ACTIVEX TECHNOLOGY ELEMENTS

Cezar Botezatu, Ph.D., Cornelia Botezatu, Ph.D.
Romanian-American University, Bucharest, Romania

ActiveX is a set of technologies from Microsoft that enables interactive content for the World Wide Web. Before ActiveX, Web content was static, 2-dimensional text and graphics. With ActiveX, Web sites come alive using multimedia effects, interactive objects, and sophisticated applications that create a user experience comparable to that of high-quality CD-ROM titles. ActiveX provides the glue that ties together a wide assortment of technology building blocks to enable these "active" Web sites.

An ActiveX control is really just another term for "OLE Object" or, more specifically, "Component Object Model (COM) Object." In other words, a control, at the very least, is some COM object that supports the **IUnknown** interface and is also self-registering. Through **QueryInterface** a container can manage the lifetime of the control, as well as dynamically discover the full extent of a control's functionality based on the available interfaces. This allows a control to implement as little functionality as it needs to, instead of supporting a large number of interfaces that actually don't do anything. In short, this minimal requirement for nothing more than **IUnknown** allows any control to be as lightweight as it can.

Other than **IUnknown** and self-registration, there are no other requirements for a control. There are, however, conventions that should be followed about what the support of an interface means in terms of functionality provided to the container by the control. This section describes what it means for a control to actually support an interface, as well as methods, properties, and events that a control should provide as a baseline if it has occasion to support methods, properties, and events.

Self-Registration

ActiveX Controls must support self-registration by implementing the DllRegisterServer and DllUnregisterServer functions. ActiveX Controls must register all the standard registry entries for embeddable objects and automation servers.

ActiveX Controls must use the Component Categories application programming interface (API) to register themselves as a control and register the component categories that they require a host to support, and any categories that the control implements. In addition, an ActiveX control might want to register the "control" keyword in order to allow older containers, such as Microsoft Visual Basic 4, to host them.

ActiveX Controls should also register the ToolBoxBitmap32 registry key, although this is not mandatory.

The Insertable component category should only be registered if the control is suitable for use as a compound document object. It is important to note that a compound document object must support certain interfaces beyond the minimum **IUnknown** required for an ActiveX control. Although an ActiveX control might qualify as a compound document object, the control's documentation should clearly state what behavior to expect under these circumstances.

The VBScript programming language is only half of Microsoft's current Internet programming strategy. The other half is ActiveX, a way to develop programmable objects that can be added to Web pages alongside images, text, Java applets, and other media.

VBScript provides access to the intrinsic HTML controls-buttons, text fields, radio buttons, and other things that are common to Web-based forms. These controls are encountered any time someone registers to join a Web site or order a product through the Web.

To create sophisticated programs to run on a Web page, somebody might want to extend the possibilities beyond intrinsic controls by using ActiveX controls.

An ActiveX control is developed using a language such as Visual C++, Visual Basic, or Delphi. Like an OCX, an ActiveX component is designed to be used by some other software - a Web browser, in this case.

More than 3,000 ActiveX controls are available for use, according to Microsoft. In addition to being usable on Web pages, these controls can be used with other types of software developed with programming languages such as Java, Borland C++ and Delphi, Visual Basic, and Visual C++.

These controls are developed in other languages, but their operation can be modified and customized with the use of HTML code and VBScript programs. ActiveX controls are placed on a Web page using a special extended HTML tag called <OBJECT> and a supporting tag called <PARAM>.

The <OBJECT> Tag

The <OBJECT> tag, proposed by the World Wide Web Consortium as a standard, is used to place an object on a Web page. The primary type of object discussed in this chapter is an ActiveX control, but the tag considers an object to be any type of media that can be put on a page. <OBJECT> was proposed by the Consortium as a way to replace several current HTML tags and attributes-the <IMG> tag, the Java <APPLET> tag, the DYNSRC attribute used for audio and video by Microsoft, and other proprietary extensions to HTML. <OBJECT> also is flexible enough in design to handle new forms of media not yet invented for the Web.

Attributes are used with the <OBJECT> tag to specify the following information:

- The object's name
- The type of object

- The URL address where the object can be found
- Layout information such as height, width, spacing, border width, alignment, and so on
- An ID code to verify the object's identity

If the object has parameters, they can be set with the <PARAM> tag. This tag has two attributes: NAME and VALUE. The NAME attribute gives the parameter a name, and VALUE sets up a value for that parameter.

Although <OBJECT> is intended to be used with a broad range of media, one example of it in current use is the ActiveX control.

Generating <OBJECT> HTML Code

When the ActiveX Control Pad is used to add an ActiveX object to a Web page, an <OBJECT> tag is added automatically to the page. Here's an example of an ActiveX control's <OBJECT> tag:

```
<OBJECT ID="SpinButton1" WIDTH=16 HEIGHT=32
 CLASSID="CLSID:79176FB0-B7f2-11CE-97EF-00AA006D2776">
   <PARAM NAME="Size" VALUE="423;846">
</OBJECT>
```

This <OBJECT> tag creates an ActiveX spin button control with up and down arrows to change a value. Java programmers will recognize the <PARAM> tag, because it has the same attributes (NAME and VALUE) and the same usage as it does with the Java <APPLET> tag.

Because this HTML code is added automatically by the ActiveX Control Pad, you do not need to enter it yourself into a Web page's HTML file.

The ID attribute of the <OBJECT> tag gives the object a name. One of the biggest advantages of VBScript and ActiveX is the capability of one object to communicate with another object. A VBScript program can be used for one element on a page-for example, a <FORM> button-to modify another program, such as an ActiveX control. ID is needed for one object to know how to contact another.

The CLSID attribute identifies the type of object and provides some identifying characteristics of the object. In the preceding example, the CLSID was set to a complicated string of numbers and letters:

CLSID:79176FB0-B7f2-11CE-97EF-00AA006D2776

This has two parts. The section before the colon, CLSID, identifies this object as an ActiveX control. Another example of an identifier would be java:, representing an applet programmed in that language.

The section after the colon indicates some registration information that reveals where the ActiveX control can be found on the user's Windows system. ActiveX controls are downloaded to the user's system and run locally. The CLSID gives the browser enough information to find, identify, and run the control. It also creates a unique identifier for the ActiveX control. No matter how many ActiveX controls are implemented across the Internet, each will use part of the CLSID to establish its identity.

In addition to being usable on the World Wide Web, ActiveX controls have an advantage over other Internet programming solutions such as Java applets and Netscape plug-ins.

These controls can be used immediately in other applications. For example, a control that performs an image editing task on the Web can be plugged into a software program as easily as it was placed on a page.

An ActiveX-enabled Web browser will behave differently if it encounters a new control than if it has seen the control previously.

If you are using a browser that can handle ActiveX controls and you come to a page containing a control, a check will be made to determine whether you have downloaded the control previously. This check will use the CLSID attribute of the <OBJECT> tag to determine whether the ActiveX control is present on your system.

Because ActiveX controls are executed on the user's system, there is obvious potential for a programmer to run malicious code. In order to run an ActiveX control, you need some means of identifying the author as a trustworthy source.

The VeriSign company is handling ActiveX developer certification for a large number of the existing controls. The certificate window that opens when you encounter a new control on a Web page has a link to a control verification source such as VeriSign and probably a link to the developer's Web site.

After the control has been downloaded and executed, it remains on the user's computer so that it does not have to be reloaded each time the control is found on a Web page. The only time that an ActiveX control will be downloaded more than once is when a new version is offered that the user does not yet have.

This enables much quicker access to an ActiveX control than is possible with Java applets, which download again each time they are encountered. However, the disadvantage is that ActiveX controls take up space on a user's hard drive.

To see a sampling of the ActiveX controls that have been made available, the Internet information service CNET has introduced an ActiveX file directory and news site.

When a control is on your system, you can use it as a component in your own Web pages and software projects. CNET's ActiveX site has many controls available that cater to programmers in need of useful components.

Now, there are four ways to write an ActiveX control.

- Microsoft Foundation Classes (MFC)
- ActiveX Template Library
- BaseCtrl framework
- Visual J++™ (COM objects only)

Short for ***A**ctiveX **D**ata **O**bjects*, Microsoft's newest high-level interface for data objects, ADO is designed to eventually replace *Data Access Objects (DAO)* and *Remote Data Objects (RDO).* Unlike RDO and DAO, which are designed only for accessing relational databases, ADO is more general and can be used to access all sorts of different types of data, including web pages, spreadsheets, and other types of documents.

Together with OLE DB and ODBC, ADO is one of the main components of Microsoft's *Universal Data Access (UDA)* specification, which is designed to provide a consistent way of accessing data regardless of how the data are structured.

ADO provides developers with a powerful, logical object model for programmatically accessing, editing, and updating data from a wide variety of data sources through OLE DB system interfaces. The most common usage of ADO is to query a table or tables in a relational database, retrieve and display the results in an application, and perhaps allow users to make and save changes to the data. Other tasks include:

- Querying a database using SQL and displaying the results.
- Accessing information in a file store over the Internet.
- Manipulating messages and folders in an e-mail system.
- Saving data from a database into an XML file.
- Executing commands described with XML and retrieving an XML stream.
- Saving data into a binary or XML stream.
- Allowing a user to review and make changes to data in database tables.
- Creating and reusing parameterized database commands.
- Executing stored procedures.
- Dynamically creating a flexible structure, called a **Recordset**, to hold, navigate, and manipulate data.
- Performing transactional database operations.
- Filtering and sorting local copies of database information based on run-time criteria.
- Creating and manipulating hierarchical results from databases.
- Binding database fields to data-aware components.
- Creating remote, disconnected **Recordsets**.

ADO exposes a wide variety of options and settings in order to provide such flexibility. Therefore it's important to take a methodical approach to learning how to use ADO in an application, breaking down each of the goals into manageable pieces.

Most developers are very familiar with ADO's ancestors, DAO and RDO. A few items of note:

Cancel applies to asynchronous **Execute** and **Open**; an error will occur if called otherwise.

Recordset.Save writes the recordset to a physical file. No more writing out the data as text!

MarshalOptions controls whether a client-side recordset sends back all records or only the modified records.

PageSize, PageCount, AbsolutePage are ideally for Web sites. They allow you to request the 123rd page of records where there are 63 records on a page—no more counting through records.

Recordset.StayInSync is part of the data shaping available in ADODB. Version 2.1 adds grandchild aggregates, reshaping and parameterized commands using COMPUTE to the earlier data shaping, and hierarchical recordsets of version 2.0.

Using ADODB on the Web

ADO may be used in many environments, one of the most popular of which is Internet Information Services (IIS) Active Server Pages (ASP).

The ADO 2.8 SDK consists of the following components:

Microsoft ActiveX Data Objects (ADO) enable client applications to access and manipulate data from a variety of sources through an OLE DB provider. Its primary benefits are ease of use, high speed, low memory overhead, and a small disk footprint. ADO supports key features for building client/server and Web-based applications.

Microsoft ActiveX Data Objects (Multidimensional) (ADO MD) provides easy access to multidimensional data from languages such as Microsoft Visual Basic , Microsoft Visual C++ , and Microsoft Visual J++ . ADO MD extends Microsoft ActiveX Data Objects (ADO) to include objects specific to multidimensional data, such as the CubeDef and Cellset objects. With ADO MD you can browse multidimensional schema, query a cube, and retrieve the results.

Like ADO, ADO MD uses an underlying OLE DB provider to gain access to data. To work with ADO MD, the provider must be a multidimensional data provider (MDP) as defined by the OLE DB for OLAP specification. MDPs present data in multidimensional views as opposed to tabular data providers (TDPs) that present data in tabular views.

Remote Data Service (RDS) is a feature of ADO, with which you can move data from a server to a client application or Web page, manipulate the data on the client, and return updates to the server in a single round trip.

Microsoft ActiveX Data Objects Extensions for Data Definition Language and Security (ADOX) is an extension to the ADO objects and programming model. ADOX includes objects for schema creation and modification, as well as security. Because it is an object-based approach to schema manipulation, you can write code that will work against various data sources regardless of differences in their native syntaxes.

ADOX is a companion library to the core ADO objects. It exposes additional objects for creating, modifying, and deleting schema objects, such as tables and procedures. It also includes security objects to maintain users and groups and to grant and revoke permissions on objects.

If you are a Java or Visual Basic developer, you may have run into situations where you really wanted to use some key feature of the operating system. Let's say you want to be able to draw something using DirectX™. You can't do it directly with Visual Basic, and, if you try it in Java, you aren't going to be playing in the sandbox anymore (for those of you who aren't Java-aware, the "sandbox" is the name used for the virtual machine in which your Java applet runs). This virtual machine is not allowed access to your system's services, to protect you, the consumer of the Java applet, from inadvertently downloading a virus that reads from or writes to your hard disk. A way to get around the problem of not being able to access basic system services is to write an ActiveX control using the Win32 API and C++.

Unlike JAVA, ActiveX can also manage files.

Slowly but surely, ActiveX technology has taken over about 80% of the specific market.

BIBLIOGRAPHY

1. Kenneth Lassesen, ADODB: ActiveX Data Objects 2.1, Microsoft Corporation
2. Nancy Winnick Cluts, Creating ActiveX Components in C++, Microsoft Corporation
3. Nancy Cluts, Microsoft ActiveX Controls Overview, Microsoft Corporation
4. W Ernst, The Components of ActiveX, O'Reilly &Assoc.Press,1998
5. *** - ActiveX – http//www.Microsoft.com
6. *** - VBScript &ActiveX– http//www.Activex.org

Надійшла до редакції 23 березня 2004р