記事
Mihoko Iijima · 2022年2月4日 7m read

Embedded Python 試してみました

開発者の皆さん、Python好きの皆さん、こんにちは！

ドキュメントをみながら IRIS 2021.2 に追加された Embedded Python を試してみました！

IRIS にログインしてるのに Pythonシェルに切り替えできて Python のコードが書けたり、Python で import iris するだけで SQL を実行できたりグローバルを操作できるので、おぁ！　という感じです。

ぜひ、みなさんも体感してみてください！

では早速。

まず、IRISにログインします。Windows ならターミナルを開きます。Windows 以外は以下実行します。

> IRIS のインストール方法を確認されたい方は、【はじめての InterSystems IRIS】セルフラーニングビデオ：基本その１：InterSystems IRIS Community Edition をインストールしてみよう！をチェックしてみてください！

```
iris session iris
```

> iris session の引数はインストール時指定のインスタンス名（構成名）です。インスタンス名が不明な場合は **iris list** を打つと確認できます。以下の例の場合は **IRIS** がインスタンス名です。

```
irisowner@dc47786c4ca9:~$ iris list
Configuration 'IRIS'   (default)
        directory:    /usr/irissys/
        versionid:    2021.2.0.649.0
        datadir:      /data/config/
        conf file:    iris.cpf  (SuperServer port = 1972, WebServer = 52773)
        status:       running, since Fri Feb  4 10:32:13 2022
        state:        warn
        product:      InterSystems IRISHealth
irisowner@dc47786c4ca9:~$
```

IRIS にログインした状態で Python シェルに切り替えてみましょう。

```
USER>do ##class(%SYS.Python).Shell()
```

```
Python 3.8.10 (default, Sep 28 2021, 16:10:42)
[GCC 9.3.0] on linux
Type quit() or Ctrl-D to exit this shell.
>>> print("??????")
??????
>>>
```

Python 書けますね！

次、IRIS にSample.Person
クラスがあるとします。後でSQLも実行したいので、永続クラスとして定義してあるとします。

```
Class Sample.Person Extends %Persistent

{

Property Name As %String;

Property Email As %String;

}
```

クラス定義を作成するのが面倒な場合は、以下 CREATE 文を実行でも大丈夫です。

```
CREATE TABLE Sample.Person (Name VARCHAR(50), Email VARCHAR(50))
```

> メモ：CREATE文の場合、列を記述した順序で列番号が付与されますが、クラス定義はオブジェクト指向の考えに基づいて定義されるため、カラム番号の考えがありません。そのため、SELECT * FROM *** をしたときの表示順が異なる場合があります。

では、この Sample.Person に対して Python シェル上でインスタンスを作成して、保存してみたいと思います！

まず最初に、IRISの操作をしたいときは、 **import iris** をします。

```
>>> import iris
>>> p=iris.cls("Sample.Person")._New()
>>> p.Name="????"
>>> p.Email="taro@mail.com"
>>> status=p._Save()
>>> print(status)
1
```

インポートした iris モジュールの cls()メソッドにクラス名を指定し、New()
メソッドでインスタンスを生成します。インスタンスに値を入れて保存するときは Save()
メソッドを実行します。

IRIS内の構文に似てますね。　　set p=##class(Sample.Person).%New() 保存は set status=p.%Save()

では、ちゃんとデータが入っているかSQLで確認しましょう！

```
>>> rs=iris.sql.exec("SELECT Name,Email from Sample.Person")
>>> for idx,row in enumerate(rs):
...  print(f"[{idx}]:{row}")
...
[0]:['????', 'taro@mail.com']
>>>
```

iris.sql.exec(”実行したいSQL文”)を指定して SQL を実行しています。実行結果（例では変数 rs）はリストで返るので、enumerate()関数を使ってFOR文で中身を取り出しています。

引数がある場合は、iris.sql.prepare() メソッドを利用します。引数の置き換え記号は❓を指定します。

```
>>> stmt=iris.sql.prepare("SELECT Name,Email from Sample.Person where ID<?")
>>> rs=stmt.execute(2)
>>> for idx,row in enumerate(rs):
...  print(f"[{idx}]:{row}")
...
[0]:['????', 'taro@mail.com']
>>>
```

今度は、INSERT文を実行して一人増やしてみます。

```
>>> rs=iris.sql.exec("INSERT INTO Sample.Person (Name,Email) VALUES('????','hana@mail
.com')")
```

SQLで登録を確認します。

```
>>> rs=iris.sql.exec("SELECT Name,Email from Sample.Person")
>>> for idx,row in enumerate(rs):
...     print(f"[{idx}]:{row}")
...
[0]:['????', 'taro@mail.com']
[1]:['????', 'hana@mail.com']
>>>
```

2件目もちゃんと登録できました。

今度は、2件目の花子さんをオブジェクトの文法を使ってオープンして名前を変更して保存します。

```
>>> hanako=iris.cls("Sample.Person")._OpenId(2)
>>> hanako.Name
'????'
>>> hanako.Name="HANAKO"
>>> hanako._Save()
1
```

また SQL で確認してみましょう。

```
>>> rs=iris.sql.exec("SELECT Name,Email from Sample.Person")
```

```
>>> for idx,row in enumerate(rs):
...     print(f"[{idx}]:{row}")
...
[0]:['????', 'taro@mail.com']
[1]:['HANAKO', 'hana@mail.com']
>>>
```

ローマ字に変わりましたね！Pyhon で IRIS の ObjectScript と同じことができました！

では、次に、クラス定義にメソッドを追加します。最初は ObjectScript で書いたものを呼び出してみます。

```
ClassMethod CreateEmail(uid As %String) As %String
{
    return uid_"@mail.com"
}
```

では Python から実行してみます。

```
>>> iris.cls("Sample.Person").CreateEmail("hanahana")
'hanahana@mail.com'
>>>
```

できましたー！

次、クラス定義にPythonでメソッドのコードを書いてみましょう！（そうなんです。書けちゃうんです。**[Language = python]** を付けたらできるんです！）

```
Method PythonPrint() [ Language = python ]
{
    print("\n????????" + self.Name + " ?????"+self.Email)
}
```

インスタンスメソッドなので、今オープンしている花子さんのデータを表示してみます。

```
>>> hanako.PythonPrint()
????????HANAKO ?????hana@mail.com
>>>
```

動きましたねー！

では、次はグローバルに挑戦です。[この記事で紹介した人物相関をグローバルで表現してみようの例]を使います。

```
>>> glo=iris.gref("^Relation")
>>> glo["Eren"]="??????"
>>> glo["Eren","Armin"]=""
>>> glo["Eren","Mikasa"]=""
>>> glo["Eren","Zeke"]=""
>>> glo["Armin"]="???????????"
>>> glo["Mikasa"]="???????????"
>>> glo["Zeke"]="????????"
>>>
```

iris.gref() メソッドを使用してグローバルの参照を変数に設定しています。例では、前で設定したグローバル変数 ^Relation の参照を取得するため引数に **"^Relation"** を指定してます。

グローバルの参照を利用して、IRIS内の操作と同様に値を設定したり、サブスクリプトにデータを入れたり、好きなように操作できます。

値を参照したい場合は、こんな感じです。

```
>>> glo["Eren"]
'??????'
>>>
```

次は、登場人物 エレン のお友達を探してみましょう。

友達情報は、^Relation("Eren") の第2サブスクリプトを探せばよいのですが、Python でも $Order() 関数 のイメージのまま記述できます。（**import iris** を実行した後の記述から書いています）

```
>>> glo=iris.gref("^Relation")
>>> sub=""
>>> while True:
...  sub=glo.order(["Eren",sub])
...  if (sub==None):
...   break
...  print(sub)
...
Armin
Mikasa
Zeke
>>>
```

無事、Python からもグローバル変数の第2サブスクリプトに設定したエレンのお友達が見つかりました。

いかがでしたでしょうか。

今回は、IRIS にログインした状態で Python を実行する方法を中心に試してみました。（IRIS にログインして Python シェルを起動して IRIS のテーブルやグローバルを操作しました。）

次は、Python のプログラムを IRIS から実行する方法などご紹介する予定です。お楽しみに〜

なお、ドキュメントにいろんなパタンのサンプルコードが掲載されています。

記事より前に試された方！ぜひ感想やこんなコード動かせたよ！の情報をお寄せください！お待ちしてます！

#Embedded Python #ObjectScript #Python #SQL #オブジェクトデータモデル #初心者 #InterSystems IRIS #InterSystems IRIS for Health

ソースURL:https://jp.community.intersystems.com/post/embedded-python-%E8%A9%A6%E3%81%97%E3%81%A6%E3%81%BF%E3%81%BE%E3%81%97%E3%81%9F