記事
Hiroshi Sato · 2020年10月5日 9m read

Caché .Net BindingアプリケーションをIRISの.Net Native APIを利用して書き換える方法（その３）

ここで紹介するサンプルは、以下のGitHubから入手可能です。

IRIS .Netサンプル

jpegファイルを読んで、IRISデータベースに格納するサンプル

上記GitHub上のinsertbinary\insertbinary\binread.csというファイル名です。

処理内容は、ファイルシステム上のjpeg形式のファイルを読み込んで、BLOB形式でIRISデータベースに格納します。

Caché ではADO.NET Managed Providerを使用して実装していましたが、それをIRISのInterSystems Managed Provider for .NETを使用して書き換えました。
（名前が変わっていますが、ADO.NETに関しては、機能はほとんど同じです）

従って、厳密に言うと.Net Native APIを使用していませんが、コネクションオブジェクトの使用方法は共通なので、この部分は、Native APIを使用していると言うこともできます。

Caché での実装は、以下の通りです。

```
// binread.cs
using System;
using System.IO;

CacheCommand   spCache;
CacheConnection   cnCache;
CacheDataAdapter   daCache;
CacheTransaction   txCache=null;
DataSet   dsCache;
DataTable   dtCache;
DataRow   drCache;

class BinaryRead {
  static void Main(string[] args) {
  //存在するファイルを指定する
  FileStream fs = new FileStream(
  @"c:\temp\picture\xxx.jpeg", FileMode.Open, FileAccess.Read)
  int fileSize = (int)fs.Length; // ファイルのサイズ
```

```
  byte[] buf = new byte[fileSize]; // データ格納用配列
  int readSize; // Readメソッドで読み込んだバイト数
  int remain = fileSize; // 読み込むべき残りのバイト数
  int bufPos = 0; // データ格納用配列内の追加位置
  readSize = fs.Read(buf, 0, fs.Length)
  string cacheConnectString = "Server = localhost;Port=1972;Namespace=User;Password=SYS;User ID =
SYSTEM;";

  cnCache = new CacheConnection(cacheConnectString);
  cnCache.Open();
  spCache = new CacheCommand("Insert into MyApp.Person2(Name, Picture) Values(?, ?)", cnCache, txCache);
   CacheParameter   pName   = new CacheParameter();
  pName.ParameterName   = "Name";
  pName.CacheDbType   = CacheDbType.NVarChar;
  pName.Direction   = ParameterDirection.Input;
  pName.Value   = "Hoge Hoge;
  CacheParameter   pPicture = new CacheParameter();
  pDOB.ParameterName   = "Picture";
  pName.CacheDbType   = CacheDbType.LONGVARBINARY;
  pDOB.Direction   = ParameterDirection.Input;
  pDOB.Value   = buf;
  spCache.ExecuteNonQuery();
  fs.Dispose();
  cnCache.close();
  }
}
```

IRISで書き換えたソースは以下の通りです。

```
// binread.cs
using System;
using System.IO;
using System.Data;
using InterSystems.Data.IRISClient;
using InterSystems.Data.IRISClient.ADO;

namespace binaryfileread {
  class binaryfileread
  {
  [STAThread]
  static void Main(string[] args) {
  IRISCommand spIRIS;
  IRISConnection cnIRIS;
  IRISTransaction txIRIS = null;

  //存在するファイルを指定する
  FileStream fs = new FileStream(
  @"c:\temp\test.jpeg", FileMode.Open, FileAccess.Read);
  int fileSize = (int)fs.Length; // ファイルのサイズ
  byte[] buf = new byte[fileSize]; // データ格納用配列
  long readSize; // Readメソッドで読み込んだバイト数
  int remain = fileSize; // 読み込むべき残りのバイト数
  readSize = fs.Read(buf, 0, (int)fs.Length);
  string IRISConnectString = "Server = localhost;Port=1972;Namespace=User;Password=SYS;User ID =
SYSTEM;";

  cnIRIS = new IRISConnection(IRISConnectString);
```

```
  cnIRIS.Open();
  spIRIS = new IRISCommand("Insert into MyApp.Person2(Name, Picture) Values(?, ?)", cnIRIS, txIRIS);

  IRISParameter pName = new IRISParameter();
  pName.ParameterName = "Name";
  pName.IRISDbType = IRISDbType.NVarChar;
  pName.Direction = ParameterDirection.Input;
  pName.Value = "Hoge Hoge";
  spIRIS.Parameters.Add(pName);

  IRISParameter pPicture = new IRISParameter();
  pPicture.ParameterName = "Picture";
  pPicture.IRISDbType = IRISDbType.LongVarBinary;
  pPicture.Direction = ParameterDirection.Input;
  pPicture.Value = buf;
  spIRIS.Parameters.Add(pPicture);

  spIRIS.ExecuteNonQuery();
  fs.Dispose();
  cnIRIS.Close();
  }
  }
}
```

CachéとIRISでは、ADO.NETに関しては、関数等の名前がCacheからIRISに変更になっているという違いがあります。

参照の変更

まず以前の参照を削除します。

Visual Studioのソリューションエクスプローラーの所で参照をクリックします。

表示されるInterSystems.Data.CacheClientを削除します。
（右クリックして削除を選ぶ）

次にプロジェクトメニューから参照の追加をクリックして、以下の２つのファイルを選択します。
（プロジェクトの.Net Frameworkバージョンに合わせて、それに対応するファイルを選択する
以下の例は、v4.5を選択）

c:\InterSystems\IRIS\dev\dotnet\bin\v4.5
InterSystems.Data.IRISClient.dll

jpegファイルを読んで、フォーム上にそのjpegファイルの内容を表示するサンプル

上記GitHub上の\VBImage\VBImage\Forms1.vbというファイル名です。

処理内容は、ファイルシステム上のjpeg形式のファイルを読み込んで、VB.NETのフォーム上のPictureオブジェクトとして表示します。

Caché では.NET Managed Providerの.Net Bindingを使用して実装していましたが、それをIRISの.Net Native APIをを使用して書き換えました。

Caché での実装は以下の通りです。

```
Imports InterSystems.Data.CacheClient
Imports InterSystems.Data.CacheTypes
Imports System.Drawing
Public Class Form1
  Private Sub Button1Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
  Dim cn As CacheConnection
  cn = New
CacheConnection("Server=127.0.0.1;Port=1972;Namespace=User;Username=system;Password=SYS")
  cn.Open()
  Dim img As System.IO.FileStream
  img = New System.IO.FileStream("C:\temp\test.jpg", System.IO.FileMode.Open, IO.FileAccess.Read)
  Dim f As User.Fax
  ' NEW -----------
  f = New User.Fax(cn)
  img.CopyTo(f.pic)
  f.memo = "abcde"
  f.Save()
  f.Dispose()
  MsgBox("Done!")
  cn.Close()
  End Sub

Private Sub Button2Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click
  Dim cn As CacheConnection
  cn = New
CacheConnection("Server=127.0.0.1;Port=1972;Namespace=User;Username=system;Password=SYS")
  cn.Open()
  Dim f As User.Fax
  f = User.Fax.OpenId(cn, 1)
  PictureBox1.Image = Image.FromStream(f.pic)
  cn.Close()
  End Sub
End Class
```

IRISで書き換えたソースです。

```
Imports InterSystems.Data.IRISClient
Imports InterSystems.Data.IRISClient.ADO
Imports System.Drawing
Imports System.IO
Public Class Form1
  Private Sub Button1Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
```

```
 Dim cn As IRISConnection
 cn = New IRISConnection("Server=127.0.0.1;Port=1972;Namespace=User;Username=system;Password=SYS")
 cn.Open()
 Dim iris As IRIS
 Dim irisobject As IRISObject
 Dim memstream As New MemoryStream
 Dim str As String
 Dim buf As Byte()
 Dim pic As IRISObject
 iris = IRIS.CreateIRIS(cn)
 irisobject = iris.ClassMethodObject("User.Fax", "%New")
 Dim img As System.IO.FileStream
 img = New System.IO.FileStream("C:\temp\test.jpg", System.IO.FileMode.Open, IO.FileAccess.Read)
 buf = New Byte(img.Length) {}
 img.Read(buf, 0, img.Length)
 str = System.Text.Encoding.GetEncoding("ISO-8859-1").GetString(buf)
 pic = irisobject.GetObject("pic")
 pic.InvokeVoid("Write", str)
 irisobject.InvokeIRISStatusCode("%Save")
 'Dim f As User.Fax
 ' NEW -----------
 'f = New User.Fax(cn)
 'img.CopyTo(f.pic)
 'f.memo = "abcde"
 'f.Save()
 'f.Dispose()
 MsgBox("Done!")
 cn.Close()
 End Sub

 Private Sub Button2Click(sender As System.Object, e As System.EventArgs) Handles Button2.Click
 Dim cn As IRISConnection
 Dim iris As IRIS
 Dim irisobject As IRISObject
 Dim pic As IRISObject
 Dim memorystream As New MemoryStream
 Dim buf As Byte()
 Dim str As String
 Dim len As Long
 cn = New IRISConnection("Server=127.0.0.1;Port=1972;Namespace=User;Username=system;Password=SYS")
 cn.Open()
 iris = IRIS.CreateIRIS(cn)
 irisobject = iris.ClassMethodObject("User.Fax", "%OpenId", 1)
 pic = irisobject.GetObject("pic")
 len = pic.InvokeLong("SizeGet")
 str = pic.InvokeString("Read", len)
 buf = System.Text.Encoding.GetEncoding("ISO-8859-1").GetBytes(str)
 memorystream.Write(buf, 0, len)
 PictureBox1.Image = Image.FromStream(MemoryStream)
 cn.Close()
 End Sub
End Class
```

良くコードを見るとわかりますが、IRIS版のほうが処理がかなり長くなっています。

結論を言うとこのケースはIRIS .Net Native APIを使用して書き換えるのは得策ではありません。
ADO.NETを使用したほうが簡単に処理できます。

上記GitHub上の\VBImageADO配下にADO.NETで書き換えたサンプルも用意しましたので、ご参考下さい。

#InterSystems IRIS

ソースURL:https://jp.community.intersystems.com/post/cach%C3%A9-net-binding%E3%82%A2%E3%83%97%E3%83%AA%E3%82%B1%E3%83%BC%E3%82%B7%E3%83%A7%E3%83%B3%E3%82%92iris%E3%81%AEnet-native-api%E3%82%92%E5%88%A9%E7%94%A8%E3%81%97%E3%81%A6%E6%9B%B8%E3%81%8D%E6%8F%9B%E3%81%88%E3%82%8B%E6%96%B9%E6%B3%95%EF%BC%88%E3%81%9D%E3%81%AE%EF%BC%93%EF%BC%89