Welcome to XMLBeans

1. Latest News

Read the latest XMLBeans news!

2. Introduction to XMLBeans

XMLBeans is a tool that allows you to access the *full* power of XML in a Java friendly way. It is an XML-Java binding tool. The idea is that you can take advantage the richness and features of XML and XML Schema and have these features mapped as naturally as possible to the equivalent Java language and typing constructs. XMLBeans uses XML Schema to *compile* Java interfaces and classes that you can then use to access and modify XML instance data. Using XMLBeans is similar to using any other Java interface/class, you will see things like `getFoo` or `setFoo` just as you would expect when working with Java. While a major use of XMLBeans is to access your XML instance data with strongly typed Java classes there are also API's that allow you access to the full XML infoset (XMLBeans keeps full XML Infoset fidelity) as well as to allow you to *reflect* into the XML schema itself through an XML Schema Object model.

For more details on XMLBeans see the XMLBeans Explanation Wiki page. As well as the XMLBeans documentation (see the Documentation tab on this website as well as the docs directory in the XMLBeans Source).

2.1. What makes XMLBeans Different

There are at least two major things that make XMLBeans unique from other XML-Java binding options.

1. **Full XML Schema support.** XMLBeans fully supports XML Schema and the corresponding java classes provide constructs for all of the major functionality of XML Schema. This is critical since often times you do not have control over the features of XML Schema that you need to work with in Java. Also, XML Schema oriented applications can take full advantage of the power of XML Schema and not have to restrict themselvs to a subset.
2. **Full XML Infoset fidelity.**When unmarshalling an XML instance the full XML infoset is kept and is available to the developer. This is critical because because of the subset of

XML that is not easily represented in java. For example, order of the elements or comments might be needed in a particular application.

A major objective of XMLBeans has been to be applicable in *all* non-streaming (in memory) XML programming situations. You should be able to compile your XML Schema into a set of java classes and know that 1) you will be able to use XMLBeans for all of the schemas you encounter (even the warped ones) and 2) that you will be able to get to the XML at whatever level is necessary - and not have to resort to multple tools to do this.

To accomplish this XMLBeans provides three major APIs:

- **XmlObject** The java classes that are generated from an XML Schema are all derived from XmlObject. These provide strongly typed getters and setters for each of the elements within the defined XML. Complex types are in turn XmlObjects. For example getCustomer might return a CustomerType (which is an XmlObject). Simple types turn into simple getters and setters with the correct java type. For example getName might return a String.
- **XmlCursor** From any XmlObject you can get an XmlCursor. This provides efficient, low level access to the XML Infoset. A cursor represents a position in the XML instance. You can move the cursor around the XML instance at any level of granularity you need from individual characters to Tokens.
- **SchemaType** XMLBeans provides a full XML Schema object model that you can use to reflect on the underlying schema meta information. For example, you might want to generate a sample XML instance for an XML schema or perhaps find the enumerations for an element so that you can display them.

All of this was built with performance in mind. Informal benchmarks and user feedback indicate that XMLBeans is extremely fast.

3. History

XMLBeans was submitted to the Apache Incubator and Apache XML projects by BEA Systems in September 2003. Thanks to the support and guidance of Steven Noels (project sponsor) and Ted Leung (project mentor), XMLBeans successfully graduated from incubation in June 2004 to become a top level project and part of the Apache XML federation. XMLBeans was originally created because of the need seen by developers, in particular David Bau (David's about me post), need for a more XML centric Java binding model that nothing on the market offered. XMLBeans 1.0 has been successfully used as an underlying technology for several products as well as by a growing number of large users including some of the largest companies in the world.

4. Future

The future of XMLBeans is exciting and you are invited to contribute. XMLBeans Version 1 is a great, stable technology that will continue to improve going forward. Its emphasis on full support for XML Schema and the XML Infoset along with it's performance charactaristics make it a great choice for in-memory XML-Java programming.

Work has begun on planning XMLBeans 2.0. At this point it appears that the major emphasis for XMLBeans 2.0 are new use cases that expand the breadth of XMLBeans. The observation is that XML-Java binding occurs in situations that XMLBeans could be optimized for, areas such as Web Services (XMLBeans works well now in many Web Services situations) and Java to XML Schema scenarios . See the XMLBeans Roadmap and the XMLBeans Feature Plan wiki page for more information. XMLBeans is actively looking for contributors and committers so, if you are interested, please join in.