Artículo
Minoru Horita · 21 jul, 2020 Lectura de 10 min

Open Exchange

Python Gateway パートIII：基本機能

この連載記事では、InterSystemsデータプラットフォーム用のPython Gatewayについて説明します。
また、InterSystems IRISからPythonコードなどを実行します。 このプロジェクトは、InterSystems IRIS環境にPythonの力を与えます。

- 任意のPythonコードを実行する
- InterSystems IRISからPythonへのシームレスなデータ転送
- Python相互運用アダプタでインテリジェントな相互運用ビジネスプロセスを構築する
- InterSystems IRISからのPythonコンテキストの保存、調査、変更、復元

その他の記事

現時点での連載計画です（変更される可能性があります）。

- パート I：概要、展望、紹介
- パート II：インストールとトラブルシューティング
- パート III：基本機能 <-- 現在、この記事を参照しています
- パート IV：相互運用アダプタ
- パート V：Execute関数
- パート VI：動的ゲートウェイ
- パート VII：プロキシゲートウェイ
- パート VIII：使用事例とML Toolkit

はじめに

Pythonゲートウェイをインストールし、動作することを確認できました。早速使ってみましょう！
この記事では、Pythonのメインインターフェースである isc.py.Main について説明します。
このインターフェースは、次のような機能を持つメソッド（すべて %Status を返します）を提供します。

- コード実行
- データ転送
- 補助

コード実行

これらのメソッドを使用すると、任意のPythonコードを実行できます。

SimpleString

SimpleString は基本的なメソッドです。 次の4つのオプション引数を受け入れます。
- code - 実行するコードの文字列です。 複数の行は $c(10) で区切ってください。
- returnVariable - 返される変数の名前です。
- serialization - returnVariable のシリアライズ方法です。 0 - 文字列のシリアライズ（デフォルト）、1 - reprシリアライズ。
- result - 変数を参照で渡します。 returnVariable が書き込まれます。

以下に例を示します。

```
set sc = ##class(isc.py.Main).SimpleString("x='HELLO'", "x", , .var)
```

この例では、Pythonの x 変数の値に Hello を代入し、Pythonの x 変数を ObjectScriptの var
変数に代入して返しています。

ExecuteCode

ExecuteCode は SimpleString の制限を受けない、SimpleString の安全な代替手段です。
文字列は3 641
144文字に制限されているため、長いコードを実行する場合はストリームを使用する必要があります。
次の2つの引数を受け入れます。
- code - 実行するPythonコードの文字列またはストリームです。
- variable - （任意）code の実行結果をこのPython変数に設定します。

次のように使用します。

```
set sc = ##class(isc.py.Main).ExecuteCode("2*3","y")
```

この例では2に3を掛け、結果をPythonの y 変数に書き込んでいます。

データ転送

Pythonとの間でデータを転送します。

Python -> InterSystems IRIS

PythonからInterSystems IRISに変数の値を渡す方法は、シリアル化の必要性に応じて4種類から選べます。
- String は単純なデータ型とデバッグに使用します。
- Repr は単純なオブジェクトの保存とデバッグに使用します。
- JSON はInterSystems IRIS側でデータを処理しやすくするために使用します。
- Pickle はオブジェクトを永続化するために使用します。

以下のメソッドを使用すると、Pythonから変数を文字列またはストリームとして取得できます。

- GetVariable(variable, serialization, .stream, useString) - variable の serialization をstream に取得します。useString
  が1で、変数をシリアル化したものが文字列に収まる場合、ストリームの代わりに文字列が返されます。
- GetVariableJson(variable, .stream, useString) - 変数をJSONでシリアル化した結果を取得します。
- GetVariablePickle(variable, .stream, useString, useDill) -
  変数をPickle（またはDill）でシリアル化した結果を取得します。

y 変数を取得してみましょう。

```
set sc = ##class(isc.py.Main).GetVariable("y", , .val, 1)
w val
>6
```

InterSystems IRIS -> Python

そして最後に、InterSystems IRISから一部のデータをPythonに読み込みます。

- ExecuteQuery(query, variable, type, namespace) - SQLクエリから結果セット（pandasの データフレーム または リスト）を作成し、variable に設定します。 isc.py パッケージは namespace でアクセスできなければなりません。
- ExecuteGlobal(global, variable, type, start, end, mask, labels, namespace) - global のデータ（start から end まで）を、 type（タプルの list または pandas の dataframe）を持つPython変数に転送します。 maskおよび labels
  引数の仕様については、クラスのドキュメントとデータ転送のドキュメントを確認してください。
- ExecuteClass(class, variable, type, start, end, properties, namespace) -
  クラスデータをPythonのタプルリスト、またはpandasのデータフレームに転送します。 properties - データフレームを構成するプロパティをカンマで区切ったリストです。 * および ？
  ワイルドカードを使用できます。 デフォルトは *（すべてのプロパティ）です。 %%CLASSNAME プロパティは無視されます。 保存されているプロパティのみを使用できます。
- ExecuteTable(table, variable, type, start, end, properties, namespace) -
  クラスデータをPythonのタプルリスト、またはpandasのデータフレームに転送します。

ExecuteQuery は汎用的です（有効なSQLクエリはPythonに転送されます）。 ただし、ExecuteGlobal とそのラッパーである ExecuteClass と ExecuteTable の動作にはいくつかの制限があります。
その代わり、はるかに高速に動作します（ODBCドライバーより3〜5倍高速で、ExecuteQuery より20倍高速です）。 詳細は、データ転送のドキュメントを確認してください。
これらのメソッドはすべて、任意のローカルネームスペースからのデータ転送をサポートしています。 isc.py パッケージは namespace でアクセスできなければなりません。

ExecuteQuery

ExecuteQuery(query, variable, type, namespace) - 有効なSQLクエリの結果をPythonに転送します。
これは最も遅いデータ転送方法です。 ExecuteGlobal とそのラッパーが利用できない場合に使用してください。

引数：
- query - SQLクエリ。
- variable - Python側のターゲット変数。
- type - list または Pandasの dataframe を指定します。

ExecuteGlobal

ExecuteGlobal(global, variable, type, start, end, mask, labels, namespace) -
グローバルデータをPythonに転送します。

引数：

- global - ^を含まないグローバル名。
- variable - Python側のターゲット変数。
- type - list または Pandasの dataframe を指定します。
- start - 最初のグローバルキー。 整数でなければなりません。
- end - 最後のグローバルキー。 整数でなければなりません。
- mask - グローバル値のマスク（文字列）です。 マスクはグローバル値のフィールドの数よりも短い場合があります（この場合、最後のフィールドはスキップされます）。 マスクの書式を以下に記載します。
  - + フィールドをそのまま使用します。
  - - フィールドをスキップします。
  - b - ブール値（0 - False、それ以外 - True）。
  - d - 日付（$horologから起算。Windowsでは1970年以降、Linuxでは1900年以降の日付。詳細は注意事項を参照してください）。

    - t - 時間（$horolog、午前0時からの秒数）。
    - m - （現在）「年-月-日 時間:分:秒」形式のタイムスタンプ文字列。
- labels - カラム名の%リストです（最初の要素がキーとなるカラム名です）。
  したがって、リストの長さはマスク文字列の長さより1つ長くなければなりません。

ExecuteClass

ExecuteGlobal のラッパーです。 コンパイルされたクラス定義を効果的に解析し、ExecuteGlobal
引数を作成して呼び出します。

ExecuteClass(class, variable, type, start, end, properties, namespace) -
クラスデータをPythonのタプルリスト、またはpandasのデータフレームに転送します。 properties -
データフレームを構成するプロパティをカンマで区切ったリストです。 * および ？
ワイルドカードを使用できます。 デフォルトは *（すべてのプロパティ）です。 %%CLASSNAME
プロパティは無視されます。 保存されているプロパティのみを使用できます。

引数：
- class - クラス名。
- variable - Python側のターゲット変数。
- type - list または Pandasの dataframe を指定します。
- start - 最初のオブジェクトID。 整数でなければなりません。
- end - 最後のオブジェクトID。 整数でなければなりません。
- properties - データフレームを構成するプロパティをカンマで区切ったリストです。 * および ？
ワイルドカードを使用できます。 デフォルトは *（すべてのプロパティ）です。 %%CLASSNAME
プロパティは無視されます。 保存されているプロパティのみを使用できます。

特定のタイプ（%Date、%Time、%Boolean、%TimeStamp）のプロパティを除き、すべてのプロパティはそのまま転送されます。 これらはそれぞれのPythonデータ型に変換されます。

ExecuteTable

ExecuteClass のラッパーです。 テーブル名をクラス名に変換し、ExecuteClass を呼び出します。 署名：

ExecuteTable(table, variable, type, start, end, properties, namespace) -
クラスデータをPythonのタプルリスト、またはpandasのデータフレームに転送します。

引数：
- table - テーブル名。

その他の引数はそのまま ExecuteClass に渡されます。

注意事項

- ExecuteGlobal、ExecuteClass、ExecuteTable
  の実行速度は概して同じです（クラス定義の解析にはごくわずかな時間しかかからないため）。
- ExecuteGlobal は測定可能なワークロード（0.01秒超）で、ExecuteQuery よりも最大20倍高速です。
- ExecuteGlobal、ExecuteClass、ExecuteTable は、^global(key) = $lb(prop1, prop2, ..., propN)
  の構造を持つグローバルでのみ機能します（key は整数でなければなりません）。
- ExecuteGlobal、ExecuteClass、ExecuteTable の場合、サポートされている %Date の範囲は mktime
  の範囲（Windows：1970-01-01、Linux：1900-01-01、Mac）と同じです。
  この範囲から外れた日付を転送するには、%TimeStamp を使用してください。 または、pandas
  データフレームを使用してください（これはリストの制限であるため）。
- ExecuteGlobal、ExecuteClass、ExecuteTable
  の場合、ソース（グローバル、クラス、テーブル）および変数以外のすべての引数は省略可能です。

例

仮に isc.py.test.Person クラスがあるとしましょう。
この場合、次のようにすべてのデータ転送メソッドを使用できます。

```
// ??????????
set global = "isc.py.test.PersonD"
set class = "isc.py.test.Person"
set table = "isc_py_test.Person"
set query = "SELECT * FROM isc_py_test.Person"

// ?????
set variable = "df"
set type = "dataframe"
set start = 1
set end = $g(^isc.py.test.PersonD, start)

// ??0??????ExecuteGlobal
set sc = ##class(isc.py.Main).ExecuteGlobal(global, variable _ 0, type)

// ??1??????ExecuteGlobal
// ????????????????????????
// globalKey - ??????????
set labels = $lb("globalKey", "Name", "DOB", "TS", "RandomTime", "AgeYears", "AgeDeci
mal", "AgeDouble", "Bool")

// "globalKey" ??????????????????????????1????????
// ???? %%CLASSNAME ????????????
set mask = "-+dmt+++b"

set sc = ##class(isc.py.Main).ExecuteGlobal(global, variable _ 1, type, start, end, m
ask, labels)

// ???2ExecuteClass
set sc = ##class(isc.py.Main).ExecuteClass(class, variable _ 2, type, start, end)

// ??3?ExecuteTable
set sc = ##class(isc.py.Main).ExecuteTable(table, variable _ 3, type, start, end)

// ??4?ExecuteTable
set sc = ##class(isc.py.Main).ExecuteQuery(query, variable _ 4, type)
```

do ##class(isc.py.test.Person).Test() を呼び出すと、これらのデータ転送メソッドの動作を確認できます。

補助

補助的なメソッドです。

- GetVariableInfo(variable, serialization, .defined, .type, .length) -
  変数に関する情報（定義の有無、型、シリアライズ後の長さ）を取得します。
- GetVariableDefined(variable, .defined) - 変数定義の有無を取得します。
- GetVariableType(variable, .type) - 変数のFQCNを取得します。
- GetStatus() - Pythonで最後に発生した例外を返し、それをクリアします。
- GetModuleInfo(module, .imported, .alias) -
  モジュールのエイリアスを取得し、現在インポートされているかどうかを取得します。
- GetFunctionInfo(function, .defined, .type, .docs, .signature, .arguments) - 関数の情報を取得します。

要約

Python Gatewayを使用すると、InterSytems IRISとPythonをシームレスに統合できます。
これを使用することで、コードを実行してデータを双方向に転送できます。

リンク

- Python Gateway
- Python 3.6.7（64ビット版）のインストール
- Pythonのドキュメント/チュートリアル

イラスト付きガイド

ML Toolkitユーザーグループには、イラスト付きのガイドもあります。 ML Toolkitユーザーグループは、InterSystems社のGitHub組織の一部として設定されている非公開GitHubリポジトリです。 このリポジトリは、Python Gatewayを含むML
Toolkitコンポーネントをインストール、学習、またはすでに使用している外部ユーザーを対象としています。 ML Toolkitユーザーグループに参加するには、以下の内容を含む簡単なメールを MLToolkit@intersystems.com
宛に送信してください（グループメンバーが議論中にあなたを認識して特定するために必要です）。

- GitHubのユーザー名
- 氏名（英文字表記の名前、姓の順）
- 組織（勤務先、通学先、または在宅勤務）
- 役職（組織での実際の役職、「学生」、または「無所属」）
- 国（本拠地としている国）

#Python #Principiante #InterSystems IRIS
Ir a la aplicación en InterSystems Open Exchange