
記事

[Hiroshi Sato](#) · 2020年7月27日 13m read

Cache Active X Bindingを利用したシステムをIRISに移行する方法

初めに

Caché ActiveX Bindingは、Visual Basicでクライアント・サーバー型のアプリケーション開発を支援するためにInterSystemsが提供してきたツールです。

CacheActiveX.dllとCacheObject.dllの2種類のバージョンが存在します。

IRISでは、CacheActiveX.dllは動作可能です。

CacheObject.dllはサポートしていません。

いずれにしろ誕生から既に20年以上が経過した非常に古いテクノロジーでマイクロソフト社も非推奨の古い規格ですので、今後も使い続けるのは得策ではありません。

Caché ActiveX Bindingの機能はIRISに用意されている.Net Native APIと.Net Managed Providerの機能を使って書き換え可能です。

ここでは、Caché ActiveX Bindingを使って書かれていたサンプルアプリケーションをIRISで動作するように移植した作業内容について解説します。

このサンプルは、以下のgithubサイトから入手可能です。

[ADBKサンプル](#)

ADBKアプリケーション

このサンプルアプリケーションは、20年以上も前にVB6サンプルとして作成されました。

VB6プロジェクトを.Netプロジェクトに変換

この作業は、VisM.OCXを利用したアプリケーションをIRISに移行する方法という記事にも同じ内容が記載されていますので、そちらをご参考ください。

[VisM.OCXを利用したアプリケーションをIRISに移行する方法](#)

CacheObject.dllの参照

Visual Studioを起動すると右側に表示されるソリューションエクスプローラーから参照の部分を開きます。

CacheObjectというのが見えるので、それを選択して、右クリックのメニューから削除をクリックします。

アプリケーション修正

それでは、IRISで動作するようにサンプルアプリケーションを修正していきましょう。

参照の追加

IRIS Native APIと.Net Managed Providerを利用するためにIRISの.Netライブラリーの参照を追加します。

プロジェクトメニューから参照の追加をクリックして参照ボタンを押して以下のファイルを選択します。

c:\InterSystems\IRIS\dev\dotnet\bin\4.5

- InterSystems.Data.Gateway64.exe
- InterSystems.Data.IRISClient.dll

ADBKMain.vbの修正

それではADBKMain.vbの修正を行いきましょう。

先ほど参照設定したライブラリーをImportします。

Option Explicit Onの後ろに以下の行を追加します。

```
Imports InterSystems.Data.IRISClient
Imports InterSystems.Data.IRISClient.ADO
```

class宣言の後ろの変数宣言の所を以下のように変更します。

```
Dim iris As IRIS
Dim irisobject As IRISObject
Dim irisconn As IRISConnection = New IRISConnection
'Dim mfactory As CacheObject.Factory
'Dim mobject As CacheObject.ObjInstance
'Const mclassname As String = "User.ADBK"
Const irisclassname As String = "User.ADBK"
```

CacheObjectの関連の変数をIRIS関連の変数に置き換えます。

IRISの場合は、サーバーとの通信のコネクションが別オブジェクトになっているため追加の変数設定が必要です。

次にフォームのロード処理の修正を行います。

```
'mfactory = CreateObject("CacheObject.Factory")
irisconn.ConnectionString = "Server = localhost; Log File=cprovider.log;Port=1972; Namespace=USER;
Password = SYS; User ID = system;"
irisconn.Open()
```

```
iris = IRIS.CreateIRIS(irisconn)
```

IRISの場合は、まずコネクションオブジェクトを作成する必要があります。
そして、そのConnectionStringプロパティにサーバー接続に必要な情報を設定します。
次にOpen()メソッドで接続を確立します。
接続が確立したら、IRISのインスタンスを生成します。
CacheObjectではFactoryと呼んでいたものと同等のものとなります。

```
'sdir = mfactory.ConnectDlg  
'If sdir <> "" Then  
'mfactory.Connect(sdir)
```

CacheObjectの場合は、接続情報が指定されていない場合は、接続情報を尋ねるダイアログボックスが表示されていましたが、
IRISのコネクションオブジェクトにはその機能がないので、すべてコメントアウトします。

次に新規ボタンが押された時の処理を変更しましょう。

CmdNewClickの処理になります。

以下のように書き換えます。

```
irisobject = iris.ClassMethodObject(irisclassname, "%New")  
If irisobject Is Nothing Then  
MsgBox("新しいオブジェクトを作成できません。")  
End If  
'mobject = mfactory.New(mclassname)  
'If mobject Is Nothing Then  
'MsgBox("新しいオブジェクトを作成できません。")  
'End If
```

CacheObjectの場合は、FactoryのNewメソッドで新しいオブジェクトの生成を行いましたが、
IRISでは、ClassMethodObjectメソッドで第一パラメータで指定したクラスのクラスメソッド%Newメソッドを呼び出すように変更します。

%Newメソッドは、OREFを返すので、ClassMethodObjectメソッドを使います。
ClassMethodXXXは、戻り値のタイプ別に用意されています。

次に保存ボタンが押された時の処理を変更します。

CmdUpdateClickの処理になります。

```
'mobject.sysSave()  
irisobject.Invoke("%Save")
```

インスタンスメソッドを呼び出す方法は、Invokeメソッドでパラメータに呼び出すメソッドの名前を指定します。

次に削除ボタンが押された時の処理を変更します。

CmdDeleteClickの処理になります。

```
'mobject.sysDeleteId(id)  
irisobject.Invoke("%DeleteId", id)
```

同様にInVokeメソッドを呼び出すように変更します。

次に検索ボタンが押された時の処理を変更します。

CmdFindClickの処理になります。

```
'mobject = FindByName.ShowDialogRenamed(mfactory)
irisobject = FindByName.ShowDialogRenamed(iris, irisconn)
```

検索用のダイアログを表示する際に渡すパラメータを追加する必要があります。
コネクションオブジェクトを追加で渡す必要があります。
また戻り値もIrisobjectに変更します。

次に CmdUpdateClickの処理を変更します。

'UPGRADEWARNING: オブジェクト mobject.ANAME の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.ANAME = TxtNAME.Text
irisobject.Set("ANAME", TxtNAME.Text)
```

'UPGRADEWARNING: オブジェクト mobject.ASTREET の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.ASTREET = TxtADDRESS.Text
irisobject.Set("ASTREET", TxtADDRESS.Text)
```

'UPGRADEWARNING: オブジェクト mobject.AZIP の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.AZIP = TxtZIP.Text
irisobject.Set("AZIP", TxtZIP.Text)
```

'UPGRADEWARNING: オブジェクト mobject.ABTHDAY の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.ABTHDAY = TxtDOB.Text
irisobject.Set("ABTHDAY", TxtDOB.Text)
```

'UPGRADEWARNING: オブジェクト mobject.APHHOME の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.APHHOME = TxtTELH.Text
irisobject.Set("APHHOME", TxtTELH.Text)
```

'UPGRADEWARNING: オブジェクト mobject.APHOTH1 の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.APHOTH1 = TxtTELO.Text
irisobject.Set("APHWORK", TxtTELO.Text)
```

```
On Error GoTo actionSaveError
```

'UPGRADEWARNING: オブジェクト mobject.sysSave の既定プロパティを解決できませんでした。 詳細については、'ms-

help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"
をクリックしてください。

```
'mobject.sysSave()  
irisobject.Invoke("%Save")
```

IRISObjectでは、プロパティに直接アクセスができず代わりにSetメソッドで設定する必要があります。

また保存は、インスタンスメソッドの%Save()をInvokeメソッドで起動します。

これでADBKMain.vbの修正は終了です。

FindByNme.vbの修正

続いてFindByNme.vbの内容を変更していきましょう。
まずはIRISライブラリーのインポートが必要です。

Option Explicit Onの後ろに以下の行を追加します。

```
Imports InterSystems.Data.IRISClient  
Imports InterSystems.Data.IRISClient.ADO
```

次に変数宣言の所を以下のように変更します。

```
'Dim RS As CacheObject.ResultSet  
'Dim mfactory As CacheObject.Factory  
'Dim mobject As CacheObject.ObjInstance  
Dim iris As IRIS  
Dim irisobject As IRISObject  
Dim irisconn As IRISConnection  
'Const mclassname As String = "User.ADBK"  
Const irisclassname As String = "User.ADBK"
```

IRISにはResultSetオブジェクトがありませんので、代替の方法で処理する必要があります。
詳細は、後程説明します。

次にShowDialogRenamed関数の処理を変更します。

これはADBKMainから検索ボタンを押したときに呼ばれる処理になります。

```
'Public Function ShowDialogRenamed(ByRef factory As CacheObject.Factory) As CacheObject.ObjInstance  
Public Function ShowDialogRenamed(ByRef irisfactory As IRIS, ByRef irisconnection As IRISConnection) As  
IRISObject
```

パラメータとしてirisconnectionを追加する必要があるので、追加します。

```
'mfactory = factory  
iris = irisfactory  
irisconn = irisconnection
```

コネクションオブジェクトの設定を追加します。

```
'Set mfactory = Nothing
```

```
'ShowDialogRenamed = mobject  
ShowDialogRenamed = irisobject
```

戻り値をIRISObjectを返すように変更します。

次に検索ボタンが押された時の処理を変更します。

CmdFindClickの処理になります。

```
'RS = mfactory.ResultSet("User.ADBK", "ByName")  
'RS.Execute(TxtSNAME.Text) ' ByName takes a single argument  
Dim SQLtext As String = "call sqluser.ADBKbyname(?)"  
Dim Command As IRISCommand = New IRISCommand(SQLtext, irisconn)  
Dim Nameparam As IRISParameter = New IRISParameter("Namecol", IRISDbType.NVarChar)  
Nameparam.Value = TxtSNAME.Text  
Command.Parameters.Add(Nameparam)  
Dim Reader As IRISDataReader = Command.ExecuteReader()
```

```
While Reader.Read()  
ListLookupName.Items.Add(Reader.Item(Reader.GetOrdinal("ANAME")))  
End While  
Reader.Close()  
Command.Dispose()
```

```
' 取得した名前リストをListBoxに展開  
'While RS.Next  
'UPGRADEWARNING: オブジェクト RS.GetDataByName() の既定プロパティを解決できませんでした。 詳細  
については、 'ms-  
help://MS.VSCC.v90/dvcommoner/local/redirect.htm?keyword="6A50421D-15FE-4896-8A1B-2EC21E9037B2"  
をクリックしてください。  
'ListLookupName.Items.Add(RS.GetDataByName("ANAME"))  
'End While
```

IRISではResultSetメソッドがありませんので、代替手段で書き換えます。
クエリはSQLのcall文で置き換え可能です。
但し、サーバー側のクエリ定義にsqlProc属性をつける必要があります。

```
Query ByName(Name As %String) As %SQLQuery(CONTAINID = 1) [ SqlProc ]
```

IRISCommandオブジェクトとIRISDataReaderオブジェクトを使用してクエリを処理します。
パラメータは、IRISParameterオブジェクトを使って定義します。

このあたりは、MicrosoftのADO.NETの仕様に基づき実装されているので、詳細はドキュメントを確認してください。

次にOKボタンが押された時の処理を変更します。

CmdOKClickの処理になります。

```
'nameRenamed = VB6.GetItemString(ListLookupName, ListLookupName.SelectedIndex)  
nameRenamed = ListLookupName.Items(ListLookupName.SelectedIndex).ToString()
```

これはIRIS対応とは関係ないのですが、動作しなかったのが、変更しました。

VB6との互換性がない部分だと想定しています。

```
'RS = mfactory.DynamicSQL("SELECT * FROM ADBK WHERE ANAME = ?")
'RS.Execute(nameRenamed) ' 値を '?' にバインド
'RS.Next()
Dim SQLtext As String = "SELECT * FROM ADBK WHERE ANAME = ?"
Dim Command As IRISCommand = New IRISCommand(SQLtext, irisconn)
Dim Nameparam As IRISParameter = New IRISParameter("Name_col", IRISDbType.NVarChar)
Nameparam.Value = nameRenamed
Command.Parameters.Add(Nameparam)
Dim Reader As IRISDataReader = Command.ExecuteReader()
Reader.Read()
```

先ほどと同様にIRISCommandとIRISDataReaderを使って書き換えます。

```
'id = RS.GetDataByName("AID")
id = Reader.Item(Reader.GetOrdinal("AID"))
```

フィールドの値の取得もさきほどと同様に書き換えます。

```
'mobject = mfactory.OpenId(mclassname, id)
'If mobject Is Nothing Then
irisobject = iris.ClassMethodObject(irisclassname, "%OpenId", id)
If irisobject Is Nothing Then
```

次は、検索の結果取得されたidを使って、オブジェクトインスタンスをオープンする処理もClassMethodObjectメソッドを使って実装します。

```
'CType(ADBKMain.Controls("TxtNAME"), Object).Text = mobject.ANAME
CType(ADBKMain.Controls("TxtNAME"), Object).Text = irisobject.Get("ANAME")
'CType(ADBKMain.Controls("TxtZIP"), Object).Text = mobject.AZIP
CType(ADBKMain.Controls("TxtZIP"), Object).Text = irisobject.Get("AZIP")
'CType(ADBKMain.Controls("TxtADDRESS"), Object).Text = mobject.ASTREET
CType(ADBKMain.Controls("TxtADDRESS"), Object).Text = irisobject.Get("ASTREET")
'CType(ADBKMain.Controls("TxtTELH"), Object).Text = mobject.APHHOME
CType(ADBKMain.Controls("TxtTELH"), Object).Text = irisobject.Get("APHHOME")
'CType(ADBKMain.Controls("TxtTELO"), Object).Text = mobject.APHOTH1
CType(ADBKMain.Controls("TxtTELO"), Object).Text = irisobject.Get("APHWORK")
'CType(ADBKMain.Controls("TxtAGE"), Object).Text = mobject.AAGE
CType(ADBKMain.Controls("TxtAGE"), Object).Text = irisobject.Get("AAGE")
'CType(ADBKMain.Controls("TxtDOB"), Object).Text = mobject.ABTHDAY
CType(ADBKMain.Controls("TxtDOB"), Object).Text = irisobject.InvokeString("ABTHDAYLogicalToOdbc",
irisobject.Get("ABTHDAY"))
```

IRISObjectでは、プロパティに直接アクセスができず代わりにGetメソッドで取得する必要があります。

DOB（誕生日）は、内部値（\$Horolog）をODBC形式の日付に変換する処理を加えています。

以上で修正は終了です。

最後に

既存の資産を生かしつつ、ActiveX Bindingを使用したアプリケーションの移行が簡単にできるということをご理解いただき、アプリケーションの移行にチャレンジしていただきたいと思います。

もう一つ、サーバー側の処理は全く修正していない点も強調しておきたいと思います。

[#.NET](#) [#InterSystems](#) [IRIS](#)

ソースURL:<https://jp.community.intersystems.com/post/cache-active-x-binding%E3%82%92%E5%88%A9%E7%94%A8%E3%81%97%E3%81%9F%E3%82%B7%E3%82%B9%E3%83%86%E3%83%A0%E3%82%92iris%E3%81%AB%E7%A7%BB%E8%A1%8C%E3%81%99%E3%82%8B%E6%96%B9%E6%B3%95>