Using XML Files with DB Web Controls

The DBWebDataSource component provides a way for you to create and use XML and XSD files as the data source for an ASP.NET application. Typically, you only use these types of files with the DBWeb controls as a way of prototyping your application. By using XML files as the data source, you can eliminate potentially costly database resources during the design and development phase of your project.

This topic covers the following issues.

- XML files as data sources.
- Suggested workflow strategy.
- Authentication and caching issues.

XML Files as Data Sources

XML has become another standard data source for many applications, but for ASP.NET applications in particular. When working with data that does not require strong security and therefore can be sent over HTTP as text, XML files provide a simple solution. Because the files are text, they are easy to read. Because the XML tags describe the data, you can understand and process the data structures with little difficulty.

Despite their obvious advantages over more complex data structures, XML files do have some drawbacks. For one thing, they are not secure, therefore, it is not a good idea to pass sensitive data, such as credit card numbers or personal identification (PIN) numbers, over the Internet by way of XML files. Another drawback is the lack of concurrency control over XML records, unlike database records.

Nonetheless, the self-describing nature and the lightweight data format of XML files makes them a natural choice as data sources for ASP.NET applications. The DBWebDataSource control, in particular, has been built to handle XML files as well as other types of data sources. There are no special requirements for using XML files, no unique drivers or communication layers beyond those that come wth Delphi 2005, so you will find it easy to work with XML files as data sources.

Suggested Workflow Strategy

You use the DBWebDataSource control to create the XML file for your application and to connect the XML file with a DataSet object. The basic workflow strategy is this:

- Build an ASP.NET application, with a connection to your target database. Use DBWeb controls, including a DBWebDataSource and specify a non-existent XML file. When you run the application, your DataSet receives the result set from the target database and the DBWebDataSource then fills the XML file with tagged data representing the DataSet.
- From this point forward, you can eliminate the data adapter and data connection, keeping only a DataSet, the DBWebDataSource , and the reference to the XML file. Your DBWeb controls will pull data from the XML file and DataSet rather than from the database. For more information, follow the links to specific procedures on building and using XML files with DBWeb controls.

Authentication and Caching Issues

The DB Web Controls support automatic reading of an XML file by the DBWebDataSource component at both designtime and runtime. To support XML files, the DBWebDataSource component includes caching properties. If you use XML caching, the XML file data is automatically read into the DataSet whenever a data source is loaded.

If you do not implement user authentication in your application, you will likely only use this feature for prototyping. Otherwise, without user authentication, users may experience permissions errors when trying to access a single XML file concurrently. When multiple clients are using the application, the XML file is constantly being overwritten by different users. One way to avoid this is to write logic in your server application to check row updates and notify various clients when there is a conflict. This is similar to what a database

system does when it enforces table-level or row-level locking. When using a text file, like an XML file, this level of control is more difficult to implement.

However, if you implement user authentication, you can create a real-world application by setting the UseUniqueFileName property. This property specifies that the DBWebDataSource control will create uniquely named XML files for each client that uses accesses the XML file specified in the XMLFileName property of the DBWebDataSource . This helps avoid data collisions within a multi-user application. The drawback to this approach is that each XML file will contain different data and your server application will need built-in logic to merge the unique data from each client XML file.

Read-write applications using XMLFileName require that all web clients have write access to the XML files to which they are writing. If the web client does not have write access, the client will get a permissions error on any attempt to update the XML file. You must grant write access to the clients who will use the application.

Related Information

Borland DB Web Controls Overview
Creating a DB Web XML File
Building a Briefcase Application with DB Web Controls