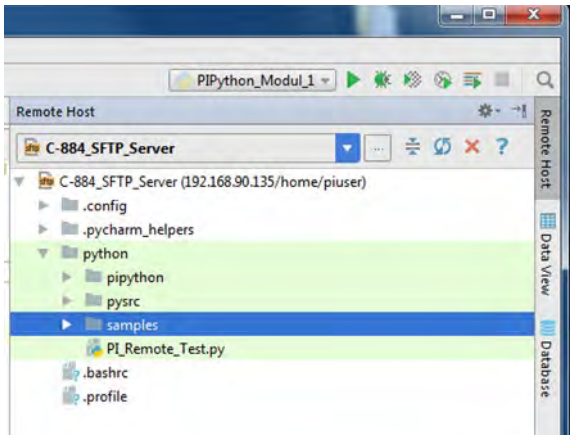


Die folgende Abbildung zeigt die Synchronisierung von Skripten mit dem Controller:

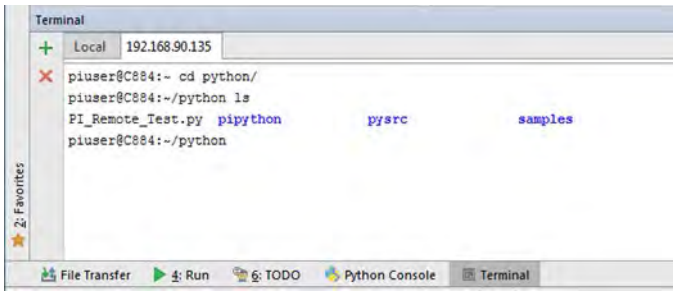


Mit Hilfe des grünen Pfeils ("Synchronize Selected") kann die ausgewählte Datei auf den Controller übertragen werden.

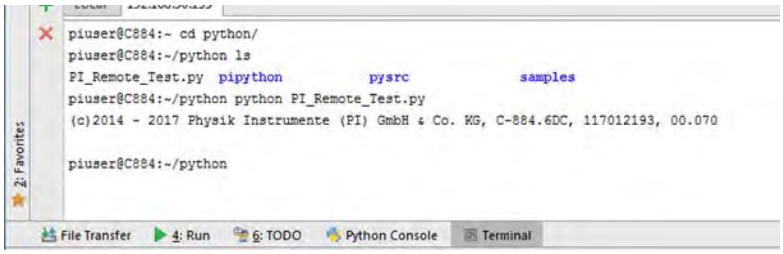
Für ein komfortables Arbeiten kann ein Dateibrowser auf dem Controller eingeblendet werden. Rufen Sie dazu in PyCharm die Menüfunktion ***Tools > Deployment > Browse Remote Host*** auf.



Befindet sich das Skript auf dem Controller, kann nun über ***Tools > Start SSH session…*** ein Terminalfenster zum Controller geöffnet werden. Nach dem Wechsel in das entsprechende Verzeichnis (home/piuser/python) können Python-Skripte durch Aufruf des Kommandos `python <Skriptname>` gestartet werden.



Im Beispiel in der folgenden Abbildung wird das Skript "PI_Remote_Test.py" gestartet. Der Aufruf lautet: `python PI_Remote_Test.py`. Die Abbildung zeigt das erfolgreich ausgeführte Skript.



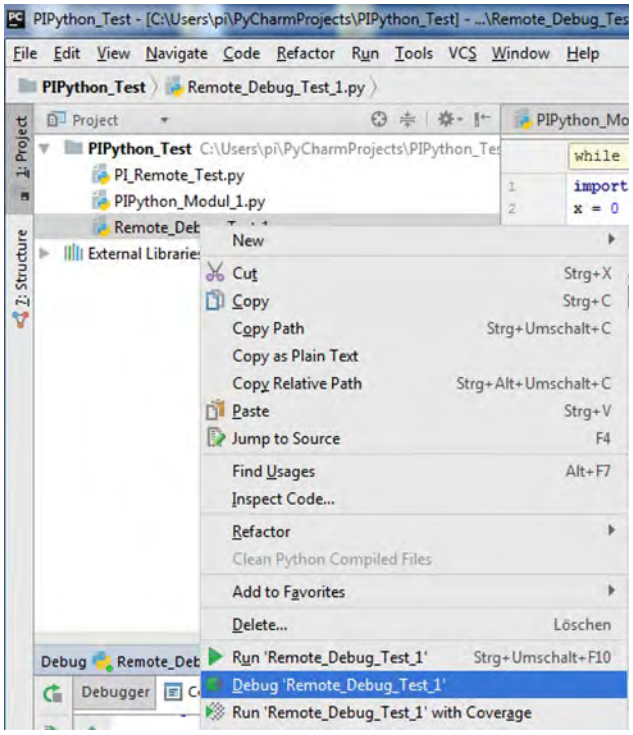
Weitere Informationen zur Konfiguration eines Remote-Interpreters in PyCharm finden Sie unter http://www.jetbrains.com.

Verwendung des Remote Debuggers in PyCharm

Es besteht die Möglichkeit, Python-Skripte remote zu debuggen. Hierbei können z. B. Variablen während der Ausführung des Skripts auf dem Controller direkt über die PyCharm IDE beobachtet werden.

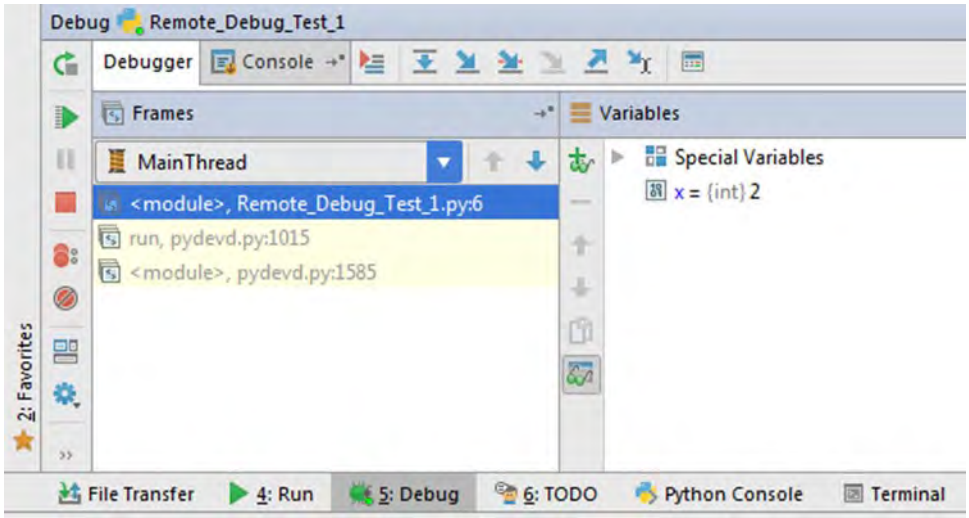
Debug-Funktion verwenden

In PyCharm kann das Remote Debugging über die Kontextmenü-Option ***Debug <Skriptname>*** gestartet werden.



Das Skript wird während der Ausführung im Tab ***Console*** dargestellt. Im Tab ***Debugger*** werden die Variablen angezeigt (in der folgenden Abbildung: im rechten Fenster).

Mit der Funktionstaste ***F9*** oder dem ***Resume Program*** Button kann das Skript nun schrittweise ausgeführt werden.



Ausführung von Skripten auf dem Controller

Automatische Ausführung von Skripten beim Systemstart

Soll ein Python-Skript automatisch beim Start des Controllers ausgeführt werden (vergleichbar zum Verhalten der Autostartfunktion von GCS-Makros), so muss dieses den Namen "autostart.py" tragen und unter /home/piuser/python abgelegt sein.

Manuelle Ausführung von Skripten

Der manuelle Aufruf eines Python-Skripts auf dem Controller kann über eine SSH-Verbindung erfolgen (z. B. mit einem Terminalprogramm wie beispielsweise "PuTTY"):

```
piuser@C884:~ python python/script.py
```

Beim Beenden der Verbindung wird auch das Skript automatisch beendet. Ist dies nicht gewünscht, muss der Aufruf ein "&" am Ende enthalten:

```
piuser@C884:~ python python/script.py &
```