http://www-labs.det.uvigo.es/documentation/LRO/jaxp/jaxp-1_3-html/index.html

for examples for DOM
http://www.java-tips.org/org.w3c.dom/

Inde

PREV NEXT FRAMES NO FRAMES All Classes

example for some area
http://www.cafeconleche.org/books/xmljava/chapters/index.html

Java API for XML Processing (JAXP) 1.4

JAXP has SIX Major parts
1. SAX, 2. DOM, 3. TrAX, 4. XPath, 5. Validation, 6. StAX

Packages	
javax.xml	Defines core XML constants and functionality from the XML specifications.
javax.xml.datatype since JAXP 1.3	XML/Java Type Mappings.
javax.xml.namespace	XML Namespace processing.
javax.xml.parsers	Provides classes allowing the processing of XML documents.
javax.xml.stream	
javax.xml.stream.events	it is Introduced by SUN
javax.xml.stream.util	
javax.xml.transform 1	This package defines the generic APIs for processing transformation instructions, and performing a transformation from source to result. 11
javax.xml.transform.dom 2	This package implements DOM-specific transformation APIs. 3
javax.xml.transform.sax 3	This package implements SAX2-specific transformation APIs. 5
javax.xml.transform.stax 4	Provides for StAX-specific transformation APIs. 2
javax.xml.transform.stream 5	This package implements stream- and URI- specific transformation APIs. 2
javax.xml.validation	This package provides an API for validation of XML documents.

some SAX example
http://www.java-tips.org/java-se-tips/org.xml.sax/

next round, please set the factory SYSTEM property in every example

1st round - SUN Material
2nd round - Deleted some of page
3rd round - deleted sun material itself
finished on 22 - Nov - 08
4th round - 24 Nov 2008

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV PACKAGE NEXT PACKAGE FRAMES NO FRAMES All Classes

Package javax.xml.transform

~~This package defines the generic APIs for processing transformation instructions, and performing a transformation from source to result.~~

it has 11 classess/interfaces

See:

Description

Source can be XML only, the XML can be in any form like sax event, dom tree.

Interface Summary

ErrorListener	To provide customized error handling, implement this interface and use the `setErrorListener` method to register an instance of the implmentation with the `Transformer`.
Result	~~An object that implements this interface contains the information needed to build a transformation result tree~~.
Source	An object that implements this interface contains the information needed to act as source input (XML source or transformation instructions).
SourceLocator	This interface is primarily for the purposes of reporting where an error occurred in the XML source or transformation instructions.
Templates	An object that implements this interface is the runtime representation of processed transformation instructions.
URIResolver	An object that implements this interface that can be called by the processor to turn a URI used in document(), xsl:import, or xsl:include into a Source object.

i.e XML file or XSL file

that is XSL file

so it is used only with XSL
(Template object)

Class Summary

OutputKeys	Provides string constants that can be used to set output properties for a Transformer, or to retrieve output properties from a Transformer or Templates object.
Transformer	An instance of this abstract class can transform a source tree into a result tree.
TransformerFactory	A TransformerFactory instance can be used to create `Transformer` and `Templates` objects.

Exception Summary

TransformerConfigurationException	Indicates a serious configuration error.

TransformerException	This class specifies an exceptional condition that occured during the transformation process.

Error Summary	
TransformerFactoryConfigurationError	Thrown when a problem with configuration with the Transformer Factories exists.

Package javax.xml.transform Description

This package defines the generic APIs for processing 1 transformation instructions, and 2 performing a transformation from source to result. These interfaces have no dependencies on SAX or the DOM standard, and try to make as few assumptions as possible about the details of the source and result of a transformation. It achieves this by defining `Source` and `Result` interfaces.

~~To define concrete classes for the user, the API defines specializations of the interfaces found at the root level. These interfaces are found in `javax.xml.transform.sax`, `javax.xml.transform.dom`, and `javax.xml.transform.stream`.~~

Creating Objects

~~The API allows a concrete `TransformerFactory` object to be created from the static function `TransformerFactory.newInstance()`.~~

Specification of Inputs and Outputs

~~This API defines two interface objects called `Source` and `Result`. In order to pass Source and Result objects to the interfaces, concrete classes must be used. Three concrete representations are defined for each of these objects: `StreamSource` and `StreamResult`, `SAXSource` and `SAXResult`, and `DOMSource` and `DOMResult`.~~ Each of these objects defines a FEATURE string (which is i the form of a URL), which can be passed into `TransformerFactory.getFeature(java.lang.String)` to see if the given type of Source or Result object is supported. For instance, to test if a DOMSource and a StreamResult is supported, you can apply the following test.

```
TransformerFactory tfactory = TransformerFactory.newInstance();
if (tfactory.getFeature(DOMSource.FEATURE) && tfactory.getFeature(StreamResult.
FEATURE)) {
...
}
```

Qualified Name Representation

~~Namespaces present something of a problem area when dealing with XML objects. Qualified Names appear in XML markup as prefixed names. But the prefixes themselves do not hold identity. Rather, it is the URIs that they contextually map to that hold the identity. Therefore, when passing a Qualified Name like "xyz:foo" among Java programs, one must provide a means to map "xyz" to a namespace.~~

~~One solution has been to create a "QName" object that holds the namespace URI, as well as the prefix and local name, but this is not always an optimal solution, as when, for example, you want to use unique strings as keys in a dictionary object. Not having a string representation also makes it difficult to specify a namespaced identity outside the context of an XML document.~~

~~In order to pass namespaced values to transformations, for instance when setting a property or a parameter on a `Transformer` object, this specification defines that a String "qname" object parameter be passed as two-part string, the namespace URI enclosed in curly braces ({}), followed by the local name. If the qname has a null URI, then the String object only contains the local name. An application can safely check for a non-null URI by testing to see if the first character of the name is a '{' character.~~

~~For example, if a URI and local name were obtained from an element defined with <xyz:foo xmlns:xyz="http://xyz.foo.com/yada/baz.html"/>, then the Qualified Name would be "{http://xyz.foo.com/yada/baz.html}foo". Note that the prefix is lost.~~

Result Tree Serialization

Serialization of the result tree to a stream can be controlled with the `Transformer.setOutputProperties(java.util.Properties)` and the `Transformer.setOutputProperty(java.lang.String, java.lang.String)` methods. These properties only apply to stream results, they have no effect when the result is a DOM tree or SAX event stream.

~~Strings that match the XSLT specification for xsl:output attributes can be referenced from the `OutputKeys` class. Other strings can be specified as well. If the transformer does not recognize an output key, a `IllegalArgumentException` is thrown, unless the key name is namespace qualified. Output key names that are namespace qualified are always allowed, although they may be ignored by some implementations.~~

If all that is desired is the simple identity transformation of a source to a result, then `TransformerFactory` provides a `TransformerFactory.newTransformer()` method with no arguments. This method creates a Transformer that effectively copies the source to the result. This method may be used to create a DOM from SAX events or to create an XML or HTML stream from a DOM or SAX events.

Oh ✓ oh

Exceptions and Error Reporting

~~The transformation API throw three types of specialized exceptions. A `TransformerFactoryConfigurationError` is parallel to the `FactoryConfigurationError`, and is thrown when a configuration problem with the TransformerFactory exists. This error will typically be thrown when~~

~~the transformation factory class specified with the "javax.xml.transform.TransformerFactory" system property cannot be found or instantiated.~~

~~A TransformerConfigurationException may be thrown if for any reason a Transformer can not be created. A TransformerConfigurationException may be thrown if there is a syntax error in the transformation instructions, for example when TransformerFactory.newTransformer(javax.xml.transform.Source) is called.~~

~~TransformerException is a general exception that occurs during the course of a transformation. A transformer exception may wrap another exception, and if any of the TransformerException.printStackTrace() methods are called on it, it will produce a list of stack dumps, starting from the most recent. The transformer exception also provides a SourceLocator object which indicates where in the source tree or transformation instructions the error occurred. TransformerException.getMessageAndLocation() may be called to get an error message with location info, and TransformerException.getLocationAsString() may be called to get just the location string.~~

Transformation warnings and errors are sent to an ErrorListener, at which point the application may decide to report the error or warning, and may decide to throw an Exception for a non-fatal error. The ErrorListener may be set via TransformerFactory.setErrorListener(javax.xml.transform.ErrorListener) for reporting errors that have to do with syntax errors in the transformation instructions, or via Transformer.setErrorListener(javax.xml.transform.ErrorListener) to report errors that occur during the transformation. The ErrorListener on both objects will always be valid and non-null, whether set by the application or a default implementation provided by the processor. The default implementation provided by the processor will report all warnings and errors to System.err and does not throw any Exceptions. Applications are *strongly* encouraged to register and use ErrorListeners that insure proper behavior for warnings and errors.

~~Resolution of URIs within a transformation~~

~~The API provides a way for URIs referenced from within the stylesheet instructions or within the transformation to be resolved by the calling application. This can be done by creating a class that implements the URIResolver interface, with its one method, URIResolver.resolve(java.lang.String, java.lang.String), and use this class to set the URI resolution for the transformation instructions or transformation with TransformerFactory.setURIResolver(javax.xml.transform.URIResolver) or Transformer.setURIResolver(javax.xml.transform.URIResolver). The URIResolver.resolve method takes two String arguments, the URI found in the stylesheet instructions or built as part of the transformation process, and the base URI against which the first argument will be made absolute if the absolute URI is required. The returned Source object must be usable by the transformer, as specified in its implemented features.~~

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV PACKAGE **NEXT PACKAGE** **FRAMES** **NO FRAMES** **All Classes**

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**
PREV PACKAGE NEXT PACKAGE FRAMES NO FRAMES All Classes

Package javax.xml.transform.dom

it has 3 classes/ interfaces

This package implements DOM-specific transformation APIs.

See:
Description

This DOM API can be used only if EITHER ONE IS AS node.

Interface Summary	
DOMLocator	Indicates the position of a node in a source DOM, intended primarily for error reporting.

Class Summary	
DOMResult	Acts as a holder for a transformation result tree in the form of a Document Object Model (DOM) tree.
DOMSource	Acts as a holder for a transformation Source tree in the form of a Document Object Model (DOM) tree.

Package javax.xml.transform.dom Description

get one sample from net

This package implements DOM-specific transformation APIs.

The `DOMSource` class allows the client of the implementation of this API to specify a DOM `Node` as the source of the input tree. The model of how the Transformer deals with the DOM tree in terms of mismatches with the XSLT data model or other data models is beyond the scope of this document. Any of the nodes derived from `Node` are legal input.

The `DOMResult` class allows a `Node` to be specified to which result DOM nodes will be appended. If an output node is not specified, the transformer will use `DocumentBuilder.newDocument()` to create an output `Document` node. If a node is specified, it should be one of the following: `Document`,

Overview **Package** Class **Use**

PREV PACKAGE **NEXT PACKAGE**

i have skipped to get THROUGH of **SAXTransformerFactory, TransformerHandler and TemplatesHandler**.

i have not find any good material. These 3 are has less number of methods. so, it wont take much time to familiar, once got example or article. SO COOL

Package javax.xml.transform.sax

This package implements SAX2-specific transformation APIs.

See:

Description

5 classess / interface

Interface Summary	
TemplatesHandler	A SAX ContentHandler that may be used to process SAX parse events (parsing transformation instructions) into a Templates object.
TransformerHandler	A TransformerHandler listens for SAX ContentHandler parse events and transforms them to a Result.

XSL file

Class Summary	
SAXResult	Acts as an holder for a transformation Result.
SAXSource	Acts as an holder for SAX-style Source.
SAXTransformerFactory	This class extends TransformerFactory to provide SAX-specific factory methods.

Package javax.xml.transform.sax Description

custom classes

This package implements SAX2-specific transformation APIs. It provides classes which allow input from `ContentHandler` events, and also classes that produce org.xml.sax.ContentHandler events. It also provides methods to set the input source as an `XMLReader`, or to use a `InputSource` as the source. It also allows the creation of a `XMLFilter`, which enables transformations to "pull" from other transformations, and lets the transformer to be used polymorphically as an `XMLReader`.

The `SAXSource` class allows the setting of an `XMLReader` to be used for "pulling" parse events, and

Overview **Package** Class **Use** **Tree** **Deprecated** **Index** **Help**

PREV PACKAGE NEXT PACKAGE FRAMES NO FRAMES All Classes

it has 2 classes / interfaces

Package javax.xml.transform.stax

Provides for StAX-specific transformation APIs.

See:
Description

As of now i am **dont** have xslt engine has been implemented this feature

```
logger.info(" StAXResult "+transformerFactory.getFeature(StAXResult.FEATURE));
logger.info(" StAXSource "+transformerFactory.getFeature(StAXSource.FEATURE));
above two statements are return false
```

Class Summary	
StAXResult	Acts as a holder for an XML `Result` in the form of a StAX writer,i.e.
StAXSource	Acts as a holder for an XML `Source` in the form of a StAX reader,i.e.

Package javax.xml.transform.stax Description

Provides for StAX-specific transformation APIs. TODO: better description(s).

Package Specification

- JSR 173: Streaming API for XML

Related Documentation

For overviews, tutorials, examples, guides, and tool documentation, please see:

- TODO: Refer to non-spec documentation

- @see XMLStreamReader
- @see XMLEventReader

Since:
1.6

Package javax.xml.transform.stream

This package implements stream- and URI- specific transformation APIs.

See:
Description

it has 2 classes

actually this one have to use if output be html /text / image

Class Summary

Class Summary	
StreamResult	Acts as an holder for a transformation result, which may be XML, plain Text, HTML, or some other form of markup.
StreamSource	Acts as an holder for a transformation Source in the form of a stream of XML markup.

Package javax.xml.transform.stream Description

This package implements stream- and URI- specific transformation APIs.

The `StreamSource` class provides methods for specifying `InputStream` input, `Reader` input, and URL input in the form of strings. Even if an input stream or reader is specified as the source, `StreamSource.setSystemId(java.lang.String)` should still be called, so that the transformer can know from where it should resolve relative URIs. The public identifier is always optional: if the application writer includes one, it will be provided as part of the `SourceLocator` information.

The `StreamResult` class provides methods for specifying `OutputStream`, `Writer`, or an output system ID, as the output of the transformation result.

Normally streams should be used rather than readers or writers, for both the Source and Result, since readers and writers already have the encoding established to and from the internal Unicode format. However, there are times when it is useful to write to a character stream, such as when using a StringWriter in order to write to a String, or in the case of reading source XML from a StringReader.

3 - Jan - 09

javax.xml.transform.dom

Interface DOMLocator

it has 1 method

All Superinterfaces:

SourceLocator

this is from TrAX package itself

```
public interface DOMLocator
```

extends SourceLocator

this will come only when error occurred.

Indicates the position of a node in a source DOM, intended primarily for error reporting. To use a DOMLocator, the receiver of an error must downcast the `SourceLocator` object returned by an exception. A `Transformer` may use this object for purposes other than error reporting, for instance, to indicate the source node that originated a result node.

Method Summary

`Node`	**`getOriginatingNode`**`()` Return the node where the event occurred.

Methods inherited from interface javax.xml.transform.SourceLocator
`getColumnNumber, getLineNumber, getPublicId, getSystemId`

Method Detail

getOriginatingNode

```
Node getOriginatingNode()
```

Return the node where the event occurred.

Returns:

The node that is the location for the event.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

3 - Jan -09

it has only 4 methods
2 - getter / setter for Node
2 - getter / setter for Sibling Node

it has 5 constructor

javax.xml.transform.dom

Class DOMResult

```
java.lang.Object
  └ javax.xml.transform.dom.DOMResult
```

All Implemented Interfaces:

Result

`public class **DOMResult**`

extends java.lang.Object
implements Result

Acts as a holder for a transformation result tree in the form of a Document Object Model (DOM) tree.

If no output DOM source is set, the transformation will create a Document node as the holder for the result of the transformation, which may be retrieved with `getNode()`.

Version:

$Revision: 1.3 $, $Date: 2005/11/03 19:34:24 $

Author:

Jeff Suttor

Field Summary

static java.lang.String	**FEATURE** If TransformerFactory.getFeature(java.lang.String) returns true when passed this value as an argument, the Transformer supports Result output of this type.

Fields inherited from interface javax.xml.transform.Result
PI_DISABLE_OUTPUT_ESCAPING, PI_ENABLE_OUTPUT_ESCAPING

Constructor Summary
DOMResult() Zero-argument default constructor. added as LAST child
DOMResult(Node node) Use a DOM node to create a new output target. added as child at before to this sibling
DOMResult(Node node, Node nextSibling) Use a DOM node to create a new output target specifying the child node where the result nodes should be inserted before.
DOMResult(Node node, Node nextSibling, java.lang.String systemId) Use a DOM node to create a new output target specifying the child node where the result nodes should be inserted before and the specified System ID.
DOMResult(Node node, java.lang.String systemId) Use a DOM node to create a new output target with the specified System ID.

Method Summary	
Node	**getNextSibling**() Get the child node before which the result nodes will be inserted.
Node	**getNode**() Get the node that will contain the result DOM tree.
java.lang.String	**getSystemId**() Get the System Identifier. by inheritance
void	**setNextSibling**(Node nextSibling) Set the child node before which the result nodes will be inserted.

`void`	`setNode(Node node)` Set the node that will contain the result DOM tree.
`void`	`setSystemId(java.lang.String systemId)` by inheritance Set the systemId that may be used in association with the node.

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(java.lang.String)` returns `true` when passed this value as an argument, the `Transformer` supports `Result` output of this type.

See Also:

Constant Field Values

Constructor Detail

DOMResult

```
public DOMResult()
```

Zero-argument default constructor.

`node`, `siblingNode` and `systemId` will be set to `null`.

DOMResult

```
public DOMResult(Node node)
```

Use a DOM node to create a new output target.

In practice, the node should be a `Document` node, a `DocumentFragment` node, or a `Element` node. In other words, a node that accepts children.

`siblingNode` and `systemId` will be set to `null`.

Parameters:

`node` - The DOM node that will contain the result tree.

DOMResult

```
public DOMResult(Node node,
                 java.lang.String systemId)
```

Use a DOM node to create a new output target with the specified System ID.

In practice, the node should be a `Document` node, a `DocumentFragment` node, or a `Element` node. In other words, a node that accepts children.

`siblingNode` will be set to `null`.

Parameters:

`node` - The DOM node that will contain the result tree.

`systemId` - The system identifier which may be used in association with this node.

DOMResult

```
public DOMResult(Node node,
                 Node nextSibling)
```

Use a DOM node to create a new output target specifying the child node where the result nodes should be inserted before.

In practice, `node` and `nextSibling` should be a `Document` node, a `DocumentFragment` node, or a `Element` node. In other words, a node that accepts children.

Use `nextSibling` to specify the child node where the result nodes should be inserted before. If `nextSibling` is not a sibling of `node`, then an `IllegalArgumentException` is thrown. If `node` is `null` and `nextSibling` is not `null`, then an `IllegalArgumentException` is thrown. If `nextSibling` is `null`, then the behavior is the same as calling `DOMResult(Node node)`, i.e. append the result nodes as the last child of the specified `node`.

`systemId` will be set to `null`.

Parameters:

`node` - The DOM node that will contain the result tree.

`nextSibling` - The child node where the result nodes should be inserted before.

Throws:

`java.lang.IllegalArgumentException` - If `nextSibling` is not a sibling of `node` or `node` is `null` and `nextSibling` is not `null`.

Since:

1.5

DOMResult

```
public DOMResult(Node node,
                 Node nextSibling,
                 java.lang.String systemId)
```

Use a DOM node to create a new output target specifying the child node where the result nodes should be inserted before and the specified System ID.

In practice, `node` and `nextSibling` should be a `Document` node, a `DocumentFragment` node, or a `Element` node. In other words, a node that accepts children.

Use `nextSibling` to specify the child node where the result nodes should be inserted before. If `nextSibling` is not a sibling of `node`, then an `IllegalArgumentException` is thrown. If `node` is `null` and `nextSibling` is not `null`, then an `IllegalArgumentException` is thrown. If `nextSibling` is `null`, then the behavior is the same as calling `DOMResult(Node node, String systemId)`, i.e. append the result nodes as the last child of the specified node and use the specified System ID.

Parameters:

`node` - The DOM node that will contain the result tree.

`nextSibling` - The child node where the result nodes should be inserted before.

`systemId` - The system identifier which may be used in association with this node.

Throws:

`java.lang.IllegalArgumentException` - If `nextSibling` is not a sibling of `node` or `node` is `null` and `nextSibling` is not `null`.

Since:

1.5

Method Detail

setNode

```
public void setNode(Node node)
```

Set the node that will contain the result DOM tree.

In practice, the node should be a `Document` node, a `DocumentFragment` node, or a `Element` node. In other words, a node that accepts children.

An `IllegalStateException` is thrown if `nextSibling` is not `null` and `node` is not a parent of `nextSibling`. An `IllegalStateException` is thrown if `node` is `null` and `nextSibling` is not `null`.

Parameters:

`node` - The node to which the transformation will be appended.

Throws:

`java.lang.IllegalStateException` - If `nextSibling` is not `null` and `nextSibling` is not a child of `node` or `node` is `null` and `nextSibling` is not `null`.

getNode

`public Node getNode()`

Get the node that will contain the result DOM tree.

If no node was set via `DOMResult(Node node)`, `DOMResult(Node node, String systeId)`, `DOMResult(Node node, Node nextSibling)`, `DOMResult(Node node, Node nextSibling, String systemId)` or `setNode(Node node)`, then the node will be set by the transformation, and may be obtained from this method once the transformation is complete. Calling this method before the transformation will return `null`.

Returns:

The node to which the transformation will be appended.

---

setNextSibling

`public void setNextSibling(Node nextSibling)`

Set the child node before which the result nodes will be inserted.

Use `nextSibling` to specify the child node before which the result nodes should be inserted. If `nextSibling` is not a descendant of `node`, then an `IllegalArgumentException` is thrown. If `node` is `null` and `nextSibling` is not `null`, then an `IllegalStateException` is thrown. If `nextSibling` is `null`, then the behavior is the same as calling `DOMResult(Node node)`, i.e. append the result nodes as the last child of the specified `node`.

Parameters:

`nextSibling` - The child node before which the result nodes will be inserted.

Throws:

`java.lang.IllegalArgumentException` - If `nextSibling` is not a descendant of `node`.

`java.lang.IllegalStateException` - If `node` is `null` and `nextSibling` is not `null`.

Since:

1.5

---

getNextSibling

```
public Node getNextSibling()
```

Get the child node before which the result nodes will be inserted.

If no node was set via `DOMResult(Node node, Node nextSibling)`, `DOMResult(Node node, Node nextSibling, String systemId)` or `setNextSibling(Node nextSibling)`, then `null` will be returned.

Returns:
The child node before which the result nodes will be inserted.

Since:
1.5

---

setSystemId

```
public void setSystemId(java.lang.String systemId)
```

Set the systemId that may be used in association with the node.

Specified by:
`setSystemId` in interface `Result`

Parameters:
`systemId` - The system identifier as a URI string.

---

getSystemId

```
public java.lang.String getSystemId()
```

Get the System Identifier.

If no System ID was set via `DOMResult(Node node, String systemId)`, `DOMResult(Node node, Node nextSibling, String systemId)` or `setSystemId(String systemId)`, then `null` will be returned.

Specified by:

`getSystemId` in interface `Result`

Returns:

The system identifier.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview Package Class Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS **FRAMES NO FRAMES All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

it has 2 method

2 - getter / setter for Node

it has 3 constructor

javax.xml.transform.dom

Class DOMSource

```
java.lang.Object
  └ javax.xml.transform.dom.DOMSource
```

All Implemented Interfaces:

Source

`public class DOMSource`

extends java.lang.Object
implements Source

Acts as a holder for a transformation Source tree in the form of a Document Object Model (DOM) tree.

Note that XSLT requires namespace support. Attempting to transform a DOM that was not contructed with a namespace-aware parser may result in errors. ~~Parsers can be made namespace aware by calling `DocumentBuilderFactory.setNamespaceAware(boolean awareness)`.~~

Version:

$Revision: 1.3 $, $Date: 2005/11/03 19:34:24 $

Author:

Jeff Suttor

See Also:

Document Object Model (DOM) Level 2 Specification

Field Summary

static java.lang.String	**FEATURE** If TransformerFactory.getFeature(java.lang.String) returns true when passed this value as an argument, the Transformer supports Source input of this type.

Constructor Summary
DOMSource() Zero-argument default constructor.
DOMSource(Node n) Create a new input source with a DOM node.
DOMSource(Node node, java.lang.String systemID) Create a new input source with a DOM node, and with the system ID also passed in as the base URI.

Method Summary	
Node	**getNode**() Get the node that represents a Source DOM tree.
java.lang.String	**getSystemId**() by inheritance Get the base ID (URL or system ID) from where URLs will be resolved.
void	**setNode**(Node node) Set the node that will represents a Source DOM tree.
void	**setSystemId**(java.lang.String systemID) by inheritance Set the base ID (URL or system ID) from where URLs will be resolved.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

FEATURE

`public static final java.lang.String` **`FEATURE`**

If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Source input of this type.

See Also:
Constant Field Values

Constructor Detail

DOMSource

`public` **`DOMSource`**`()`

Zero-argument default constructor. If this constructor is used, and no DOM source is set using `setNode(Node node)` , then the `Transformer` will create an empty source `Document` using `DocumentBuilder.newDocument()`.

See Also:
`Transformer.transform(Source xmlSource, Result outputTarget)`

DOMSource

`public` **`DOMSource`**`(Node n)`

Create a new input source with a DOM node. The operation will be applied to the subtree rooted at this node. In XSLT, a "/" pattern still means the root of the tree (not the subtree), and the evaluation of global variables and parameters is done from the root node also.

Parameters:
`n` - The DOM node that will contain the Source tree.

DOMSource

```
public DOMSource(Node node,
                 java.lang.String systemID)
```

Create a new input source with a DOM node, and with the system ID also passed in as the base URI.

the remaining path will be appended from root node to it current node

Parameters:

`node` - The DOM node that will contain the Source tree.

`systemID` - Specifies the base URI associated with node.

Method Detail

setNode

```
public void setNode(Node node)
```

~~Set the node that will represents a Source DOM tree.~~

~~**Parameters:**~~

~~`node` - The node that is to be transformed.~~

---

getNode

```
public Node getNode()
```

~~Get the node that represents a Source DOM tree.~~

~~**Returns:**~~

~~The node that is to be transformed.~~

---

setSystemId

base id means- just full path of document. but, node path with in document will be appended dynamically

```
public void setSystemId(java.lang.String systemID)
```

Set the base ID (URL or system ID) from where URLs will be resolved.

Specified by:

`setSystemId` in interface `Source`

Parameters:

`systemID` - Base URL for this DOM tree.

---

getSystemId

```
public java.lang.String getSystemId()
```

~~Get the base ID (URL or system ID) from where URLs will be resolved.~~

~~**Specified by:**~~

~~`getSystemId` in interface `Source`~~

~~**Returns:**~~

~~Base URL for this DOM tree.~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

---

3 - Jan - 09

Only it has 3 methods

javax.xml.transform

Interface ErrorListener

`public interface ErrorListener`

~~To provide customized error handling, implement this interface and use the `setErrorListener` method to register an instance of the implmentation with the `Transformer`. The `Transformer` then reports all errors and warnings through this interface.~~

~~If an application does *not* register its own custom `ErrorListener`, the default `ErrorListener` is used which reports all warnings and errors to `System.err` and does not throw any `Exception`s. Applications are *strongly* encouraged to register and use `ErrorListener`s that insure proper behavior for warnings and errors.~~

~~For transformation errors, a `Transformer` must use this interface instead of throwing an `Exception`: it is up to the application to decide whether to throw an `Exception` for different types of errors and warnings. Note however that the `Transformer` is not required to continue with the transformation after a call to `fatalError(TransformerException exception)`.~~

~~`Transformer`s~~ may use ~~this mechanism to report XML parsing errors as well as transformation errors~~.

Method Summary

`void`	`error(TransformerException exception)` Receive notification of a recoverable error.
`void`	`fatalError(TransformerException exception)` Receive notification of a non-recoverable error.

void	**warning**(TransformerException exception) Receive notification of a warning.

Method Detail

warning

so, if error handler dont want to continue just throw exception. this rule apply all SAX / DOM / TrAX error handlers

```
void warning(TransformerException exception)
             throws TransformerException
```

I think, This method will be used only with DTD.

Receive notification of a warning.

Transformer can use this method to report conditions that are not errors or fatal errors. The default behaviour is to take no action.

After invoking this method, the Transformer must continue with the transformation. It should still be possible for the application to process the document through to the end.

Parameters:

exception - The warning information encapsulated in a transformer exception.

Throws:

TransformerException - if the application chooses to discontinue the transformation.

See Also:

TransformerException

---

error

so, if error handler dont want to continue just throw exception. this rule apply all SAX / DOM / TrAX error handlers

```
void error(TransformerException exception)
           throws TransformerException
```

Receive notification of a recoverable error.

~~The transformer must continue to try and provide normal transformation after invoking this method. It should still be possible for the application to process the document through to the end~~

~~if no other errors are encountered.~~

Parameters:

`exception` - The error information encapsulated in a transformer exception.

Throws:

`TransformerException` - if the application chooses to discontinue the transformation.

See Also:

`TransformerException`

---

so, if error handler dont want to continue just throw exception. this rule apply all SAX / DOM / TrAX error handlers

fatalError

```
void fatalError(TransformerException exception)
                throws TransformerException
```

~~Receive notification of a non-recoverable error.~~

~~The processor may choose to continue, but will not normally proceed to a successful completion.~~

~~The method should throw an exception if it is unable to process the error, or if it wishes execution to terminate immediately. The processor will not necessarily honor this request.~~

~~**Parameters:**~~

~~`exception` - The error information encapsulated in a `TransformerException`.~~

~~**Throws:**~~

~~`TransformerException` - if the application chooses to discontinue~~ the ~~transformation.~~

~~**See Also:**~~

~~`TransformerException`~~

Overview Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

3 - Jan - 09

javax.xml.transform

Class OutputKeys

These are the output properties, we can set to XSLT Engine

```
java.lang.Object
  └ javax.xml.transform.OutputKeys
```

```
public class OutputKeys
```

it deals with different attribute of <xsl:output> tag. this tag has around 10 attribute

i have skipped. Gets hands on at the time usage (3 - jan -09)

extends java.lang.Object

Provides string constants that can be used to set output properties for a Transformer, or to retrieve output properties from a Transformer or Templates object.

All the fields in this class are read-only.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation

Field Summary

`static java.lang.String`	**`CDATA_SECTION_ELEMENTS`**
	cdata-section-elements = *expanded names*.

`static java.lang.String`	**`DOCTYPE_PUBLIC`**
	doctype-public = *string*.

`static java.lang.String`	**`DOCTYPE_SYSTEM`**
	doctype-system = *string*.

`static java.lang.String`	**`ENCODING`**
	encoding = *string*.

`static java.lang.String`	**INDENT** indent = "yes" | "no".
`static java.lang.String`	**MEDIA_TYPE** media-type = *string*.
`static java.lang.String`	**METHOD** method = "xml" | "html" | "text" | *expanded name*.
`static java.lang.String`	**OMIT_XML_DECLARATION** omit-xml-declaration = "yes" | "no".
`static java.lang.String`	**STANDALONE** standalone = "yes" | "no".
`static java.lang.String`	**VERSION** version = *nmtoken*.

Method Summary

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

METHOD

`public static final java.lang.String` **`METHOD`**

method = "xml" | "html" | "text" | *expanded name*.

The value of the method property identifies the overall method that should be used for outputting the result tree. Other non-namespaced values may be used, such as "xhtml", but, if accepted, the handling of such values is implementation defined. ~~If any of the method values are not accepted and are not namespace qualified, then `Transformer.setOutputProperty(java.lang.String, java.lang.String)` or `Transformer.setOutputProperties(java.util.Properties)` will throw a `IllegalArgumentException`.~~

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

---

VERSION

```
public static final java.lang.String VERSION
```

version = *nmtoken*.

`version` specifies the version of the output method.

When the output method is "xml", the version value specifies the version of XML to be used for outputting the result tree. The default value for the xml output method is 1.0. When the output method is "html", the version value indicates the version of the HTML. The default value for the xml output method is 4.0, which specifies that the result should be output as HTML conforming to the HTML 4.0 Recommendation [HTML]. If the output method is "text", the version property is ignored.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

---

ENCODING

```
public static final java.lang.String ENCODING
```

encoding = *string*.

`encoding` specifies the preferred character encoding that the Transformer should use to encode sequences of characters as sequences of bytes. The value of the encoding property should be treated case-insensitively. The value must only contain characters in the range #x21 to #x7E (i.e., printable ASCII characters). The value should either be a `charset` registered with the Internet Assigned Numbers Authority [IANA], [RFC2278] or start with `X-`.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

---

OMIT_XML_DECLARATION

```
public static final java.lang.String OMIT_XML_DECLARATION
```

omit-xml-declaration = "yes" | "no".

`omit-xml-declaration` specifies whether the XSLT processor should output an XML declaration; the value must be `yes` or `no`.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

---

STANDALONE

```
public static final java.lang.String STANDALONE
```

standalone = "yes" | "no".

`standalone` specifies whether the Transformer should output a standalone document declaration; the value must be `yes` or `no`.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

---

DOCTYPE_PUBLIC

```
public static final java.lang.String DOCTYPE_PUBLIC
```

doctype-public = *string*.

See the documentation for the `DOCTYPE_SYSTEM` property for a description of what the value of the key should be.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

DOCTYPE_SYSTEM

```
public static final java.lang.String DOCTYPE_SYSTEM
```

doctype-system = *string*.

`doctype-system` specifies the system identifier to be used in the document type declaration.

If the doctype-system property is specified, the xml output method should output a document type declaration immediately before the first element. The name following <!DOCTYPE should be the name of the first element. If doctype-public property is also specified, then the xml output method should output PUBLIC followed by the public identifier and then the system identifier; otherwise, it should output SYSTEM followed by the system identifier. The internal subset should be empty. The value of the doctype-public property should be ignored unless the doctype-system property is specified.

If the doctype-public or doctype-system properties are specified, then the html output method should output a document type declaration immediately before the first element. The name following <!DOCTYPE should be HTML or html. If the doctype-public property is specified, then the output method should output PUBLIC followed by the specified public identifier; if the doctype-system property is also specified, it should also output the specified system identifier following the public identifier. If the doctype-system property is specified but the doctype-public property is not specified, then the output method should output SYSTEM followed by the specified system identifier.

`doctype-system` specifies the system identifier to be used in the document type declaration.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field

Values

CDATA_SECTION_ELEMENTS

```
public static final java.lang.String CDATA_SECTION_ELEMENTS
```

cdata-section-elements = *expanded names*.

`cdata-section-elements` specifies a whitespace delimited list of the names of elements whose text node children should be output using CDATA sections. Note that these names must use the format described in the section Qualfied Name Representation in `javax.xml.transform`.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation., Constant Field Values

---

INDENT

```
public static final java.lang.String INDENT
```

indent = "yes" | "no".

`indent` specifies whether the Transformer may add additional whitespace when outputting the result tree; the value must be `yes` or `no`.

See Also:

section 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

---

MEDIA_TYPE

```
public static final java.lang.String MEDIA_TYPE
```

media-type = *string*.

`media-type` specifies the media type (MIME content type) of the data that results from outputting the result tree. The `charset` parameter should not be specified explicitly; instead, when the top-level media type is `text`, a `charset` parameter should be added according to the character encoding actually used by the output method.

See Also:

s ection 16 of the XSL Transformations (XSLT) W3C Recommendation, Constant Field Values

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

3 - Jan - 09

javax.xml.transform

Interface Result

only it has 2 methods

All Known Implementing Classes:

DOMResult, SAXResult, StAXResult, StreamResult

```
public interface Result
```

An object that implements this interface contains the information needed to build a transformation result tree.

Author:

Jeff Suttor

~~Field Summary~~

~~static java.lang.String~~	~~PI_DISABLE_OUTPUT_ESCAPING~~ ~~The name of the processing instruction that is sent if the result tree disables output escaping.~~
~~static java.lang.String~~	~~PI_ENABLE_OUTPUT_ESCAPING~~ ~~The name of the processing instruction that is sent if the result tree enables output escaping at some point after having received a PI_DISABLE_OUTPUT_ESCAPING processing instruction.~~

Method Summary

java.lang.String	getSystemId() Get the system identifier that was set with setSystemId.

`void`	`setSystemId(java.lang.String systemId)` Set the system identifier for this Result.

Field Detail

PI_DISABLE_OUTPUT_ESCAPING

~~`static final java.lang.String` **`PI_DISABLE_OUTPUT_ESCAPING`**~~

~~The name of the processing instruction that is sent if the result tree disables output escaping.~~

~~Normally, result tree serialization escapes & and < (and possibly other characters) when outputting text nodes. This ensures that the output is well-formed XML. However, it is sometimes convenient to be able to produce output that is almost, but not quite well-formed XML; for example, the output may include ill-formed sections that will be transformed into well-formed XML by a subsequent non-XML aware process. If a processing instruction is sent with this name, serialization should be output without any escaping.~~

~~Result DOM trees may also have PI_DISABLE_OUTPUT_ESCAPING and PI_ENABLE_OUTPUT_ESCAPING inserted into the tree.~~

~~**See Also:**~~

~~disable-output-~~escaping in XSLT Specification, Constant Field Values

---

PI_ENABLE_OUTPUT_ESCAPING

`static final java.lang.String` **`PI_ENABLE_OUTPUT_ESCAPING`**

~~The name of the processing instruction that is sent if the result tree enables output escaping at some point after having received a PI_~~DISABLE_OUTPUT_ESCAPING processing instruction.

See Also:

disable-output-escaping in XSLT Specification, Constant Field Values

Method Detail

setSystemId

```
void setSystemId(java.lang.String systemId)
```

Set the system identifier for this Result.

If the Result is not to be written to a file, the system identifier is optional. The application may still want to provide one, however, for use in error messages and warnings, or to resolve relative output identifiers.

super debuging style..

Parameters:

`systemId` - The system identifier as a URI string.

getSystemId

```
java.lang.String getSystemId()
```

~~Get the system identifier that was set with setSystemId.~~

~~**Returns:**~~

~~The system identifier that was set with setSystemId, or null if setSystemId was not called.~~

Overview Package Class Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS **FRAMES NO FRAMES All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

1 - Nov - 08

it has 4 methods

2 - getter / setter for ContentHandler
2 - getter / setter for LexicalHandler

3 - Jan - 09

nsform.sax

Class SAXResult

this is simply like
Domain object

```
java.lang.Object
  └javax.xml.transform.sax.SAXResult
```

All Implemented Interfaces:

Result

```
public class SAXResult
```

extends java.lang.Object
implements Result

Acts as an holder for a transformation Result.

Author:

Jeff Suttor

Field Summary

`static java.lang.String`	**`FEATURE`** If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Result output of this type.

Fields inherited from interface javax.xml.transform.Result
`PI_DISABLE_OUTPUT_ESCAPING`, `PI_ENABLE_OUTPUT_ESCAPING`

Constructor Summary

SAXResult() Zero-argument default constructor.
SAXResult(ContentHandler handler) Create a SAXResult that targets a SAX2 ContentHandler.

Method Summary

ContentHandler	**getHandler**() ~~Get the ContentHandler that is the Result.~~
LexicalHandler	**getLexicalHandler**() ~~Get a SAX2 LexicalHandler for the output.~~
java.lang. by inheritance String	**getSystemId**() ~~Get the system identifier that was set with setSystemId.~~
void	**setHandler**(ContentHandler handler) ~~Set the target to be a SAX2 ContentHandler.~~
void	**setLexicalHandler**(LexicalHandler handler) ~~Set the SAX2 LexicalHandler for the output.~~
void by inheritance	**setSystemId**(java.lang.String systemId) ~~Method setSystemId Set the systemID that may be used in association with the ContentHandler.~~

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Result output of this type.

See Also:

Constant Field Values

Constructor Detail

SAXResult

```
public SAXResult()
```

~~Zero-argument default constructor.~~

SAXResult

```
public SAXResult(ContentHandler handler)
```

~~Create a SAXResult that targets a SAX2 `ContentHandler`.~~

~~**Parameters:**~~

~~`handler` - Must be a non-null ContentHandler reference.~~

Method Detail

setHandler

```
public void setHandler(ContentHandler handler)
```

~~Set the target to be a SAX2 `ContentHandler`.~~

~~**Parameters:**~~

~~`handler` - Must be a non-null ContentHandler reference.~~

---

getHandler

```
public ContentHandler getHandler()
```

~~Get the ContentHandler that is the Result.~~

~~**Returns:**~~
~~The ContentHandler that is to be transformation output~~.

---

setLexicalHandler

```
public void setLexicalHandler(LexicalHandler handler)
```

Set the SAX2 `LexicalHandler` for the output.

This is needed to handle XML comments and the like. If the lexical handler is not set, an attempt should be made by the transformer to cast the `ContentHandler` to a `LexicalHandler`.

[illegible]

Parameters:
`handler` - A non-null `LexicalHandler` for handling lexical parse events.

---

getLexicalHandler

```
public LexicalHandler getLexicalHandler()
```

~~Get a SAX2 LexicalHandler for the output.~~

~~**Returns:**~~
~~A `LexicalHandler`, or null~~.

---

setSystemId

```
public void setSystemId(java.lang.String systemId)
```

~~Method setSystemId Set the systemID that may be used in association with the `ContentHandler`.~~

Specified by:

`setSystemId` in interface `Result`

Parameters:

`systemId` - The system identifier as a URI string.

---

getSystemId

```
public java.lang.String getSystemId()
```

~~Get the system identifier that was set with setSystemId.~~

~~**Specified by:**~~

~~`getSystemId` in interface `Result`~~

~~**Returns:**~~

~~The system identifier that was set with setSystemId, or null if setSystemId was not called.~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

1 - Nov - 08

3 - Jan - 09

it has 5 methods

2 - getter / setter for XMLReader
2 - getter / setter for InputSource (XML)
1 - utility method

it has 3 constructor

javax.[illegible].sax

Class SAXSource

```
java.lang.Object
  └ javax.xml.transform.sax.SAXSource
```

For normal process
just SAXSource and SAXResult with normal sax content handler is enough.

Then why SAXTransformerFactory / TransormerHandler, TemplateHanlder in this package ????

ANS :To transform SAXEvent

All Implemented Interfaces:

Source

this is simply like Domain object

`public class SAXSource`

this is another one way to transform SAX

extends java.lang.Object
implements Source

Here Not talks about XSL file.
only XML file

Acts as an holder for SAX-style Source.

Note that XSLT requires namespace support. Attempting to transform an input source that is not generated with a namespace-aware parser may result in errors. Parsers can be made namespace aware by calling the `SAXParserFactory.setNamespaceAware(boolean awareness)` method.

this is not mandatory. (i tested)

but mandatory if result is SAXResult of transform method

Version:

$Revision: 1.4 $, $Date: 2006/04/06 00:26:38 $

Author:

Jeff Suttor

Field Summary

static java.lang.String	**FEATURE** If TransformerFactory.getFeature(java.lang.String) returns true when passed this value as an argument, the Transformer supports Source input of this type.

Constructor Summary

SAXSource() Zero-argument default constructor.
SAXSource(InputSource inputSource) this is the xml file Create a SAXSource, using a SAX InputSource.
SAXSource(XMLReader reader, InputSource inputSource) Create a SAXSource, using an XMLReader and a SAX InputSource.

Method Summary

InputSource	**getInputSource**() ~~Get the SAX InputSource to be used for the Source.~~
java.lang.String by inheritance	**getSystemId**() ~~Get the base ID (URI or system ID) from where URIs will be resolved.~~
XMLReader	**getXMLReader**() ~~Get the XMLReader to be used for the Source.~~
void	**setInputSource**(InputSource inputSource) ~~Set the SAX InputSource to be used for the Source.~~
void by inheritance	**setSystemId**(java.lang.String systemId) ~~Set the system identifier for this Source.~~
void	**setXMLReader**(XMLReader reader) ~~Set the XMLReader to be used for the Source.~~
static InputSource	**sourceToInputSource**(Source source) Attempt to obtain a SAX InputSource object from a Source object.

this is utility method

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll,
toString, wait, wait, wait
```

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Source input of this type.

See Also:
Constant Field Values

Constructor Detail

SAXSource

```
public SAXSource()
```

Zero-argument default constructor. If this constructor is used, and no SAX source is set using `setInputSource(InputSource inputSource)` , then the `Transformer` will create an empty source `InputSource` using `new InputSource()`.

See Also:
`Transformer.transform(Source xmlSource, Result outputTarget)`

SAXSource

```
public SAXSource(XMLReader reader,
                 InputSource inputSource)
```

Create a `SAXSource`, using an `XMLReader` and a SAX InputSource. The `Transformer` or

SAXTransformerFactory will set itself to be the reader's ContentHandler, and then will call reader.parse(inputSource).

Parameters:

reader - An XMLReader to be used for the parse.
inputSource - A SAX input source reference that must be non-null and that will be passed to the reader parse method.

SAXSource

public **SAXSource**(InputSource inputSource)

wow

Create a SAXSource, using a SAX InputSource. The Transformer or SAXTransformerFactory creates a reader via XMLReaderFactory (if setXMLReader is not used), sets itself as the reader's ContentHandler, and calls reader.parse(inputSource).

Parameters:

inputSource - An input source reference that must be non-null and that will be passed to the parse method of the reader.

Method Detail

setXMLReader

public void **setXMLReader**(XMLReader reader)

~~Set the XMLReader to be used for the Source.~~

Parameters:

reader - A valid XMLReader or XMLFilter reference.

---

getXMLReader

public XMLReader **getXMLReader**()

~~Get the XMLReader to be used for the Source~~.

Returns:

A valid XMLReader or XMLFilter reference, or null.

---

setInputSource

```
public void setInputSource(InputSource inputSource)
```

~~Set the SAX InputSource to be used for the Source~~.

Parameters:

`inputSource` - A valid InputSource reference.

---

getInputSource

```
public InputSource getInputSource()
```

~~Get the SAX InputSource to be used for the Source.~~

~~**Returns:**~~

~~A valid InputSource reference, or null.~~

---

setSystemId

```
public void setSystemId(java.lang.String systemId)
```

Set the system identifier for this Source. If an input source has already been set, it will set the system ID or that input source, otherwise it will create a new input source.

The system identifier is optional if there is a byte stream or a character stream, but it is still useful to provide one, since the application can use it to resolve relative URIs and can include it in error messages and warnings (the parser will attempt to open a connection to the URI only if no byte stream or character stream is specified).

Specified by:

`setSystemId` in interface `Source`

Parameters:

`systemId` - The system identifier as a URI string.

---

getSystemId

```
public java.lang.String getSystemId()
```

~~Get the base ID (URI or system ID) from where URIs will be resolved.~~

~~**Specified by:**~~

~~`getSystemId` in interface `Source`~~

~~**Returns:**~~

~~Base URL for the `Source`, or `null`.~~

---

sourceToInputSource

```
public static InputSource sourceToInputSource(Source source)
```

Attempt to obtain a SAX InputSource object from a Source object.

Parameters:

`source` - Must be a non-null Source reference.

Returns:

An InputSource, or null if Source can not be converted.

can i get from DOMSource ???

i dont know...

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

3 - Jan -09

Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONS

it has 6 methods

3 - create TransformerHandler
2 - create XMLFilter
1 - create TemplatesHandler

javax.xml.transform.sax

Class SAXTransformerFactory

```
java.lang.Object
  └ javax.xml.transform.TransformerFactory
      └ javax.xml.transform.sax.SAXTransformerFactory
```

how to create Templates for given XSL file ???

ANS: use TransformerFactory

`public abstract class` **`SAXTransformerFactory`**

extends TransformerFactory

this factory is not having newInstance() methods,

so we have to cast based feature support

This class extends TransformerFactory to provide SAX-specific factory methods. It provides two types of ContentHandlers, one for creating Transformers, the other for creating Templates objects.

If an application wants to set the ErrorHandler or EntityResolver for an XMLReader used during a transformation, it should use a URIResolver to return the SAXSource which provides (with getXMLReader) a reference to the XMLReader.

Field Summary

`static java.lang.String`	**`FEATURE`** If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the TransformerFactory returned from `TransformerFactory.newInstance()` may be safely cast to a SAXTransformerFactory.
`static java.lang.String`	**`FEATURE_XMLFILTER`** If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the `newXMLFilter(Source src)` and `newXMLFilter(Templates templates)` methods are supported.

Constructor Summary

we have to call XMLReader.parser() method explictly when using SAXTransformerFactory and no need to call transform method .
But in SAXSource we never call parse method of XMLReader explictly and but have to call transform() method on transormer

protected	SAXT

Method Summary

abstract TemplatesHandler	**newTemplatesHandler**() if want parse XSL file use this Templatehandler Get a TemplatesHandler object that can process SAX ContentHandler events into a Templates object. but not transfer to Result
abstract TransformerHandler	**newTransformerHandler**() this is xml file Get a TransformerHandler object that can process SAX ContentHandler events into a Result.
abstract TransformerHandler	**newTransformerHandler**(Source src) XSL file Get a TransformerHandler object that can process SAX ContentHandler events into a Result, based on the transformation instructions specified by the argument.
abstract TransformerHandler	**newTransformerHandler**(Templates templates) memory rep. of XSL file Get a TransformerHandler object that can process SAX ContentHandler events into a Result, based on the Templates argument.
abstract XMLFilter	**newXMLFilter**(Source src) XSL file Create an XMLFilter that uses the given Source as the transformation instructions.
abstract XMLFilter	**newXMLFilter**(Templates templates) memory rep. of XSL file Create an XMLFilter, based on the Templates argument..

Methods inherited from class javax.xml.transform.TransformerFactory
getAssociatedStylesheet, getAttribute, getErrorListener, getFeature, getURIResolver, newInstance, newInstance, newTemplates, newTransformer, newTransformer, setAttribute, setErrorListener, setFeature, setURIResolver

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If TransformerFactory.getFeature(java.lang.String) returns true when passed this value as an argument, the TransformerFactory returned from TransformerFactory.newInstance() may be safely cast to a SAXTransformerFactory.

See Also:
Constant Field Values

FEATURE_XMLFILTER

```
public static final java.lang.String FEATURE_XMLFILTER
```

If TransformerFactory.getFeature(java.lang.String) returns true when passed this value as an argument, the newXMLFilter(Source src) and newXMLFilter(Templates templates) methods are supported.

See Also:
Constant Field Values

Constructor Detail

SAXTransformerFactory

```
protected SAXTransformerFactory()
```

The default constructor is protected on purpose.

Method Detail

newTransformerHandler

```
public abstract TransformerHandler newTransformerHandler(Source src)
                                                  throws
TransformerConfigurationException
```

XSL file

Get a TransformerHandler object that can process SAX ContentHandler events into a Result, based on the transformation instructions specified by the argument.

Parameters:

`src` - The Source of the transformation instructions. XSL file

Returns:

TransformerHandler ready to transform SAX events.

Throws:

`TransformerConfigurationException` - If for some reason the TransformerHandler can not be created.

---

newTransformerHandler

super method. use this method.

```
public abstract TransformerHandler newTransformerHandler(Templates templates)
                                                  throws TransformerConfigurationException
```

Get a TransformerHandler object that can process SAX ContentHandler events into a Result, based on the Templates argument.

Parameters:

`templates` - The compiled transformation instructions.

Returns:

TransformerHandler ready to transform SAX events.

Throws:

`TransformerConfigurationException` - If for some reason the TransformerHandler can not be created.

---

newTransformerHandler

```
public abstract TransformerHandler newTransformerHandler()
                                                  throws TransformerConfigurationException
```

Get a TransformerHandler object that can process SAX ContentHandler events into a Result. The transformation is defined as an identity (or copy) transformation, for example to copy a series of SAX parse events into a DOM tree.

Returns:

A non-null reference to a TransformerHandler, that may be used as a ContentHandler for SAX parse events.

Throws:

`TransformerConfigurationException` - If for some reason the TransformerHandler cannot be created.

---

newTemplatesHandler

```
public abstract TemplatesHandler newTemplatesHandler()
                                          throws
TransformerConfigurationException
```

to create **Templates** object from SAX events..

Get a TemplatesHandler object that can process SAX ContentHandler events into a Templates object.

Returns:

A non-null reference to a TransformerHandler, that may be used as a ContentHandler for SAX parse events.

Throws:

TransformerConfigurationException - If for some reason the TemplatesHandler cannot be created.

---

newXMLFilter

XSL file

```
public abstract XMLFilter newXMLFilter(Source src)
                                throws TransformerConfigurationException
```

Create an XMLFilter that uses the given Source as the transformation instructions.

Parameters:

src - The Source of the transformation instructions.

Returns:

An XMLFilter object, or null if this feature is not supported.

Throws:

TransformerConfigurationException - If for some reason the TemplatesHandler cannot be created.

---

newXMLFilter

```
public abstract XMLFilter newXMLFilter(Templates templates)
                                throws TransformerConfigurationException
```

Create an XMLFilter, based on the Templates argument..

Parameters:

`templates` - The compiled transformation instructions.

Returns:

An XMLFilter object, or null if this feature is not supported.

Throws:

`TransformerConfigurationException` - If for some reason the TemplatesHandler cannot be created.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

3 - Jan - 09

javax.xml.transform

only it has 2 methods

Interface Source

All Known Implementing Classes:

DOMSource, SAXSource, StAXSource, StreamSource

```
public interface Source
```

An object that implements this interface contains the information needed to act as source input (XML source or transformation instructions).

Method Summary

java.lang.String	getSystemId() ~~Get the system identifier that was set with setSystemId~~.
void	setSystemId(java.lang.String systemId) ~~Set the system identifier for this Source~~.

Method Detail

setSystemId

```
void setSystemId(java.lang.String systemId)
```

Set the system identifier for this Source.

The system identifier is optional if the source does not get its data from a URL, but it may still be useful to provide one. The application can use a system identifier, for example, to resolve relative URIs and to include in error messages and warnings. ✓ Ok

Parameters:

`systemId` - The system identifier as a URL string.

---

getSystemId

```
java.lang.String getSystemId()
```

~~Get the system identifier that was set with setSystemId.~~

~~**Returns:**~~

~~The system identifier that was set with setSystemId, or null if setSystemId was not called.~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

---

3 - Jan - 09

javax.xml.transform

Interface SourceLocator

it has only 4 methods

All Known Subinterfaces:

DOMLocator

you can access this object ONLY when ERROR occurred.

that is any one method of ErrorListener ONLY. NOT even exception at transform() method calling . (i tested.)

```
public interface SourceLocator
```

This interface is primarily for the purposes of reporting where an error occurred in the XML source or transformation instructions.

error in Both file......

Method Summary

`int`	`getColumnNumber()` ~~Return the character position where the current document event ends.~~
`int`	`getLineNumber()` ~~Return the line number where the current document event ends.~~
`java.lang.String`	`getPublicId()` ~~Return the public identifier for the current document event.~~
`java.lang.String`	`getSystemId()` ~~Return the system identifier for the current document event.~~

Method Detail

getPublicId

java.lang.String **getPublicId**()

URL path

~~Return the public identifier for the current document event.~~

~~The return value is the public identifier of the document entity or of the external parsed entity in which the markup that triggered the event appears.~~

~~**Returns:**~~

~~A string containing the public identifier, or null if none is available.~~

~~**See Also:**~~

~~getSystemId()~~

---

getSystemId

java.lang.String **getSystemId**()

hard disk path

~~Return the system identifier for the current document event.~~

~~The return value is the system identifier of the document entity or of the external parsed entity in which the markup that triggered the event appears.~~

~~If the system identifier is a URL, the parser must resolve it fully before passing it to the application.~~

~~**Returns:**~~

~~A string containing the system identifier, or null if none is available.~~

~~**See Also:**~~

~~getPublicId()~~

---

getLineNumber

int **getLineNumber**()

~~Return the line number where the current document event ends.~~

~~**Warning:** The return value from the method is intended only as an approximation for the sake of~~

~~error reporting; it is not intended to provide sufficient information to edit the character content of the original XML document.~~

~~The return value is an approximation of the line number in the document entity or external parsed entity where the markup that triggered the event appears.~~

~~**Returns:**~~
~~The line number, or -1 if none is available.~~
~~**See Also:**~~
~~`getColumnNumber()`~~

---

getColumnNumber

`int` **`getColumnNumber`**`()`

~~Return the character position where the current document event ends.~~

~~**Warning:** The return value from the method is intended only as an approximation for the sake of error reporting; it is not intended to provide sufficient information to edit the character content of the original XML document.~~

~~The return value is an approximation of the column number in the document entity or external parsed entity where the markup that triggered the event appears.~~

~~**Returns:**~~
~~The column number, or -1 if none is available.~~
~~**See Also:**~~
~~`getLineNumber()`~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

As of now i am **dont** have xslt engine has been implemented this feature

```
logger.info(" StAXResult "+transformerFactory.getFeature(StAXResult.FEATURE));
logger.info(" StAXSource "+transformerFactory.getFeature(StAXSource.FEATURE));
above two statements are return false
```

4 - Jan - 09

1 - Nov - 08

2 - methods

1 - getter StreamWriter
1 - getter EventWriter

javax.xml.transform.stax

Class StAXResult

```
java.lang.Object
  └ javax.xml.transform.stax.StAXResult
```

All Implemented Interfaces:

Result

`public class` **`StAXResult`**

extends java.lang.Object
implements Result

Acts as a holder for an XML `Result` in the form of a StAX writer,i.e. `XMLStreamWriter` or `XMLEventWriter`. `StAXResult` can be used in all cases that accept a `Result`, e.g. `Transformer`, `Validator` which accept `Result` as input.

Since:
1.6

Version:
$Revision: 1.5 $, $Date: 2006/06/28 15:00:35 $

Author:
Neeraj Bajaj, Jeff Suttor

See Also:
JSR 173: Streaming API for XML, `XMLStreamWriter`, `XMLEventWriter`

Field Summary

static java.lang.String	**FEATURE** If TransformerFactory.getFeature(String name) returns true when passed this value as an argument, the Transformer supports Result output of this type.

Fields inherited from interface javax.xml.transform.Result
PI_DISABLE_OUTPUT_ESCAPING, PI_ENABLE_OUTPUT_ESCAPING

Constructor Summary

StAXResult(XMLEventWriter xmlEventWriter) Creates a new instance of a StAXResult by supplying an XMLEventWriter.
StAXResult(XMLStreamWriter xmlStreamWriter) Creates a new instance of a StAXResult by supplying an XMLStreamWriter.

Method Summary

java.lang.String	**getSystemId**() by inheritance The returned system identifier is always null.
XMLEventWriter	**getXMLEventWriter**() Get the XMLEventWriter used by this StAXResult.
XMLStreamWriter	**getXMLStreamWriter**() Get the XMLStreamWriter used by this StAXResult.
void	**setSystemId**(java.lang.String systemId) In the context of a StAXResult, it is not appropriate to explicitly set the system identifier. by inheritance

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(String name)` returns true when passed this value as an argument, the Transformer supports Result output of this type.

See Also:

Constant Field Values

Constructor Detail

StAXResult

```
public StAXResult(XMLEventWriter xmlEventWriter)
```

Creates a new instance of a `StAXResult` by supplying an `XMLEventWriter`.

`XMLEventWriter` must be a non-`null` reference.

Parameters:

`xmlEventWriter` - `XMLEventWriter` used to create this `StAXResult`.

Throws:

`java.lang.IllegalArgumentException` - If `xmlEventWriter` == `null`.

StAXResult

```
public StAXResult(XMLStreamWriter xmlStreamWriter)
```

Creates a new instance of a `StAXResult` by supplying an `XMLStreamWriter`.

`XMLStreamWriter` must be a non-`null` reference.

Parameters:

`xmlStreamWriter` - `XMLStreamWriter` used to create this `StAXResult`.

Throws:

`java.lang.IllegalArgumentException` - If `xmlStreamWriter` == `null`.

Method Detail

getXMLEventWriter

```
public XMLEventWriter getXMLEventWriter()
```

~~Get the `XMLEventWriter` used by this `StAXResult`.~~

~~`XMLEventWriter` will be `null` if this `StAXResult` was created with a `XMLStreamWriter`.~~

~~**Returns:**~~

~~`XMLEventWriter` used by this `StAXResult`.~~

---

getXMLStreamWriter

```
public XMLStreamWriter getXMLStreamWriter()
```

~~Get the `XMLStreamWriter` used by this `StAXResult`.~~

~~`XMLStreamWriter` will be `null` if this `StAXResult` was created with a `XMLEventWriter`.~~

~~**Returns:**~~

~~`XMLStreamWriter` used by this `StAXResult`~~.

~~setSystemId~~

~~`public void setSystemId(java.lang.String systemId)`~~

~~In the context of a `StAXResult`, it is not appropriate to explicitly set the system identifier. The~~

~~`XMLEventWriter` or `XMLStreamWriter` used to construct this `StAXResult` determines the system identifier of the XML result.~~

~~An `UnsupportedOperationException` is **always** thrown by this method.~~

~~**Specified by:**~~

~~`setSystemId` in interface `Result`~~

~~**Parameters:**~~

~~`systemId` - Ignored.~~

~~**Throws:**~~

~~`java.lang.UnsupportedOperationException` - Is **always** thrown by this method.~~

---

getSystemId

```
public java.lang.String getSystemId()
```

~~The returned system identifier is always `null`.~~

~~**Specified by:**~~

~~`getSystemId` in interface `Result`~~

~~**Returns:**~~

~~The returned system identifier is always `null`.~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

As of now i am **dont** have xslt engine has been implemented this feature

```
logger.info(" StAXResult "+transformerFactory.getFeature(StAXResult.FEATURE));
logger.info(" StAXSource "+transformerFactory.getFeature(StAXSource.FEATURE));
above two statements are return false
```

1 - Nov - 08

javax.x 4 - Jan - 09 .stax

Class StAXSource

2 - methods

1 - getter for StreamReader
1 - getter for EventReader

```
java.lang.Object
  └ javax.xml.transform.stax.StAXSource
```

All Implemented Interfaces:

Source

`public class` **`StAXSource`**

extends java.lang.Object
implements Source

Acts as a holder for an XML `Source` in the form of a StAX reader,i.e. `XMLStreamReader` or `XMLEventReader`. `StAXSource` can be used in all cases that accept a `Source`, e.g. `Transformer`, `Validator` which accept `Source` as input.

`StAXSource`s are consumed during processing and are not reusable.

Since:
1.6

Version:
$Revision: 1.9 $, $Date: 2007/04/13 20:22:33 $

Author:
Neeraj Bajaj, Jeff Suttor

See Also:
JSR 173: Streaming API for XML, `XMLStreamReader`, `XMLEventReader`

Field Summary	
static java.lang.String	**FEATURE** If TransformerFactory.getFeature(String name) returns true when passed this value as an argument, the Transformer supports Source input of this type.

Constructor Summary
StAXSource(XMLEventReader xmlEventReader) Creates a new instance of a StAXSource by supplying an XMLEventReader.
StAXSource(XMLStreamReader xmlStreamReader) Creates a new instance of a StAXSource by supplying an XMLStreamReader.

Method Summary	
java.lang.String	**getSystemId**() by inheritance Get the system identifier used by this StAXSource.
XMLEventReader	**getXMLEventReader**() Get the XMLEventReader used by this StAXSource.
XMLStreamReader	**getXMLStreamReader**() Get the XMLStreamReader used by this StAXSource.
void	**setSystemId**(java.lang.String systemId) In the context of a StAXSou by inheritance ppropriate to explicitly set the system identifier.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(String name)` returns true when passed this value as an argument, the Transformer supports Source input of this type.

See Also:

Constant Field Values

Constructor Detail

StAXSource

```
public StAXSource(XMLEventReader xmlEventReader)
           throws XMLStreamException
```

Creates a new instance of a `StAXSource` by supplying an `XMLEventReader`.

`XMLEventReader` must be a non-`null` reference.

`XMLEventReader` must be in `XMLStreamConstants.START_DOCUMENT` or `XMLStreamConstants.START_ELEMENT` state.

Parameters:

`xmlEventReader` - `XMLEventReader` used to create this `StAXSource`.

Throws:

`XMLStreamException` - If `xmlEventReader` access throws an `Exception`.

`java.lang.IllegalArgumentException` - If `xmlEventReader` == `null`.

`java.lang.IllegalStateException` - If `xmlEventReader` is not in `XMLStreamConstants.START_DOCUMENT` or `XMLStreamConstants.START_ELEMENT` state.

StAXSource

```
public StAXSource(XMLStreamReader xmlStreamReader)
```

Creates a new instance of a `StAXSource` by supplying an [`XMLStreamReader`].

`XMLStreamReader` must be a non-`null` reference.

`XMLStreamReader` must be in [`XMLStreamConstants.START_DOCUMENT`] or [`XMLStreamConstants.START_ELEMENT`] state.

Parameters:

`xmlStreamReader` - `XMLStreamReader` used to create this `StAXSource`.

Throws:

`java.lang.IllegalArgumentException` - If `xmlStreamReader` == `null`.
`java.lang.IllegalStateException` - If `xmlStreamReader` is not in `XMLStreamConstants.START_DOCUMENT` or `XMLStreamConstants.START_ELEMENT` state.

Method Detail

getXMLEventReader

`public` [`XMLEventReader`] **`getXMLEventReader`**`()`

~~Get the `XMLEventReader` used by this `StAXSource`.~~

~~`XMLEventReader` will be `null`. if this `StAXSource` was created with a `XMLStreamReader`.~~

~~**Returns:**~~

~~`XMLEventReader` used by this `StAXSource`.~~

---

getXMLStreamReader

`public` [`XMLStreamReader`] **`getXMLStreamReader`**`()`

~~Get the `XMLStreamReader` used by this `StAXSource`.~~

~~`XMLStreamReader` will be `null` if this `StAXSource` was created with a `XMLEventReader`.~~

~~**Returns:**~~

~~`XMLStreamReader` used by this `StAXSource`.~~

---

setSystemId

```
public void setSystemId(java.lang.String systemId)
```

~~In the context of a `StAXSource`, it is not appropriate to explicitly set the system identifier. The `XMLStreamReader` or `XMLEventReader` used to construct this `StAXSource` determines the system identifier of the XML source.~~

~~An `UnsupportedOperationException` is **always** thrown by this method.~~

~~**Specified by:**~~

`setSystemId` ~~in interface~~ `Source`

~~**Parameters:**~~

~~`systemId` - Ignored.~~

~~**Throws:**~~

~~`java.lang.UnsupportedOperationException` - Is **always** thrown by this~~ method.

---

getSystemId

```
public java.lang.String getSystemId()
```

~~Get the system identifier used by this `StAXSource`.~~

~~The `XMLStreamReader` or `XMLEventReader` used to construct this `StAXSource` is queried to determine the system identifier of the XML source.~~

~~The system identifier may be `null` or an empty `""` `String`.~~

~~Specified by:~~

getSystemId ~~in interface~~ Source

~~Returns:~~

~~System identifier used by this `StAXSource`.~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

4 - Jan -09

it has only 4 methods

2 - getter / setter for Writer
2 - getter / setter for outputstream
it has 5 constructor

javax.xml.transform.stream

Class StreamResult

```
java.lang.Object
  └ javax.xml.transform.stream.StreamResult
```

All Implemented Interfaces:

Result

`public class` **`StreamResult`**

extends java.lang.Object
implements Result

actually this one have to use if output be html /text / image

Acts as an holder for a transformation result, which may be XML, plain Text, HTML, or some other form of markup.

Author:

Jeff Suttor

Field Summary

`static java.lang.String`	**`FEATURE`** If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Result output of this type.

Fields inherited from interface javax.xml.transform.Result
`PI_DISABLE_OUTPUT_ESCAPING`, `PI_ENABLE_OUTPUT_ESCAPING`

Constructor Summary

StreamResult() Zero-argument default constructor.
StreamResult(java.io.File f) ✓ 1 Construct a StreamResult from a File.
StreamResult(java.io.OutputStream outputStream) ✓ 2 Construct a StreamResult from a byte stream.
StreamResult(java.lang.String systemId) ✓ 3 Construct a StreamResult from a URL.
StreamResult(java.io.Writer writer) ✓ 4 Construct a StreamResult from a character stream.

Method Summary

java.io.OutputStream	**getOutputStream**() 1 Get the byte stream that was set with setOutputStream.
java.lang.String	**getSystemId**() by inheritance Get the system identifier that was set with setSystemId.
java.io.Writer	**getWriter**() 3 Get the character stream that was set with setWriter.
void	**setOutputStream**(java.io.OutputStream outputStream) 4 Set the ByteStream that is to be written to.
void	**setSystemId**(java.io.File f) where is the getter method for "File" Set the system ID from a File reference.
void	**setSystemId**(java.lang.String systemId) Set the systemID that may be used in assoc by inheritance [illegible] byte or character stream, or, if neither is set, use this value as a writeable URI (probably a file name).
void	**setWriter**(java.io.Writer writer) Set the writer that is to receive the result.

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll,
toString, wait, wait, wait
```

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Result output of this type.

See Also:

Constant Field Values

Constructor Detail

StreamResult

```
public StreamResult()
```

Zero-argument default constructor.

StreamResult

```
public StreamResult(java.io.OutputStream outputStream)
```

Construct a StreamResult from a byte stream. Normally, a stream should be used rather than a reader, so that the transformer may use instructions contained in the transformation instructions to control the encoding.

Parameters:

`outputStream` - A valid OutputStream reference.

StreamResult

```
public StreamResult(java.io.Writer writer)
```

Construct a StreamResult from a character stream. Normally, a stream should be used rather than a reader, so that the transformer may use instructions contained in the transformation instructions to control the encoding. However, there are times when it is useful to write to a character stream, such as when using a StringWriter.

Parameters:
`writer` - A valid Writer reference.

StreamResult

```
public StreamResult(java.lang.String systemId)
```

Construct a StreamResult from a URL.

Parameters:
`systemId` - Must be a String that conforms to the URI syntax.

StreamResult

```
public StreamResult(java.io.File f)
```

Construct a StreamResult from a File.

Parameters:
`f` - Must a non-null File reference.

Method Detail

setOutputStream

```
public void setOutputStream(java.io.OutputStream outputStream)
```

Set the ByteStream that is to be written to. Normally, a stream should be used rather than a reader, so that the transformer may use instructions contained in the transformation instructions to control the encoding.

Parameters:

`outputStream` - A valid OutputStream reference.

---

getOutputStream

```
public java.io.OutputStream getOutputStream()
```

Get the byte stream that was set with setOutputStream.

Returns:

The byte stream that was set with setOutputStream, or null if setOutputStream or the ByteStream constructor was not called.

---

setWriter

```
public void setWriter(java.io.Writer writer)
```

Set the writer that is to receive the result. Normally, a stream should be used rather than a writer, so that the transformer may use instructions contained in the transformation instructions to control the encoding. However, there are times when it is useful to write to a writer, such as when using a StringWriter.

Parameters:

`writer` - A valid Writer reference.

---

getWriter

```
public java.io.Writer getWriter()
```

Get the character stream that was set with setWriter.

Returns:

The character stream that was set with setWriter, or null if setWriter or the Writer constructor was not called.

---

setSystemId

```
public void setSystemId(java.lang.String systemId)
```

Set the systemID that may be used in association with the byte or character stream, or, if neither is set, use this value as a writeable URI (probably a file name).

Specified by:

`setSystemId` in interface `Result`

Parameters:

`systemId` - The system identifier as a URI string.

---

setSystemId

```
public void setSystemId(java.io.File f)
```

Set the system ID from a `File` reference.

Parameters:

`f` - Must a non-null File reference.

---

getSystemId

```
public java.lang.String getSystemId()
```

Get the system identifier that was set with setSystemId.

Specified by:

`getSystemId` in interface `Result`

Returns:

The system identifier that was set with setSystemId, or null if setSystemId was not called.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

4 - Jan -09

it has 7 methods

2 - setter / getter for InputStream
2 - getter / setter for Reader
3 - getter / setter for public id

javax.xml.transform.stream

Class StreamSource

```
java.lang.Object
  └ javax.xml.transform.stream.StreamSource
```

All Implemented Interfaces:

Source

`public class` **`StreamSource`**

extends java.lang.Object
implements Source

Acts as an holder for a transformation Source in the form of a stream of XML markup.

Note: Due to their internal use of either a `Reader` or `InputStream` instance, `StreamSource` instances may only be used once. oK

Version:
$Revision: 1.5 $, $Date: 2005/11/21 05:57:19 $

Author:
Jeff Suttor

Field Summary

static java.lang.String	**FEATURE** If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Source input of this type.

Constructor Summary
StreamSource() 1 ~~Zero-argument default constructor~~.
StreamSource(java.io.File f) 2 ~~Construct a StreamSource from a File.~~
StreamSource(java.io.InputStream inputStream) 3 ~~Construct a StreamSource from a byte stream~~.
StreamSource(java.io.InputStream inputStream, java.lang.String systemId) ~~Construct a StreamSource from a byte stream~~.
StreamSource(java.io.Reader reader) 4 ~~Construct a StreamSource from a character reader~~.
StreamSource(java.io.Reader reader, java.lang.String systemId) ~~Construct a StreamSource from a character reader~~.
StreamSource(java.lang.String systemId) 5 ~~Construct a StreamSource from a URL~~.

Method Summary	
java.io.InputStream	getInputStream() ~~Get the byte stream that was set with setByteStream~~.
java.lang.String	getPublicId() ~~Get the public identifier that was set with setPublicId.~~
java.io.Reader	getReader() ~~Get the character stream that was set with setReader~~.
java.lang.String	getSystemId() by inheritance Get the system identifier that was set with setSystemId.
void	setInputStream(java.io.InputStream inputStream) ~~Set the byte stream to be used as input~~.
void	setPublicId(java.lang.String publicId) ~~Set the public identifier for this Source.~~

`void`	**setReader**(java.io.Reader reader) ~~Set the input to be a character reader.~~
`void`	**setSystemId**(java.io.File f) ← useful for SourceLocator ~~Set the system ID from a File reference.~~
`void`	**setSystemId**(java.lang.String systemId) Set the system identifier for this Source. by inheritance

Methods inherited from class java.lang.Object
`clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait`

Field Detail

FEATURE

```
public static final java.lang.String FEATURE
```

If `TransformerFactory.getFeature(java.lang.String)` returns true when passed this value as an argument, the Transformer supports Source input of this type.

See Also:

Constant Field Values

Constructor Detail

StreamSource

```
public StreamSource()
```

Zero-argument default constructor. If this constructor is used, and no Stream source is set using `setInputStream(java.io.InputStream inputStream)` or `setReader(java.io.Reader reader)`, then the `Transformer` will create an empty source `InputStream` using `new InputStream()`.

See Also:

`Transformer.transform(Source xmlSource, Result outputTarget)`

StreamSource

```
public StreamSource(java.io.InputStream inputStream)
```

Construct a StreamSource from a byte stream. Normally, a stream should be used rather than a reader, so the XML parser can resolve character encoding specified by the XML declaration.

If this constructor is used to process a stylesheet, normally setSystemId should also be called, so that relative URI references can be resolved.

Parameters:

`inputStream` - A valid InputStream reference to an XML stream.

StreamSource

this is path 0f XSL file

```
public StreamSource(java.io.InputStream inputStream,
                    java.lang.String systemId)
```

Construct a StreamSource from a byte stream. Normally, a stream should be used rather than a reader, so that the XML parser can resolve character encoding specified by the XML declaration.

This constructor allows the systemID to be set in addition to the input stream, which allows relative URIs to be processed.

Parameters:

`inputStream` - A valid InputStream reference to an XML stream.

`systemId` - Must be a String that conforms to the URI syntax.

StreamSource

```
public StreamSource(java.io.Reader reader)
```

Construct a StreamSource from a character reader. Normally, a stream should be used rather than a reader, so that the XML parser can resolve character encoding specified by the XML declaration. However, in many cases the encoding of the input stream is already resolved, as in the case of reading XML from a StringReader.

Slow

Parameters:

`reader` - A valid Reader reference to an XML character stream.

StreamSource

```
public StreamSource(java.io.Reader reader,
                    java.lang.String systemId)
```

Construct a StreamSource from a character reader. Normally, a stream should be used rather than a reader, so that the XML parser may resolve character encoding specified by the XML declaration. However, in many cases the encoding of the input stream is already resolved, as in the case of reading XML from a StringReader.

Parameters:

`reader` - A valid Reader reference to an XML character stream.

`systemId` - Must be a String that conforms to the URI syntax.

StreamSource

```
public StreamSource(java.lang.String systemId)
```

Construct a StreamSource from a URL.

must call this method if the actual source is XSL rather than XML

Parameters:

`systemId` - Must be a String that conforms to the URI syntax.

StreamSource

```
public StreamSource(java.io.File f)
```

Construct a StreamSource from a File.

Parameters:

`f` - Must a non-null File reference.

Method Detail

setInputStream

```
public void setInputStream(java.io.InputStream inputStream)
```

~~Set the byte stream to be used as input. Normally, a stream should be used rather than a reader, so that the XML parser can resolve character encoding specified by the XML declaration.~~

If this Source object is used to process a stylesheet, normally setSystemId should also be called, so that relative URL references can be resolved.

Parameters:

`inputStream` - A valid InputStream reference to an XML stream.

---

getInputStream

```
public java.io.InputStream getInputStream()
```

~~Get the byte stream that was set with setByteStream.~~

~~**Returns:**~~

~~The byte stream that was set with setByteStream, or null if setByteStream or the ByteStream constructor was not called.~~

---

setReader

```
public void setReader(java.io.Reader reader)
```

~~Set the input to be a character reader. Normally, a stream should be used rather than a reader, so that the XML parser can resolve character encoding specified by the XML declaration. However, in many cases the encoding of the input stream is already resolved, as in the case of reading XML from a StringReader.~~

~~**Parameters:**~~

~~`reader` - A valid Reader reference to an XML CharacterStream~~.

---

getReader

```
public java.io.Reader getReader()
```

~~Get the character stream that was set with setReader.~~

~~**Returns:**~~

~~The character stream that was set with setReader, or null if setReader or the Reader constructor was not called.~~

---

setPublicId

```
public void setPublicId(java.lang.String publicId)
```

~~Set the public identifier for this Source.~~

~~The public identifier is always optional: if the application writer includes one, it will be provided as part of the location information.~~

~~**Parameters:**~~

~~`publicId` - The public identifier as a string.~~

---

getPublicId

~~`public java.lang.String getPublicId()`~~

~~Get the public identifier that was set with setPublicId.~~

~~**Returns:**~~

~~The public identifier that was set with setPublicId, or null if setPublicId was not called.~~

setSystemId

```
public void setSystemId(java.lang.S                  d)
```

it means file path can get from stream or reader object itself

Set the system identifier for this Source.

The system identifier is optional if there is a byte stream or a character stream, but it is still useful to provide one, since the application can use it to resolve relative URIs and can include it in error messages and warnings (the parser will attempt to open a connection to the URI only if there is no byte stream or character stream specified).

Specified by:

`setSystemId` in interface `Source`

Parameters:

`systemId` - The system identifier as a URL string.

getSystemId

```
public java.lang.String getSystemId()
```

~~Get the system identifier that was set with setSystemId.~~

~~**Specified by:**~~

~~`getSystemId` in interface `Source`~~

~~**Returns:**~~

~~The system identifier that was set with setSystemId, or null if setSystemId was not called.~~

setSystemId

```
public void setSystemId(java.io.File f)
```

~~Set the system ID from a File reference.~~

~~**Parameters:**~~

~~`f` - Must a non-null File reference.~~

Overview Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

it has only 2 methods

the implementation of this template is optional

javax.xml.transform

Interface Templates

when the same XSL file used for many time, it can be saved into templates. it is like load only one time and use many time. Then create transformer from this templates and use. it is threadsafe. This one of best practice

To reuse a single `Template` instance in multiple concurrent threads, multiple `Transformer` instances would have to be created via the `Templates.newTransformer()` factory method

pu es

An object that implements this interface is the runtime representation of processed transformation instructions.

that is XSL file

Templates must be threadsafe for a given instance over multiple threads running concurrently, and may be used multiple times in a given session.

returned property object will have entries those are declared in given XSL file's xsl: output tag (So, only existing output properties in XSL file)

Method Summary

`java.util. Properties`	**`getOutputProperties`**`()` Get the properties corresponding to the effective xsl:output element.
`Transformer`	**`newTransformer`**`()` Create a new transformation context for this Templates object.

Method Detail

newTransformer

```
Transformer newTransformer()
                           throws TransformerConfigurationException
```

Create a new transformation context for this Templates object.

Returns:

A valid non-null instance of a Transformer.

Throws:

TransformerConfigurationException - if a Transformer can not be created.

getOutputProperties

i have skipped. Gets hands on at the time usage (3 - jan -09)

```
java.util.Properties getOutputProperties()
```

Get the properties corresponding to the effective xsl:output element. The object returned will be a clone of the internal values. Accordingly, it can be mutated without mutating the Templates object, and then handed in to `Transformer.setOutputProperties(java.util.Properties)`.

The properties returned should contain properties set by the stylesheet, and these properties are "defaulted" by default properties specified by section 16 of the XSL Transformations (XSLT) W3C Recommendation. The properties that were specifically set by the stylesheet should be in the base Properties list, while the XSLT default properties that were not specifically set should be in the "default" Properties list. Thus, getOutputProperties().getProperty(String key) will obtain any property in that was set by the stylesheet, *or* the default properties, while getOutputProperties ().get(String key) will only retrieve properties that were explicitly set in the stylesheet.

For XSLT, Attribute Value Templates attribute values will be returned unexpanded (since there is no context at this point). The namespace prefixes inside Attribute Value Templates will be unexpanded, so that they remain valid XPath values.

Returns:

A Properties object, never null.

we have to call XMLReader.parser() method explictly when using SAXTransformerFactory and no need to call transform method . But in SAXSource we never call parse method of XMLReader explictly and but have to call transform() method on transormer

javax.xml.transform.sax

Interface TemplatesHandler

it has 3 methods

2 - getter /setter for System id
1 - getter for Templates

All Superinterfaces:

ContentHandler

Parsing an XSL file by SAX API and create Templates from that.

Actually this idea no need. already super methods are available in TransformerFactory

```
public interface TemplatesHa
```

so, this is one of way to load XSL file and represent as Templates object . But Dont use this API

extends ContentHandler

it used to parse an XSL file not xml file using SAX

A SAX ContentHandler that may be used to process SAX parse events (parsing transformation instructions) into a Templates object.

Note that TemplatesHandler does not need to implement LexicalHandler.

Method Summary

`java.lang.String`	**`getSystemId`**`()` ~~Get the base ID (URI or system ID) from where relative URLs will be resolved.~~
`Templates`	**`getTemplates`**`()` When a TemplatesHandler object is used as a ContentHandler for the parsing of transformation instructions, it creates a Templates object, which the caller can get once the SAX events have been completed.
`void`	**`setSystemId`**`(java.lang.String systemID)` ~~Set the base ID (URI or system ID) for the Templates object created by this builder.~~

Methods inherited from interface org.xml.sax.ContentHandler

`characters, endDocument, endElement, endPrefixMapping, ignorableWhitespace, processingInstruction, setDocumentLocator, skippedEntity, startDocument, startElement, startPrefixMapping`

Method Detail

getTemplates

call this method, after paring of XSL file over.

```
Templates getTemplates()
```

When a TemplatesHandler object is used as a ContentHandler for the parsing of transformation instructions, it creates a Templates object, which the caller can get once the SAX events have been completed.

Returns:

The Templates object that was created during the SAX event process, or null if no Templates object has been created.

---

setSystemId

```
void setSystemId(java.lang.String systemID)
```

Set the base ID (URI or system ID) for the Templates object created by this builder. This must be set in order to resolve relative URIs in the stylesheet. This must be called before the startDocument event.

ok. careful param.

Parameters:

`systemID` - Base URI for this stylesheet.

---

getSystemId

```
java.lang.String getSystemId()
```

Get the base ID (URI or system ID) from where relative URLs will be resolved.

Returns:

The systemID that was set with `setSystemId(java.lang.String)`.

Overview Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

3 - Jan - 09

it has 13 methods

4 - getter/setter for output properties
3 - getter/setter/clear template parameter
2 - setter/getter URIResolver
2 - setter/getter ErrorListener
2 - Transformer related (reset, transform)

javax.xml.transform

Class Transformer

```
java.lang.Object
  └javax.xml.transform.Transformer
```

i have skipped Parameter <xsl:param> related methods.

learn at the time usage (3 - jan - 08)

```
public abstract class Transformer
```

extends java.lang.Object

~~An instance of this abstract class can transform a source tree into a result tree.~~

~~An instance of this class can be obtained with the~~ `TransformerFactory.newTransformer` ~~method. This instance may then be used to process XML from a variety of sources and write the transformation output to a variety of sinks.~~

An object of this class may not be used in multiple threads running concurrently. Different Transformers may be used concurrently by different threads.

A `Transformer` may be used multiple times. Parameters and output properties are preserved across transformations.

Version:
$Revision: 1.4 $, $Date: 2005/11/03 19:34:22 $

Author:
Jeff Suttor

Constructor Summary

protected	**Transformer**() Default constructor is protected on purpose.

Method Summary

abstract void	**clearParameters**() deals with <xsl:param> ~~Clear all parameters set with setParameter.~~
abstract ErrorListener	**getErrorListener**() ~~Get the error event handler in effect for the transformation.~~

so,changing value of property in this returned Properties object, wont affect actual Transformation process ??????

deals with <xsl:output /> abstract java.util.Properties	**getOutputProperties**() ~~Get a copy of the output properties for the transformation.~~
deals with <xsl:output /> abstract java.lang.String	**getOutputProperty**(java.lang.String name) ~~Get an output property that is in effect for the transformer.~~
abstract java.lang.Object	**getParameter**(java.lang.String name) deals with <xsl:param> ~~Get a parameter that was explicitly set with setParameter.~~
abstract URIResolver	**getURIResolver**() ~~Get an object that will be used to resolve URIs used in document().~~
void	**reset**() ~~Reset this Transformer to its original configuration.~~
abstract void	**setErrorListener**(ErrorListener listener) ~~Set the error event listener in effect for the transformation.~~
abstract void deals with <xsl:output />	**setOutputProperties**(java.util.Properties oformat) ~~Set the output properties for the transformation.~~
abstract void deals with <xsl:output />	**setOutputProperty**(java.lang.String name, java.lang.String value) ~~Set an output property that will be in effect for the transformation.~~
abstract void	**setParameter**(java.lang.String name, java.lang.Object value) ~~Add a parameter for the transformation.~~ deals with <xsl:param>
abstract void	**setURIResolver**(URIResolver resolver) ~~Set an object that will be used to resolve URIs used in document().~~
abstract void	**transform**(Source xmlSource, Result outputTarget) ~~Transform the XML Source to a Result.~~

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Constructor Detail

Transformer

```
protected Transformer()
```

~~Default constructor is protected on purpose.~~

Method Detail

reset

```
public void reset()
```

~~Reset this `Transformer` to its original configuration.~~

~~`Transformer` is reset to the same state as when it was created with `TransformerFactory.newTransformer()`, `TransformerFactory.newTransformer(Source source)` or `Templates.newTransformer()`. `reset()` is designed to allow the reuse of existing `Transformer`s thus saving resources associated with the creation of new `Transformer`s.~~

The reset `Transformer` is not guaranteed to have the same `URIResolver` or `ErrorListener` `Object`s, e.g. `Object.equals(Object obj)`. It is guaranteed to have a functionally equal `URIResolver` and `ErrorListener`.

Throws:

`java.lang.UnsupportedOperationException` - When implementation does not override this method.

Since:

1.5

---

transform

```
public abstract void transform(Source xmlSource,
                               Result outputTarget)
                        throws TransformerException
```

~~Transform the XML `Source` to a `Result`. Specific transformation behavior is determined by the settings of the `TransformerFactory` in effect when the `Transformer` was instantiated and any modifications made to the `Transformer` instance.~~

An empty `Source` is represented as an empty document as constructed by `DocumentBuilder.newDocument()`. The result of transforming an empty `Source` depends on the transformation behavior; it is not always an empty `Result`.

but for StreamSource the behavior different.
StreamSource cannot be empty

Parameters:

`xmlSource` - The XML input to transform.

`outputTarget` - The `Result` of transforming the `xmlSource`.

Throws:

`TransformerException` - If an unrecoverable error occurs during the course of the transformation.

---

setParameter

```
public abstract void setParameter(java.lang.String name,
                                  java.lang.Object value)
```

i hope, i can validate
at the time of usage.
now leave....

deals with <xsl:param>

Add a parameter for the transformation.

Pass a qualified name as a two-part string, the namespace URI enclosed in curly braces ({}), followed by the local name. If the name has a null URL, the String only contain the local name. An application can safely check for a non-null URI by testing to see if the first character of the name is a '{' character.

For example, if a URI and local name were obtained from an element defined with <xyz:foo xmlns:xyz="http://xyz.foo.com/yada/baz.html"/>, then the qualified name would be "{http://xyz.foo.com/yada/baz.html}foo". Note that no prefix is used.

Parameters:

`name` - The name of the parameter, which may begin with a namespace URI in curly braces ({}).

`value` - The value object. This can be any valid Java object. It is up to the processor to provide the proper object coersion or to simply pass the object on for use in an extension.

Throws:

`java.lang.NullPointerException` - If value is null.

---

getParameter

i hope, i can validate
at the time of usage.
now leave....

```
public abstract java.lang.Object getParameter(java.lang.String name)
```

deals with <xsl:param>

Get a parameter that was explicitly set with setParameter.

This method does not return a default parameter value, which cannot be determined until the node context is evaluated during the transformation process.

Parameters:

`name` - of `Object` to get

Returns:

A parameter that has been set with setParameter.

---

clearParameters

i hope, i can validate
at the time of usage.
now leave....

```
public abstract void clearParameters()
```

deals with <xsl:param>

Clear all parameters set with setParameter.

---

setURIResolver

```
public abstract void setURIResolver(URIResolver resolver)
```

Set an object that will be used to resolve URIs used in document().

If the resolver argument is null, the URIResolver value will be cleared and the transformer will no longer have a resolver.

by default, this will be null

Parameters:

`resolver` - An object that implements the URIResolver interface, or null.

---

getURIResolver

```
public abstract URIResolver getURIResolver()
```

Get an object that will be used to resolve URIs used in document().

Returns:

An object that implements the URIResolver interface, or null.

---

setOutputProperties

deals with <XSL:output./ >

i have skipped. hands on at the time usage (3 - jan -09)

```
public abstract void setOutputProperties(java.util.Properties oformat)
```

Set the output properties for the transformation. These properties will override properties set in the Templates with xsl:output.

If argument to this function is null, any properties previously set are removed, and the value will revert to the value defined in the templates object.

Pass a qualified property key name as a two-part string, the namespace URI enclosed in curly braces ({}), followed by the local name. If the name has a null URL, the String only contain the local name. An application can safely check for a non-null URI by testing to see if the first character of the name is a '{' character.

For example, if a URI and local name were obtained from an element defined with <xyz:foo xmlns:xyz="http://xyz.foo.com/yada/baz.html"/>, then the qualified name would be "{http://xyz.foo.com/yada/baz.html}foo". Note that no prefix is used.

An `IllegalArgumentException` is thrown if any of the argument keys are not recognized and are not namespace qualified.

Parameters:

`oformat` - A set of output properties that will be used to override any of the same properties in affect for the transformation.

Throws:

`java.lang.IllegalArgumentException` - When keys are not recognized and are not namespace qualified.

See Also:

OutputKeys, Properties

---

getOutputProperties

deals with <XSL:output./ >

i have skipped. Gets hands on at the time usage (3 - jan -09)

```
public abstract java.util.Properties getOutputProperties()
```

Get a copy of the output properties for the transformation.

The properties returned should contain 1 properties set by the user, and 2 properties set by the stylesheet, and these 3 properties are "defaulted" by default properties specified by section 16 of the XSL Transformations (XSLT) W3C Recommendation. The properties that were specifically set by the user or the stylesheet should be in the base Properties list, while the XSLT default properties that were not specifically set should be the default Properties list. Thus, getOutputProperties().getProperty(String key) will obtain any property in that was set by `setOutputProperty(java.lang.String, java.lang.String)`, `setOutputProperties(java.util.Properties)`, in the stylesheet, *or* the default properties, while getOutputProperties().get(String key) will only retrieve properties that were explicitly set by `setOutputProperty(java.lang.String, java.lang.String)`, `setOutputProperties(java.util.Properties)`, or in the stylesheet.

Note that mutation of the Properties object returned will not effect the properties that the transformer contains.

If any of the argument keys are not recognized and are not namespace qualified, the property will be ignored and not returned. In other words the behaviour is not orthogonal with `setOutputProperties`.

Returns:

A copy of the set of output properties in effect for the next transformation.

See Also:

`OutputKeys`, `Properties`, XSL Transformations (XSLT) Version 1.0

---

setOutputProperty

i have skipped. Gets hands on at the time usage (3 - jan -09)

```
public abstract void setOutputProperty(java.lang.String name,
                                       java.lang.String value)
                                throws java.lang.IllegalArgumentException
```

deals with <XSL:output./ >

Set an output property that will be in effect for the transformation.

Pass a qualified property name as a two-part string, the namespace URI enclosed in curly braces ({}), followed by the local name. If the name has a null URL, the String only contain the local name. An application can safely check for a non-null URI by testing to see if the first character of the name is a '{' character.

For example, if a URI and local name were obtained from an element defined with <xyz:foo xmlns:xyz="http://xyz.foo.com/yada/baz.html"/>, then the qualified name would be "{http://xyz.foo.com/yada/baz.html}foo". Note that no prefix is used.

The Properties object that was passed to setOutputProperties(java.util.Properties) won't be effected by calling this method.

Parameters:

name - A non-null String that specifies an output property name, which may be namespace qualified.

value - The non-null string value of the output property.

Throws:

java.lang.IllegalArgumentException - If the property is not supported, and is not qualified with a namespace.

See Also:

OutputKeys

---

deals with <XSL:output./ >

getOutputProperty

i have skipped. hands on at the time usage (3 - jan -09)

```
public abstract java.lang.String getOutputProperty(java.lang.String name)
                                            throws java.lang.IllegalArgumentException
```

Get an output property that is in effect for the transformer.

If a property has been set using setOutputProperty(java.lang.String, java.lang.String), that value will be returned. Otherwise, if a property is explicitly specified in the stylesheet, that value will be returned. If the value of the property has been defaulted, that is, if no value has been set explicitly either with setOutputProperty(java.lang.String, java.lang.String) or in the stylesheet, the result may vary depending on implementation and input stylesheet.

Parameters:

name - A non-null String that specifies an output property name, which may be namespace qualified.

Returns:

The string value of the output property, or null if no property was found.

Throws:

java.lang.IllegalArgumentException - If the property is not supported.

See Also:

OutputKeys

---

setErrorListener

By, default, transformer has NON-NULL errorListener

```
public abstract void setErrorListener(ErrorListener listener)
                               throws java.lang.IllegalArgumentException
```

ErrorListener cannot be null

Set the error event listener in effect for the transformation.

Parameters:

`listener` - The new error listener.

Throws:

`java.lang.IllegalArgumentException` - if listener is null.

---

getErrorListener

By, default,
transformer has
NON-NULL
errorListener

```
public abstract ErrorListener getErrorListener()
```

Get the error event handler in effect for the transformation. Implementations must provide a default error listener.

Returns:

The current error handler, which should never be null.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

javax.xml.transform

Class TransformerConfigurationException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └javax.xml.transform.TransformerException
              └javax.xml.transform.TransformerConfigurationException
```

All Implemented Interfaces:

java.io.Serializable

public class **TransformerConfigurationException**

extends TransformerException

Indicates a serious configuration error.

See Also:

Serialized Form

Constructor Summary
TransformerConfigurationException()   Create a new `TransformerConfigurationException` with no detail mesage.
TransformerConfigurationException(java.lang.String msg)   Create a new `TransformerConfigurationException` with the `String` specified as an error message.

TransformerConfigurationException(java.lang.String message, SourceLocator locator) Create a new TransformerConfigurationException from a message and a Locator.
TransformerConfigurationException(java.lang.String message, SourceLocator locator, java.lang.Throwable e) Wrap an existing exception in a TransformerConfigurationException.
TransformerConfigurationException(java.lang.String msg, java.lang.Throwable e) Create a new `TransformerConfigurationException` with the given `Exception` base cause and detail message.
TransformerConfigurationException(java.lang.Throwable e) Create a new `TransformerConfigurationException` with a given `Exception` base cause of the error.

Method Summary

Methods inherited from class javax.xml.transform.TransformerException
getCause, getException, getLocationAsString, getLocator, getMessageAndLocation, initCause, printStackTrace, printStackTrace, printStackTrace, setLocator

Methods inherited from class java.lang.Throwable
fillInStackTrace, getLocalizedMessage, getMessage, getStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

TransformerConfigurationException

```
public TransformerConfigurationException()
```

Create a new `TransformerConfigurationException` with no detail mesage.

TransformerConfigurationException

```
public TransformerConfigurationException(java.lang.String msg)
```

Create a new `TransformerConfigurationException` with the `String` specified as an error message.

Parameters:
`msg` - The error message for the exception.

TransformerConfigurationException

```
public TransformerConfigurationException(java.lang.Throwable e)
```

Create a new `TransformerConfigurationException` with a given `Exception` base cause of the error.

Parameters:
`e` - The exception to be encapsulated in a TransformerConfigurationException.

TransformerConfigurationException

```
public TransformerConfigurationException(java.lang.String msg,
                                         java.lang.Throwable e)
```

Create a new `TransformerConfigurationException` with the given `Exception` base cause and detail message.

Parameters:

`e` - The exception to be encapsulated in a TransformerConfigurationException
`msg` - The detail message.

TransformerConfigurationException

```
public TransformerConfigurationException(java.lang.String message,
                                         SourceLocator locator)
```

Create a new TransformerConfigurationException from a message and a Locator.

This constructor is especially useful when an application is creating its own exception from within a DocumentHandler callback.

Parameters:
`message` - The error or warning message.
`locator` - The locator object for the error or warning.

TransformerConfigurationException

```
public TransformerConfigurationException(java.lang.String message,
                                         SourceLocator locator,
                                         java.lang.Throwable e)
```

Wrap an existing exception in a TransformerConfigurationException.

Parameters:
`message` - The error or warning message, or null to use the message from the embedded exception.
`locator` - The locator object for the error or warning.
`e` - Any exception.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**
PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**
SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview Package **Class** Use Tree Deprecated Index Help

PREV CLASS NEXT CLASS FRAMES NO FRAMES All Classes

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

javax.xml.transform

Class TransformerException

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Exception
          └javax.xml.transform.TransformerException
```

All Implemented Interfaces:

java.io.Serializable

Direct Known Subclasses:

TransformerConfigurationException

```
public class TransformerException
```

extends java.lang.Exception

This class specifies an exceptional condition that occured during the transformation process.

See Also:

Serialized Form

Constructor Summary

`TransformerException(java.lang.String message)` Create a new TransformerException.

`TransformerException``(java.lang.String message, SourceLocator locator)` Create a new TransformerException from a message and a Locator.
`TransformerException``(java.lang.String message, SourceLocator locator, java.lang.Throwable e)` Wrap an existing exception in a TransformerException.
`TransformerException``(java.lang.String message, java.lang.Throwable e)` Wrap an existing exception in a TransformerException.
`TransformerException``(java.lang.Throwable e)` Create a new TransformerException wrapping an existing exception.

Method Summary

`java.lang.Throwable`	**`getCause`**`()` Returns the cause of this throwable or `null` if the cause is nonexistent or unknown.
`java.lang.Throwable`	**`getException`**`()` This method retrieves an exception that this exception wraps.
`java.lang.String`	**`getLocationAsString`**`()` Get the location information as a string.
`SourceLocator`	**`getLocator`**`()` Method getLocator retrieves an instance of a SourceLocator object that specifies where an error occured.
`java.lang.String`	**`getMessageAndLocation`**`()` Get the error message with location information appended.
`java.lang.Throwable`	**`initCause`**`(java.lang.Throwable cause)` Initializes the *cause* of this throwable to the specified value.
`void`	**`printStackTrace`**`()` Print the the trace of methods from where the error originated.
`void`	**`printStackTrace`**`(java.io.PrintStream s)` Print the the trace of methods from where the error originated.

void	**printStackTrace**(java.io.PrintWriter s) Print the the trace of methods from where the error originated.
void	**setLocator**(SourceLocator location) Method setLocator sets an instance of a SourceLocator object that specifies where an error occured.

Methods inherited from class java.lang.Throwable
fillInStackTrace, getLocalizedMessage, getMessage, getStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

TransformerException

```
public TransformerException(java.lang.String message)
```

Create a new TransformerException.

Parameters:

`message` - The error or warning message.

TransformerException

```
public TransformerException(java.lang.Throwable e)
```

Create a new TransformerException wrapping an existing exception.

Parameters:

`e` - The exception to be wrapped.

TransformerException

```
public TransformerException(java.lang.String message,
                            java.lang.Throwable e)
```

Wrap an existing exception in a TransformerException.

This is used for throwing processor exceptions before the processing has started.

Parameters:

`message` - The error or warning message, or null to use the message from the embedded exception.
`e` - Any exception

TransformerException

```
public TransformerException(java.lang.String message,
                            SourceLocator locator)
```

Create a new TransformerException from a message and a Locator.

This constructor is especially useful when an application is creating its own exception from within a DocumentHandler callback.

Parameters:

`message` - The error or warning message.
`locator` - The locator object for the error or warning.

TransformerException

```
public TransformerException(java.lang.String message,
                            SourceLocator locator,
                            java.lang.Throwable e)
```

Wrap an existing exception in a TransformerException.

Parameters:

`message` - The error or warning message, or null to use the message from the embedded exception.

`locator` - The locator object for the error or warning.

`e` - Any exception

Method Detail

getLocator

```
public SourceLocator getLocator()
```

Method getLocator retrieves an instance of a SourceLocator object that specifies where an error occured.

Returns:

A SourceLocator object, or null if none was specified.

---

setLocator

```
public void setLocator(SourceLocator location)
```

Method setLocator sets an instance of a SourceLocator object that specifies where an error occured.

Parameters:

`location` - A SourceLocator object, or null to clear the location.

---

getException

```
public java.lang.Throwable getException()
```

This method retrieves an exception that this exception wraps.

Returns:

An Throwable object, or null.

See Also:

`getCause()`

---

getCause

```
public java.lang.Throwable getCause()
```

Returns the cause of this throwable or `null` if the cause is nonexistent or unknown. (The cause is the throwable that caused this throwable to get thrown.)

Overrides:

`getCause` in class `java.lang.Throwable`

---

initCause

```
public java.lang.Throwable initCause(java.lang.Throwable cause)
```

Initializes the *cause* of this throwable to the specified value. (The cause is the throwable that caused this throwable to get thrown.)

This method can be called at most once. It is generally called from within the constructor, or immediately after creating the throwable. If this throwable was created with `TransformerException(Throwable)` or `TransformerException(String, Throwable)`, this method cannot be called even once.

Overrides:

`initCause` in class `java.lang.Throwable`

Parameters:

`cause` - the cause (which is saved for later retrieval by the `getCause()` method). (A `null` value is permitted, and indicates that the cause is nonexistent or unknown.)

Returns:

a reference to this `Throwable` instance.

Throws:

`java.lang.IllegalArgumentException` - if `cause` is this throwable. (A throwable cannot be its own cause.)
`java.lang.IllegalStateException` - if this throwable was created with `TransformerException(Throwable)` or `TransformerException(String,Throwable)`, or this method has already been called on this throwable.

---

getMessageAndLocation

```
public java.lang.String getMessageAndLocation()
```

Get the error message with location information appended.

Returns:
A `String` representing the error message with location information appended.

---

getLocationAsString

```
public java.lang.String getLocationAsString()
```

Get the location information as a string.

Returns:
A string with location info, or null if there is no location information.

---

printStackTrace

```
public void printStackTrace()
```

Print the the trace of methods from where the error originated. This will trace all nested exception objects, as well as this object.

Overrides:
`printStackTrace` in class `java.lang.Throwable`

---

printStackTrace

```
public void printStackTrace(java.io.PrintStream s)
```

Print the the trace of methods from where the error originated. This will trace all nested exception objects, as well as this object.

Overrides:

`printStackTrace` in class `java.lang.Throwable`

Parameters:

`s` - The stream where the dump will be sent to.

---

printStackTrace

```
public void printStackTrace(java.io.PrintWriter s)
```

Print the the trace of methods from where the error originated. This will trace all nested exception objects, as well as this object.

Overrides:

`printStackTrace` in class `java.lang.Throwable`

Parameters:

`s` - The writer where the dump will be sent to.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

---

3 - Jan - 09

javax.xml.transform

Class TransformerFactory

```
java.lang.Object
  └ javax.xml.transform.TransformerFactory
```

Direct Known Subclasses:

SAXTransformerFactory

SchemaFactory compiles XSD file, where as TransformerFactory compiles XSL file

it has 14 method

4 - getter / setter ErrorListener / URIResolver
4 - getter / setter for feature / property
2 - get the same instance
2 - get transformer
1 ~ get template
1 - retrieve style sheet associated with XML

the main purpose of this factory is to CREATE
1. Transformer
2. Templates

```
public abstract class TransformerFactory
```

extends java.lang.Object

A TransformerFactory instance can be used to create `Transformer` and `Templates` objects.

~~The system property that determines which Factory implementation to create is named "`javax.xml.transform.TransformerFactory`". This property names a concrete subclass of the `TransformerFactory` abstract class. If the property is not defined, a platform default is be used.~~

JAXP can have one utility class which can have string variable to hold keys of factory.

i have created a utility class in my JAXP archicture project

Author:

Jeff Suttor, Neeraj Bajaj

Constructor Summary

`protected`	**`TransformerFactory`**`()` ~~Default constructor is protected on purpose.~~

Method Summary

`abstract Source`
retrieve if any Style sheet associated with given XML file.
input source --> XML file
return source --> XSL file
	`getAssociatedStylesheet``(Source source, java.lang.String media, java.lang.String title, java.lang.String charset)`
	Get the stylesheet specification(s) associated with the XML `Source` document via the xml-stylesheet processing instruction that match the given criteria.

ALL ARE ABSTRACT method

abstract java.lang.Object	**getAttribute**(java.lang.String name) Allows the user to retrieve specific attributes on the underlying implementation.
abstract ErrorListener	**getErrorListener**() ~~Get the error event handler for the TransformerFactory.~~
abstract boolean	**getFeature**(java.lang.String name) ~~Look up the value of a feature.~~
abstract URIResolver	**getURIResolver**() Get the object that is used by default during the transformation to resolve URIs used in document(), xsl:import, or xsl:include.
static TransformerFactory	**newInstance**() ~~Obtain a new instance of a TransformerFactory.~~
static TransformerFactory	**newInstance**(java.lang.String factoryClassName, java.lang.ClassLoader classLoader) ~~Obtain a new instance of a TransformerFactory from factory class name.~~
abstract Templates	**newTemplates**(Source source) this is XSL file ~~Process the Source in~~ ~~a compiled representation of the source~~ The only way to associate XSL file with transformer while object creation
abstract Transformer	**newTransformer**() Create a new Transformer that performs a copy of the Source to the Result. use this when DONT have xsl file like converting SAX source to DOM result and vice versa
abstract Transformer	**newTransformer**(Source source) this is XSL file Process the Source into a Transformer Object.
abstract void	**setAttribute**(java.lang.String name, java.lang.Object value) ~~Allows the user to set specific attributes on the underlying implementation.~~
ract void	**setErrorListener**(ErrorListener listener) Set the error event listener for the TransformerFactory, which is used for the processing of transformation instructions, and not for the transformation itself. oh.... it is used WHILE processing XSL file. not FOR transformation
abstract void	**setFeature**(java.lang.String name, boolean value) Set a feature for this TransformerFactory and Transformers or Templates created by this factory.
abstract void	**setURIResolver**(URIResolver resolver) Set an object that is used by default during the transformation to resolve URIs used in document(), xsl:import, or xsl:include.

Methods inherited from class java.lang.Object

```
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString,
wait, wait, wait
```

Constructor Detail

TransformerFactory

```
protected TransformerFactory()
```

~~Default constructor is protected on purpose~~.

Method Detail

newInstance

```
public static TransformerFactory newInstance()
                                      throws TransformerFactoryConfigurationError
```

~~Obtain a new instance of a `TransformerFactory`. This static method creates a new factory instance This method uses the following ordered lookup procedure to determine the `TransformerFactory` implementation class to load:~~

1 - ~~Use the `javax.xml.transform.TransformerFactory` system property.~~
2 - ~~Use the properties file "lib/jaxp.properties" in the JRE directory. This configuration file is in standard `java.util.Properties` format and contains the fully qualified name of the implementation class with the key being the system property defined above. The jaxp.properties file is read only once by the JAXP implementation and it's values are then cached for future use. If the file does not exist when the first attempt is made to read from it, no further attempts are made to check for its existence. It is not possible to change the value of any property in jaxp.properties after it has been read for the first time.~~
3 - ~~Use the Services API (as detailed in the JAR specification), if available, to determine the classname. The Services API will look for a classname in the file `META-INF/services/javax.xml.transform.TransformerFactory` in jars available to the runtime.~~
- ~~Platform default `TransformerFactory` instance.~~

~~Once an application has obtained a reference to a `TransformerFactory` it can use the factory to configure and obtain transformer instances.~~

Returns:

new TransformerFactory instance, never null.

Throws:

`TransformerFactoryConfigurationError` - Thrown if the implementation is not available or cannot be instantiated.

newInstance

```
public static TransformerFactory newInstance(java.lang.String factoryClassName,
                                             java.lang.ClassLoader classLoader)
                                      throws TransformerFactoryConfigurationError
```

~~Obtain a new instance of a `TransformerFactory` from factory class name. This function is useful when there are multiple providers in the classpath. It gives more control to the application as it can specify which provider should be loaded.~~

~~Once an application has obtained a reference to a `TransformerFactory` it can use the factory to configure and obtain transformer instances.~~

Tip for Trouble-shooting

~~Setting the `jaxp.debug` system property will cause this method to print a lot of debug messages to `System.err` about what it is doing and where it is looking at.~~

If you have problems try:

```
 java -Djaxp.debug=1 YourProgram ....
```

Parameters:

~~`factoryClassName` - fully qualified factory class name that provides implementation of `javax.xml.transform.TransformerFactory`.~~

`classLoader` - `ClassLoader` used to load the factory class. If `null` current `Thread`'s context classLoader is used to load the factory class.

~~**Returns:**~~

~~new TransformerFactory instance, never null.~~

~~**Throws:**~~

~~`TransformerFactoryConfigurationError` - if `factoryClassName` is `null`, or the factory class cannot be loaded, instantiated.~~

~~**Since:**~~

~~1.6~~

~~**See Also:**~~

~~`newInstance()`~~

---

newTransformer

use this method to associate a XSL file for the transformation

```
public abstract Transformer newTransformer(Source source)
                                    throws TransformerConfigurationException
```

Process the Source into a Transformer Object. The Source is an XSLT document that ~~conforms to XSL Transformations (XSLT) Version 1.0. Care must be taken not to use this Transformer in multiple Threads running concurrently. Different TransformerFactories can be used concurrently by different Threads.~~

Parameters:

source - Source of XSLT document used to create Transformer. Examples ~~of XML Sources include DOMSource, SAXSource, and StreamSource.~~

Returns:

A Transformer object that may be used to perform a transformation in a single Thread, never null.

~~**Throws:**~~

~~TransformerConfigurationException - Thrown if there are errors when parsing the Source or it is not possible to create a Transformer instance.~~

~~**See Also:**~~

~~XSL Transformations (XSLT) Version 1.0~~

---

newTransformer

use this method, when DONT have any XSL file for this transformation

source and result can be mixed like sax and dom

```
public abstract Transformer newTransformer()
                                    throws TransformerConfigurationException
```

Create a new Transformer that performs a copy of the Source to the Result. i.e. the "*identity transform*".

Returns:

A Transformer object that may be used to perform a transformation in a single thread, never null.

~~**Throws:**~~

~~TransformerConfigurationException - When it is not possible to create a Transformer instance~~.

---

newTemplates

Source cannot be either null or empty object. (i tested)

this is XSL file

```
public abstract Templates newTemplates(Source source)
                                throws TransformerConfigurationException
```

Process the Source into a Templates object, which is a a compiled representation of the source. This Templates object may then be used concurrently across multiple threads. Creating a Templates object allows the TransformerFactory to do detailed performance optimization of transformation instructions, without penalizing runtime transformation.

~~**Parameters:**~~

~~source - An object that holds a URL, input stream, etc.~~

~~**Returns:**~~

~~A Templates object capable of being used for transformation purposes, never `null`.~~

~~**Throws:**~~

TransformerConfigurationException ~~- When parsing to construct the Templates object fails.~~

---

getAssociatedStylesheet

```
public abstract Source getAssociatedStylesheet(Source source,
                                               java.lang.String media,
                                               java.lang.String title,
                                               java.lang.String charset)
                                        s TransformerConfigurationException
```

this is another way to apply XSL to xml.

1. apply xsl by programming
2. apply xsl by xml file itself

so, we mostly dont use this method

Get the stylesheet specification(s) associated with the XML `Source` document via the xml-stylesheet processing instruction that match the given criteria. Note that it is possible to return several stylesheets, in which case they are applied as if they were a list of imports or cascades in a single stylesheet.

Parameters:

`source` - The XML source document.

`media` - The media attribute to be matched. May be null, in which case the prefered templates will be used (i.e. alternate = no).

`title` - The value of the title attribute to match. May be null.

`charset` - The value of the charset attribute to match. May be null.

~~**Returns:**~~

~~A `Source Object` suitable for passing to the `TransformerFactory`.~~

~~**Throws:**~~

TransformerConfigurationException ~~- An `Exception` is thrown if an error occurings during parsing of the `source`.~~

~~**See Also:**~~

~~Associating Style Sheets with XML documents Version 1.0~~

---

setURIResolver

```
public abstract void setURIResolver(URIResolver resolver)
```

Set an object that is used by default during the transformation to resolve URIs used in document(), xsl:import, or xsl:include.

yes, it can be null

Parameters:

`resolver` - An object that implements the URIResolver interface, or null.

---

getURIResolver

```
public abstract URIResolver getURIResolver()
```

~~Get the object that is used by default during the transformation to resolve URIs used in document(), xsl:import, or xsl:include.~~

~~**Returns:**~~

~~The URIResolver that was set with setURIResolver.~~

---

setFeature

```
public abstract void setFeature(java.lang.String name,
                                boolean value)
                         throws TransformerConfigurationException
```

Set a feature for this `TransformerFactory` and `Transformer`s or `Template`s created by this factory.

~~Feature names are fully qualified `URI`s. Implementations may define their own features. An `TransformerConfigurationException` is thrown if this `TransformerFactory` or the `Transformer`s or `Template`s it creates cannot support the feature. It is possible for an `TransformerFactory` to expose a feature value but be unable to change its state.~~

~~All implementations are required to support the `XMLConstants.FEATURE_SECURE_PROCESSING` feature. When the feature is:~~

- ~~`true`: the implementation will limit XML processing to conform to implementation limits and behave in a secure fashion as defined by the implementation. Examples include resolving user defined style sheets and functions. If XML processing is limited for security reasons, it will be reported via a call to the registered `ErrorListener.fatalError(TransformerException exception)`. See `setErrorListener(ErrorListener listener)`.~~
- ~~`false`: the implementation will processing XML according to the XML specifications without regard to possible implementation limits.~~

Parameters:

`name` - Feature name.

`value` - Is feature state `true` or `false`.

Throws:

`TransformerConfigurationException` - if this `TransformerFactory` or the `Transformer`s or `Template`s it creates cannot support this feature.

`java.lang.NullPointerException` - If the `name` parameter is null.

getFeature

```
public abstract boolean getFeature(java.lang.String name)
```

~~Look up the value of a feature.~~

Defined features:
(key : url-value)
StreamSource.FEATURE: http://javax.xml.transform.stream.StreamSource/feature
StreamResult.FEATURE: http://javax.xml.transform.stream.StreamResult/feature
DOMSource.FEATURE: http://javax.xml.transform.dom.DOMSource/feature
DOMResult.FEATURE: http://javax.xml.transform.dom.DOMResult/feature
SAXSource.FEATURE: http://javax.xml.transform.dom.SAXSource/feature
SAXResult.FEATURE: http://javax.xml.transform.dom.SAXResult/feature
SAXTransformerFactory.FEATURE: http://javax.xml.transform.sax.SAXTransformerFactory/feature
SAXTransformerFactory.FEATURE_XMLFILTER: http://javax.xml.transform.sax.
SAXTransformerFactory/feature/xmlfilter
The boolean values of these features for the current XSLT engine can be tested with the getFeature() method in the TransformerFactory class:

~~is~~

setAttribute

```
public abstract void setAttribute(java.lang.String name,
                                  java.lang.Object value)
```

~~Allows the user to set specific attributes on the underlying implementation. An attribute in this context is defined to be an option that the implementation provides. An IllegalArgumentException is thrown if the underlying implementation doesn't recognize the attribute.~~

~~**Parameters:**~~
- ~~name - The name of the attribute.~~
- ~~value - The value of the attribute.~~

~~**Throws:**~~
- ~~java.lang~~.IllegalArgumentException - When implementation does not recognize the attribute.

getAttribute

```
public abstract java.lang.Object getAttribute(java.lang.String name)
```

~~Allows the user to retrieve specific attributes on the underlying implementation. An IllegalArgumentException is thrown if the underlying implementation doesn't recognize the attribute.~~

~~**Parameters:**~~

~~`name` - The name of the attribute.~~

~~**Returns:**~~

~~value The value of the attribute.~~

~~**Throws:**~~

~~`java.lang.IllegalArgumentException` - When implementation does not recognize the attribute.~~

---

setErrorListener

ErrorListener Cannot be Null as
URIResolver. (i tested)

```
public abstract void setErrorListener(ErrorListener listener)
```

Set the error event listener for the TransformerFactory, which is used for the processing of transformation instructions, and not for the transformation itself. An `IllegalArgumentException` is thrown if the `ErrorListener` listener is `null`.

this listener will be notified ONLY while
processing XSL. But not while
transformation of a XML file

Parameters:

`listener` - The new error listener.

Throws:

`java.lang.IllegalArgumentException` - When `listener` is `null`

---

getErrorListener

```
public abstract ErrorListener getErrorListener()
```

~~Get the error event handler for the TransformerFactory.~~

~~**Returns:**~~

~~The current error handler, which should never be null.~~

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

javax.xml.transform

Class TransformerFactoryConfigurationError

```
java.lang.Object
  └java.lang.Throwable
      └java.lang.Error
          └javax.xml.transform.TransformerFactoryConfigurationError
```

All Implemented Interfaces:

java.io.Serializable

`public class` **`TransformerFactoryConfigurationError`**

extends java.lang.Error

Thrown when a problem with configuration with the Transformer Factories exists. This error will typically be thrown when the class of a transformation factory specified in the system properties cannot be found or instantiated.

See Also:

Serialized Form

Constructor Summary
`TransformerFactoryConfigurationError``()` Create a new `TransformerFactoryConfigurationError` with no detail mesage.
`TransformerFactoryConfigurationError``(java.lang.Exception e)` Create a new `TransformerFactoryConfigurationError` with a given `Exception` base cause of the error.

TransformerFactoryConfigurationError(java.lang.Exception e, java.lang.String msg) Create a new `TransformerFactoryConfigurationError` with the given `Exception` base cause and detail message.
TransformerFactoryConfigurationError(java.lang.String msg) Create a new `TransformerFactoryConfigurationError` with the `String` specified as an error message.

Method Summary

java.lang.Exception	**getException**() Return the actual exception (if any) that caused this exception to be raised.
java.lang.String	**getMessage**() Return the message (if any) for this error .

Methods inherited from class java.lang.Throwable
fillInStackTrace, getCause, getLocalizedMessage, getStackTrace, initCause, printStackTrace, printStackTrace, printStackTrace, setStackTrace, toString

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, wait, wait, wait

Constructor Detail

TransformerFactoryConfigurationError

```
public TransformerFactoryConfigurationError()
```

Create a new `TransformerFactoryConfigurationError` with no detail mesage.

TransformerFactoryConfigurationError

```
public TransformerFactoryConfigurationError(java.lang.String msg)
```

Create a new `TransformerFactoryConfigurationError` with the `String` specified as an error message.

Parameters:

`msg` - The error message for the exception.

TransformerFactoryConfigurationError

```
public TransformerFactoryConfigurationError(java.lang.Exception e)
```

Create a new `TransformerFactoryConfigurationError` with a given `Exception` base cause of the error.

Parameters:

`e` - The exception to be encapsulated in a TransformerFactoryConfigurationError.

TransformerFactoryConfigurationError

```
public TransformerFactoryConfigurationError(java.lang.Exception e,
                                            java.lang.String msg)
```

Create a new `TransformerFactoryConfigurationError` with the given `Exception` base cause and detail message.

Parameters:

`e` - The exception to be encapsulated in a TransformerFactoryConfigurationError

`msg` - The detail message.

Method Detail

getMessage

```
public java.lang.String getMessage()
```

Return the message (if any) for this error . If there is no message for the exception and there is an encapsulated exception then the message of that exception will be returned.

Overrides:

`getMessage` in class `java.lang.Throwable`

Returns:

The error message.

---

getException

```
public java.lang.Exception getException()
```

Return the actual exception (if any) that caused this exception to be raised.

Returns:

The encapsulated exception, or null if there is none.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS **NEXT CLASS** **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NE

FRAMES **NO FRAMES** **All Classes**

SUMMARY: NESTED | ETHOD

Register this object into SAXParser as contentHanlder.
So, really it is hander :) :)

see my JAXP code to see example of this class

3 - jan - 09

javax.xml.transform.sax

Interface TransformerHandler

This is Transfer SAXEvents into Result object

All Superinterfaces:

ContentHandler, DTDHand

we have to call XMLReader.parser() method explictly when using SAXTransformerFactory and no need to call transform method .
But in SAXSource we never call parse method of XMLReader explicitly and but have to call transform() method on transormer

```
public interface TransformerHandler
```

extends ContentHandler, LexicalHandler, DTDHandler

it has 4 methods
2 - setter /getter for system id
1 - getter for transformer
1 - setter for Result

A TransformerHandler listens for SAX ContentHandler parse events and transforms them to a Result.

if you want to call transform() method on Transformer for sax then must use SAXSource (this is another api for SAX)

Method Summary

java.lang.String	**getSystemId**() ~~Get the base ID (URI or syst~~ ~~resolved~~.
Transformer	**getTransformer**() Get the `Transformer` associated with this handler, which is needed in order to set parameters and output properties.
void	**setResult**(Result result) Set the `Result` associated with this `TransformerHandler` to be used for the transformation.
void	**setSystemId**(java.lang.String systemID) ~~Set the base ID (URI or system ID) from where relative URLs will be resolved.~~

it is used **ONLY** for setting output properties and paremeters. But dont call *(actually cannot call .. think arguments of transform() method)* transform method on this returned Transformer. Simply call parse method on xmlreader / saxparser to execute transformation process

Methods inherited from interface org.xml.sax.ContentHandler

characters, endDocument, endElement, endPrefixMapping, ignorableWhitespace, processingInstruction, setDocumentLocator, skippedEntity, startDocument, startElement, startPrefixMapping

Methods inherited from interface org.xml.sax.ext.LexicalHandler
comment, endCDATA, endDTD, endEntity, startCDATA, startDTD, startEntity

Methods inherited from interface org.xml.sax.DTDHandler
notationDecl, unparsedEntityDecl

Method Detail

setResult

```
void setResult(Result result)
               throws java.lang.IllegalArgumentException
```

~~Set the `Result` associated with this `TransformerHandler` to be used for the transformation.~~

~~**Parameters:**~~

~~`result` - A `Result` instance, should not be `null`.~~

~~**Throws:**~~

~~`java.lang.IllegalArgumentException` - if result is invalid for some reason.~~

---

setSystemId

```
void setSystemId(java.lang.String systemID)
```

~~Set the base ID (URI or system ID) from where relative URLs will be resolved.~~

~~**Parameters:**~~

~~`systemID` - Base URI for the source tree.~~

getSystemId

```
java.lang.String getSystemId()
```

~~Get the base ID (URI or system ID) from where relative URLs will be resolved.~~

~~**Returns:**~~

~~The systemID that was set with setSystemId(java.lang.String).~~

getTransformer

```
Transformer getTransformer()
```

it is used **ONLY** for setting output properties and paremeters. But dont call *(actually cannot call .. think arguments of transform() method)* transform method on this returned Transformer. Simply call parse method on xmlreader / saxparser to execute transformation process

Get the `Transformer` associated with this handler, which is needed in order to set parameters and output properties.

Returns:

`Transformer` associated with this `TransformerHandler`.

3 - Jan - 09

Package **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS

SUMMARY: NESTED | FIELD | CONSTR | METHOD

FRAMES **NO FRAMES** **All Classes**

DETAIL: FIELD | CONSTR | METHOD

---

javax.xml.transform

Interface URIResolver

it has 1 method

so it is used only with XSL
(Template object)

```
public interface URIResolver
```

it is not for
DTD :)

An object that implements this interface that can be called by the processor to turn a URI used in document(), xsl:import, or xsl:include into a Source object.

Method Summary

Source	**resolve**(java.lang.String href, java.lang.String base) Called by the processor when it encounters an xsl:include, xsl:import, or document() function.

Method Detail

resolve

```
Source resolve(java.lang.String href,
               java.lang.String base)
               throws TransformerException
```

Called by the processor when it encounters an xsl:include, xsl:import, or document() function.

Parameters:

`href` - An href attribute, which may be relative or absolute.

`base` - The base URI against which the first argument will be made absolute if the absolute URI is required.

Returns:

A Source object, or null if the href cannot be resolved, and the processor should try to resolve the URI itself.

Throws:

`TransformerException` - if an error occurs when trying to resolve the URI.

Overview **Package** **Class** **Use** **Tree** **Deprecated** **Index** **Help**

PREV CLASS NEXT CLASS **FRAMES** **NO FRAMES** **All Classes**

SUMMARY: NESTED | FIELD | CONSTR | METHOD DETAIL: FIELD | CONSTR | METHOD