

Proc Python - Zwei Sprachen, ein Workflow

David Weik, SAS

Was ist eigentlich die SAS Sprache?

Wie viele Sprachen/Syntaxen gibt es?

- Data Step
- Standard Proc's
- SQL
- TPL/PCL
- DS2
- Lua
- Groovy
- X-Kommando
- Java Object
- IML
- IML-R Interface
- REST
- SCL
- CASL
- Macro
- Optmodel
- LITI
- Proc FCMP

18

Eine Mehr...

Proc Python – Die Prozedur der Kokosnuss

Timeline

Proc FCMP kann doch Python?

- Proc FCMP kann bereits seit SAS 9.4 M3 Python aufrufen
- Aufruf als Funktion z.B. im Data Step
- Keine tiefere Integration, reine Wertübergabe

Und von Python nach SAS

Das geht doch auch schon lange

- saspy, ermöglicht es SAS Code auszuführen
- Bietet Schnittstellen für den Austausch von Daten
- SAS Funktionalität in Pythonic Syntax
- Aber nur der Aufruf von SAS aus Python, nicht die andere Richtung

Timeline

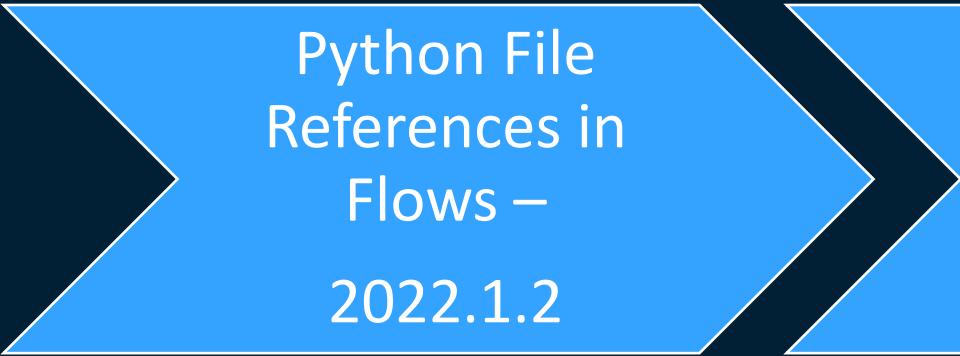
SAS Viya 2021.1.3+

Proc Python –
2021.1.3

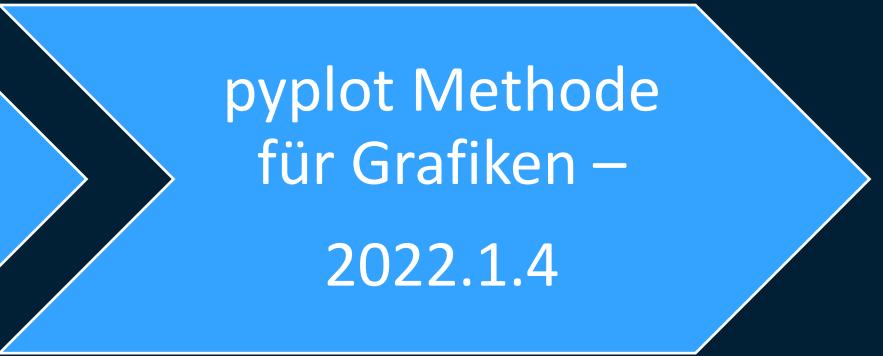
Python Editor +
Transformer –
2021.1.5

Python in Flow
& Lineage –
2021.1.6

Timeline (2)



Python File
References in
Flows –
2022.1.2



pyplot Methode
für Grafiken –
2022.1.4

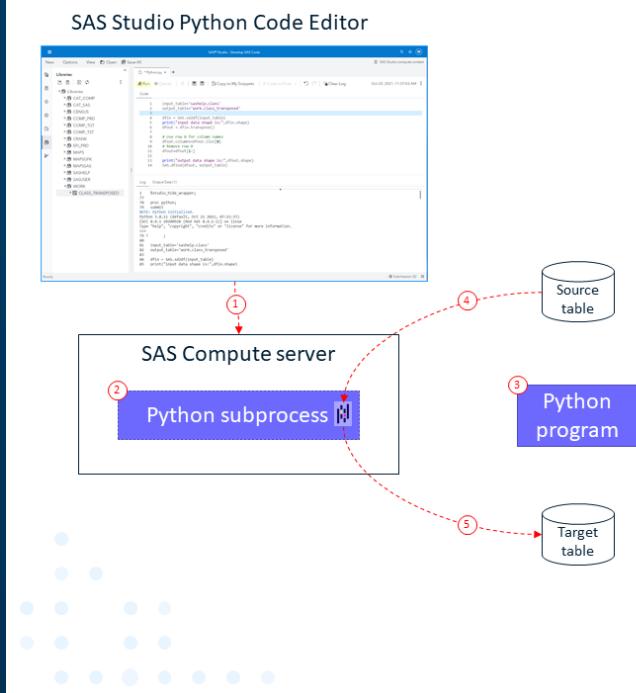
PROC PYTHON

Wie funktioniert es?

Starten eines Python Subprozesses

Hinzufügen des SAS-Moduls mit Funktionen zum Austausch von Daten und Informationen zwischen einem SAS-Prozess und einem Python-Unterprozess

Enthält eine Funktion zur Übermittlung von SAS-Code aus einem Python-Unterprozess



```
proc python; ②
submit;
[...]
input_table='sashelp.class'
output_table='work.class_transposed'

# get input data from SAS into Pandas DataFrame
dfin = SAS.sd2df(input_table) ④
print("input data shape is:",dfin.shape)
dfout = dfin.transpose()

# Use row 0 for column names
dfout.columns=dfout.iloc[0]
# Remove row 0
dfout=dfout[1:]

print("output data shape is:",dfout.shape)
# Write Pandas DataFrame to SAS
SAS.df2sd(dfout, output_table) ⑤

endsubmit;
quit;
```

[Open](#) [Save All](#)

« PY * Python.py x +

[Run](#) [Cancel](#)

Copy to My Snippets

+ Code to Flow



Clear Log

Code

```
1 input_table='sashelp.class'
2 output_table='work.class_transposed'
3
4 dfin = SAS.sd2df(input_table)
5 print("input data shape is:",dfin.shape)
6 dfout = dfin.transpose()
7
8 # Use row 0 for column names
9 dfout.columns=dfout.iloc[0]
10 # Remove row 0
11 dfout=dfout[1:]
12
13 print("output data shape is:",dfout.shape)
14 SAS.df2sd(dfout, output_table)
```

Log Output Data (1)

```
RANPOSED
1 %studio_hide_wrapper;
77
78 proc python;
79 submit
NOTE: Python initialized.
Python 3.8.11 (default, Oct 21 2021, 07:21:37)
[GCC 8.4.1 20200928 (Red Hat 8.4.1-1)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
79 ! ;
80
81 input_table='sashelp.class'
82 output_table='work.class_transposed'
83
84 dfin = SAS.sd2df(input_table)
85 print("input data shape is:",dfin.shape)
```

Ermöglicht es SAS-Benutzern
Python-Code in SAS-Jobs
einzubinden

Run

Cancel



Copy to My Snippets

Code

```
1 ⊕ proc python;
2   submit;
3   print('hello world')
4   enssubmit;
5   run;
```

```
+ 1 /* region: Generated preamble */
79
80  proc python;
81  submit
NOTE: Python initialized.
Python 3.9.10 (main, Apr  5 2022, 17:25:54)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-44)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
81 !      ;
82  print('hello world')
83  enssubmit;
84  run;
85
86 /* region: Generated postamble */
98
```

Start Page * Python 1.py +

Run Cancel | ⌂ | ⌂ | ⌂ | ⌂ Copy to M

Code

```
1 print('hello world')
```

```
1 /* region: Generated preamble */
79
80 proc python;
81 submit
NOTE: Resuming Python state from previous PROC PYTHON invocation.
81 !      ;
82
83 print('hello world')
84
85 endsubmit;
86 quit;
>>>
hello world
>>>
>>>
```

Prozedur Argumente

- Restart
- Terminate
- Timeout
- Infile

Callback Methods

- sasfnc
- submit
- symget
- symput
- pyplot
- df2sd
- sd2df
- hideLOG
- printLOG

SAS Studio

Der Python Code Editor

Python Code
Editor

Daten direkt
aus SAS als
Dataframe

Dataframe
zurück zu SAS

Syntax-Hilfe
für Python

The screenshot shows the SAS Studio Python Code Editor interface. On the left, a code editor window titled "chainladder.py" displays Python code for reading a CSV file, creating a triangle object, and outputting it as a DataFrame. A red box highlights the line "data = SAS.df2sd('work.clrd')". Below the code editor is a syntax help panel showing suggestions like "id", "if", "import", etc. A red box highlights the "import" suggestion. On the right, the "Output Data (1)" tab of the Log window shows the resulting DataFrame. A red box highlights the "proc python;" and "endsubmit;" statements. The SAS logo is in the bottom right corner.

```
import pandas as pd
import chainladder as cl

#data = pd.read_csv('/sasdata/data/Insurance/Chainladder/clrd.csv')
data = SAS.df2sd('work.clrd')

# Create a triangle
triangle = cl.Triangle(
    data, origin='AccidentYear', development='DevelopmentYear',
    index=['GRNAME'], columns=['IncurLoss', 'CumPaidLoss', 'EarnedPremDIR'])

# Output
out_df = triangle.to_frame()
rc = SAS.df2sd(out_df, 'work.cl_triangle')
```

Log Output Data (1)
Errors (0) Warnings (0) Notes (27)

Mar 29, 2022, 5:22:36 PM

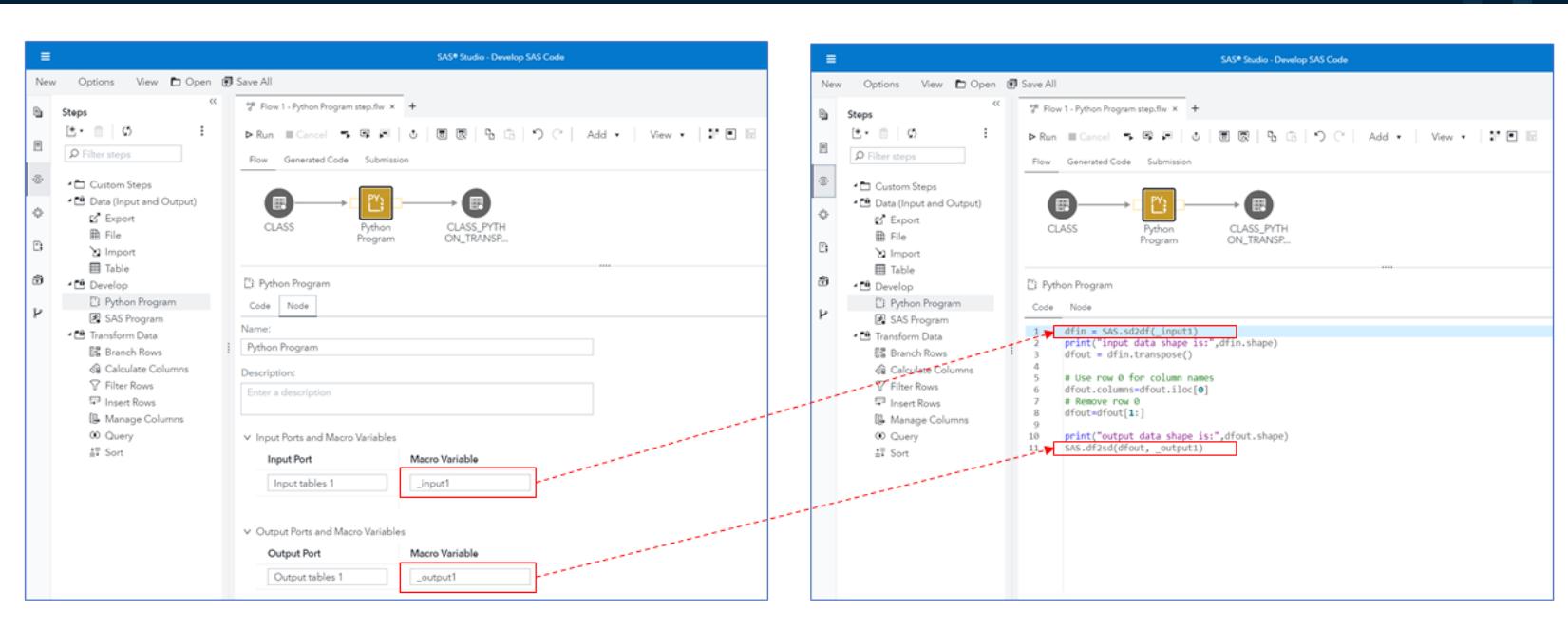
There are no messages.

```
proc python;
submit
NOTE: Python initialized.
Python 3.8.8 (default, Apr 13 2021, 19:56:36)
[GCC 7.3.0] :: Anaconda, Inc. on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
79 ! ;
80
81 import pandas as pd
82 import chainladder as cl
83
84 #data = pd.read_csv('/sasdata/data/Insurance/Chainladder/clrd.csv')
85 data = SAS.df2sd('work.clrd')
86
87 # Create a triangle
88 triangle = cl.Triangle(
89     data, origin='AccidentYear', development='DevelopmentYear',
90     index=['GRNAME'], columns=['IncurLoss', 'CumPaidLoss', 'EarnedPremDIR'])
91
92 # Output
93 out_df = triangle.to_frame()
94 rc = SAS.df2sd(out_df, 'work.cl_triangle')
95
96 endsubmit;
97 quit;
```

Ausgabe-
Tabellen von
Python
anzeigen

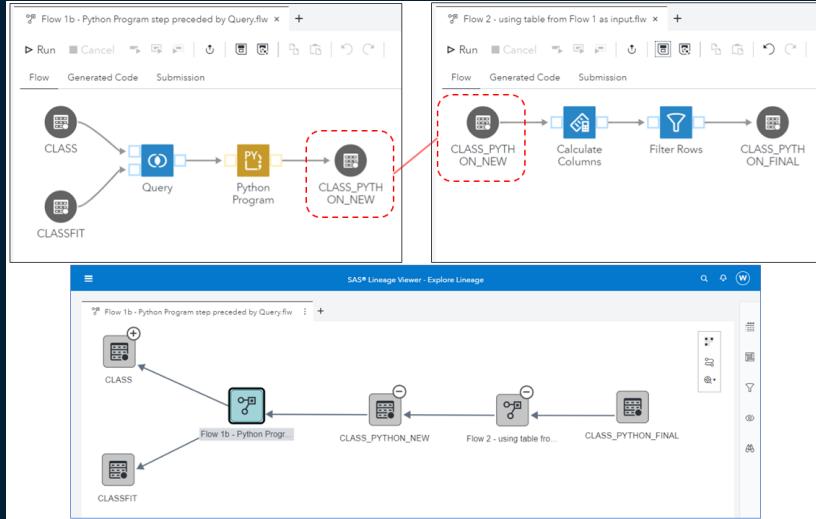
Implizites
Wrapping in
Proc Python

Python Programm Schritt - Studio Flow



- Eingangsport(s) und Ausgangsport(s) sind als Variablen im Python-Prozess verfügbar
- Diese Variablen enthalten Namen von verbundenen Tabellen (libref.tabellenName-Notation)

Integration mit SAS Lineage



- Wo werden die Daten genutzt?
- Woher kommen die Daten?
- In welchen Flows werden die Daten genutzt?

Und das gilt auch wenn Python-Programme als Schritt genutzt werden

```
%macro _etm_py_link_meta;
  %do i = 1 %to &_etm_n_links. / 25;
    %let _etm_firstob = %eval(&i. * 25);
    %let _etm_obsCount = %eval(&_etm_firstob. + 25);

    %if &i. = 1 %then %do;
      data work._etm_links;
        set work._etm_distinct_links(obs=25);
      run;

      proc python restart infile=pgm;
      run;

      data work._etm_link_meta_all;
        set work._etm_link_meta;
      run;
    %end;
```

```
data _null_;
file pgm;

put "import pandas as pd";
put "import requests";
put "from bs4 import BeautifulSoup";

if &allowUnverifiedRequests. then do;
    put "from urllib3.exceptions import InsecureRequestWarning";
    put "requests.packages.urllib3.disable_warnings(category=InsecureRequestWarning)";
end;

put " ";
put "# Get the unique link table from SAS";
put "_etm_df = SAS.sd2df('work._etm_links')";
put " ";
put "# Function to gather link information for each link";
put "def get_link_metadata(row):";
put "    try:";

if &allowUnverifiedRequests. then do;
    put "        r = requests.get(row['_etm_lnks'], verify=False)";
end;
else do;
    put "        r = requests.get(row['_etm_lnks'])";
end;
```

```
end;

put "      html = BeautifulSoup(r, 'html.parser')";
put "      _etm_status_code = r.status_code";
put "      _etm_title = html.find('meta', attrs={'property': 'og:title'})";
put "      _etm_description = html.find('meta', attrs={'property': 'og:description'})";
put "      _etm_url = html.find('meta', attrs={'property': 'og:url'})";
put "      _etm_site_name = html.find('meta', attrs={'property': 'og:site_name'})";
put "    except:";
put "      _etm_status_code = 404";
put "      _etm_title = 'Not available'";
put "      _etm_description = 'Not available'";
put "      _etm_url = 'Not available'";
put "      _etm_site_name = 'Not available'";
put "    return _etm_status_code, _etm_title, _etm_description, _etm_url, _etm_site_name";
put "  ";
put "# Get the information for each link";
put "_etm_df_meta = _etm_df.apply(get_link_metadata, axis='columns', result_type='expand')";
put "_etm_df_all = pd.concat([_etm_df, _etm_df_meta], axis='columns')";
put "_etm_df_all.rename(columns = {0:'_etm_status_code', 1:'_etm_title', 2:'_etm_description', 3:'_etm_url', 4:'_etm_site_name'}, inplace = True)";
put "  ";
put "# Return the information to SAS";
put "SAS.df2sd(_etm_df_all, 'work._etm_link_meta')";

run;
```



④ Extract Text Features

< Base Metadata Custom RegEx Pattern Link Data Text Analytics - Start Text Analytics - Topic Creation Text Analytics - Bool Rule > ✖

Collect additional information from the links in the tweets. This option requires you to have enabled the Options for concatenated and separated columns.

Please note for this step to work your environment needs to be able to make calls to the open internet.

Please be also aware that this step can take a lot of time to run as the individual sites have to be called and their output need to be parsed.

The following five features are extracted for each link:

- HTTP Status Code (basically is it reachable or not)
- Title of the Webpage
- Description of the Webpage
- URL of the Webpage (handy if URL shorteners were used)
- Owner of the site

Do you want to collect metadata from Links in the text?

Do you want to allow Unverified Requests (Warning potential impact: Breach of Confidentiality & Breach of Integrity)?

Run

Cancel



Copy to My Snippets

Code to Flow



Clear Log

Sep 12, 2022, 5:35:40 PM

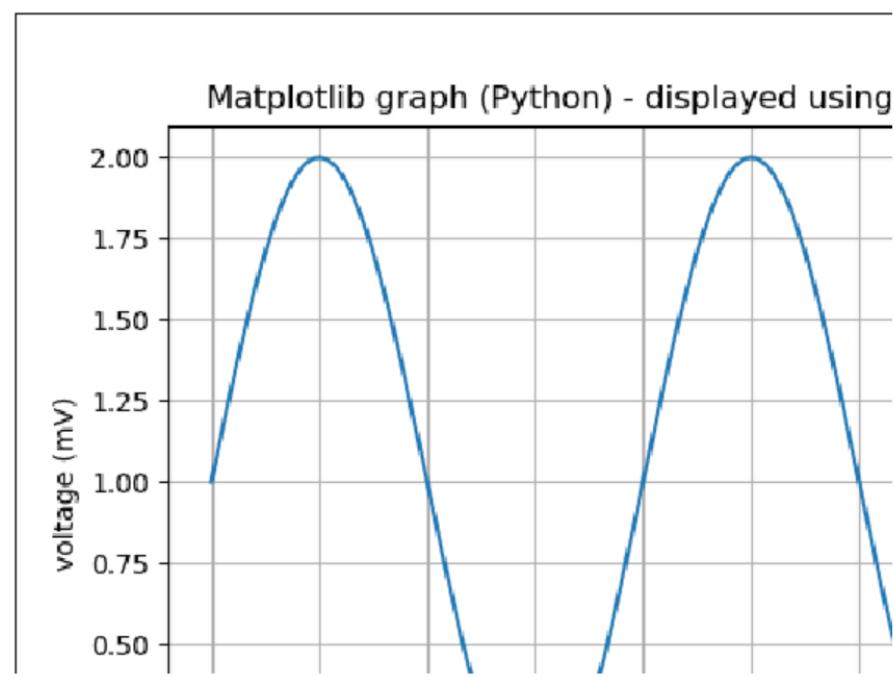


Code

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 # might have to clear the graph using plt.clf() if you run your code multiple times
5 plt.clf()
6
7 t = np.arange(0.0, 2.0, 0.01)
8 s = 1 + np.sin(2*np.pi*t)
9 plt.plot(t, s)
10
11 plt.xlabel('time (s)')
12 plt.ylabel('voltage (mV)')
13 plt.title('Matplotlib graph (Python) - displayed using proc gslice')
14 plt.grid(True)
15
16 SAS.pyplot(plt)
```

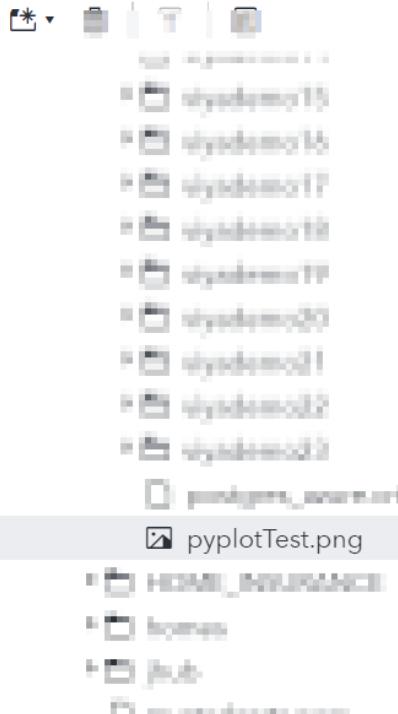
Log

Results



New Options View Open Save All

Explorer



Start Page

* Python.py x



Cancel



Copy to My Snippets

+ Code to Flow



Code

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 # might have to clear the graph using plt.clf() if you run your (changed) Python code
5 plt.clf()
6
7 t = np.arange(0.0, 2.0, 0.01)
8 s = 1 + np.sin(2*np.pi*t)
9 plt.plot(t, s)
10
11 plt.xlabel('time (s)')
12 plt.ylabel('voltage (mV)')
13 plt.title('Matplotlib graph (Python) - displayed using proc gslide')
14 plt.grid(True)
15
16 SAS.pyplot(plt, filepath='/export/pvs/sasdata/data', filename='pyplotTest')
```

Was kommt noch?

Fehler Erkennung im SAS Log

```
ERROR: Python Exception.  
Traceback (most recent call last):  
  File "<stdin>", line 5, in <module>  
  File "<stdin>", line 2, in <module>  
  File "<string>", line 6, in <module>  
ZeroDivisionError: division by zero  
>>>  
NOTE: The SAS System stopped processing this step because of errors.  
NOTE: PROCEDURE PYTHON used (Total process time):  
      real time            2.32 seconds  
      cpu time             0.40 seconds
```