RepVB Object

The Repository Add-in for Visual Basic (RepVB) manages interaction between Visual Basic and the repository. To initiate the mediation, you must register RepVB as an add-in. Thereafter, each time you create or modify a Visual Basic project, RepVB can respond to the events that Visual Basic triggers. When a Visual Basic event indicates a change in the structure of a project, RepVB responds by updating the contents of the repository. In this way, RepVB ensures that the repository reflects the current structure of your projects.

RepVB also triggers events to which *your* Visual Basic add-ins can respond. To respond to RepVB events, you must implement an event sink object. RepVB supports five different event sink objects:

1. **MdoAddInEvents Object**
2. **MdoComponentEvents Object**
3. **MdoControlEvents Object**
4. **MdoProjectEvents Object**
5. **MdoReferenceEvents Object**

To connect or disconnect your event sink object from RepVB events, use the connection and disconnection methods that are provided by the RepVB object.

Properties

Property	Description
IsSynchronizing	Indicates whether or not RepVB is currently performing synchronization processing.

Methods

Method	Description
ConnectAddInEvents	Connects the event sink object supplied by the caller to events that are related to the state of the RepVB Add-in.
ConnectComponentEvents	Connects the event sink object supplied by the caller to component-related RepVB events.
ConnectControlEvents	Connects the event sink object supplied by the caller to control-related RepVB events.
ConnectProjectEvents	Connects the event sink object supplied by the caller to project-related RepVB events.
ConnectReferenceEvents	Connects the event sink object supplied by the caller to reference-related RepVB events.
ConnectRepository	Connects the RepVB Add-in to the specified repository.
DisconnectAddInEvents	Disconnects the specified event sink object from events that are related to the state of the RepVB Add-in.
DisconnectComponent	Disconnects the specified event sink

Events	object from component-related RepVB events.
DisconnectControlEvents	Disconnects the specified event sink object from control-related RepVB events.
DisconnectProjectEvents	Disconnects the specified event sink object from project-related RepVB events.
DisconnectReference Events	Disconnects the specified event sink object from reference-related RepVB events.
GetMDOObject	Retrieves the MDO object that is associated with the specified Visual Basic object.
GetRepository	Retrieves the opened repository instance that is currently being used by the RepVB Add-in.
GetVBObject	Retrieves the Visual Basic object that is associated with the specified MDO object.
SynchronizeRepository	Synchronizes the repository with all currently open projects.

RepVB ConnectAddInEvents Method

See Also

This method connects the specified event sink object to events that are related to the state of the RepVB Add-in.

Syntax

variable **=** *object***.ConnectAddInEvents(** *eventSinkObj* **)**

The **ConnectAddInEvents** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives a number that identifies the specific connection that has been made.
object	The variable name of the RepVB object.
eventSinkObj	The event sink object that is to receive add-in related events. This event sink implements methods defined for the MdoAddInEvents object.

RepVB ConnectComponentEvents Method

See Also

This method connects the specified event sink object to RepVB for receiving component-related events.

Syntax

variable **=** *object***.ConnectComponentEvents(** *eventSinkObj* **)**

The **ConnectComponentEvents** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives a number that identifies the specific connection that has been made.
object	The variable name of the RepVB object.
eventSinkObj	The event sink object that is to receive component-related events. This event sink implements methods defined for the MdoComponentEvents object.

RepVB ConnectControlEvents Method

See Also

This method connects the specified event sink object to RepVB for receiving control-related events.

Syntax

variable **=** *object***.ConnectControlEvents(** *eventSinkObj, progId* **)**

The **ConnectControlEvents** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives a number that identifies the specific connection that has been made.
object	The variable name of the RepVB object.
eventSinkObj	The event sink object that is to receive control-related events. This event sink implements methods defined for the MdoControlEvents object.
progId	(Optional). A string containing the programmatic identifier that represents the kind of control for which you wish to receive events. If this parameter is not specified, you will receive all control-related events.

RepVB ConnectProjectEvents Method

See Also

This method connects the specified event sink object to RepVB for receiving project-related events.

Syntax

variable **=** *object***.ConnectProjectEvents(** *eventSinkObj* **)**

The **ConnectProjectEvents** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives a number that identifies the specific connection that has been made.
object	The variable name of the RepVB object.
eventSinkObj	The event sink object that is to receive project-related events. This event sink implements methods defined for the MdoProjectEvents object.

RepVB ConnectReferenceEvents Method

See Also

This method connects the specified event sink object to RepVB for receiving reference-related events.

Syntax

variable **=** *object***.ConnectReferenceEvents(** *eventSinkObj* **)**

The **ConnectReferenceEvents** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives a number that identifies the specific connection that has been made.
object	The variable name of the RepVB object.
eventSinkObj	The event sink object that is to receive reference-related events. This event sink implements methods defined for the MdoReferenceEvents object.

RepVB ConnectRepository Method

See Also

This method connects the RepVB Add-in to the specified repository. If another repository was previously connected, it is disconnected, and a RepositoryInvalid event is fired.

Syntax

variable **=** *object***.ConnectRepository(** *connectOptions, user, password* **)**

The **ConnectRepository** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives the value zero if the connect was successful, nonzero otherwise.
object	The variable name of the RepVB object.
connectOptions	The ODBC connection string to be used for accessing the database server that hosts your repository.
user	The user name to use for identification to the database server. This parameter is optional.
password	The password that matches the *user* input parameter. This parameter is optional.

Remarks

When you connect to a repository, RepVB performs an automatic project synchronization for all open Visual Basic projects. This effectively adds the projects to the repository, if the repository to which you are connecting does not already contain information about the currently open projects.

RepVB DisconnectAddInEvents Method

See Also

This method disconnects the specified event sink object from RepVB. The event sink object will not be sent any more events related to the state of the RepVB Add-in.

Syntax

Call *object*.**DisconnectAddInEvents(** *connection* **)**

The **DisconnectAddInEvents** method has these parts:

Part	Description
object	The variable name of the RepVB object.
connection	The connection number for the event sink object that is to be disconnected from RepVB Add-in related events.

RepVB DisconnectComponentEvents Method

See Also

This method disconnects the specified event sink object from RepVB. The event sink object will not be sent any more component-related events.

Syntax

Call *object*.**DisconnectComponentEvents(** *connection* **)**

The **DisconnectComponentEvents** method has these parts:

Part	Description
object	The variable name of the RepVB object.
connection	The connection number for the event sink object that is to be disconnected from component-related events.

RepVB DisconnectControlEvents Method

See Also

This method disconnects the specified event sink object from RepVB. The event sink object will not be sent any more control-related events.

Syntax

Call *object*.**DisconnectControlEvents(** *connection* **)**

The **DisconnectControlEvents** method has these parts:

Part	Description
object	The variable name of the RepVB object.
connection	The connection number for the event sink object that is to be disconnected from control-related events.

RepVB DisconnectProjectEvents Method

See Also

This method disconnects the specified event sink object from RepVB. The event sink object will not be sent any more project-related events.

Syntax

Call *object*.**DisconnectProjectEvents(** *connection* **)**

The **DisconnectProjectEvents** method has these parts:

Part	Description
object	The variable name of the RepVB object.
connection	The connection number for the event sink object that is to be disconnected from project-related events.

RepVB DisconnectReferenceEvents Method

See Also

This method disconnects the specified event sink object from RepVB. The event sink object will not be sent any more reference-related events.

Syntax

Call *object*.**DisconnectReferenceEvents(** *connection* **)**

The **DisconnectReferenceEvents** method has these parts:

Part	Description
object	The variable name of the RepVB object.
connection	The connection number for the event sink object that is to be disconnected from reference-related events.

RepVB GetMDOObject Method

See Also

When you add Visual Basic objects to a project, RepVB automatically adds corresponding MDO objects to the repository. Use this method to retrieve the MDO object that corresponds to a particular Visual Basic object.

Syntax

Set *variable* **=** *object***.GetMDOObject(** *vbObject* **)**

The **GetMDOObject** method has these parts:

Part	Description
variable	A variable declared as a MDO object. Receives the MDO object that represents the *vbObject* Visual Basic object.
object	The variable name of the RepVB object.
vbObject	The Visual Basic object whose corresponding MDO object is to be retrieved.

RepVB GetRepository Method

See Also

Retrieves the opened repository instance that is currently being used by the RepVB Add-in.

Syntax

Set *variable* **=** *object***.GetRepository**

The **GetRepository** method has these parts:

Part	Description
variable	A variable declared as a **Repository** object. Receives the instance of the open repository that RepVB is currently using.
object	The variable name of the RepVB object.

RepVB GetVBObject Method

See Also

When you add Visual Basic objects to a project, RepVB automatically adds corresponding MDO objects to the repository. Use this method to retrieve the Visual Basic object that corresponds to a particular MDO object.

Syntax

Set *variable* **=** *object***.GetVBObject(** *mdoObject* **)**

The **GetVBObject** method has these parts:

Part	Description
variable	A variable declared as a Visual Basic object of the appropriate type. Receives the Visual Basic object that is represented by the *mdoObject* object in the repository.
object	The variable name of the RepVB object.
mdoObject	The MDO object whose corresponding Visual Basic object is to be retrieved.

RepVB IsSynchronizing Property

See Also

This property indicates whether or not RepVB is currently performing a project synchronization. A nonzero value indicates that RepVB is performing a synchronization. A zero value indicates that RepVB is not performing a synchronization. For more information about repository synchronization, see Synchronizing the Repository.

Syntax

object.**IsSynchronizing**

The **IsSynchronizing** property syntax has these parts:

Part	Description
object	The variable name of the RepVB object.

RepVB SynchronizeRepository Method

See Also

RepVB provides the capability to force a synchronization of currently open projects with the repository. This method performs that synchronization. For more information about repository synchronization, see Synchronizing the Repository.

Syntax

variable **=** *object***.SynchronizeRepository**

The **SynchronizeRepository** method has these parts:

Part	Description
variable	A variable declared as a long integer. Receives the value zero if the synchronization was successful, nonzero otherwise.
object	The variable name of the RepVB object.

Remarks

If this method is unsuccessful in its attempt to synchronize the currently open projects with the repository, **RepVB** will shut itself down.

Synchronizing the Repository

See Also

The structure of your Visual Basic projects is maintained in two places; in the Visual Basic project itself, and, if the Repository Add-in for Visual Basic (RepVB) is registered, in the repository.

Whenever the same information is maintained in two locations, there exist possibilities for the two copies of the information to become unsynchronized with respect to each other. These are the most common possibilities:

- If you copy someone else's Visual Basic project into your environment and open it. In this case, the repository does not have any information at all about the project.
- If you open a Visual Basic project, modify one or more of the project components, and then quit the Visual Basic environment without saving the project. In this case, RepVB has already committed the corresponding repository changes, so the repository reflects the state of your Visual Basic project *as if you had saved the changes*.
- If multiple users are working on the same Visual Basic project, but each user has a local repository. In this case, changes made by other users will not be reflected in your repository.
- If you open a Visual Basic project, modify one or more of the project components, and a power outage or system failure occurs. In this case, RepVB has already committed the corresponding repository changes, so the repository reflects the state of your Visual Basic project as if the changes had been successfully saved.

In order to keep the repository information about a project synchronized with the Visual Basic project itself, RepVB performs an *automatic project synchronization* each time a project is opened in the Visual Basic environment. Automatic project synchronization consists of checking the last-modified time associated with each project component with the TimeStamp property of the corresponding MDO Model component. (Visual Basic project components are stored as separate files in the supporting file system. The last-modified time for a Visual Basic project component is the time and date that the project component's corresponding file was modified.) When a project component is found that has been modified either *before* or *after* the last time that it was synchronized with the repository, the repository copy of the information for that component is refreshed.

Automatic project synchronization is a very effective mechanism to keep the two copies of your Visual Basic project structure synchronized. However, it relies on the last-modified times that are maintained by the file system. If these times are not accurately maintained, then the automatic project synchronization feature of RepVB may not detect that a project has become unsynchronized.

As a last resort, RepVB provides a *manual project synchronization* capability. A manual project synchronization is only performed when you explicitly request it. Manual project synchronization consists of forcing the refresh of *all* project components for all open projects, regardless of their last-modified time and TimeStamp property values.

MdoComponentEvents Object

An MDO component object is a repository object that implements the **IMdoComponent** interface. MDO Model classes that implement this interface are:

- **MdoClassModule**
- **MdoDesigner**
- **MdoMDIForm**
- **MdoMSForm**
- **MdoPropertyPage**
- **MdoRelatedDocument**
- **MdoResourceFile**
- **MdoStdModule**
- **MdoUserDocument**
- **MdoUserControl**
- **MdoVBForm**

To be able to respond to component-related Repository Add-in (RepVB) events, you must supply an MdoComponentEvents event sink object that implements the following methods:

Methods

Method	Description
ObjectAdded	Invoked when an MDO component object has been added to the repository.
ObjectChangedFileName	Invoked when the file associated with a Visual Basic component has been renamed.
ObjectRemoved	Invoked when an MDO component object has been removed from the repository.
ObjectRenamed	Invoked when an MDO component object has been renamed in the repository.
ObjectSynchronized	Invoked when an MDO component object has been synchronized with the corresponding Visual Basic component.

Remarks

To begin receiving events, register your MdoComponentEvents event sink object with the **ConnectComponentEvents** Repository Add-in method.

MdoComponentEvents ObjectAdded Method

See Also

The Repository Add-in invokes this method when an MDO component object has been added to the repository. The method is invoked for each event sink object that has been registered for component-related Repository Add-in events.

Syntax

Sub ObjectAdded(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectAdded** method has these parts:

Part	Description
MdoObject	The MDO component object that has been added to the repository.
VBObject	The corresponding Visual Basic component.

MdoComponentEvents ObjectChangedFileName Method

See Also

The Repository Add-in invokes this method when the FileName property of an MDO component object has been changed in the repository. The method is invoked for each event sink object that has been registered for component-related Repository Add-in events.

Syntax

Sub ObjectChangedFileName(*MdoObject* **As Object,** *VBObject* **As Object,** *OldFileName* **As String)**

The **ObjectChangedFileName** method has these parts:

Part	Description
MdoObject	The MDO component object whose corresponding Visual Basic component file has been renamed.
VBObject	The corresponding Visual Basic component.
OldFileName	The old file name for the Visual Basic component file.

Remarks

Do not confuse the name of an MDO component object with the name of the *file* that is associated with the corresponding Visual Basic component. If the *object name* is changed, the **ObjectRenamed** method is invoked, not this method.

MdoComponentEvents ObjectRemoved Method

See Also

The Repository Add-in invokes this method when an MDO component object has been removed from the repository. The method is invoked for each event sink object that has been registered for component-related Repository Add-in events.

Syntax

Sub ObjectRemoved(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectRemoved** method has these parts:

Part	Description
MdoObject	The MDO component object that has been removed from the repository.
VBObject	The corresponding Visual Basic component.

MdoComponentEvents ObjectRenamed Method

See Also

The Repository Add-in invokes this method when the name of an MDO component object has been changed in the repository. The method is invoked for each event sink object that has been registered for component-related Repository Add-in events.

Syntax

Sub ObjectRenamed(*MdoObject* **As Object,** *VBObject* **As Object,** *OldName* **As String)**

The **ObjectRenamed** method has these parts:

Part	Description
MdoObject	The MDO component object that has been renamed in the repository.
VBObject	The corresponding Visual Basic component.
OldName	The name that this object had before it was renamed.

Remarks

Do not confuse the name of an MDO component object with the name of the *file* that is associated with the corresponding Visual Basic component. If the *file name* is changed, the **ObjectChangedFileName** method is invoked, not this method.

MdoComponentEvents ObjectSynchronized Method

See Also

The Repository Add-in invokes this method when an MDO component object has been synchronized with its corresponding Visual Basic component. The method is invoked for each event sink object that has been registered for component-related Repository Add-in events.

Syntax

Sub ObjectSynchronized(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectSynchronized** method has these parts:

Part	Description
MdoObject	The MDO component object that has been synchronized.
VBObject	The corresponding Visual Basic component.

MdoControlEvents Object

To be able to respond to control-related Repository Add-in (RepVB) events, you must supply an MdoControlEvents event sink object that implements these methods:

Methods

Method	Description
ObjectAdded	Invoked when an MdoControl object has been added to the repository.
ObjectRemoved	Invoked when an MdoControl object has been removed from the repository.
ObjectRenamed	Invoked when an MdoControl object has been renamed in the repository.
ObjectSynchronized	Invoked when an MDO control object has been synchronized with the corresponding Visual Basic control.

Remarks

To begin receiving events, register your MdoControlEvents event sink object with the **ConnectControlEvents** Repository Add-in method.

MdoControlEvents ObjectAdded Method

See Also

The Repository Add-in invokes this method when an MdoControl object has been added to the repository. The method is invoked for each event sink object that has been registered for control-related Repository Add-in events.

Syntax

Sub ObjectAdded(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectAdded** method has these parts:

Part	Description
MdoObject	The MDO control object that has been added to the repository.
VBObject	The corresponding Visual Basic control.

MdoControlEvents ObjectRemoved Method

See Also

The Repository Add-in invokes this method when an MdoControl object has been removed from the repository. The method is invoked for each event sink object that has been registered for control-related Repository Add-in events.

Syntax

Sub ObjectRemoved(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectRemoved** method has these parts:

Part	Description
MdoObject	The MDO control object that has been removed from the repository.
VBObject	The corresponding Visual Basic control.

MdoControlEvents ObjectRenamed Method

See Also

The Repository Add-in invokes this method when the name of an MdoControl object has been changed in the repository. The method is invoked for each event sink object that has been registered for control-related Repository Add-in events.

Syntax

Sub ObjectRenamed(*MdoObject* **As Object,** *VBObject* **As Object,** *OldName* **As String)**

The **ObjectRenamed** method has these parts:

Part	Description
MdoObject	The MDO control object that has been renamed in the repository.
VBObject	The corresponding Visual Basic control.
OldName	The name that this object had before it was renamed.

MdoControlEvents ObjectSynchronized Method

See Also

The Repository Add-in invokes this method when an MDO control object has been synchronized with its corresponding Visual Basic control. The method is invoked for each event sink object that has been registered for control-related Repository Add-in events.

Syntax

Sub ObjectSynchronized(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectSynchronized** method has these parts:

Part	Description
MdoObject	The MDO control object that has been synchronized.
VBObject	The corresponding Visual Basic control.

MdoProjectEvents Object

To be able to respond to project-related Repository Add-in (RepVB) events, you must supply an MdoProjectEvents event sink object that implements these methods:

Methods

Method	Description
ObjectAdded	Invoked when an MdoProject object has been added to the repository.
ObjectChangedFileName	Invoked when the file associated with a Visual Basic project has been renamed.
ObjectRemoved	Invoked when an MdoProject object has been removed from the repository.
ObjectRenamed	Invoked when an MdoProject object has been renamed in the repository.
ObjectSynchronized	Invoked when an MDO project object has been synchronized with the corresponding Visual Basic project.

Remarks

To begin receiving events, register your MdoProjectEvents event sink object with the **ConnectProjectEvents** Repository Add-in method.

MdoProjectEvents ObjectAdded Method

See Also

The Repository Add-in invokes this method when an MdoProject object has been added to the repository. The method is invoked for each event sink object that has been registered for project-related Repository Add-in events.

Syntax

Sub ObjectAdded(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectAdded** method has these parts:

Part	Description
MdoObject	The MDO project object that has been added to the repository.
VBObject	The corresponding Visual Basic project.

MdoProjectEvents ObjectChangedFileName Method

See Also

The Repository Add-in invokes this method when the FileName property of an MdoProject object has been changed in the repository. The method is invoked for each event sink object that has been registered for project-related Repository Add-in events.

Syntax

Sub ObjectChangedFileName(*MdoObject* **As Object,** *VBObject* **As Object,** *OldFileName* **As String)**

The **ObjectChangedFileName** method has these parts:

Part	Description
MdoObject	The MDO project object whose corresponding Visual Basic project file has been renamed.
VBObject	The corresponding Visual Basic project.
OldFileName	The old file name for the Visual Basic project file.

Remarks

Do not confuse the name of an MDO project object with the name of the *file* that is associated with the corresponding Visual Basic project. If the *object name* is changed, the **ObjectRenamed** method is invoked, not this method.

MdoProjectEvents ObjectRemoved Method

See Also

The Repository Add-in invokes this method when an MdoProject object has been removed from the repository. The method is invoked for each event sink object that has been registered for project-related Repository Add-in events.

Syntax

Sub ObjectRemoved(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectRemoved** method has these parts:

Part	Description
MdoObject	The MDO project object that has been removed from the repository.
VBObject	The corresponding Visual Basic project.

MdoProjectEvents ObjectRenamed Method

See Also

The Repository Add-in invokes this method when the name of an MdoProject object has been changed in the repository. The method is invoked for each event sink object that has been registered for project-related Repository Add-in events.

Syntax

Sub ObjectRenamed(*MdoObject* **As Object,** *VBObject* **As Object,** *OldName* **As String)**

The **ObjectRenamed** method has these parts:

Part	Description
MdoObject	The MDO project object that has been renamed in the repository.
VBObject	The corresponding Visual Basic project.
OldName	The name that the MDO project object had before it was renamed.

Remarks

Do not confuse the name of an MDO project object with the name of the *file* that is associated with the corresponding Visual Basic project. If the *file name* is changed, the **ObjectChangedFileName** method is invoked, not this method.

MdoProjectEvents ObjectSynchronized Method

See Also

The Repository Add-in invokes this method when an MDO project object has been synchronized with its corresponding Visual Basic project. The method is invoked for each event sink object that has been registered for project-related Repository Add-in events.

Syntax

Sub ObjectSynchronized(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectSynchronized** method has these parts:

Part	Description
MdoObject	The MDO project object that has been synchronized.
VBObject	The corresponding Visual Basic project.

MdoReferenceEvents Object

To be able to respond to reference-related Repository Add-in (RepVB) events, you must supply an MdoReferenceEvents event sink object that implements the following methods:

Methods

Method	Description
ObjectAdded	Invoked when an MdoTypeLib or MdoProjRef object has been added to the repository.
ObjectRemoved	Invoked when an MdoTypeLib or MdoProjRef object has been removed from the repository.

Remarks

To begin receiving events, register your MdoReferenceEvents event sink object with the **ConnectReferenceEvents** Repository Add-in method.

MdoReferenceEvents ObjectAdded Method

See Also

The Repository Add-in invokes this method when an MdoTypeLib or MdoProjRef object has been added to the repository. The method is invoked for each event sink object that has been registered for reference-related Repository Add-in events.

Syntax

Sub ObjectAdded(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectAdded** method has these parts:

Part	Description
MdoObject	The MDO reference object that has been added to the repository.
VBObject	The corresponding Visual Basic reference.

MdoReferenceEvents ObjectRemoved Method

See Also

The Repository Add-in invokes this method when an MdoTypeLib or MdoProjRef object has been removed from the repository. The method is invoked for each event sink object that has been registered for reference-related Repository Add-in events.

Syntax

Sub ObjectRemoved(*MdoObject* **As Object,** *VBObject* **As Object)**

The **ObjectRemoved** method has these parts:

Part	Description
MdoObject	The MDO reference object that has been removed from the repository.
VBObject	The corresponding Visual Basic reference.

MdoAddInEvents Object

To be able to respond to events related to the state of the Repository Add-in for Visual Basic (RepVB), you must supply an MdoAddInEvents event sink object that implements these methods:

Methods

Method	Description
RepositoryInvalid	Invoked when the Repository Add-in is switching to another repository.
Shutdown	Invoked when the Repository Add-in module is stopping.

Remarks

To begin receiving events, register your MdoAddInEvents event sink object with the **ConnectAddInEvents** Repository Add-in method.

MdoAddInEvents RepositoryInvalid Method

See Also

The Repository Add-in invokes this method when it is disconnecting from one repository and connecting to another repository. Any repository instance that your add-in may have obtained from the repository is no longer the currently open repository for the Repository Add-in.

Syntax

Sub RepositoryInvalid

MdoAddInEvents Shutdown Method

See Also

The Repository Add-in invokes this method when it is preparing to stop processing and unload itself. This is the last event that the Repository Add-in will send prior to shut down.

Syntax

Sub Shutdown

MDO Model Automation Objects

The Microsoft Development Objects Model (MDO Model) is a tool information model; it describes *kinds* of data that Visual Basic stores in the repository as you create Visual Basic projects. These kinds of objects are defined in the MDO Model:

- Visual Basic projects
- Standard modules
- Class modules
- Form modules
- MDI forms
- Dialogs
- Property pages
- Custom controls
- Document objects
- Custom base class modules
- Related documents
- Resource files
- Type libraries
- External project references

Use the MDO Model Automation objects to access and manipulate repository information about Visual Basic projects.

Objects	
MdoClassModule	**MdoRelatedDocument**
MdoControl	**MdoResourceFile**
MdoDesigner	**MdoStdModule**
MdoMDIForm	**MdoTypeLibRef**
MdoMSForm	**MdoUserControl**
MdoProject	**MdoUserDocument**
MdoProjectRef	**MdoVBForm**
MdoPropertyPage	**ReposRoot Object Modifications**

Accessing Automation Object Members

See Also

The MDO Model contains a number of Automation objects that support multiple interfaces. With each Automation object, one interface is defined to be the default interface, and the members (the properties, methods, and collections) that are attached to that interface are accessible via the standard Visual Basic mechanisms.

When accessing members that are attached to an interface that is *not* the default interface for an Automation object, a different access technique must be used. An additional reference to the object must be declared that explicitly calls for the non-default interface. The non-default interface members can then be accessed via the new object reference.

The following example illustrates how to access a collection that is attached to an interface that is not the default interface for an Automation object.

In this example, the repository object that represents the first control in a Visual Basic form is retrieved. **MdoVBForm** objects implement the **IMdoControlContainer** interface; this interface is not the default interface. The **MdoControls** collection is attached to the IMdoControlContainer interface. The MdoControls collection is the collection of controls that are contained in the form.

```
Dim vbFormObj    As MdoVBForm
Dim nonDefIfcObj As IMdoControlContainer
Dim controlObj   As MdoControl

'  Initialize the vbFormObj object by retrieving it
'  from the collection of components that are part
'  of a particular Visual Basic project.

Set nonDefIfcObj = vbFormObj
Set controlObj = nonDefIntObj.MdoControls(1)
```

ReposRoot Object Modifications

See Also Properties Methods Collections

The MDO Model modifies the repository root object of the repository to implement one additional repository interface: the IMpoProjectItemContainer interface. The IMpoProjectItemContainer interface maintains the relationship between a repository and the projects that it contains. The IMpoProjectItemContainer interface has one member: the Contents collection.

This modification enables you to use the repository root object as the starting point for navigating to MDO Model objects in the repository.

Collections

Collection	Description
Contents	The collection of all projects that are contained within a repository. This is not a default object member.

ReposRoot Contents Collection

See Also

The collection of all projects that are contained within a repository. This property is not attached to the default interface for the repository root object; it is attached to the **IMpoProjectItemContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	ProjectItemContainer-Contains-ProjectItems
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**Contents**(*index*)

The **Contents** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements IMpoProjectItemContainer as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.Contents.Count.

Remarks

The Contents collection is a heterogeneous collection; it can contain any kind of project item that implements the IMpoProjectItem interface. If you are using the collection to enumerate *a particular kind of project item*, be sure to check the **Type** property of each project item to skip over project items that are not of the desired type.

MdoClassModule Object

An MdoClassModule object represents a Visual Basic class module in the repository.

An MdoClassModule object is also a RepositoryObject. In addition to the members described here, MdoClassModule objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.

MdoClassModule Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoClassModule object.

MdoClassModule FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoClassModule object.

MdoClassModule Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoClassModule object.

MdoClassModule MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the class module belongs.

This collection is not attached to the default interface for the MdoClassModule Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoControl Object

An MdoControl object represents a Visual Basic control in the repository.

An MdoControl object is also a RepositoryObject. In addition to the members described here, MdoControl objects also provide the members that are defined for repository objects.

Properties

Property	Description
ClassName	Name of the Visual Basic class module that defines this control.
Description	Description of the Visual Basic control.
Index	Index value for the Visual Basic control, if the control is a Visual Basic control array. If the control is not a control array, this property is set to negative one.
ProgID	Program identifier for the Visual Basic control.

Collections

Collection	Description
MdoControlContainer	The collection of one Visual Basic container that contains this Visual Basic control.
MdoControls	The collection of all Visual Basic controls that are contained within *this* Visual Basic control. This is not a default object member.
MdoISVControl	The collection of all Visual Basic controls that are contained within *this* Visual Basic control. This is not a default object member.

MdoControl ClassName Property

See Also

A character string that contains the name of the Visual Basic class module that defines this control. This is a read-only property.

Syntax

object.**ClassName**

The **ClassName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoControl object.

MdoControl Description Property

See Also

A character string that contains the textual description that Visual Basic maintains for each control in a project. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoControl object.

MdoControl Index Property

See Also

A long integer that contains the index value for the Visual Basic control, if the control is a Visual Basic control array. If the control is not a control array, this property is set to negative one. This is a read-only property.

Syntax

object.**Index**

The **Index** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoControl object.

MdoControl ProgID Property

See Also

A character string that contains the program identifier for the Visual Basic control. This is a read-only property.

Syntax

object.**ProgID**

The **ProgID** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoControl object.

MdoControl MdoControlContainer Collection

See Also

The collection of one Visual Basic container that contains this Visual Basic control.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	No
Case Sensitive Names	Not Applicable
Unique Names	Not Applicable

Syntax

object.**MdoControlContainer**(*index*)

The **MdoControlContainer** collection syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoControl object.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControlContainer.Count.

MdoControl MdoControls Collection

See Also

MdoControls is the collection of all Visual Basic controls that are contained within *this* Visual Basic control. This collection does not recursively include controls that are contained within controls.

This collection is not attached to the default interface for the MdoControl Automation object; it is attached to the **IMdoControlContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

Syntax

object.**MdoControls**(*index*)

The **MdoControls** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlContainer** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControls.Count.

MdoControl MdoISVControl Collection

See Also

MdoISVControl is the collection of one ISV control object that is associated with a particular Visual Basic control.

This collection is not attached to the default interface for the MdoControl Automation object; it is attached to the **IMdoControlUsage** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Control-Uses-ISVControl
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	No
Case Sensitive Names	Not Applicable
Unique Names	Not Applicable

Syntax

object.**MdoISVControl**(*index*)

The **MdoISVControl** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlUsage** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoISVControl.Count.

MdoDesigner Object

An MdoDesigner object represents a Visual Basic custom base class in the repository.

An MdoDesigner object is also a RepositoryObject. In addition to the members described here, MdoDesigner objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.

MdoDesigner Description Property

See Also

The description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoDesigner object.

MdoDesigner FileName Property

See Also

The name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoDesigner object.

MdoDesigner Timestamp Property

See Also

The time at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoDesigner object.

MdoDesigner MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the custom base class module belongs.

This collection is not attached to the default interface for the MdoDesigner Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoMDIForm Object

An MdoMDIForm object represents a multiple document interface (MDI) form in the repository.

An MdoMDIForm object is also a RepositoryObject. In addition to the members described here, MdoMDIForm objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.
MdoControls	The collection of all Visual Basic controls that are contained within this MDI form. This is not a default object member.

MdoMDIForm Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoMDIForm object.

MdoMDIForm FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoMDIForm object.

MdoMDIForm Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoMDIForm object.

MdoMDIForm MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the MDI form belongs.

This collection is not attached to the default interface for the MdoMDIForm Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoMDIForm MdoControls Collection

See Also

MdoMDIForm is the collection of all Visual Basic controls that are contained within this MDI form. This collection does not recursively include controls that are contained within controls.

This collection is not attached to the default interface for the MdoMDIForm Automation object; it is attached to the **IMdoControlContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

Syntax

object.**MdoControls**(*index*)

The **MdoControls** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlContainer** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControls.Count.

MdoMSForm Object

An MdoMSForm object represents a dialog form in the repository.

An MdoMSForm object is also a RepositoryObject. In addition to the members described here, MdoMSForm objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.

MdoMSForm Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoMSForm object.

MdoMSForm FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoMSForm object.

MdoMSForm Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoMSForm object.

MdoMSForm MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the dialog form belongs.

This collection is not attached to the default interface for the MdoMSForm Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoProject Object

See Also Properties Methods Collections

An MdoProject object represents a Visual Basic project in the repository.

An MdoProject object is also a RepositoryObject. In addition to the members described here, MdoProject objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponents	The collection of all Visual Basic components that are contained within this project. This is not a default object member.
Containers	The collection of one repository root object that contains this Visual Basic project. This is not a default object member.
MdoReferences	The collection of all type libraries to which this Visual Basic project refers. This is not a default object member.

MdoProject Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoProject object.

MdoProject FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoProject object.

MdoProject Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoProject object.

MdoProject MdoComponents Collection

See Also

The collection of all Visual Basic components that are contained within this project.

This collection is not attached to the default interface for the MdoProject Automation object; it is attached to the **IMdoProject** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponents**(*index*)

The **MdoComponents** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoProject** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponents.Count.

MdoProject Containers Collection

See Also

The collection of one repository root object that contains this Visual Basic project.

This collection is not attached to the default interface for the MdoProject Automation object; it is attached to the **IMpoProjectItem** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	ProjectItemContainer-Contains-ProjectItems
Source Is Origin	No
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**Containers**(*index*)

The **Containers** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMpoProjectItem** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.Containers.Count.

MdoProject MdoReferences Collection

See Also

The collection of all type libraries to which this Visual Basic project refers.

This collection is not attached to the default interface for the MdoProject Automation object; it is attached to the **IMdoProject** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-References
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoReferences**(*index*)

The **MdoReferences** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoProject** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoReferences.Count.

MdoProjectRef Object

A Visual Basic project can reference another Visual Basic project, if the second project exposes a type library. An MdoProjectRef object represents a Visual Basic project reference in the repository.

An MdoProjectRef object is also a RepositoryObject. In addition to the members described here, MdoProjectRef objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the type library that is exported by a Visual Basic project.
FileName	Name of the type library file associated with this project reference.
TimeStamp	The time this MdoProjectRef object was last synchronized with the repository.

Collections

Collection	Description
MdoReferenceProject	The collection of one Visual Basic project that references this type library.

MdoProjectRef Description Property

See Also

Character string containing a description of the type library that is exported by a Visual Basic project. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoProjectRef object.

MdoProjectRef FileName Property

See Also

Character string containing the name of the type library that is associated with this project reference. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoProjectRef object.

MdoProjectRef Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoProjectRef object.

MdoProjectRef MdoReferenceProject Collection

See Also

The collection of one Visual Basic project that references this type library.

This collection is not attached to the default interface for the MdoProjectRef Automation object; it is attached to the **IMdoReference** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-References
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoReferenceProject**(*index*)

The **MdoReferenceProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoReference** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoReferenceProject.Count.

MdoPropertyPage Object

An MdoPropertyPage object represents a property page object in the repository.

An MdoPropertyPage object is also a RepositoryObject. In addition to the members described here, MdoPropertyPage objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.
MdoControls	The collection of all Visual Basic controls that are contained within this property page. This is not a default object member.

MdoPropertyPage Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoPropertyPage object.

MdoPropertyPage FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoPropertyPage object.

MdoPropertyPage Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoPropertyPage object.

MdoPropertyPage MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the property page belongs.

This collection is not attached to the default interface for the MdoPropertyPage Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoPropertyPage MdoControls Collection

See Also

MdoPropertyPage is the collection of all Visual Basic controls that are contained within this property page. This collection does not recursively include controls that are contained within controls.

This collection is not attached to the default interface for the MdoPropertyPage Automation object; it is attached to the **IMdoControlContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

Syntax

object.**MdoControls**(*index*)

The **MdoControls** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlContainer** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControls.Count.

MdoRelatedDocument Object

An MdoRelatedDocument object represents a document that is related to a Visual Basic project; for example, a functional specification or design document.

An MdoRelatedDocument object is also a RepositoryObject. In addition to the members described here, MdoRelatedDocument objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the related document.
FileName	Name of the related document file.
TimeStamp	The time this MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the document is related. This is not a default object member.

MdoRelatedDocument Description Property

See Also

Character string containing a description of the related document. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoRelatedDocument object.

MdoRelatedDocument FileName Property

See Also

Character string containing the name of the related document file. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoRelatedDocument object.

MdoRelatedDocument Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoRelatedDocument object.

MdoRelatedDocument MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the document is related.

This collection is not attached to the default interface for the MdoRelatedDocument Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoResourceFile Object

Resource files contain bitmaps, text strings, and other presentation data. An MdoResourceFile object represents a resource file in the repository.

An MdoResourceFile object is also a RepositoryObject. In addition to the members described here, MdoResourceFile objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.

MdoResourceFile Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoResourceFile object.

MdoResourceFile FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoResourceFile object.

MdoResourceFile Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoResourceFile object.

MdoResourceFile MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the resource file belongs.

This collection is not attached to the default interface for the MdoResourceFile Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoStdModule Object

An MdoStdModule object represents a Visual Basic standard module in the repository.

An MdoStdModule object is also a RepositoryObject. In addition to the members described here, MdoStdModule objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.

MdoStdModule Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoStdModule object.

MdoStdModule FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoStdModule object.

MdoStdModule Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoStdModule object.

MdoStdModule MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the standard module belongs.

This collection is not attached to the default interface for the MdoStdModule Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoTypeLibRef Object

An MdoTypeLibRef object represents an external type library that is referenced from a Visual Basic project.

An MdoTypeLibRef object is also a RepositoryObject. In addition to the members described here, MdoTypeLibRef objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the type library that is represented by this MDO object.
FileName	Name of the type library file associated with this MDO object.
TimeStamp	The time this MdoTypeLibRef object was last synchronized with the repository.

Collections

Collection	Description
MdoReferenceProject	The collection of one Visual Basic project that references this type library.

MdoTypeLibRef Description Property

See Also

Character string containing a description of the type library that is represented by this MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoTypeLibRef object.

MdoTypeLibRef FileName Property

See Also

Character string containing the name of the type library file that is associated with this MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoTypeLibRef object.

MdoTypeLibRef Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoTypeLibRef object.

MdoTypeLibRef MdoReferenceProject Collection

See Also

The collection of one Visual Basic project that references this type library.

This collection is not attached to the default interface for the MdoTypeLibRef Automation object; it is attached to the **IMdoReference** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-References
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoReferenceProject**(*index*)

The **MdoReferenceProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoReference** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoReferenceProject.Count.

MdoUserControl Object

An MdoUserControl object represents a custom control in the repository. You can build your own custom controls, or use controls that are supplied by an independent software vendor (ISV).

An MdoUserControl object is also a RepositoryObject. In addition to the members described here, MdoUserControl objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.
MdoControls	The collection of all Visual Basic controls that are contained within this user control. This is not a default object member.

MdoUserControl Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoUserControl object.

MdoUserControl FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoUserControl object.

MdoUserControl Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoUserControl object.

MdoUserControl MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the user control belongs.

This collection is not attached to the default interface for the MdoUserControl Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoUserControl MdoControls Collection

See Also

MdoUserControl is the collection of all Visual Basic controls that are contained within this user control. This collection does not recursively include controls that are contained within controls.

This collection is not attached to the default interface for the MdoUserControl Automation object; it is attached to the **IMdoControlContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

Syntax

object.**MdoControls**(*index*)

The **MdoControls** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlContainer** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControls.Count.

MdoUserDocument Object

An MdoUserDocument object represents a document object in the repository.

An MdoUserDocument object is also a RepositoryObject. In addition to the members described here, MdoUserDocument objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.
MdoControls	The collection of all Visual Basic controls that are contained within this user document. This is not a default object member.

MdoUserDocument Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoUserDocument object.

MdoUserDocument FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoUserDocument object.

MdoUserDocument Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoUserDocument object.

MdoUserDocument MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the user document belongs.

This collection is not attached to the default interface for the MdoUserDocument Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoUserDocument MdoControls Collection

See Also

MdoUserDocument is the collection of all Visual Basic controls that are contained within this user document. This collection does not recursively include controls that are contained within controls.

This collection is not attached to the default interface for the MdoUserDocument Automation object; it is attached to the **IMdoControlContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

Syntax

object.**MdoControls**(*index*)

The **MdoControls** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlContainer** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControls.Count.

MdoVBForm Object

An MdoVBForm object represents a Visual Basic form module in the repository.

An MdoVBForm object is also a RepositoryObject. In addition to the members described here, MdoVBForm objects also provide the members that are defined for repository objects.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file associated with the MDO object.
TimeStamp	The time the MDO object was last synchronized with the repository.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs. This is not a default object member.
MdoControls	The collection of all Visual Basic controls that are contained within this Visual Basic form. This is not a default object member.

MdoVBForm Description Property

See Also

Character string containing the description of the MDO object. This is a read-only property.

Syntax

object.**Description**

The **Description** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoVBForm object.

MdoVBForm FileName Property

See Also

Character string containing the name of the file that is associated with the MDO object. This is a read-only property.

Syntax

object.**FileName**

The **FileName** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoVBForm object.

MdoVBForm Timestamp Property

See Also

A **Date** value that specifies the time and date at which the MDO object was last synchronized with the repository. This is a read-only property.

Syntax

object.**TimeStamp**

The **TimeStamp** property syntax has these parts:

Part	Description
object	An object expression that evaluates to an MdoVBForm object.

MdoVBForm MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the Visual Basic form belongs.

This collection is not attached to the default interface for the MdoVBForm Automation object; it is attached to the **IMdoComponent** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Syntax

object.**MdoComponentProject**(*index*)

The **MdoComponentProject** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoComponent** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoComponentProject.Count.

MdoVBForm MdoControls Collection

See Also

MdoVBForm is the collection of all Visual Basic controls that are contained within this Visual Basic form. This collection does not recursively include controls that are contained within controls.

This collection is not attached to the default interface for the MdoVBForm Automation object; it is attached to the **IMdoControlContainer** interface. For details on how to access a member of an interface that is not the default interface, see Accessing Automation Object Members.

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

Syntax

object.**MdoControls**(*index*)

The **MdoControls** collection syntax has these parts:

Part	Description
object	An object expression; evaluates to an object that implements **IMdoControlContainer** as the default interface.
index	An integer index that identifies which element in the collection is to be addressed. The valid range is from one to the number of elements in the collection. The number of elements in the collection is specified by *object*.MdoControls.Count.

MDO Model Classes

The Microsoft Development Objects Model (MDO Model) is a tool information model; it describes *kinds* of data that Visual Basic stores in the repository as you create Visual Basic projects. These kinds of objects are defined in the MDO Model:

- Visual Basic projects
- Standard modules
- Class modules
- Form modules
- MDI forms
- Dialogs
- Property pages
- Custom controls
- Document objects
- Custom base class modules
- Related documents
- Resource files
- Type libraries
- External project references

Use the MDO Model classes to access and manipulate repository information about MDO objects.

Classes

MdoClassModule
MdoControl
MdoDesigner
MdoMDIForm
MdoMSForm
MdoProject
MdoProjectRef
MdoPropertyPage
MdoRelatedDocument
MdoResourceFile
MdoStdModule
MdoTypeLibRef
MdoUserControl
MdoUserDocument
MdoVBForm
ReposRoot Class Modifications

MdoClassModule Class

See Also Interfaces

An instance of the MdoClassModule class represents a Visual Basic class module in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoControl Class

See Also Interfaces

An instance of the MdoControl class represents a Visual Basic control.

Interfaces

Interface	Description
IMdoControl	Maintains information about a Visual Basic control.
IMdoControlContainer	Maintains the relationship between a Visual Basic control container and the Visual Basic controls that it contains.
IMdoControlUsage	Maintains the relationship between a Visual Basic control and the corresponding custom control.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoDesigner Class

See Also Interfaces

An instance of the MdoDesigner class represents a Visual Basic custom base class in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoMDIForm Class

See Also Interfaces

An instance of the MdoMDIForm class represents a multiple document interface (MDI) form in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoControlContainer	Maintains the relationship between a Visual Basic control container and the controls that it contains.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoMSForm Class

See Also Interfaces

An instance of the MdoMSForm class represents a dialog form in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoProject Class

See Also Interfaces

An instance of the MdoProject class represents a Visual Basic project in the repository.

Interfaces

Interface	Description
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IMdoProject	Maintains the relationships between a Visual Basic project and its components and type library references.
IMpoProjectItem	Maintains the relationship between a Visual Basic project and its containing repository.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoProjectRef Class

See Also Interfaces

A Visual Basic project can reference another Visual Basic project, if the second project exposes a type library. An instance of the MdoProjectRef class represents a Visual Basic project reference in the repository.

Interfaces

Interface	Description
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IMdoReference	Maintains the relationship between two Visual Basic projects, where one project references the other project.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoPropertyPage Class

See Also Interfaces

An instance of the MdoPropertyPage class represents a property page object in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoControlContainer	Maintains the relationship between a Visual Basic control container and the Visual Basic controls that it contains.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoRelatedDocument Class

See Also Interfaces

An instance of the MdoRelatedDocument class represents a document that is related to a Visual Basic project; for example, a functional specification or design document.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoResourceFile Class

See Also Interfaces

Resource files contain bitmaps, text strings, and other presentation data. An instance of the MdoResourceFile class represents a resource file in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoStdModule Class

See Also Interfaces

An instance of the MdoStdModule class represents a Visual Basic standard module in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoTypeLibRef Class

See Also Interfaces

An instance of the MdoTypeLibRef class represents an external type library that is referenced from one or more Visual Basic projects.

Interfaces

Interface	Description
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IMdoReference	Maintains the relationship between a type library and the Visual Basic projects that reference it.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoUserControl Class

See Also Interfaces

An instance of the MdoUserControl class represents a custom control in the repository. You can build your own custom controls, or use controls that are supplied by an independent software vendor (ISV).

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoControlContainer	Maintains the relationship between a Visual Basic control container and the Visual Basic controls that it contains.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoUserDocument Class

See Also Interfaces

An instance of the MdoUserDocument class represents a document object in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoControlContainer	Maintains the relationship between a Visual Basic control container and the Visual Basic controls that it contains.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

MdoVBForm Class

See Also Interfaces

An instance of the MdoVBForm class represents a Visual Basic form module in the repository.

Interfaces

Interface	Description
IMdoComponent	Maintains the relationship between a Visual Basic component and the Visual Basic project to which it belongs.
IMdoControlContainer	Maintains the relationship between a Visual Basic control container and the Visual Basic controls that it contains.
IMdoModelItem	Maintains properties that are common to many different types of MDO Model objects.
IRepositoryItem	Manages repository objects and relationships.
IRepositoryObject	Retrieves repository object identifiers.
IRepositoryObjectStorage	Creates and loads repository objects.

ReposRoot Class Modifications

The MDO Model modifies the **ReposRoot** class of the repository to implement one additional repository interface; the **IMpoProjectItemContainer** interface. With this interface, you can use the repository root as the starting point for navigating to MdoProject objects in the repository.

Added Interfaces

Interface	Description
IMpoProjectItemContainer	Maintains the relationship between a repository root object and the projects that it contains.

IMdoComponent Interface

See Also Properties Methods Collections

The IMdoComponent Interface maintains the relationship between a Visual Basic component and the Visual Basic project that contains it.

When to Use

Use the IMdoComponent Interface to determine the Visual Basic project to which a particular Visual Basic component belongs.

Collections

Collection	Description
MdoComponentProject	The collection of one Visual Basic project to which the component belongs.

IMdoComponent MdoComponentProject Collection

See Also

The collection of one Visual Basic project to which the component belongs.

Dispatch Identifier: DISPID_ComponentProject (53)

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

IMdoControl Interface

See Also Properties Methods Collections

The IMdoControl Interface maintains properties and collection information for Visual Basic control objects.

When to Use

Use the IMdoControl Interface to access information about a particular control.

Properties

Property	Description
ClassName	Name of the Visual Basic class module that defines this control.
Description	Description of the Visual Basic control.
Index	Index value for the Visual Basic control, if the control is a Visual Basic control array. If the control is not a control array, this property is set to negative one.
ProgID	Program identifier for the Visual Basic control.

Collections

Collection	Description
MdoControlContainer	The collection of one Visual Basic container that contains this Visual Basic control.

IMdoControl ClassName Property

See Also

The name of the Visual Basic class module that defines this control.

Dispatch Identifier: DISPID_PropClassName (112)

Property Data Type: character string

IMdoControl Description Property

See Also

The brief textual description that Visual Basic maintains for each control in a project.

Dispatch Identifier: DISPID_PropCtlDescription (114)

Property Data Type: character string

IMdoControl Index Property

See Also

Index value for the Visual Basic control, if the control is a Visual Basic control array. If the control is not a control array, this property is set to negative one.

Dispatch Identifier: DISPID_PropIndex (111)

Property Data Type: long

IMdoControl ProgID Property

See Also

The program identifier for the Visual Basic control.

Dispatch Identifier: DISPID_PropProgId (113)

Property Data Type: character string

IMdoControl MdoControlContainer Collection

See Also

The collection of one Visual Basic container that contains this Visual Basic control.

Dispatch Identifier: DISPID_ControlContainer (57)

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	No
Case Sensitive Names	Not Applicable
Unique Names	Not Applicable

IMdoControlContainer Interface

See Also Properties Methods Collections

The IMdoControlContainer Interface maintains the relationship between a Visual Basic container and the Visual Basic controls that it contains.

When to Use

Use the IMdoControlContainer Interface to determine the set of Visual Basic controls that are contained by a particular Visual Basic container.

Collections

Collection	Description
MdoControls	The collection of all Visual Basic controls that are contained within this Visual Basic container.

IMdoControlContainer MdoControls Collection

See Also

The collection of all Visual Basic controls that are contained within this Visual Basic container. This collection does not recursively include controls that are contained within controls.

Dispatch Identifier: DISPID_Controls (56)

Collection Descriptor	Descriptor Value
Relationship Type	Component-Has-Controls
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	No

IMdoControlUsage Interface

The IMdoControlUsage Interface maintains the relationship between a Visual Basic control and an independent software vendor (ISV) control. If a particular control is not implemented as an ISV control, this collection is empty.

When to Use

Use the IMdoControlUsage Interface to determine the ISV control that is associated with a Visual Basic control.

Collections

Collection	Description
MdoISVControl	The collection of one ISV control object.

IMdoControlUsage MdoISVControl Collection

See Also

The collection of one ISV control object that is associated with a particular Visual Basic control.

Dispatch Identifier: DISPID_ISVControls (102)

Collection Descriptor	Descriptor Value
Relationship Type	Control-Uses-ISVControl
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	No
Case Sensitive Names	Not Applicable
Unique Names	Not Applicable

IMdoISVControl Interface

The IMdoISVControl Interface maintains the relationship between an independent software vendor (ISV) control and its corresponding Visual Basic control object.

When to Use

Use the IMdoISVControl Interface to determine the Visual Basic control that is associated with a particular ISV control.

Collections

Collection	Description
MdoControlUsage	The collection of one Visual Basic control that uses this ISV control.

IMdoISVControl MdoControlUsage Collection

See Also

The collection of one Visual Basic control that uses this ISV control.

Dispatch Identifier: DISPID_ISVUsage (59)

Collection Descriptor	Descriptor Value
Relationship Type	Control-Uses-ISVControl
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	No
Case Sensitive Names	Not Applicable
Unique Names	Not Applicable

IMdoModelItem Interface

The IMdoModelItem Interface maintains properties that are common to many different types of MDO Model objects.

When to Use

Use the IMdoModelItem Interface to access information about a particular MDO object.

Properties

Property	Description
Description	Description of the MDO object.
FileName	Name of the file that is associated with the MDO object.
TimeStamp	The time at which the MDO object was last synchronized with the repository.

IMdoModelItem Description Property

See Also

The description of the MDO object.

Dispatch Identifier: DISPID_PropDescription (102)

Property Data Type: character string

IMdoModelItem FileName Property

See Also

The name of the file that is associated with the MDO object.

Dispatch Identifier: DISPID_FileName (103)

Property Data Type: character string

IMdoModelItem Timestamp Property

See Also

The time at which the MDO object was last synchronized with the repository.

Dispatch Identifier: DISPID_Timestamp (104)

Property Data Type: DATE

IMdoProject Interface

See Also Properties Methods Collections

The IMdoProject Interface maintains the relationships between a Visual Basic project and:

- The Visual Basic components that it contains.
- Its references, if there are any, to external type libraries.

When to Use

Use the IMdoProject Interface to determine:

- The set of Visual Basic components that are contained by a project.
- The references that a Visual Basic project has, if any, to type libraries.

Collections

Collection	Description
MdoComponents	The collection of all Visual Basic components that are contained within this project.
MdoReferences	The collection of all type libraries to which this Visual Basic project refers.

IMdoProject MdoComponents Collection

See Also

The collection of all Visual Basic components that are contained within this project.

Dispatch Identifier: DISPID_Components (52)

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-Components
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

IMdoProject MdoReferences Collection

See Also

The collection of all type libraries to which this Visual Basic project refers.

Dispatch Identifier: DISPID_References (54)

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-References
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

IMdoReference Interface

See Also Properties Methods Collections

The IMdoReference Interface maintains the relationship between a type library and the Visual Basic project that references it. The type library may represent another Visual Basic project, or any other executable component that exposes a type library.

When to Use

Use the IMdoReference Interface to determine which Visual Basic project references a particular type library.

Collections

Collection	Description
MdoReferenceProject	The collection of one Visual Basic project that references this type library.

IMdoReference MdoReferenceProject Collection

See Also

The collection of one Visual Basic project that references this type library.

Dispatch Identifier: DISPID_ReferenceProject (55)

Collection Descriptor	Descriptor Value
Relationship Type	Project-Has-References
Source Is Origin	No
Minimum Collection Size	One
Maximum Collection Size	One
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

IMpoProjectItemContainer Interface

See Also Properties Methods Collections

A project consists of containers and items. Some items are also containers, and contain other items. The **IMpoProjectItemContainer** interface maintains the relationship between a container and the items it contains.

For Visual Basic projects, the repository root object is modified to support this interface. The project items that are contained by the repository root object are projects. So the IMpoProjectItemContainer interface maintains the relationship between a repository and the projects that it contains.

When to Use

Use the IMpoProjectItemContainer interface to determine the set of project items that are contained within a project item container. For Visual Basic projects, use this interface to determine the set of projects that are contained within a repository.

Collections

Collection	Description
Contents	The collection of all project items that are contained within a particular project item container.

IMpoProjectItemContainer Contents Collection

See Also

The collection of all project items that are contained within a particular project item container.

Dispatch Identifier: DISPID_Contents (70)

Collection Descriptor	Descriptor Value
Relationship Type	ProjectItemContainer-Contains-ProjectItems
Source Is Origin	Yes
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

Remarks

The Contents collection is a heterogeneous collection; it can contain any kind of project item that implements the **IMpoProjectItem** interface. If you are using the collection to enumerate *a particular kind of project item*, be sure to check the type (via the **get_Type** method) of each project item to skip over project items that are not of the desired type.

IMpoProjectItem Interface

See Also Properties Methods Collections

The **IMpoProjectItem** interface maintains the relationship between a project item and the project item container that contains it.

For Visual Basic projects, the IMpoProjectItem interface maintains the relationship between a project and the repository that contains it.

When to Use

Use the IMpoProjectItemContainer interface to determine the project item container that contains a particular project item. For Visual Basic projects, use this interface to determine the repository that contains a particular project.

Collections

Collection	Description
Containers	The collection of all project item containers that are contain a particular project item.

IMpoProjectItem Containers Collection

See Also

The collection of all project item containers that contain a particular project item.

Dispatch Identifier: DISPID_Containers (71)

Collection Descriptor	Descriptor Value
Relationship Type	ProjectItemContainer-Contains-ProjectItems
Source Is Origin	No
Minimum Collection Size	Zero
Maximum Collection Size	Many
Sequenced Collection	No
Deletes Propagated	Yes
Destinations Named	Yes
Case Sensitive Names	No
Unique Names	Yes

MDO Model SQL Schema

See Also

SQL tables are used by the repository to store the instance information for tool information models. SQL schema modifications are automatically performed when you create or extend a tool information model.

Normally, one table is created for each interface that is defined in the repository. The table contains the instance data for persistent properties that are attached to the interface. One column in the table is created for each property. If an interface is defined that has no member properties, a table is not created.

You can construct a SQL query to extract specific tool information from the repository. You need to be familiar with the SQL schema for a tool information model to build such a query.

The MDO Model is a tool information model, and therefore, its instance information is stored in SQL tables. In the MDO Model (without any extensions), there are only two interfaces that have member properties: the **IMdoModelItem** interface and the **IMdoControl** interface. The SQL table names for these two interfaces are shown below.

Interface	SQL Table Name
IMdoControl	TblMdoControl
IMdoModelItem	TblMdoModelItem

TblMdoControl SQL Table

See Also

This SQL table contains the properties that are attached to the **IMdoControl** interface of the MDO Model. The IntID column is the primary key for this table.

Column Name	Data Type	Description
IntID	8 byte SQL_BINARY	The internal identifier for the control object.
CtlIx	SQL_INTEGER	The index value for the control, if it is a member of a control array. Otherwise, this field is not used.
ClassName	SQL_VARCHAR	The class name of the control.
ProgID	SQL_VARCHAR	The Visual Basic programmatic identifier for the control.
Description	SQL_VARCHAR	A description of the control.

TblMdoModelItem SQL Table

See Also

This SQL table contains the properties that are attached to the **IMdoModelItem** interface of the MDO Model. The IntID column is the primary key for this table.

Column Name	Data Type	Description
IntID	8 byte SQL_BINARY	The internal identifier for the model item.
Description	SQL_VARCHAR	A description of the model item.
FileName	SQL_LONG_VARCHAR	The name of the file that is associated with this model item.
tsTimeStamp	SQL_TIMESTAMP	The time at which the MDO object was last synchronized with the repository.