Call Python from Stata

This cheat sheet demonstrates how to call Python from Stata. To learn about calling Stata from Python, type the following in Stata: **`help pystata module`**

Setup

Search for Python installations on the current system
`python search`

Set the Python version to be used
`python set exec` *pyexecutable*

List Python settings and system information
`python query`

Executing Python code in Stata

Enter the Python environment
`python``[:]`

Execute a Python script file
`python script` *filename(.py)*

*Type **`end`** in Python to exit Python environment
** **`python`** without a colon allows you to remain in the Python environment despite any errors

Executing Stata code in Python

`stata:` *command*

This syntax only works in the Python interactive environment. To issue a Stata command within a Python script, or embed a Stata command within a Python compound statement, see the **`stata()`** function on page 2.

Python modules

Check availability of a Python module
`python which` *modname*

Set additional module search paths
`python set userpath` *paths*

SFI module (Bidirectional connection between Stata and Python)

Import the SFI module
`import sfi`

Read values from the current Stata dataset
`sfi.Data.get()`

Import all classes of the SFI module
`from sfi import *`

Read values from the current Stata dataset
`Data.get()`

Import specified class(es) of the SFI module
`from sfi import Data`

Read values from the current Stata dataset
`Data.get()`

ValueLabel class

Create value label *lblname*
`ValueLabel.createLabel(`*lblname***`)`**

Set a value and label for value label *lblname*
`ValueLabel.setLabelValue(`*lblname, value, label***`)`**

Set value label for a variable
`ValueLabel.setVarValueLabel(`*var, lblname***`)`**

Data class

Read values from the current Stata dataset
`Data.get(``[`*var,obs,selectvar,valuelabel,…*`]`**`)`**

Store values in the current Stata dataset
`Data.store(`*var,obs,val*`[,`*selectvar*`]`**`)`**

Get total # of observations in the current Stata dataset
`Data.getObsTotal()`

Set the # of observations in the current Stata dataset
`Data.setObsTotal()`

Add a variable of type byte to the current Stata dataset
`Data.addVarByte(`*name***`)`**

Add a variable of type str to the current Stata dataset
`Data.addVarStr(`*name,length***`)`**

Frame class

Connect instance fr to an existing Stata frame *name*
fr **`= Frame.connect(`***name***`)`**

Create a new frame in Stata and return a new frame instance, *fr* ,that can be used to access it
fr **`= Frame.create(`***name***`)`**

Read values from the frame
fr`.`**`get(`**`[`*var,obs,selectvar,valuelabel,…*`]`**`)`**

Store values in the frame
fr`.`**`store(`***var,obs,val*`[,`*selectvar*`]`**`)`**

Get the # of observations in the frame
fr`.`**`getObsTotal()`**

**fr* is an example of an instance; replace *fr* with instance name of your choice

Matrix class

Get the data in a Stata matrix
`Matrix.get(``name[,rows,cols]`**`)`**

Store elements in a Stata matrix
`Matrix.store(`*`name,val`***`)`**

Create a Stata matrix
`Matrix.create(`*`name,nrows,ncols,initVal[, …]`***`)`**

Display a Stata matrix
`Matrix.list(``name[,rows,cols]`**`)`**

Macro class

Get the contents of a global macro
`Macro.getGlobal(`*`name`***`)`**

Get the contents of a local macro
`Macro.getLocal(`*`name`***`)`**

Set the value of a global macro
`Macro.setGlobal(`*`name,value[,vtype]`***`)`**

Set the value of a local macro
`Macro.setLocal(`*`name,value`***`)`**

Scalar class

Get the contents of a Stata string scalar
`Scalar.getString(`*`name`***`)`**

Get the value of a Stata numeric scalar
`Scalar.getValue(`*`name`***`)`**

Set the contents of a Stata string scalar
`Scalar.setString(`*`name,value`***`)`**

Set the value of a Stata numeric scalar
`Scalar.setValue(`*`name,value[,vtype]`***`)`**

SFIToolkit class

Output a string to the Stata Results window
`SFIToolkit.display(`*`str[, asis]`***`)`**

Output a string to the Stata Results window as an error
`SFIToolkit.errprint(`*`str[, asis]`***`)`**

Output standard Stata error message associated with return code *rc* to the Stata Results window
`SFIToolkit.error(`*`rc`***`)`**

Execute a Stata command
`SFIToolkit.stata(`*`command[, echo]`***`)`**

Example 1: Read Stata content in Python

```
webuse lbw, clear
python:
from sfi import Scalar, Matrix
stata: regress bwt i.smoke age
Scalar.getValue('e(r2_a)')
Matrix.list('e(b)')
end
```

is a comment indicator in Python

Example 2: Store Python calculations in Stata

```
sysuse auto, clear
python:
from sfi import Data,Scalar,SFIToolkit
fprice = Data.get('price', None, 'foreign')   # Read values of price for foreign cars
Scalar.setValue('forsum', sum(fprice))        # Store cumulative sum in scalar forsum
SFIToolkit.stata('display as text "Cumulative sum of price for foreign cars = " forsum')
pymake = Data.get('make')
pybrand = [m.split(" ")[0] for m in pymake]
Data.addVarStr('brand', 9)
Data.store('brand', None, pybrand)
end
```

Example 3: Define Stata's frame contents in Python

```
python:
from sfi import Frame
# Create the Stata frame speed with one variable and 3 observations
spd = Frame.create('speed')
spd.setObsTotal(3)
spd.addVarByte('spdlimit')
spd.store('spdlimit',None,[50, 60, 45])   # Store values in this variable

# Below, we make speed the current working frame and list the contents
stata: frame change speed
stata: list
end
```