Technical Note

Creating and Customizing REP++ Typed Instances

Author: R&D Department

Publication date: May 3, 2006

Revised: September 2006

Revised: December 2006

Revised: May 2010

Creating and Customizing REP++ Typed Instances

Overview

REP++ provides a library of meta-objects used to represent different types of data. A meta-object is a generic object that can stand for different object types according to the metadata. A concrete object is an object that can represent only one entity of the system.

A typed instance is a wrapper object that encapsulates a meta-object, transforming it into a concrete object. It contains the values of the object but delegates all the work to the meta-object. Typed instances are defined by subclassing REP++ objects. They can be generated automatically using the **REP++Typed Instance Wizard**. These generated classes can then be customized.

This document focuses on the way the REP++ typed instances are created and customized.

Note: code samples are given in Visual Basic®.

Subclassing the REP++ objects

REP++ provides a mechanism to subclass the different objects in its hierarchy. Subclassing allows the user to customize these objects by adding methods and properties or by overriding the basic methods to change the default behaviour of the object.

This subclassing mechanism is implemented using the class factory design pattern. The class factory allows the creation of the different custom objects. To use this mechanism, you should define your REP++ subclass object, define a class factory and register it in the REP++ **Application** object. Once registered, the class factory becomes responsible for creating the REP++ objects, and the REP++ **Application** will delegate the task of creating new instances to this class. For more details on subclassing REP++ objects, please refer to the technical note *Subclassing and Customizing REP++ Objects*.

Using the REP++ typed instance factory

REP++ class library defines a built-in class factory, **TypedInstanceFactory**. This class is a factory for the RowsetTree and Rowset objects. The typed instances are subclasses that use the REP++ subclassing mechanism to define a concrete object with typed properties. These typed instances can be generated using the **REP++Typed Instances Wizard** (see below).

The **TypedInstanceFactory** class implements the **IObjFactory** interface and defines methods for the registration of the subclasses for the **RowsetTree** and **Rowset** objects. Each **RowsetTreeDef** and **RowsetDef** can be associated with a different subclass manually or declaratively using a custom attribute. The **AddRowsetTreeDef** and **AddRowsetDef** methods allow the manual registration of subclasses, while the **RegisterAll** methods allow the declarative registration of subclasses.

- **AddRowsetTreeDef**(*rowsetTreeDef* As RowsetTreeDef, *type* As Type). Registers *type* as the type of **RowsetTree** to create from the *rowsetTreeDef* parameter.

- **AddRowsetDef**(*rowsetDef* As RowsetDef, *type* As Type). Registers *type* as the type of the **Rowset** to create from the *rowsetDef* parameter.
- **RegisterAll**(*assembly* As System.Reflection.Assembly). Registers each typed instance class defined in an assembly as the type of the object to create for the **RowsetTreeDef** or **RowsetDef** object represented by this typed instance. A typed instance class is identified by the ***TypedInstanceClass*** attribute. This attribute holds two parameters. The first parameter defines the name of the **RowsetTreeDef** or **RowsetDef** attached to the typed instance. The second parameter represents the priority of the typed instance class. It is used when more than one typed instance class is defined for the same **RowsetTreeDef** or **RowsetDef**. In this case, the class having the highest priority will be registered.

```
        ''' <summary>
        ''' Typed Instance of RowsetTree CLIENT
        ''' </summary>
        <RepPP.TypedInstanceClass("CLIENT", 100)> _
        Public MustInherit Class BaseRTCCLIENT
            Inherits RepPP.TypedRowsetTreeBase

        End Class
```

- **RegisterAll**(*OnlyMarked* As Boolean). Registers each typed instance class defined in all the assemblies of the project as the type of the object to create for the **RowsetTreeDef** the **RowsetDef** object represented by this typed instance. Calling the method with the parameter *OnlyMarked* as true finds the typed instances only in the marked assemblies. An assembly is marked as an assembly that contains typed instances when it has the ***AssemblyIncludeTypedInstances*** attribute.

```
<Assembly: RepPP.AssemblyIncludeTypes>
```

Generating typed instances

To generate a typed instance class, you can use the **REP++Typed Instances Wizard**. This wizard allows you to generate typed instances for RowsetDef or RowsetTreeDef objetcs. When you generate a typed instance for a RowsetTreeDef, a typed instance is automatically generated for each RowsetDef of this RowsetTreeDef. Here is an example of a generated RowsetTreeDef typed instance and the typed instance of its root RowsetDef.

```
' Rep++ Generated file. Do not modify

Imports System

    ''' <summary>
    ''' Typed Instance of RowseTree CLIENT
    ''' </summary>
    <RepPP.TypedInstanceClass("CLIENT", 100)> _
    Public MustInherit Class BaseRTCCLIENT
        Inherits RepPP.TypedRowsetTreeBase

        ''' <summary>
        ''' Class constructor.
        ''' </summary>
        ''' <param name="pobj">    Reference to the internal object</param>
        Public Sub New(pobj As IntPtr)
            MyBase.New(pobj)
        End Sub

        ''' <summary>
        ''' Create:               Create a new instance.
        ''' </summary>
        ''' <param name="app">    Rep++ application object</param>
```

```
        ''' <returns>
        ''' New instance
        ''' </returns>
        Public Shared Function Create(app As RepPP.Application) As RTCCLIENT
            Return CType(app.RowsetTreeDefs("CLIENT").RowsetTrees.Add(), RTCCLIENT)
        End Function

        ''' <summary>
        ''' Create:                     Create a new instance of the specified type.
        ''' </summary>
        ''' <param name="app">          Rep++ application object</param>
        ''' <param name="typeWrapper">  Type of the wrapper class to create</param>
        ''' <returns>
        ''' New instance
        ''' </returns>
        Public Shared Function Create(app As RepPP.Application, _
                                      typeWrapper As System.Type) As RTCCCLIENT
            Dim rowsetTreeRetVal  As RepPP.RowsetTree
            Dim rowsetTreeDef     As RepPP.RowsetTreeDef
            Dim typeOld           As System.Type
            Dim typedInstFact     As RepPP.TypedInstanceFactory

            rowsetTreeDef = app.RowsetTreeDefs("CLIENT")
            typedInstFact = CType(app.GetObjFactory(GetType(RepPP.RowsetTree)), _
                                  RepPP.TypedInstanceFactory)
            typeOld      = typedInstFact.GetTypeAssociateWithRowsetTreeDef(rowsetTreeDef)
            typedInstFact.AddRowsetTreeDef(rowsetTreeDef, typeWrapper)
            Try
                ciRetVal = rowsetTreeDef.RowsetTrees.Add()
            Finally
                typedInstFact.AddRowsetTreeDef(rowsetTreeDef, typeOld)
            End Try
            Return CType(rowsetTreeRetVal, RTCCLIENT)
        End Function

        ''' <summary>
        ''' Gets the wrapper for the instance of rowset CLIENT
        ''' </summary>
        Public ReadOnly Property rsCLIENT As RCCLIENT
            Get
                Return(CType(RootRowset, RCCLIENT))
            End Get
        End Property

        ''' <summary>
        ''' Gets the wrapper for the instance of rowset ADDRESS
        ''' </summary>
        Public ReadOnly Property rsADDRESS As RCADDRESS
            Get
                Return(CType(GetCurrentRowset(RowsetTreeDef.FindNode("ADDRESS")), _
                             RCADDRESS))
            End Get
        End Property

        ''' <summary>
        ''' Gets the wrapper for the instance of rowset PHONE
        ''' </summary>
        Public ReadOnly Property rsPHONE As RCPHONE
            Get
                Return(CType(GetCurrentRowset(RowsetTreeDef.FindNode("PHONE")), _
                             RCPHONE))
            End Get
        End Property
    End Class ' class BaseRTCLIENT


' Rep++ Generated file. Do not modify

Imports System

    ''' <summary>
```

```
    ''' Typed Instance of rowset CLIENT
    ''' </summary>
    <RepPP.TypedInstanceClass("CLIENT", 100)>
    Public MustInherit Class RCCLIENT
        Inherits RepPP.TypedRowsetBase
        ''' <summary>
        ''' Class constructor.
        ''' </summary>
        ''' <param name="pobj">    Reference to the internal object</param>
        Public Sub New(pobj As IntPtr)
            MyBase.New(pobj,
                   New string() { _
                                                "CLIENTCODE", _
                                                "CLIENTFIRSTNAME", _
                                                "CLIENTLASTNAME", _
                                                "CLIENTTYPE", _
                                                "CLIENTSALESTODATE", _
                                                "CIECODE", _
                                                "CIENAME", _
                                                "CREATIONDATE", _
                                                "MODIFICATIONDATE", _
                                                "DESCRIPTION" _
                                               })
        End Sub

        ''' <summary>
        ''' Create:               Create a new instance.
        ''' </summary>
        ''' <param name="app">    Rep++ application object</param>
        ''' <returns>
        ''' New instance
        ''' </returns>
        Public Shared Function Create(app As RepPP.Application) As RCCLIENT
            Return CType(app.RowsetDefs("CLIENT").Rowsets.Create(RepPP.RowsetType.sdHorizontal), _
                         RCCLIENT)
        End Function

        ''' <summary>
        ''' Create:               Create a new instance of the specified type.
        ''' </summary>
        ''' <param name="app">    Rep++ application object</param>
        ''' <param name="type">   Type of the wrapper class to create</param>
        ''' <returns>
        ''' New instance
        ''' </returns>
        Protected Shared Function Create(ByVal app As RepPP.Application, type As Type) _
                                         As RCCLIENT
            Dim rowsetRetVal    As RepPP.Rowset
            Dim rowsetDef       As RepPP.RowsetDef
            Dim typeOld         As Type
            Dim typedInstFact   As RepPP.TypedInstanceFactory

            rowsetDef       = app.RowsetDefs("CLIENT")
            typedInstFact   = CType(app.GetObjFactory(GetType(RepPP.Rowset)), _
                                    RepPP.TypedInstanceFactory)
            typeOld         = typedInstFact.GetTypeAssociateWithRowsetDef(rowsetDef)
            typedInstFact.AddRowsetDef(rowsetDef, type)
            Try
                rowsetRetVal = grp.Rowsets.Create(RepPP.RowsetType.sdHorizontal)
            Finally
                typedInstFact.AddRowsetDef(rowsetDef, typeOld)
            End Try
            Return CType(rowsetRetVal, RCCLIENT)
        End Function

        ''' <summary>
        ''' Client Code
        ''' </summary>
        Public ReadOnly Property fldCLIENTCODE As RepPP.Field
            Get
                Return(CType(Fields("CLIENTCODE"), RepPP.Field))
```

```
            End Get
        End Property

        ''' <summary>
        ''' First Name
        ''' </summary>
        Public ReadOnly Property fldCLIENTFIRSTNAME As RepPP.Field
            Get
                Return(CType(Fields("CLIENTFIRSTNAME"), RepPP.Field))
            End Get
        End Property

        ''' <summary>
        ''' Last Name
        ''' </summary>
        Public ReadOnly Property fldCLIENTLASTNAME As RepPP.Field
            Get
                Return(CType(Fields("CLIENTLASTNAME"), RepPP.Field))
            End Get
        End Property

    . . .

    End Class ' class GCCLIENT
```

Defining your custom typed instance class

Your custom typed instance class should inherit from the generated typed instance. You can register this new class manually or declaratively by specifying a priority greater than 100 for the ***TypedInstanceClassAttribute*** attribute. When you subclass a RowsetDef typed instance, the new class will be used by all RowsetTrees that include this RowsetDef. If you want to subclass a RowsetDef typed instance for a specific RowsetTreeDef, you should define a new class that inherits from the generated RowsetTreeDef typed instance.

Defining a custom typed instance for the CLIENT RowsetDef

1. Define a new class that inherits from the generated typed instance for the CLIENT RowsetDef.
2. Add a constructor to this class: the constructor should have a single parameter of type integer.
3. Set the priority of this class to 200.

```
''' <summary>
''' Custom typed instance of rowset CLIENT
''' </summary>
<RepPP.TypedInstanceClass("CLIENT", 200)> _
Public Class MyRCClient
    Inherits Internal.BaseRCCLIENT
    ''' <summary>
    ''' Class constructor.
    ''' </summary>
    ''' <param name="pobj">    Reference to the internal object</param>
    Public Sub New(ByVal pobj As IntPtr)
        MyBase.New(pobj)
    End Sub
End Class
```

Defining a custom typed instance for the CLIENT RowsetDef only when it is used in the CLIENT RowsetTreeDef

1. Define a new class that inherits from the generated typed instance for the CLIENT RowsetTreeDef.

2. Add a constructor to this class.
3. Set the priority of this class to 200.

```
''' <summary>
''' Custom typed instance for the RowsetTree CLIENT
''' </summary>
<RepPP.TypedInstanceClass("CLIENT", 200)> _
Public Class MyRTClient
    Inherits Internal.BaseRTCCLIENT
    ''' <summary>
    ''' Class constructor.
    ''' </summary>
    ''' <param name="pobj">    Reference to the internal object</param>
    Public Sub New(ByVal pobj As IntPtr)
        MyBase.New(pobj)
    End Sub
End Class
```

Deciding at runtime which wrapper class to create

In the CLIENT contact program, suppose that you want to define a **ClientCompany** class or a **ClientPerson** class, depending if the client is a company or a person. These two classes are subclasses of the generated RowsetTreeDef typed instance class. To be able to decide at runtime which wrapper class to create, the following method is generated in the client typed instance:

```
''' <summary>
''' Create:             Create a new instance of the specified type.
''' </summary>
''' <param name="app">          Rep++ application object</param>
''' <param name="typeWrapper">  Type of the wrapper class to create</param>
''' <returns>
''' New instance
''' </returns>
Public Shared Function Create(app As RepPP.Application, _
                              typeWrapper As System.Type) As RTCCCLIENT
    Dim rowsetTreeRetVal  As RepPP.RowsetTree
    Dim rowsetTreeDef     As RepPP.RowsetTreeDef
    Dim typeOld           As System.Type
    Dim typedInstFact     As RepPP.TypedInstanceFactory

    rowsetTreeDef = app.RowsetTreeDefs("CLIENT")
    typedInstFact = CType(app.GetObjFactory(GetType(RepPP.RowsetTree)), _
                          RepPP.TypedInstanceFactory)
    typeOld       = typedInstFact.GetTypeAssociateWithRowsetTreeDef(rowsetTreeDef)
    typedInstFact.AddRowsetTreeDef(rowsetTreeDef, typeWrapper)
    Try
        ciRetVal = rowsetTreeDef.RowsetTrees.Add()
    Finally
        typedInstFact.AddRowsetTreeDef(rowsetTreeDef, typeOld)
    End Try
    Return CType(rowsetTreeRetVal, RTCCLIENT)
End Function
```

In this function, the typed instance associated with the CLIENT RowsetTreeDef is saved. The type for the new wrapper class is then registered, and the RowsetTree created. Finally, the old type is restored.

The following example shows you how to create an object for the wrapper class **ClientPerson:**

```
Dim person As ClientPerson
```

```
person = CType(RTCLIENT.Create(app), ClientPerson)
```