Article
Heloisa Paiva · Sep 22 4m read

Python and IRIS in practice - with examples!

Here you'll find a simple program that uses Python in an IRIS environment and another simple program that uses ObjectScript in a Python environment. Also, I'd like to share a few of the troubles I went trough while learning to implement this.

Python in IRIS environment

Let's say, for example, you're in an IRIS environment and you want to solve a problem that you find easy, or more efficient with Python.

You can simply change the environment: create your method as any other, and in the end of it's name and specifications, you add [Language = python]:

```
Debug this method
ClassMethod Example(Arg As %String, OtherArg As Demo.Books.PD.Books) As %Status [ Language = python ]
{
    # Code anything you want in python here!
}
```

You can use any kinds of arguments in the method, and to access them you do the exact same thing you would in COS:

Say you have this %String argument, Arg, and an argument that comes from a custom class, OtherArg. This other class may have properties such as Title, and Author. You will want to access them like this:

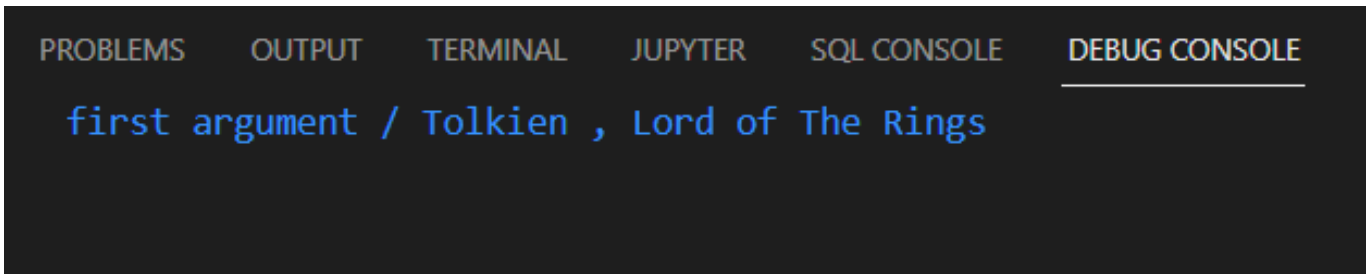
```
Debug this method
ClassMethod Example(Arg As %String, OtherArg As Demo.Books.PD.Books) [ Language = python ]
{
    # Code anything you want in python here!

    # Get the Arg argument:
    firstArgument = Arg

    # Get the OtherArg properties:
    secondArgumentTitle = OtherArg.Title
    secondArgumentAuthor = OtherArg.Author

    print(firstArgument, "/", secondArgumentAuthor, ",", secondArgumentTitle)
}
```

This method provides an output such as:



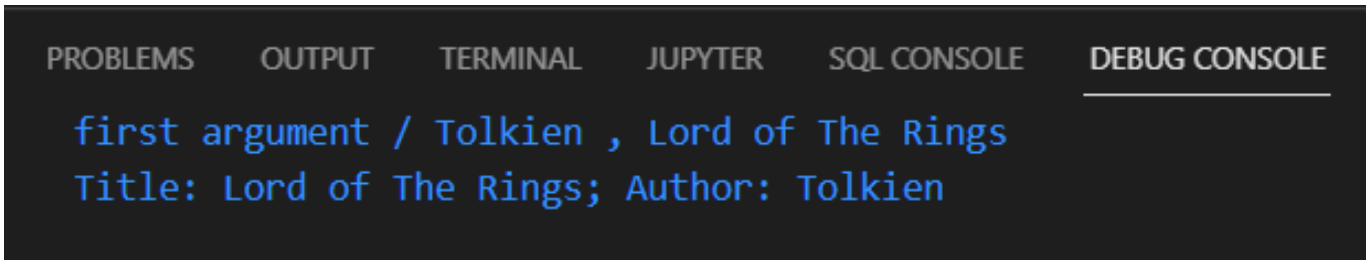
And, for accessing class methods, it's pretty much the same. Let's say we have a method in Demo.Books.PD.Books called "CreateString" that concatenates the title and author into something like "Title: <Title>; Author: <Author>".

Adding this to the end of our python method:

```
methodString = OtherArg.CreateString(OtherArg)

print(methodString)
```

will provide the following output:



(To access the method, you use OtherArg.CreateString(), but I chose to pass the same values in OtherArg to the CreateString method so the outputs would look alike and for the code to look simpler)

ObjectScript in Python Environment

Also, there might exist situations where you are in a Python environment, but you want ObjectScript's supplies to help you.

First, you will want to check a few items off of this list to be able to access your COS files from a Python environment in many ways (I won't necessarily use all of it here):

- Do you have the prerequisites? Check here
- Can you use a python external server? Check here
- Do you have the drivers you need? Download them here or Learn more here

You can always come back to those links or check the errors I ran into while first creating my python files with COS, if you find it helpful.

So let's start coding!

First, we will have to adapt a few things from COS to Python. Luckily, InterSystems already did it and we only have to type "import iris" to access everything!

```
1   # imports all python's methods and classes to adapt from COS
2   import iris
3
```

On the next step, we create a connection to the desired namespace, using a path with host, port and namespace (host:port/namespace), and providing user and password:

```
# info for connecting into the namespace you want: mine is SAMPLE
connection_string = "localhost:1972/SAMPLE"
username = "_system"
password = "sys"

# create the connection
connection = iris.connect(connection_string, username, password)

# create an iris object
irispy = iris.createIRIS(connection)
```

Notice how we created an IRIS OBJECT in the end, so we can use this connection to have access to everything we want in that namespace.

Finally, you code everything you want:

```
15  # access your classes and methods with .classMethod and other functions
16  request = irispy.classMethodObject('Demo.Books.PD.Books', '%New')
17  request.set('Title', 'Lord of The Rings')
18  request.set('Author', 'Tolkien')
19
20  status = irispy.classMethodValue('Demo.Python.Test', 'Insert', request)
21
22  # while also coding in python anything you want
23  print(status)
```

You can access methods with the irispy.classMethodValue() by providing the class' name, the method's name and arguments, manipulate objects with .set() (for properties) and many other possibilites, while treating everything on python the way you prefer.

For more information on functions provided by iris and how to use them, check Introduction to the Native SDK for Python

In this example, on line 16 I instanciated a Persistent class, to set it's properties Title and Author to Lord of the Rings and Tolkien on the following lines.

On line 20 I called a method from another class that saves the object to a table and returns a status if it worked. Finally , I print the status on line 23.

Mix!

Within an ObjectScript environment, you might want to use python's known libraries or your own custom files with functions and routines.

You can use the "import" command with Numpy, SciPy and everything you want (provided that you have them correctly installed: check here how to do that)

But also, if you want to access your local files, there are several ways to do that, and it's easy to find tutorials for that, since Python is so popular.

To me, the easiest to work with is the following:

```
    Debug this method
40  ClassMethod TesteImport() [ Language = python ]
41  {
42      import os
43      import sys
44      sys.path.append(os.path.abspath("C:/python/"))
45
46      import testesql
47
48      print(testesql.select())
49  }
50
```

Here I imported everything from the file testesql.py located in C:/python, and printed the results of the select() function

Extras - troubles I ran into

- **SHELLS**: When using Shells, remember that the Windows PowerShell works as an UNIX-based system (since it is based on .NET) and the Command Prompt is the one that will work with the Windows' examples in the official documentation. This can sound basic for some more experienced programmers, but if you're not paying enough attention you can lose quite some time on this so I found important to write a bit about it.
- **USER AND PRIVILEGES**: The current user for coding with ObjectScript from a Python environment needs to have privileges for the Namespace's resources. Remember that if you don't select any user, the current is UnknownUser, and if you don't select any namespaces, the current is USER. So, in the most simple access, you might want to follow: Management Portal > System Management > Security > Users > UnknownUser > Roles and select %DB_USER and save.
- **I DON'T KNOW WHAT'S GOING ON:** To check more information on errors you're having, you might want to follow Management Portal > System Explorer > SQL and type "SELECT * FROM %SYS.Audit ORDER BY UTCTimeStamp Desc" to get the most recent audit. There you'll find causes for errors such as IRIS_ACCESSDENIED() and much more you might be getting even outside IRIS environment.
- **PYTHON COMPILING ERROR:** You will want to avoid Methods names such as try() or functions that are already built in python. The compiler won't understand the difference from the Method to the function.

Thank you for reading, and please feel free to share suggestions, comments, doubts or just whatever you're developing!

#Python #InterSystems IRIS