| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Unmarshaller.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/xml/bind/UnmarshalException.html)   [**NEXT CLASS**](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) | [**FRAMES**](http://docs.google.com/index.html?javax/xml/bind/Unmarshaller.html)    [**NO FRAMES**](http://docs.google.com/Unmarshaller.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2s8eyo1) | FIELD | CONSTR | [METHOD](#17dp8vu) | DETAIL: FIELD | CONSTR | [METHOD](#3rdcrjn) |

## **javax.xml.bind**

Interface Unmarshaller

**All Known Implementing Classes:** [AbstractUnmarshallerImpl](http://docs.google.com/javax/xml/bind/helpers/AbstractUnmarshallerImpl.html)

public interface **Unmarshaller**

The Unmarshaller class governs the process of deserializing XML data into newly created Java content trees, optionally validating the XML data as it is unmarshalled. It provides an overloading of unmarshal methods for many different input kinds.

Unmarshalling from a File:

JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
 Object o = u.unmarshal( new File( "nosferatu.xml" ) );

Unmarshalling from an InputStream:

InputStream is = new FileInputStream( "nosferatu.xml" );  
 JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
 Object o = u.unmarshal( is );

Unmarshalling from a URL:

JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
 URL url = new URL( "http://beaker.east/nosferatu.xml" );  
 Object o = u.unmarshal( url );

Unmarshalling from a StringBuffer using a javax.xml.transform.stream.StreamSource:

JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
 StringBuffer xmlStr = new StringBuffer( "<?xml version="1.0"?>..." );  
 Object o = u.unmarshal( new StreamSource( new StringReader( xmlStr.toString() ) ) );

Unmarshalling from a org.w3c.dom.Node:

JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
   
 DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();  
 dbf.setNamespaceAware(true);  
 DocumentBuilder db = dbf.newDocumentBuilder();  
 Document doc = db.parse(new File( "nosferatu.xml"));  
  
 Object o = u.unmarshal( doc );

Unmarshalling from a javax.xml.transform.sax.SAXSource using a client specified validating SAX2.0 parser:

// configure a validating SAX2.0 parser (Xerces2)  
 static final String JAXP\_SCHEMA\_LANGUAGE =  
 "http://java.sun.com/xml/jaxp/properties/schemaLanguage";  
 static final String JAXP\_SCHEMA\_LOCATION =  
 "http://java.sun.com/xml/jaxp/properties/schemaSource";  
 static final String W3C\_XML\_SCHEMA =  
 "http://www.w3.org/2001/XMLSchema";  
  
 System.setProperty( "javax.xml.parsers.SAXParserFactory",  
 "org.apache.xerces.jaxp.SAXParserFactoryImpl" );  
  
 SAXParserFactory spf = SAXParserFactory.newInstance();  
 spf.setNamespaceAware(true);  
 spf.setValidating(true);  
 SAXParser saxParser = spf.newSAXParser();  
   
 try {  
 saxParser.setProperty(JAXP\_SCHEMA\_LANGUAGE, W3C\_XML\_SCHEMA);  
 saxParser.setProperty(JAXP\_SCHEMA\_LOCATION, "http://....");  
 } catch (SAXNotRecognizedException x) {  
 // exception handling omitted  
 }  
  
 XMLReader xmlReader = saxParser.getXMLReader();  
 SAXSource source =   
 new SAXSource( xmlReader, new InputSource( "http://..." ) );  
  
 // Setup JAXB to unmarshal  
 JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
 ValidationEventCollector vec = new ValidationEventCollector();  
 u.setEventHandler( vec );  
   
 // turn off the JAXB provider's default validation mechanism to   
 // avoid duplicate validation  
 u.setValidating( false )  
  
 // unmarshal  
 Object o = u.unmarshal( source );  
  
 // check for events  
 if( vec.hasEvents() ) {  
 // iterate over events  
 }

Unmarshalling from a StAX XMLStreamReader:

JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
   
 javax.xml.stream.XMLStreamReader xmlStreamReader =   
 javax.xml.stream.XMLInputFactory().newInstance().createXMLStreamReader( ... );  
   
 Object o = u.unmarshal( xmlStreamReader );

Unmarshalling from a StAX XMLEventReader:

JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
   
 javax.xml.stream.XMLEventReader xmlEventReader =   
 javax.xml.stream.XMLInputFactory().newInstance().createXMLEventReader( ... );  
   
 Object o = u.unmarshal( xmlEventReader );

**Unmarshalling XML Data**

Unmarshalling can deserialize XML data that represents either an entire XML document or a subtree of an XML document. Typically, it is sufficient to use the unmarshalling methods described by [Unmarshal root element that is declared globally](#2et92p0). These unmarshal methods utilize [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html)'s mapping of global XML element declarations and type definitions to JAXB mapped classes to initiate the unmarshalling of the root element of XML data. When the [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html)'s mappings are not sufficient to unmarshal the root element of XML data, the application can assist the unmarshalling process by using the [unmarshal by declaredType methods](#tyjcwt). These methods are useful for unmarshalling XML data where the root element corresponds to a local element declaration in the schema.An unmarshal method never returns null. If the unmarshal process is unable to unmarshal the root of XML content to a JAXB mapped object, a fatal error is reported that terminates processing by throwing JAXBException.

**Unmarshal a root element that is globally declared**

The unmarshal methods that do not have an declaredType parameter use [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html) to unmarshal the root element of an XML data. The [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html) instance is the one that was used to create this Unmarshaller. The [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html) instance maintains a mapping of globally declared XML element and type definition names to JAXB mapped classes. The unmarshal method checks if [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html) has a mapping from the root element's XML name and/or @xsi:type to a JAXB mapped class. If it does, it umarshalls the XML data using the appropriate JAXB mapped class. Note that when the root element name is unknown and the root element has an @xsi:type, the XML data is unmarshalled using that JAXB mapped class as the value of a [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html). When the [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html) object does not have a mapping for the root element's name nor its @xsi:type, if it exists, then the unmarshal operation will abort immediately by throwing a [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html). This exception scenario can be worked around by using the unmarshal by declaredType methods described in the next subsection.

**Unmarshal by Declared Type**

The unmarshal methods with a declaredType parameter enable an application to deserialize a root element of XML data, even when there is no mapping in [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html) of the root element's XML name. The unmarshaller unmarshals the root element using the application provided mapping specified as the declaredType parameter. Note that even when the root element's element name is mapped by [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html), the declaredType parameter overrides that mapping for deserializing the root element when using these unmarshal methods. Additionally, when the root element of XML data has an xsi:type attribute and that attribute's value references a type definition that is mapped to a JAXB mapped class by [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html), that the root element's xsi:type attribute takes precedence over the unmarshal methods declaredType parameter. These methods always return a JAXBElement<declaredType> instance. The table below shows how the properties of the returned JAXBElement instance are set.

| Unmarshal By Declared Type returned JAXBElement | |
| --- | --- |
| JAXBElement Property | Value |
| name | xml element name |
| value | instanceof declaredType |
| declaredType | unmarshal method declaredType parameter |
| scope | null *(actual scope is unknown)* |

The following is an example of [unmarshal by declaredType method](#tyjcwt).

Unmarshal by declaredType from a org.w3c.dom.Node:

Schema fragment for example  
 <xs:schema>  
 <xs:complexType name="FooType">...<\xs:complexType>  
 <!-- global element declaration "PurchaseOrder" -->  
 <xs:element name="PurchaseOrder">  
 <xs:complexType>  
 <xs:sequence>  
 <!-- local element declaration "foo" -->  
 <xs:element name="foo" type="FooType"/>  
 ...  
 </xs:sequence>  
 </xs:complexType>  
 </xs:element>  
 </xs:schema>  
  
 JAXBContext jc = JAXBContext.newInstance( "com.acme.foo" );  
 Unmarshaller u = jc.createUnmarshaller();  
   
 DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();  
 dbf.setNamespaceAware(true);  
 DocumentBuilder db = dbf.newDocumentBuilder();  
 Document doc = db.parse(new File( "nosferatu.xml"));  
 Element fooSubtree = ...; // traverse DOM till reach xml element foo, constrained by a   
 // local element declaration in schema.  
   
 // FooType is the JAXB mapping of the type of local element declaration foo.  
 JAXBElement<FooType> foo = u.unmarshal( fooSubtree, FooType.class);

**Support for SAX2.0 Compliant Parsers**

A client application has the ability to select the SAX2.0 compliant parser of their choice. If a SAX parser is not selected, then the JAXB Provider's default parser will be used. Even though the JAXB Provider's default parser is not required to be SAX2.0 compliant, all providers are required to allow a client application to specify their own SAX2.0 parser. Some providers may require the client application to specify the SAX2.0 parser at schema compile time. See [unmarshal(Source)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.transform.Source)) for more detail.

**Validation and Well-Formedness**

A client application can enable or disable JAXP 1.3 validation mechanism via the setSchema(javax.xml.validation.Schema) API. Sophisticated clients can specify their own validating SAX 2.0 compliant parser and bypass the JAXP 1.3 validation mechanism using the [unmarshal(Source)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.transform.Source)) API.

Since unmarshalling invalid XML content is defined in JAXB 2.0, the Unmarshaller default validation event handler was made more lenient than in JAXB 1.0. When schema-derived code generated by JAXB 1.0 binding compiler is registered with [JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html), the default unmarshal validation handler is [DefaultValidationEventHandler](http://docs.google.com/javax/xml/bind/helpers/DefaultValidationEventHandler.html) and it terminates the marshal operation after encountering either a fatal error or an error. For a JAXB 2.0 client application, there is no explicitly defined default validation handler and the default event handling only terminates the marshal operation after encountering a fatal error.

**Supported Properties**

There currently are not any properties required to be supported by all JAXB Providers on Unmarshaller. However, some providers may support their own set of provider specific properties.

**Unmarshal Event Callbacks**

The [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html) provides two styles of callback mechanisms that allow application specific processing during key points in the unmarshalling process. In 'class defined' event callbacks, application specific code placed in JAXB mapped classes is triggered during unmarshalling. 'External listeners' allow for centralized processing of unmarshal events in one callback method rather than by type event callbacks.

'Class defined' event callback methods allow any JAXB mapped class to specify its own specific callback methods by defining methods with the following method signature:

// This method is called immediately after the object is created and before the unmarshalling of this   
 // object begins. The callback provides an opportunity to initialize JavaBean properties prior to unmarshalling.  
 void beforeUnmarshal(Unmarshaller, Object parent);  
   
 //This method is called after all the properties (except IDREF) are unmarshalled for this object,   
 //but before this object is set to the parent object.  
 void afterUnmarshal(Unmarshaller, Object parent);

The class defined callback methods should be used when the callback method requires access to non-public methods and/or fields of the class.

The external listener callback mechanism enables the registration of a [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) instance with an [setListener(Listener)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setListener(javax.xml.bind.Unmarshaller.Listener)). The external listener receives all callback events, allowing for more centralized processing than per class defined callback methods. The external listener receives events when unmarshalling proces is marshalling to a JAXB element or to JAXB mapped class.

The 'class defined' and external listener event callback methods are independent of each other, both can be called for one event. The invocation ordering when both listener callback methods exist is defined in [Unmarshaller.Listener.beforeUnmarshal(Object, Object)](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html#beforeUnmarshal(java.lang.Object,%20java.lang.Object)) and [Unmarshaller.Listener.afterUnmarshal(Object, Object)](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html#afterUnmarshal(java.lang.Object,%20java.lang.Object)).

An event callback method throwing an exception terminates the current unmarshal process.

**Since:** JAXB1.0 **See Also:**[JAXBContext](http://docs.google.com/javax/xml/bind/JAXBContext.html), [Marshaller](http://docs.google.com/javax/xml/bind/Marshaller.html), [Validator](http://docs.google.com/javax/xml/bind/Validator.html)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**Unmarshaller.Listener**](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html)    Register an instance of an implementation of this class with [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html) to externally listen for unmarshal events. |

| **Method Summary** | |
| --- | --- |
| | <A extends [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html)>  A | | --- | | [**getAdapter**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getAdapter(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<A> type)            Gets the adapter associated with the specified type. |
| [AttachmentUnmarshaller](http://docs.google.com/javax/xml/bind/attachment/AttachmentUnmarshaller.html) | [**getAttachmentUnmarshaller**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getAttachmentUnmarshaller())() |
| [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) | [**getEventHandler**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getEventHandler())()            Return the current event handler or the default event handler if one hasn't been set. |
| [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) | [**getListener**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getListener())()            Return [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) registered with this [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html). |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getProperty**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getProperty(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Get the particular property in the underlying implementation of Unmarshaller. |
| [Schema](http://docs.google.com/javax/xml/validation/Schema.html) | [**getSchema**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getSchema())()            Get the JAXP 1.3 [Schema](http://docs.google.com/javax/xml/validation/Schema.html) object being used to perform unmarshal-time validation. |
| [UnmarshallerHandler](http://docs.google.com/javax/xml/bind/UnmarshallerHandler.html) | [**getUnmarshallerHandler**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getUnmarshallerHandler())()            Get an unmarshaller handler object that can be used as a component in an XML pipeline. |
| boolean | [**isValidating**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#isValidating())()  **Deprecated.** *since JAXB2.0, please see* [*getSchema()*](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getSchema()) |
| | <A extends [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html)>  void | | --- | | [**setAdapter**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setAdapter(java.lang.Class,%20A))([Class](http://docs.google.com/java/lang/Class.html)<A> type, A adapter)            Associates a configured instance of [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html) with this unmarshaller. |
| void | [**setAdapter**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setAdapter(javax.xml.bind.annotation.adapters.XmlAdapter))([XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html) adapter)            Associates a configured instance of [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html) with this unmarshaller. |
| void | [**setAttachmentUnmarshaller**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setAttachmentUnmarshaller(javax.xml.bind.attachment.AttachmentUnmarshaller))([AttachmentUnmarshaller](http://docs.google.com/javax/xml/bind/attachment/AttachmentUnmarshaller.html) au)            Associate a context that resolves cid's, content-id URIs, to binary data passed as attachments. |
| void | [**setEventHandler**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setEventHandler(javax.xml.bind.ValidationEventHandler))([ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) handler)            Allow an application to register a ValidationEventHandler. |
| void | [**setListener**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setListener(javax.xml.bind.Unmarshaller.Listener))([Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) listener)             Register unmarshal event callback [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) with this [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html). |
| void | [**setProperty**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setProperty(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) name, [Object](http://docs.google.com/java/lang/Object.html) value)            Set the particular property in the underlying implementation of Unmarshaller. |
| void | [**setSchema**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setSchema(javax.xml.validation.Schema))([Schema](http://docs.google.com/javax/xml/validation/Schema.html) schema)            Specify the JAXP 1.3 [Schema](http://docs.google.com/javax/xml/validation/Schema.html) object that should be used to validate subsequent unmarshal operations against. |
| void | [**setValidating**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setValidating(boolean))(boolean validating)  **Deprecated.** *since JAXB2.0, please see* [*setSchema(javax.xml.validation.Schema)*](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setSchema(javax.xml.validation.Schema)) |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(java.io.File))([File](http://docs.google.com/java/io/File.html) f)            Unmarshal XML data from the specified file and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(org.xml.sax.InputSource))([InputSource](http://docs.google.com/org/xml/sax/InputSource.html) source)            Unmarshal XML data from the specified SAX InputSource and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(java.io.InputStream))([InputStream](http://docs.google.com/java/io/InputStream.html) is)            Unmarshal XML data from the specified InputStream and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(org.w3c.dom.Node))([Node](http://docs.google.com/org/w3c/dom/Node.html) node)            Unmarshal global XML data from the specified DOM tree and return the resulting content tree. |
| | <T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> | | --- | | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(org.w3c.dom.Node,%20java.lang.Class))([Node](http://docs.google.com/org/w3c/dom/Node.html) node, [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)            Unmarshal XML data by JAXB mapped declaredType and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(java.io.Reader))([Reader](http://docs.google.com/java/io/Reader.html) reader)            Unmarshal XML data from the specified Reader and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.transform.Source))([Source](http://docs.google.com/javax/xml/transform/Source.html) source)            Unmarshal XML data from the specified XML Source and return the resulting content tree. |
| | <T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> | | --- | | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.transform.Source,%20java.lang.Class))([Source](http://docs.google.com/javax/xml/transform/Source.html) source, [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)            Unmarshal XML data from the specified XML Source by declaredType and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(java.net.URL))([URL](http://docs.google.com/java/net/URL.html) url)            Unmarshal XML data from the specified URL and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.stream.XMLEventReader))([XMLEventReader](http://docs.google.com/javax/xml/stream/XMLEventReader.html) reader)            Unmarshal XML data from the specified pull parser and return the resulting content tree. |
| | <T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> | | --- | | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.stream.XMLEventReader,%20java.lang.Class))([XMLEventReader](http://docs.google.com/javax/xml/stream/XMLEventReader.html) reader, [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)            Unmarshal root element to JAXB mapped declaredType and return the resulting content tree. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.stream.XMLStreamReader))([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader)            Unmarshal XML data from the specified pull parser and return the resulting content tree. |
| | <T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> | | --- | | [**unmarshal**](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.stream.XMLStreamReader,%20java.lang.Class))([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader, [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)            Unmarshal root element to JAXB mapped declaredType and return the resulting content tree. |

| **Method Detail** |
| --- |

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([File](http://docs.google.com/java/io/File.html) f)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified file and return the resulting content tree.

Implements [Unmarshal Global Root Element](#2et92p0).

**Parameters:**f - the file to unmarshal XML data from **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the file parameter is null

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([InputStream](http://docs.google.com/java/io/InputStream.html) is)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified InputStream and return the resulting content tree. Validation event location information may be incomplete when using this form of the unmarshal API.

Implements [Unmarshal Global Root Element](#2et92p0).

**Parameters:**is - the InputStream to unmarshal XML data from **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the InputStream parameter is null

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([Reader](http://docs.google.com/java/io/Reader.html) reader)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified Reader and return the resulting content tree. Validation event location information may be incomplete when using this form of the unmarshal API, because a Reader does not provide the system ID.

Implements [Unmarshal Global Root Element](#2et92p0).

**Parameters:**reader - the Reader to unmarshal XML data from **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the InputStream parameter is null**Since:** JAXB2.0

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([URL](http://docs.google.com/java/net/URL.html) url)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified URL and return the resulting content tree.

Implements [Unmarshal Global Root Element](#2et92p0).

**Parameters:**url - the url to unmarshal XML data from **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the URL parameter is null

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([InputSource](http://docs.google.com/org/xml/sax/InputSource.html) source)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified SAX InputSource and return the resulting content tree.

Implements [Unmarshal Global Root Element](#2et92p0).

**Parameters:**source - the input source to unmarshal XML data from **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the InputSource parameter is null

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([Node](http://docs.google.com/org/w3c/dom/Node.html) node)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal global XML data from the specified DOM tree and return the resulting content tree.

Implements [Unmarshal Global Root Element](#2et92p0).

**Parameters:**node - the document/element to unmarshal XML data from. The caller must support at least Document and Element. **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the Node parameter is null**See Also:**[unmarshal(org.w3c.dom.Node, Class)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(org.w3c.dom.Node,%20java.lang.Class))

### unmarshal

<T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> **unmarshal**([Node](http://docs.google.com/org/w3c/dom/Node.html) node,  
 [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data by JAXB mapped declaredType and return the resulting content tree.

Implements [Unmarshal by Declared Type](#tyjcwt)

**Parameters:**node - the document/element to unmarshal XML data from. The caller must support at least Document and Element.declaredType - appropriate JAXB mapped class to hold node's XML data. **Returns:**[JAXB Element](#3dy6vkm) representation of node **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If any parameter is null**Since:** JAXB2.0

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([Source](http://docs.google.com/javax/xml/transform/Source.html) source)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified XML Source and return the resulting content tree.

Implements [Unmarshal Global Root Element](#2et92p0).

**SAX 2.0 Parser Pluggability**

A client application can choose not to use the default parser mechanism supplied with their JAXB provider. Any SAX 2.0 compliant parser can be substituted for the JAXB provider's default mechanism. To do so, the client application must properly configure a SAXSource containing an XMLReader implemented by the SAX 2.0 parser provider. If the XMLReader has an org.xml.sax.ErrorHandler registered on it, it will be replaced by the JAXB Provider so that validation errors can be reported via the ValidationEventHandler mechanism of JAXB. If the SAXSource does not contain an XMLReader, then the JAXB provider's default parser mechanism will be used.

This parser replacement mechanism can also be used to replace the JAXB provider's unmarshal-time validation engine. The client application must properly configure their SAX 2.0 compliant parser to perform validation (as shown in the example above). Any SAXParserExceptions encountered by the parser during the unmarshal operation will be processed by the JAXB provider and converted into JAXB ValidationEvent objects which will be reported back to the client via the ValidationEventHandler registered with the Unmarshaller. *Note:* specifying a substitute validating SAX 2.0 parser for unmarshalling does not necessarily replace the validation engine used by the JAXB provider for performing on-demand validation.

The only way for a client application to specify an alternate parser mechanism to be used during unmarshal is via the unmarshal(SAXSource) API. All other forms of the unmarshal method (File, URL, Node, etc) will use the JAXB provider's default parser and validator mechanisms.

**Parameters:**source - the XML Source to unmarshal XML data from (providers are only required to support SAXSource, DOMSource, and StreamSource) **Returns:**the newly created root object of the java content tree **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the Source parameter is null**See Also:**[unmarshal(javax.xml.transform.Source, Class)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.transform.Source,%20java.lang.Class))

### unmarshal

<T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> **unmarshal**([Source](http://docs.google.com/javax/xml/transform/Source.html) source,  
 [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified XML Source by declaredType and return the resulting content tree.

Implements [Unmarshal by Declared Type](#tyjcwt)

See [SAX 2.0 Parser Pluggability](#1y810tw)

**Parameters:**source - the XML Source to unmarshal XML data from (providers are only required to support SAXSource, DOMSource, and StreamSource)declaredType - appropriate JAXB mapped class to hold source's xml root element **Returns:**Java content rooted by [JAXB Element](#3dy6vkm) **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If any parameter is null**Since:** JAXB2.0

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified pull parser and return the resulting content tree.

Implements [Unmarshal Global Root Element](#2et92p0).

This method assumes that the parser is on a START\_DOCUMENT or START\_ELEMENT event. Unmarshalling will be done from this start event to the corresponding end event. If this method returns successfully, the reader will be pointing at the token right after the end event.

**Parameters:**reader - The parser to be read. **Returns:**the newly created root object of the java content tree. **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the reader parameter is null [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - If reader is not pointing to a START\_DOCUMENT or START\_ELEMENT event.**Since:** JAXB2.0 **See Also:**[unmarshal(javax.xml.stream.XMLStreamReader, Class)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.stream.XMLStreamReader,%20java.lang.Class))

### unmarshal

<T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> **unmarshal**([XMLStreamReader](http://docs.google.com/javax/xml/stream/XMLStreamReader.html) reader,  
 [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal root element to JAXB mapped declaredType and return the resulting content tree.

This method implements [unmarshal by declaredType](#tyjcwt).

This method assumes that the parser is on a START\_DOCUMENT or START\_ELEMENT event. Unmarshalling will be done from this start event to the corresponding end event. If this method returns successfully, the reader will be pointing at the token right after the end event.

**Parameters:**reader - The parser to be read.declaredType - appropriate JAXB mapped class to hold reader's START\_ELEMENT XML data. **Returns:**content tree rooted by [JAXB Element representation](#3dy6vkm) **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If any parameter is null**Since:** JAXB2.0

### unmarshal

[Object](http://docs.google.com/java/lang/Object.html) **unmarshal**([XMLEventReader](http://docs.google.com/javax/xml/stream/XMLEventReader.html) reader)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal XML data from the specified pull parser and return the resulting content tree.

This method is an [Unmarshal Global Root method](#2et92p0).

This method assumes that the parser is on a START\_DOCUMENT or START\_ELEMENT event. Unmarshalling will be done from this start event to the corresponding end event. If this method returns successfully, the reader will be pointing at the token right after the end event.

**Parameters:**reader - The parser to be read. **Returns:**the newly created root object of the java content tree. **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the reader parameter is null [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - If reader is not pointing to a START\_DOCUMENT or START\_ELEMENT event.**Since:** JAXB2.0 **See Also:**[unmarshal(javax.xml.stream.XMLEventReader, Class)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#unmarshal(javax.xml.stream.XMLEventReader,%20java.lang.Class))

### unmarshal

<T> [JAXBElement](http://docs.google.com/javax/xml/bind/JAXBElement.html)<T> **unmarshal**([XMLEventReader](http://docs.google.com/javax/xml/stream/XMLEventReader.html) reader,  
 [Class](http://docs.google.com/java/lang/Class.html)<T> declaredType)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Unmarshal root element to JAXB mapped declaredType and return the resulting content tree.

This method implements [unmarshal by declaredType](#tyjcwt).

This method assumes that the parser is on a START\_DOCUMENT or START\_ELEMENT event. Unmarshalling will be done from this start event to the corresponding end event. If this method returns successfully, the reader will be pointing at the token right after the end event.

**Parameters:**reader - The parser to be read.declaredType - appropriate JAXB mapped class to hold reader's START\_ELEMENT XML data. **Returns:**content tree rooted by [JAXB Element representation](#3dy6vkm) **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - If any unexpected errors occur while unmarshalling [UnmarshalException](http://docs.google.com/javax/xml/bind/UnmarshalException.html) - If the [ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) returns false from its handleEvent method or the Unmarshaller is unable to perform the XML to Java binding. See [Unmarshalling XML Data](#3znysh7) [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If any parameter is null**Since:** JAXB2.0

### getUnmarshallerHandler

[UnmarshallerHandler](http://docs.google.com/javax/xml/bind/UnmarshallerHandler.html) **getUnmarshallerHandler**()

Get an unmarshaller handler object that can be used as a component in an XML pipeline.

The JAXB Provider can return the same handler object for multiple invocations of this method. In other words, this method does not necessarily create a new instance of UnmarshallerHandler. If the application needs to use more than one UnmarshallerHandler, it should create more than one Unmarshaller.

**Returns:**the unmarshaller handler object**See Also:**[UnmarshallerHandler](http://docs.google.com/javax/xml/bind/UnmarshallerHandler.html)

### setValidating

void **setValidating**(boolean validating)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

**Deprecated.** *since JAXB2.0, please see* [*setSchema(javax.xml.validation.Schema)*](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setSchema(javax.xml.validation.Schema))

Specifies whether or not the default validation mechanism of the Unmarshaller should validate during unmarshal operations. By default, the Unmarshaller does not validate.

This method may only be invoked before or after calling one of the unmarshal methods.

This method only controls the JAXB Provider's default unmarshal-time validation mechanism - it has no impact on clients that specify their own validating SAX 2.0 compliant parser. Clients that specify their own unmarshal-time validation mechanism may wish to turn off the JAXB Provider's default validation mechanism via this API to avoid "double validation".

This method is deprecated as of JAXB 2.0 - please use the new [setSchema(javax.xml.validation.Schema)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setSchema(javax.xml.validation.Schema)) API.

**Parameters:**validating - true if the Unmarshaller should validate during unmarshal, false otherwise **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - if an error occurred while enabling or disabling validation at unmarshal time [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - could be thrown if this method is invoked on an Unmarshaller created from a JAXBContext referencing JAXB 2.0 mapped classes

### isValidating

boolean **isValidating**()  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

**Deprecated.** *since JAXB2.0, please see* [*getSchema()*](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getSchema())

Indicates whether or not the Unmarshaller is configured to validate during unmarshal operations.

This API returns the state of the JAXB Provider's default unmarshal-time validation mechanism.

This method is deprecated as of JAXB 2.0 - please use the new [getSchema()](http://docs.google.com/javax/xml/bind/Unmarshaller.html#getSchema()) API.

**Returns:**true if the Unmarshaller is configured to validate during unmarshal operations, false otherwise **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - if an error occurs while retrieving the validating flag [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - could be thrown if this method is invoked on an Unmarshaller created from a JAXBContext referencing JAXB 2.0 mapped classes

### setEventHandler

void **setEventHandler**([ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) handler)  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Allow an application to register a ValidationEventHandler.

The ValidationEventHandler will be called by the JAXB Provider if any validation errors are encountered during calls to any of the unmarshal methods. If the client application does not register a ValidationEventHandler before invoking the unmarshal methods, then ValidationEvents will be handled by the default event handler which will terminate the unmarshal operation after the first error or fatal error is encountered.

Calling this method with a null parameter will cause the Unmarshaller to revert back to the default event handler.

**Parameters:**handler - the validation event handler **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - if an error was encountered while setting the event handler

### getEventHandler

[ValidationEventHandler](http://docs.google.com/javax/xml/bind/ValidationEventHandler.html) **getEventHandler**()  
 throws [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html)

Return the current event handler or the default event handler if one hasn't been set.

**Returns:**the current ValidationEventHandler or the default event handler if it hasn't been set **Throws:** [JAXBException](http://docs.google.com/javax/xml/bind/JAXBException.html) - if an error was encountered while getting the current event handler

### setProperty

void **setProperty**([String](http://docs.google.com/java/lang/String.html) name,  
 [Object](http://docs.google.com/java/lang/Object.html) value)  
 throws [PropertyException](http://docs.google.com/javax/xml/bind/PropertyException.html)

Set the particular property in the underlying implementation of Unmarshaller. This method can only be used to set one of the standard JAXB defined properties above or a provider specific property. Attempting to set an undefined property will result in a PropertyException being thrown. See  [Supported Properties](#1t3h5sf).

**Parameters:**name - the name of the property to be set. This value can either be specified using one of the constant fields or a user supplied string.value - the value of the property to be set **Throws:** [PropertyException](http://docs.google.com/javax/xml/bind/PropertyException.html) - when there is an error processing the given property or value [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the name parameter is null

### getProperty

[Object](http://docs.google.com/java/lang/Object.html) **getProperty**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [PropertyException](http://docs.google.com/javax/xml/bind/PropertyException.html)

Get the particular property in the underlying implementation of Unmarshaller. This method can only be used to get one of the standard JAXB defined properties above or a provider specific property. Attempting to get an undefined property will result in a PropertyException being thrown. See  [Supported Properties](#1t3h5sf).

**Parameters:**name - the name of the property to retrieve **Returns:**the value of the requested property **Throws:** [PropertyException](http://docs.google.com/javax/xml/bind/PropertyException.html) - when there is an error retrieving the given property or value property name [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - If the name parameter is null

### setSchema

void **setSchema**([Schema](http://docs.google.com/javax/xml/validation/Schema.html) schema)

Specify the JAXP 1.3 [Schema](http://docs.google.com/javax/xml/validation/Schema.html) object that should be used to validate subsequent unmarshal operations against. Passing null into this method will disable validation.

This method replaces the deprecated [setValidating(boolean)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setValidating(boolean)) API.

Initially this property is set to null.

**Parameters:**schema - Schema object to validate unmarshal operations against or null to disable validation **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - could be thrown if this method is invoked on an Unmarshaller created from a JAXBContext referencing JAXB 1.0 mapped classes**Since:** JAXB2.0

### getSchema

[Schema](http://docs.google.com/javax/xml/validation/Schema.html) **getSchema**()

Get the JAXP 1.3 [Schema](http://docs.google.com/javax/xml/validation/Schema.html) object being used to perform unmarshal-time validation. If there is no Schema set on the unmarshaller, then this method will return null indicating that unmarshal-time validation will not be performed.

This method provides replacement functionality for the deprecated [isValidating()](http://docs.google.com/javax/xml/bind/Unmarshaller.html#isValidating()) API as well as access to the Schema object. To determine if the Unmarshaller has validation enabled, simply test the return type for null:

boolean isValidating = u.getSchema()!=null;

**Returns:**the Schema object being used to perform unmarshal-time validation or null if not present **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - could be thrown if this method is invoked on an Unmarshaller created from a JAXBContext referencing JAXB 1.0 mapped classes**Since:** JAXB2.0

### setAdapter

void **setAdapter**([XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html) adapter)

Associates a configured instance of [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html) with this unmarshaller.

This is a convenience method that invokes setAdapter(adapter.getClass(),adapter);.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the adapter parameter is null. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if invoked agains a JAXB 1.0 implementation.**Since:** JAXB2.0 **See Also:**[setAdapter(Class,XmlAdapter)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setAdapter(java.lang.Class,%20A))

### setAdapter

<A extends [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html)> void **setAdapter**([Class](http://docs.google.com/java/lang/Class.html)<A> type,  
 A adapter)

Associates a configured instance of [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html) with this unmarshaller.

Every unmarshaller internally maintains a [Map](http://docs.google.com/java/util/Map.html)<[Class](http://docs.google.com/java/lang/Class.html),[XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html)>, which it uses for unmarshalling classes whose fields/methods are annotated with [XmlJavaTypeAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlJavaTypeAdapter.html).

This method allows applications to use a configured instance of [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html). When an instance of an adapter is not given, an unmarshaller will create one by invoking its default constructor.

**Parameters:**type - The type of the adapter. The specified instance will be used when [XmlJavaTypeAdapter.value()](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlJavaTypeAdapter.html#value()) refers to this type.adapter - The instance of the adapter to be used. If null, it will un-register the current adapter set for this type. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the type parameter is null. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if invoked agains a JAXB 1.0 implementation.**Since:** JAXB2.0

### getAdapter

<A extends [XmlAdapter](http://docs.google.com/javax/xml/bind/annotation/adapters/XmlAdapter.html)> A **getAdapter**([Class](http://docs.google.com/java/lang/Class.html)<A> type)

Gets the adapter associated with the specified type. This is the reverse operation of the [setAdapter(javax.xml.bind.annotation.adapters.XmlAdapter)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setAdapter(javax.xml.bind.annotation.adapters.XmlAdapter)) method.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the type parameter is null. [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if invoked agains a JAXB 1.0 implementation.**Since:** JAXB2.0

### setAttachmentUnmarshaller

void **setAttachmentUnmarshaller**([AttachmentUnmarshaller](http://docs.google.com/javax/xml/bind/attachment/AttachmentUnmarshaller.html) au)

Associate a context that resolves cid's, content-id URIs, to binary data passed as attachments.

Unmarshal time validation, enabled via [setSchema(Schema)](http://docs.google.com/javax/xml/bind/Unmarshaller.html#setSchema(javax.xml.validation.Schema)), must be supported even when unmarshaller is performing XOP processing.

**Throws:** [IllegalStateException](http://docs.google.com/java/lang/IllegalStateException.html) - if attempt to concurrently call this method during a unmarshal operation.

### getAttachmentUnmarshaller

[AttachmentUnmarshaller](http://docs.google.com/javax/xml/bind/attachment/AttachmentUnmarshaller.html) **getAttachmentUnmarshaller**()

### setListener

void **setListener**([Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) listener)

Register unmarshal event callback [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) with this [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html).

There is only one Listener per Unmarshaller. Setting a Listener replaces the previous set Listener. One can unregister current Listener by setting listener to null.

**Parameters:**listener - provides unmarshal event callbacks for this [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html)**Since:** JAXB2.0

### getListener

[Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) **getListener**()

Return [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) registered with this [Unmarshaller](http://docs.google.com/javax/xml/bind/Unmarshaller.html).

**Returns:**registered [Unmarshaller.Listener](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) or null if no Listener is registered with this Unmarshaller.**Since:** JAXB2.0

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Unmarshaller.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/xml/bind/UnmarshalException.html)   [**NEXT CLASS**](http://docs.google.com/javax/xml/bind/Unmarshaller.Listener.html) | [**FRAMES**](http://docs.google.com/index.html?javax/xml/bind/Unmarshaller.html)    [**NO FRAMES**](http://docs.google.com/Unmarshaller.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#2s8eyo1) | FIELD | CONSTR | [METHOD](#17dp8vu) | DETAIL: FIELD | CONSTR | [METHOD](#3rdcrjn) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).