Schema

Both a sample XML Schema definition (*binding.xsd*) and a sample DTD (*binding.dtd*) for JiBX binding definitions are included in the */docs* directory of the JiBX distribution. These grammars will normally be kept current, and if you find any errors please report them so they can be fixed. However, the HTML documentation supplied on these pages is the definitive description of the binding definition format.

Element Summary

Below is a quick summary of the types of elements used by the binding definition, along with their functions and possible child elements. Most child elements are optional and used only where needed. All are ordered, except where "any combination" is specified.

Elements

binding

The root element of the binding definition, with optional attributes for binding name and global settings.

Children: **namespace**, **format**, **include**, **mapping** (at least 1 **mapping** required)

namespace

Namespace declaration that defines a namespace URI and associated prefix (with the prefix used for marshalling).

Children: none

format

Format definition for converting simple values to and from text. This is needed only if you want to use nonstandard conversions.

Children: none

include

Include another binding definition within this binding definition.

Children: none

mapping

Defines how objects of a particular class are converted to and from XML. Each **mapping** defines a rule that applies within a particular context, associating a particular element name with objects of a particular Java class. The mapping definition may be global or local. It's possible to redefine a mapping within the context of another mapping, for instance. Abstract mappings can be used to define handling for base classes which can be extended by mappings for subclasses. Abstract mappings can also be used to define a reusable binding structure for a Java class, which doesn't need to correspond to a particular element name or even to a single element of the XML representation.

Children: **namespace**, **format**, **mapping**, followed by any combination of **value**, **structure**, and **collection** elements

value

Gives the conversion handling for a simple value (a primitive, or an object type with **format** supplied to convert it to and from text. The XML representation can be an attribute or simple element, or text or CDATA content within a sequence of elements.

Children: none

structure

Structure component of binding, which can represent a Java object, an XML element, or both. The usual case is where this represents both a Java object and an XML element linked to that object. A structure mapping is defined when either the object or the XML element is missing from the definition. The child elements of the structure may be ordered or unordered.

Children: any combination of **value**, **structure**, and **collection** elements

collection

Representation of a collection object. The collection may define the name for an element wrapper that contains the XML representations of the items in the collection. If different types of items are included in the collection the

items may be ordered or unordered.

Children: any combination of **value**, **structure**, and **collection** elements

Attribute Group Summary

The binding definition elements define a number of different attributes. In many cases these attributes are grouped and the attribute groups apply to more than one element type. The table below gives a list of the basic attribute groups and the elements that support them. The elements also have unique attributes in addition to these common groups. In almost all cases the attributes are optional and are used to qualify or add details to the basic function of the element.

Attribute Groups

style

This group consists of only one attribute, the **value-style** attribute. It's used to control whether a simple value is expressed in XML as an element or an attribute.

Used with: **binding**, **mapping**, **structure**, and **collection**

name

The name attributes define an XML element or attribute name.

Used with: **mapping**, **value**, **structure**, and **collection**

object

The object group of attributes provide handling options for Java objects. These include factory, pre- and post-unmarshal, and pre-marshal methods for objects.

Used with: **mapping**, **structure**, and **collection**

structure

The structure attributes deal with the children of a binding component. They include control over whether children are ordered or unordered, as well as linkages that allow portions of a binding definition to be referenced and reused within the binding.

Used with: **value**, **structure**, and **collection**

property	The property attributes control how a value is accessed from a Java object. These include field and/or get/set/test method names. Used with: **value**, **structure**, and **collection**
string	The string attributes define conversions between Java primitive or object values and text representations. Used with: **format** and **value**