| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TypeInfo.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/org/w3c/dom/Text.html)   [**NEXT CLASS**](http://docs.google.com/org/w3c/dom/UserDataHandler.html) | [**FRAMES**](http://docs.google.com/index.html?org/w3c/dom/TypeInfo.html)    [**NO FRAMES**](http://docs.google.com/TypeInfo.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#tyjcwt) | CONSTR | [METHOD](#17dp8vu) |

## **org.w3c.dom**

Interface TypeInfo

public interface **TypeInfo**

The TypeInfo interface represents a type referenced from Element or Attr nodes, specified in the schemas associated with the document. The type is a pair of a namespace URI and name properties, and depends on the document's schema.

If the document's schema is an XML DTD [[XML 1.0](http://www.w3.org/TR/2004/REC-xml-20040204)], the values are computed as follows:

* If this type is referenced from an Attr node, typeNamespace is "http://www.w3.org/TR/REC-xml" and typeName represents the **[attribute type]** property in the [[XML Information Set](http://www.w3.org/TR/2004/REC-xml-infoset-20040204/)] . If there is no declaration for the attribute, typeNamespace and typeName are null.
* If this type is referenced from an Element node, typeNamespace and typeName are null.

If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , the values are computed as follows using the post-schema-validation infoset contributions (also called PSVI contributions):

* If the **[validity]** property exists AND is *"invalid"* or *"notKnown"*: the {target namespace} and {name} properties of the declared type if available, otherwise null.  
  **Note:** At the time of writing, the XML Schema specification does not require exposing the declared type. Thus, DOM implementations might choose not to provide type information if validity is not valid.
* If the **[validity]** property exists and is *"valid"*:
  1. If **[member type definition]** exists:
     1. If {name} is not absent, then expose {name} and {target namespace} properties of the **[member type definition]** property;
     2. Otherwise, expose the namespace and local name of the corresponding anonymous type name.
  2. If the **[type definition]** property exists:
     1. If {name} is not absent, then expose {name} and {target namespace} properties of the **[type definition]** property;
     2. Otherwise, expose the namespace and local name of the corresponding anonymous type name.
  3. If the **[member type definition anonymous]** exists:
     1. If it is false, then expose **[member type definition name]** and **[member type definition namespace]** properties;
     2. Otherwise, expose the namespace and local name of the corresponding anonymous type name.
  4. If the **[type definition anonymous]** exists:
     1. If it is false, then expose **[type definition name]** and **[type definition namespace]** properties;
     2. Otherwise, expose the namespace and local name of the corresponding anonymous type name.

**Note:** Other schema languages are outside the scope of the W3C and therefore should define how to represent their type systems using TypeInfo.

See also the [Document Object Model (DOM) Level 3 Core Specification](http://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407).

**Since:** DOM Level 3

| **Field Summary** | |
| --- | --- |
| static int | [**DERIVATION\_EXTENSION**](http://docs.google.com/org/w3c/dom/TypeInfo.html#DERIVATION_EXTENSION)            If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the derivation by  [extension](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#key-typeExtension). |
| static int | [**DERIVATION\_LIST**](http://docs.google.com/org/w3c/dom/TypeInfo.html#DERIVATION_LIST)            If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the [list](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#element-list). |
| static int | [**DERIVATION\_RESTRICTION**](http://docs.google.com/org/w3c/dom/TypeInfo.html#DERIVATION_RESTRICTION)            If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the derivation by  [restriction](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#key-typeRestriction) if complex types are involved, or a  [restriction](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#element-restriction) if simple types are involved. |
| static int | [**DERIVATION\_UNION**](http://docs.google.com/org/w3c/dom/TypeInfo.html#DERIVATION_UNION)            If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the  [union](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#element-union) if simple types are involved. |

| **Method Summary** | |
| --- | --- |
| [String](http://docs.google.com/java/lang/String.html) | [**getTypeName**](http://docs.google.com/org/w3c/dom/TypeInfo.html#getTypeName())()            The name of a type declared for the associated element or attribute, or null if unknown. |
| [String](http://docs.google.com/java/lang/String.html) | [**getTypeNamespace**](http://docs.google.com/org/w3c/dom/TypeInfo.html#getTypeNamespace())()            The namespace of the type declared for the associated element or attribute or null if the element does not have declaration or if no namespace information is available. |
| boolean | [**isDerivedFrom**](http://docs.google.com/org/w3c/dom/TypeInfo.html#isDerivedFrom(java.lang.String,%20java.lang.String,%20int))([String](http://docs.google.com/java/lang/String.html) typeNamespaceArg, [String](http://docs.google.com/java/lang/String.html) typeNameArg, int derivationMethod)            This method returns if there is a derivation between the reference type definition, i.e. |

| **Field Detail** |
| --- |

### DERIVATION\_RESTRICTION

static final int **DERIVATION\_RESTRICTION**

If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the derivation by  [restriction](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#key-typeRestriction) if complex types are involved, or a  [restriction](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#element-restriction) if simple types are involved.

The reference type definition is derived by restriction from the other type definition if the other type definition is the same as the reference type definition, or if the other type definition can be reached recursively following the {base type definition} property from the reference type definition, and all the *derivation methods* involved are restriction.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.TypeInfo.DERIVATION_RESTRICTION)

### DERIVATION\_EXTENSION

static final int **DERIVATION\_EXTENSION**

If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the derivation by  [extension](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#key-typeExtension).

The reference type definition is derived by extension from the other type definition if the other type definition can be reached recursively following the {base type definition} property from the reference type definition, and at least one of the *derivation methods* involved is an extension.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.TypeInfo.DERIVATION_EXTENSION)

### DERIVATION\_UNION

static final int **DERIVATION\_UNION**

If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the  [union](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#element-union) if simple types are involved.

The reference type definition is derived by union from the other type definition if there exists two type definitions T1 and T2 such as the reference type definition is derived from T1 by DERIVATION\_RESTRICTION or DERIVATION\_EXTENSION, T2 is derived from the other type definition by DERIVATION\_RESTRICTION, T1 has {variety} *union*, and one of the {member type definitions} is T2. Note that T1 could be the same as the reference type definition, and T2 could be the same as the other type definition.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.TypeInfo.DERIVATION_UNION)

### DERIVATION\_LIST

static final int **DERIVATION\_LIST**

If the document's schema is an XML Schema [[XML Schema Part 1](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/)] , this constant represents the [list](http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/#element-list).

The reference type definition is derived by list from the other type definition if there exists two type definitions T1 and T2 such as the reference type definition is derived from T1 by DERIVATION\_RESTRICTION or DERIVATION\_EXTENSION, T2 is derived from the other type definition by DERIVATION\_RESTRICTION, T1 has {variety} *list*, and T2 is the {item type definition}. Note that T1 could be the same as the reference type definition, and T2 could be the same as the other type definition.

**See Also:**[Constant Field Values](http://docs.google.com/constant-values.html#org.w3c.dom.TypeInfo.DERIVATION_LIST)

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

### getTypeName

[String](http://docs.google.com/java/lang/String.html) **getTypeName**()

The name of a type declared for the associated element or attribute, or null if unknown.

### getTypeNamespace

[String](http://docs.google.com/java/lang/String.html) **getTypeNamespace**()

The namespace of the type declared for the associated element or attribute or null if the element does not have declaration or if no namespace information is available.

### isDerivedFrom

boolean **isDerivedFrom**([String](http://docs.google.com/java/lang/String.html) typeNamespaceArg,  
 [String](http://docs.google.com/java/lang/String.html) typeNameArg,  
 int derivationMethod)

This method returns if there is a derivation between the reference type definition, i.e. the TypeInfo on which the method is being called, and the other type definition, i.e. the one passed as parameters.

**Parameters:**typeNamespaceArg - the namespace of the other type definition.typeNameArg - the name of the other type definition.derivationMethod - the type of derivation and conditions applied between two types, as described in the list of constants provided in this interface. **Returns:**If the document's schema is a DTD or no schema is associated with the document, this method will always return false . If the document's schema is an XML Schema, the method will true if the reference type definition is derived from the other type definition according to the derivation parameter. If the value of the parameter is 0 (no bit is set to 1 for the derivationMethod parameter), the method will return true if the other type definition can be reached by recursing any combination of {base type definition}, {item type definition}, or {member type definitions} from the reference type definition.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TypeInfo.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/org/w3c/dom/Text.html)   [**NEXT CLASS**](http://docs.google.com/org/w3c/dom/UserDataHandler.html) | [**FRAMES**](http://docs.google.com/index.html?org/w3c/dom/TypeInfo.html)    [**NO FRAMES**](http://docs.google.com/TypeInfo.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | CONSTR | [METHOD](#2et92p0) | DETAIL: [FIELD](#tyjcwt) | CONSTR | [METHOD](#17dp8vu) |

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

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

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