| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TypeCode.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/omg/CORBA/TRANSIENT.html)   [**NEXT CLASS**](http://docs.google.com/org/omg/CORBA/TypeCodeHolder.html) | [**FRAMES**](http://docs.google.com/index.html?org/omg/CORBA/TypeCode.html)    [**NO FRAMES**](http://docs.google.com/TypeCode.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

## **org.omg.CORBA**

Class TypeCode

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **org.omg.CORBA.TypeCode**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [IDLEntity](http://docs.google.com/org/omg/CORBA/portable/IDLEntity.html)

public abstract class **TypeCode**extends [Object](http://docs.google.com/java/lang/Object.html)implements [IDLEntity](http://docs.google.com/org/omg/CORBA/portable/IDLEntity.html)

A container for information about a specific CORBA data type.

TypeCode objects are used:

* in the Dynamic Invocation Interface -- to indicate the types of the actual arguments or the type of the return value.  
  NamedValue objects are used to represent arguments and return values. One of their components is an Any object, which in turn has as one of its components a TypeCode object.
* by an Interface Repository to represent the type specifications that are part of many OMG IDL declarations

The representation of a TypeCode object is opaque, but abstractly, a TypeCode object consists of:

* a kind field, which is set to an instance of the class TCKind
* zero or more additional fields appropriate for the particular kind. For example, the TypeCode object describing the OMG IDL type 1ong has kind TCKind.tk\_long and no additional fields. The TypeCode describing OMG IDL type sequence<boolean, 10> has a kind field with the value TCKind.tk\_sequence and also fields with the values boolean and 10 for the type of sequence elements and the length of the sequence.

TypeCode objects can be obtained in various ways:

1. from a call to the method Any.insert\_X, where X is a basic IDL type. This method creates a TypeCode object for type X and assigns it to the Any object's type field.
2. from invocations of methods in the ORB class  
   For example, the following creates a TypeCode object for a string with a maximum of 30 characters:  
    org.omg.CORBA.TypeCode tcString = orb.create\_string\_tc(30);  
      
   The following creates a TypeCode object for an array of five strings:  
    org.omg.CORBA.TypeCode tcArray = orb.create\_array\_tc(  
    5, TCKind.tk\_string);  
      
   The following creates a TypeCode object for an interface named "Account":  
    org.omg.CORBA.TypeCode tcInterface = orb.create\_interface\_tc(  
    "thisId", "Account");
3. as the return value from the \_type method in Holder classes for user-defined IDL types. These Holder classes are generated by the idltojava compiler.
4. from a CORBA Interface Repository

Most of the methods in the class TypeCode are accessors, and the information contained in a TypeCode object is specific to a particular type. Therefore, methods must be invoked only on the kind of type codes to which they apply. If an accessor method tries to access information from an inappropriate kind of type code, it will throw the exception TypeCodePackage.BadKind. For example, if the method discriminator\_type is called on anything other than a union, it will throw BadKind because only unions have a discriminator. The following list shows which methods apply to which kinds of type codes:

These methods may be invoked on all TypeCode kinds:

* equal
* kind

These methods may be invoked on objref, struct, union, enum, alias, exception, value, value\_box, native, and abstract\_interface:

* id
* name

These methods may be invoked on struct, union, enum, and exception:

* member\_count
* member\_name

These methods may be invoked on struct, union, and exception:

* member\_type(int index)

These methods may be invoked on union:

* member\_label
* discriminator\_type
* default\_index

These methods may be invoked on string, sequence, and array:

* length

These methods may be invoked on alias, sequence, array, and value\_box:

* content\_type

Unlike other CORBA pseudo-objects, TypeCode objects can be passed as general IDL parameters.

The methods parameter and param\_count, which are deprecated, are not mapped.

Java IDL extends the CORBA specification to allow all operations permitted on a struct TypeCode to be permitted on an exception TypeCode as well.

| **Constructor Summary** | |
| --- | --- |
| [**TypeCode**](http://docs.google.com/org/omg/CORBA/TypeCode.html#TypeCode())() |

| **Method Summary** | |
| --- | --- |
| abstract  [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) | [**concrete\_base\_type**](http://docs.google.com/org/omg/CORBA/TypeCode.html#concrete_base_type())()            Returns the TypeCode object that describes the concrete base type of the value type that this TypeCode object describes. |
| abstract  [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) | [**content\_type**](http://docs.google.com/org/omg/CORBA/TypeCode.html#content_type())()            Returns the TypeCode object representing the IDL type for the members of the object described by this TypeCode object. |
| abstract  int | [**default\_index**](http://docs.google.com/org/omg/CORBA/TypeCode.html#default_index())()            Returns the index of the default member, or -1 if there is no default member. |
| abstract  [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) | [**discriminator\_type**](http://docs.google.com/org/omg/CORBA/TypeCode.html#discriminator_type())()            Returns a TypeCode object describing all non-default member labels. |
| abstract  boolean | [**equal**](http://docs.google.com/org/omg/CORBA/TypeCode.html#equal(org.omg.CORBA.TypeCode))([TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) tc)            Compares this TypeCode object with the given one, testing for equality. |
| abstract  boolean | [**equivalent**](http://docs.google.com/org/omg/CORBA/TypeCode.html#equivalent(org.omg.CORBA.TypeCode))([TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) tc)            Tests to see if the given TypeCode object is equivalent to this TypeCode object. |
| abstract  short | [**fixed\_digits**](http://docs.google.com/org/omg/CORBA/TypeCode.html#fixed_digits())()            Returns the number of digits in the fixed type described by this TypeCode object. |
| abstract  short | [**fixed\_scale**](http://docs.google.com/org/omg/CORBA/TypeCode.html#fixed_scale())()            Returns the scale of the fixed type described by this TypeCode object. |
| abstract  [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) | [**get\_compact\_typecode**](http://docs.google.com/org/omg/CORBA/TypeCode.html#get_compact_typecode())()            Strips out all optional name and member name fields, but leaves all alias typecodes intact. |
| abstract  [String](http://docs.google.com/java/lang/String.html) | [**id**](http://docs.google.com/org/omg/CORBA/TypeCode.html#id())()            Retrieves the RepositoryId globally identifying the type of this TypeCode object. |
| abstract  [TCKind](http://docs.google.com/org/omg/CORBA/TCKind.html) | [**kind**](http://docs.google.com/org/omg/CORBA/TypeCode.html#kind())()            Retrieves the kind of this TypeCode object. |
| abstract  int | [**length**](http://docs.google.com/org/omg/CORBA/TypeCode.html#length())()            Returns the number of elements in the type described by this TypeCode object. |
| abstract  int | [**member\_count**](http://docs.google.com/org/omg/CORBA/TypeCode.html#member_count())()            Retrieves the number of members in the type described by this TypeCode object. |
| abstract  [Any](http://docs.google.com/org/omg/CORBA/Any.html) | [**member\_label**](http://docs.google.com/org/omg/CORBA/TypeCode.html#member_label(int))(int index)            Retrieves the label of the union member identified by the given index. |
| abstract  [String](http://docs.google.com/java/lang/String.html) | [**member\_name**](http://docs.google.com/org/omg/CORBA/TypeCode.html#member_name(int))(int index)            Retrieves the simple name of the member identified by the given index. |
| abstract  [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) | [**member\_type**](http://docs.google.com/org/omg/CORBA/TypeCode.html#member_type(int))(int index)            Retrieves the TypeCode object describing the type of the member identified by the given index. |
| abstract  short | [**member\_visibility**](http://docs.google.com/org/omg/CORBA/TypeCode.html#member_visibility(int))(int index)            Returns the constant that indicates the visibility of the member at the given index. |
| abstract  [String](http://docs.google.com/java/lang/String.html) | [**name**](http://docs.google.com/org/omg/CORBA/TypeCode.html#name())()            Retrieves the simple name identifying this TypeCode object within its enclosing scope. |
| abstract  short | [**type\_modifier**](http://docs.google.com/org/omg/CORBA/TypeCode.html#type_modifier())()            Returns a constant indicating the modifier of the value type that this TypeCode object describes. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [clone](http://docs.google.com/java/lang/Object.html#clone()), [equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [hashCode](http://docs.google.com/java/lang/Object.html#hashCode()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [toString](http://docs.google.com/java/lang/Object.html#toString()), [wait](http://docs.google.com/java/lang/Object.html#wait()), [wait](http://docs.google.com/java/lang/Object.html#wait(long)), [wait](http://docs.google.com/java/lang/Object.html#wait(long,%20int)) |

| **Constructor Detail** |
| --- |

### TypeCode

public **TypeCode**()

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

### equal

public abstract boolean **equal**([TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) tc)

Compares this TypeCode object with the given one, testing for equality. TypeCode objects are equal if they are interchangeable and give identical results when TypeCode operations are applied to them.

**Parameters:**tc - the TypeCode object to compare against **Returns:**true if the type codes are equal; false otherwise

### equivalent

public abstract boolean **equivalent**([TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) tc)

Tests to see if the given TypeCode object is equivalent to this TypeCode object.

**Parameters:**tc - the typecode to compare with this typecode **Returns:**true if the given typecode is equivalent to this typecode; false otherwise

### get\_compact\_typecode

public abstract [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) **get\_compact\_typecode**()

Strips out all optional name and member name fields, but leaves all alias typecodes intact.

**Returns:**a TypeCode object with optional name and member name fields stripped out, except for alias typecodes, which are left intact**See Also:**[CORBA package comments for unimplemented features](http://docs.google.com/package-summary.html#unimpl)

### kind

public abstract [TCKind](http://docs.google.com/org/omg/CORBA/TCKind.html) **kind**()

Retrieves the kind of this TypeCode object. The kind of a type code determines which TypeCode methods may legally be invoked on it.

The method kind may be invoked on any TypeCode object.

**Returns:**the TCKind instance indicating the value of the kind field of this TypeCode object

### id

public abstract [String](http://docs.google.com/java/lang/String.html) **id**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Retrieves the RepositoryId globally identifying the type of this TypeCode object.

The method id can be invoked on object reference, structure, union, enumeration, alias, exception, valuetype, boxed valuetype, native, and abstract interface type codes. Object reference, exception, valuetype, boxed valuetype, native, and abstract interface TypeCode objects always have a RepositoryId. Structure, union, enumeration, and alias TypeCode objects obtained from the Interface Repository or the method ORB.create\_operation\_list also always have a RepositoryId. If there is no RepositoryId, the method can return an empty string.

**Returns:**the RepositoryId for this TypeCode object or an empty string if there is no RepositoryID **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind ofTypeCode object

### name

public abstract [String](http://docs.google.com/java/lang/String.html) **name**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Retrieves the simple name identifying this TypeCode object within its enclosing scope. Since names are local to a Repository, the name returned from a TypeCode object may not match the name of the type in any particular Repository, and may even be an empty string.

The method name can be invoked on object reference, structure, union, enumeration, alias, exception, valuetype, boxed valuetype, native, and abstract interface TypeCode objects.

**Returns:**the name identifying this TypeCode object or an empty string **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind ofTypeCode object

### member\_count

public abstract int **member\_count**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Retrieves the number of members in the type described by this TypeCode object.

The method member\_count can be invoked on structure, union, and enumeration TypeCode objects. Java IDL extends the CORBA specification to allow this method to operate on exceptions as well.

**Returns:**the number of members constituting the type described by this TypeCode object **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind of TypeCode object

### member\_name

public abstract [String](http://docs.google.com/java/lang/String.html) **member\_name**(int index)  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html),  
 [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html)

Retrieves the simple name of the member identified by the given index. Since names are local to a Repository, the name returned from a TypeCode object may not match the name of the member in any particular Repository, and may even be an empty string.

The method member\_name can be invoked on structure, union, and enumeration TypeCode objects. Java IDL extends the CORBA specification to allow this method to operate on exceptions as well.

**Parameters:**index - index of the member for which a name is being reqested **Returns:**simple name of the member identified by the index or an empty string **Throws:** [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html) - if the index is equal to or greater than the number of members constituting the type [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind of TypeCode object

### member\_type

public abstract [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) **member\_type**(int index)  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html),  
 [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html)

Retrieves the TypeCode object describing the type of the member identified by the given index.

The method member\_type can be invoked on structure and union TypeCode objects. Java IDL extends the CORBA specification to allow this method to operate on exceptions as well.

**Parameters:**index - index of the member for which type information is begin requested **Returns:**the TypeCode object describing the member at the given index **Throws:** [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html) - if the index is equal to or greater than the number of members constituting the type [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind of TypeCode object

### member\_label

public abstract [Any](http://docs.google.com/org/omg/CORBA/Any.html) **member\_label**(int index)  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html),  
 [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html)

Retrieves the label of the union member identified by the given index. For the default member, the label is the zero octet.

The method member\_label can only be invoked on union TypeCode objects.

**Parameters:**index - index of the union member for which the label is being requested **Returns:**an Any object describing the label of the requested union member or the zero octet for the default member **Throws:** [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html) - if the index is equal to or greater than the number of members constituting the union [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on a non-union TypeCode object

### discriminator\_type

public abstract [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) **discriminator\_type**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns a TypeCode object describing all non-default member labels. The method discriminator\_type can be invoked only on union TypeCode objects.

**Returns:**the TypeCode object describing the non-default member labels **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on a non-union TypeCode object

### default\_index

public abstract int **default\_index**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns the index of the default member, or -1 if there is no default member.

The method default\_index can be invoked only on union TypeCode objects.

**Returns:**the index of the default member, or -1 if there is no default member **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on a non-union TypeCode object

### length

public abstract int **length**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns the number of elements in the type described by this TypeCode object. For strings and sequences, it returns the bound, with zero indicating an unbounded string or sequence. For arrays, it returns the number of elements in the array.

The method length can be invoked on string, sequence, and array TypeCode objects.

**Returns:**the bound for strings and sequences, or the number of elements for arrays **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind of TypeCode object

### content\_type

public abstract [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) **content\_type**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns the TypeCode object representing the IDL type for the members of the object described by this TypeCode object. For sequences and arrays, it returns the element type. For aliases, it returns the original type. Note that multidimensional arrays are represented by nesting TypeCode objects, one per dimension. For boxed valuetypes, it returns the boxed type.

The method content\_type can be invoked on sequence, array, alias, and boxed valuetype TypeCode objects.

**Returns:**a TypeCode object representing the element type for sequences and arrays, the original type for aliases, or the boxed type for boxed valuetypes. **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if the method is invoked on an inappropriate kind of TypeCode object

### fixed\_digits

public abstract short **fixed\_digits**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns the number of digits in the fixed type described by this TypeCode object. For example, the typecode for the number 3000.275d could be fixed<7,3>, where 7 is the precision and 3 is the scale.

**Returns:**the total number of digits **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if this method is invoked on an inappropriate kind of TypeCode object

### fixed\_scale

public abstract short **fixed\_scale**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns the scale of the fixed type described by this TypeCode object. A positive number indicates the number of digits to the right of the decimal point. For example, the number 3000d could have the typecode fixed<4,0>, where the first number is the precision and the second number is the scale. A negative number is also possible and adds zeroes to the left of the decimal point. In this case, fixed<1,-3>, could be the typecode for the number 3000d.

**Returns:**the scale of the fixed type that this TypeCode object describes **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if this method is invoked on an inappropriate kind of TypeCode object

### member\_visibility

public abstract short **member\_visibility**(int index)  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html),  
 [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html)

Returns the constant that indicates the visibility of the member at the given index. This operation can only be invoked on non-boxed value TypeCode objects.

**Parameters:**index - an int indicating the index into the value **Returns:**either PRIVATE\_MEMBER.value or PUBLIC\_MEMBER.value **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if this method is invoked on a non-value type TypeCode object [Bounds](http://docs.google.com/org/omg/CORBA/TypeCodePackage/Bounds.html) - if the given index is out of bounds**See Also:**[CORBA package comments for unimplemented features](http://docs.google.com/package-summary.html#unimpl)

### type\_modifier

public abstract short **type\_modifier**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns a constant indicating the modifier of the value type that this TypeCode object describes. The constant returned must be one of the following: VM\_NONE.value, VM\_ABSTRACT.value, VM\_CUSTOM.value, or VM\_TRUNCATABLE.value,

**Returns:**a constant describing the value type that this TypeCode object describes **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if this method is invoked on a non-value type TypeCode object**See Also:**[CORBA package comments for unimplemented features](http://docs.google.com/package-summary.html#unimpl)

### concrete\_base\_type

public abstract [TypeCode](http://docs.google.com/org/omg/CORBA/TypeCode.html) **concrete\_base\_type**()  
 throws [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html)

Returns the TypeCode object that describes the concrete base type of the value type that this TypeCode object describes. Returns null if it doesn't have a concrete base type.

**Returns:**the TypeCode object that describes the concrete base type of the value type that this TypeCode object describes **Throws:** [BadKind](http://docs.google.com/org/omg/CORBA/TypeCodePackage/BadKind.html) - if this method is invoked on a non-boxed value type TypeCode object**See Also:**[CORBA package comments for unimplemented features](http://docs.google.com/package-summary.html#unimpl)

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/TypeCode.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/omg/CORBA/TRANSIENT.html)   [**NEXT CLASS**](http://docs.google.com/org/omg/CORBA/TypeCodeHolder.html) | [**FRAMES**](http://docs.google.com/index.html?org/omg/CORBA/TypeCode.html)    [**NO FRAMES**](http://docs.google.com/TypeCode.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

[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).