| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Constructor.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/java/lang/reflect/Array.html)   [**NEXT CLASS**](http://docs.google.com/java/lang/reflect/Field.html) | [**FRAMES**](http://docs.google.com/index.html?java/lang/reflect/Constructor.html)    [**NO FRAMES**](http://docs.google.com/Constructor.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [METHOD](#4d34og8) |

## **java.lang.reflect**

Class Constructor<T>

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 [java.lang.reflect.AccessibleObject](http://docs.google.com/java/lang/reflect/AccessibleObject.html)  
 **java.lang.reflect.Constructor<T>**

**Type Parameters:**T - the class in which the constructor is declared **All Implemented Interfaces:** [AnnotatedElement](http://docs.google.com/java/lang/reflect/AnnotatedElement.html), [GenericDeclaration](http://docs.google.com/java/lang/reflect/GenericDeclaration.html), [Member](http://docs.google.com/java/lang/reflect/Member.html)

public final class **Constructor<T>**extends [AccessibleObject](http://docs.google.com/java/lang/reflect/AccessibleObject.html)implements [GenericDeclaration](http://docs.google.com/java/lang/reflect/GenericDeclaration.html), [Member](http://docs.google.com/java/lang/reflect/Member.html)

Constructor provides information about, and access to, a single constructor for a class.

Constructor permits widening conversions to occur when matching the actual parameters to newInstance() with the underlying constructor's formal parameters, but throws an IllegalArgumentException if a narrowing conversion would occur.

**See Also:**[Member](http://docs.google.com/java/lang/reflect/Member.html), [Class](http://docs.google.com/java/lang/Class.html), [Class.getConstructors()](http://docs.google.com/java/lang/Class.html#getConstructors()), [Class.getConstructor(Class[])](http://docs.google.com/java/lang/Class.html#getConstructor(java.lang.Class...)), [Class.getDeclaredConstructors()](http://docs.google.com/java/lang/Class.html#getDeclaredConstructors())

| **Field Summary** | |
| --- | --- |

| **Fields inherited from interface java.lang.reflect.**[**Member**](http://docs.google.com/java/lang/reflect/Member.html) |
| --- |
| [DECLARED](http://docs.google.com/java/lang/reflect/Member.html#DECLARED), [PUBLIC](http://docs.google.com/java/lang/reflect/Member.html#PUBLIC) |

| **Method Summary** | |
| --- | --- |
| boolean | [**equals**](http://docs.google.com/java/lang/reflect/Constructor.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Compares this Constructor against the specified object. |
| | <T extends [Annotation](http://docs.google.com/java/lang/annotation/Annotation.html)>  T | | --- | | [**getAnnotation**](http://docs.google.com/java/lang/reflect/Constructor.html#getAnnotation(java.lang.Class))([Class](http://docs.google.com/java/lang/Class.html)<T> annotationClass)            Returns this element's annotation for the specified type if such an annotation is present, else null. |
| [Annotation](http://docs.google.com/java/lang/annotation/Annotation.html)[] | [**getDeclaredAnnotations**](http://docs.google.com/java/lang/reflect/Constructor.html#getDeclaredAnnotations())()            Returns all annotations that are directly present on this element. |
| [Class](http://docs.google.com/java/lang/Class.html)<[T](http://docs.google.com/java/lang/reflect/Constructor.html)> | [**getDeclaringClass**](http://docs.google.com/java/lang/reflect/Constructor.html#getDeclaringClass())()            Returns the Class object representing the class that declares the constructor represented by this Constructor object. |
| [Class](http://docs.google.com/java/lang/Class.html)<?>[] | [**getExceptionTypes**](http://docs.google.com/java/lang/reflect/Constructor.html#getExceptionTypes())()            Returns an array of Class objects that represent the types of exceptions declared to be thrown by the underlying constructor represented by this Constructor object. |
| [Type](http://docs.google.com/java/lang/reflect/Type.html)[] | [**getGenericExceptionTypes**](http://docs.google.com/java/lang/reflect/Constructor.html#getGenericExceptionTypes())()            Returns an array of Type objects that represent the exceptions declared to be thrown by this Constructor object. |
| [Type](http://docs.google.com/java/lang/reflect/Type.html)[] | [**getGenericParameterTypes**](http://docs.google.com/java/lang/reflect/Constructor.html#getGenericParameterTypes())()            Returns an array of Type objects that represent the formal parameter types, in declaration order, of the method represented by this Constructor object. |
| int | [**getModifiers**](http://docs.google.com/java/lang/reflect/Constructor.html#getModifiers())()            Returns the Java language modifiers for the constructor represented by this Constructor object, as an integer. |
| [String](http://docs.google.com/java/lang/String.html) | [**getName**](http://docs.google.com/java/lang/reflect/Constructor.html#getName())()            Returns the name of this constructor, as a string. |
| [Annotation](http://docs.google.com/java/lang/annotation/Annotation.html)[][] | [**getParameterAnnotations**](http://docs.google.com/java/lang/reflect/Constructor.html#getParameterAnnotations())()            Returns an array of arrays that represent the annotations on the formal parameters, in declaration order, of the method represented by this Constructor object. |
| [Class](http://docs.google.com/java/lang/Class.html)<?>[] | [**getParameterTypes**](http://docs.google.com/java/lang/reflect/Constructor.html#getParameterTypes())()            Returns an array of Class objects that represent the formal parameter types, in declaration order, of the constructor represented by this Constructor object. |
| [TypeVariable](http://docs.google.com/java/lang/reflect/TypeVariable.html)<[Constructor](http://docs.google.com/java/lang/reflect/Constructor.html)<[T](http://docs.google.com/java/lang/reflect/Constructor.html)>>[] | [**getTypeParameters**](http://docs.google.com/java/lang/reflect/Constructor.html#getTypeParameters())()            Returns an array of TypeVariable objects that represent the type variables declared by the generic declaration represented by this GenericDeclaration object, in declaration order. |
| int | [**hashCode**](http://docs.google.com/java/lang/reflect/Constructor.html#hashCode())()            Returns a hashcode for this Constructor. |
| boolean | [**isSynthetic**](http://docs.google.com/java/lang/reflect/Constructor.html#isSynthetic())()            Returns true if this constructor is a synthetic constructor; returns false otherwise. |
| boolean | [**isVarArgs**](http://docs.google.com/java/lang/reflect/Constructor.html#isVarArgs())()            Returns true if this constructor was declared to take a variable number of arguments; returns false otherwise. |
| [T](http://docs.google.com/java/lang/reflect/Constructor.html) | [**newInstance**](http://docs.google.com/java/lang/reflect/Constructor.html#newInstance(java.lang.Object...))([Object](http://docs.google.com/java/lang/Object.html)... initargs)            Uses the constructor represented by this Constructor object to create and initialize a new instance of the constructor's declaring class, with the specified initialization parameters. |
| [String](http://docs.google.com/java/lang/String.html) | [**toGenericString**](http://docs.google.com/java/lang/reflect/Constructor.html#toGenericString())()            Returns a string describing this Constructor, including type parameters. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/java/lang/reflect/Constructor.html#toString())()            Returns a string describing this Constructor. |

| **Methods inherited from class java.lang.reflect.**[**AccessibleObject**](http://docs.google.com/java/lang/reflect/AccessibleObject.html) |
| --- |
| [getAnnotations](http://docs.google.com/java/lang/reflect/AccessibleObject.html#getAnnotations()), [isAccessible](http://docs.google.com/java/lang/reflect/AccessibleObject.html#isAccessible()), [isAnnotationPresent](http://docs.google.com/java/lang/reflect/AccessibleObject.html#isAnnotationPresent(java.lang.Class)), [setAccessible](http://docs.google.com/java/lang/reflect/AccessibleObject.html#setAccessible(java.lang.reflect.AccessibleObject%5B%5D,%20boolean)), [setAccessible](http://docs.google.com/java/lang/reflect/AccessibleObject.html#setAccessible(boolean)) |

| **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()), [finalize](http://docs.google.com/java/lang/Object.html#finalize()), [getClass](http://docs.google.com/java/lang/Object.html#getClass()), [notify](http://docs.google.com/java/lang/Object.html#notify()), [notifyAll](http://docs.google.com/java/lang/Object.html#notifyAll()), [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)) |

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

### getDeclaringClass

public [Class](http://docs.google.com/java/lang/Class.html)<[T](http://docs.google.com/java/lang/reflect/Constructor.html)> **getDeclaringClass**()

Returns the Class object representing the class that declares the constructor represented by this Constructor object.

**Specified by:**[getDeclaringClass](http://docs.google.com/java/lang/reflect/Member.html#getDeclaringClass()) in interface [Member](http://docs.google.com/java/lang/reflect/Member.html) **Returns:**an object representing the declaring class of the underlying member

### getName

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

Returns the name of this constructor, as a string. This is always the same as the simple name of the constructor's declaring class.

**Specified by:**[getName](http://docs.google.com/java/lang/reflect/Member.html#getName()) in interface [Member](http://docs.google.com/java/lang/reflect/Member.html) **Returns:**the simple name of the underlying member

### getModifiers

public int **getModifiers**()

Returns the Java language modifiers for the constructor represented by this Constructor object, as an integer. The Modifier class should be used to decode the modifiers.

**Specified by:**[getModifiers](http://docs.google.com/java/lang/reflect/Member.html#getModifiers()) in interface [Member](http://docs.google.com/java/lang/reflect/Member.html) **Returns:**the Java language modifiers for the underlying member**See Also:**[Modifier](http://docs.google.com/java/lang/reflect/Modifier.html)

### getTypeParameters

public [TypeVariable](http://docs.google.com/java/lang/reflect/TypeVariable.html)<[Constructor](http://docs.google.com/java/lang/reflect/Constructor.html)<[T](http://docs.google.com/java/lang/reflect/Constructor.html)>>[] **getTypeParameters**()

Returns an array of TypeVariable objects that represent the type variables declared by the generic declaration represented by this GenericDeclaration object, in declaration order. Returns an array of length 0 if the underlying generic declaration declares no type variables.

**Specified by:**[getTypeParameters](http://docs.google.com/java/lang/reflect/GenericDeclaration.html#getTypeParameters()) in interface [GenericDeclaration](http://docs.google.com/java/lang/reflect/GenericDeclaration.html) **Returns:**an array of TypeVariable objects that represent the type variables declared by this generic declaration **Throws:** [GenericSignatureFormatError](http://docs.google.com/java/lang/reflect/GenericSignatureFormatError.html) - if the generic signature of this generic declaration does not conform to the format specified in the Java Virtual Machine Specification, 3rd edition**Since:** 1.5

### getParameterTypes

public [Class](http://docs.google.com/java/lang/Class.html)<?>[] **getParameterTypes**()

Returns an array of Class objects that represent the formal parameter types, in declaration order, of the constructor represented by this Constructor object. Returns an array of length 0 if the underlying constructor takes no parameters.

**Returns:**the parameter types for the constructor this object represents

### getGenericParameterTypes

public [Type](http://docs.google.com/java/lang/reflect/Type.html)[] **getGenericParameterTypes**()

Returns an array of Type objects that represent the formal parameter types, in declaration order, of the method represented by this Constructor object. Returns an array of length 0 if the underlying method takes no parameters.

If a formal parameter type is a parameterized type, the Type object returned for it must accurately reflect the actual type parameters used in the source code.

If a formal parameter type is a type variable or a parameterized type, it is created. Otherwise, it is resolved.

**Returns:**an array of Types that represent the formal parameter types of the underlying method, in declaration order **Throws:** [GenericSignatureFormatError](http://docs.google.com/java/lang/reflect/GenericSignatureFormatError.html) - if the generic method signature does not conform to the format specified in the Java Virtual Machine Specification, 3rd edition [TypeNotPresentException](http://docs.google.com/java/lang/TypeNotPresentException.html) - if any of the parameter types of the underlying method refers to a non-existent type declaration [MalformedParameterizedTypeException](http://docs.google.com/java/lang/reflect/MalformedParameterizedTypeException.html) - if any of the underlying method's parameter types refer to a parameterized type that cannot be instantiated for any reason**Since:** 1.5

### getExceptionTypes

public [Class](http://docs.google.com/java/lang/Class.html)<?>[] **getExceptionTypes**()

Returns an array of Class objects that represent the types of exceptions declared to be thrown by the underlying constructor represented by this Constructor object. Returns an array of length 0 if the constructor declares no exceptions in its throws clause.

**Returns:**the exception types declared as being thrown by the constructor this object represents

### getGenericExceptionTypes

public [Type](http://docs.google.com/java/lang/reflect/Type.html)[] **getGenericExceptionTypes**()

Returns an array of Type objects that represent the exceptions declared to be thrown by this Constructor object. Returns an array of length 0 if the underlying method declares no exceptions in its throws clause.

If an exception type is a parameterized type, the Type object returned for it must accurately reflect the actual type parameters used in the source code.

If an exception type is a type variable or a parameterized type, it is created. Otherwise, it is resolved.

**Returns:**an array of Types that represent the exception types thrown by the underlying method **Throws:** [GenericSignatureFormatError](http://docs.google.com/java/lang/reflect/GenericSignatureFormatError.html) - if the generic method signature does not conform to the format specified in the Java Virtual Machine Specification, 3rd edition [TypeNotPresentException](http://docs.google.com/java/lang/TypeNotPresentException.html) - if the underlying method's throws clause refers to a non-existent type declaration [MalformedParameterizedTypeException](http://docs.google.com/java/lang/reflect/MalformedParameterizedTypeException.html) - if the underlying method's throws clause refers to a parameterized type that cannot be instantiated for any reason**Since:** 1.5

### equals

public boolean **equals**([Object](http://docs.google.com/java/lang/Object.html) obj)

Compares this Constructor against the specified object. Returns true if the objects are the same. Two Constructor objects are the same if they were declared by the same class and have the same formal parameter types.

**Overrides:**[equals](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) in class [Object](http://docs.google.com/java/lang/Object.html) **Parameters:**obj - the reference object with which to compare. **Returns:**true if this object is the same as the obj argument; false otherwise.**See Also:**[Object.hashCode()](http://docs.google.com/java/lang/Object.html#hashCode()), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### hashCode

public int **hashCode**()

Returns a hashcode for this Constructor. The hashcode is the same as the hashcode for the underlying constructor's declaring class name.

**Overrides:**[hashCode](http://docs.google.com/java/lang/Object.html#hashCode()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a hash code value for this object.**See Also:**[Object.equals(java.lang.Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)), [Hashtable](http://docs.google.com/java/util/Hashtable.html)

### toString

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

Returns a string describing this Constructor. The string is formatted as the constructor access modifiers, if any, followed by the fully-qualified name of the declaring class, followed by a parenthesized, comma-separated list of the constructor's formal parameter types. For example:

public java.util.Hashtable(int,float)

The only possible modifiers for constructors are the access modifiers public, protected or private. Only one of these may appear, or none if the constructor has default (package) access.

**Overrides:**[toString](http://docs.google.com/java/lang/Object.html#toString()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a string representation of the object.

### toGenericString

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

Returns a string describing this Constructor, including type parameters. The string is formatted as the constructor access modifiers, if any, followed by an angle-bracketed comma separated list of the constructor's type parameters, if any, followed by the fully-qualified name of the declaring class, followed by a parenthesized, comma-separated list of the constructor's generic formal parameter types. A space is used to separate access modifiers from one another and from the type parameters or return type. If there are no type parameters, the type parameter list is elided; if the type parameter list is present, a space separates the list from the class name. If the constructor is declared to throw exceptions, the parameter list is followed by a space, followed by the word "throws" followed by a comma-separated list of the thrown exception types.

The only possible modifiers for constructors are the access modifiers public, protected or private. Only one of these may appear, or none if the constructor has default (package) access.

**Returns:**a string describing this Constructor, include type parameters**Since:** 1.5

### newInstance

public [T](http://docs.google.com/java/lang/reflect/Constructor.html) **newInstance**([Object](http://docs.google.com/java/lang/Object.html)... initargs)  
 throws [InstantiationException](http://docs.google.com/java/lang/InstantiationException.html),  
 [IllegalAccessException](http://docs.google.com/java/lang/IllegalAccessException.html),  
 [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html),  
 [InvocationTargetException](http://docs.google.com/java/lang/reflect/InvocationTargetException.html)

Uses the constructor represented by this Constructor object to create and initialize a new instance of the constructor's declaring class, with the specified initialization parameters. Individual parameters are automatically unwrapped to match primitive formal parameters, and both primitive and reference parameters are subject to method invocation conversions as necessary.

If the number of formal parameters required by the underlying constructor is 0, the supplied initargs array may be of length 0 or null.

If the constructor's declaring class is an inner class in a non-static context, the first argument to the constructor needs to be the enclosing instance; see *The Java Language Specification*, section 15.9.3.

If the required access and argument checks succeed and the instantiation will proceed, the constructor's declaring class is initialized if it has not already been initialized.

If the constructor completes normally, returns the newly created and initialized instance.

**Parameters:**initargs - array of objects to be passed as arguments to the constructor call; values of primitive types are wrapped in a wrapper object of the appropriate type (e.g. a float in a [Float](http://docs.google.com/java/lang/Float.html)) **Returns:**a new object created by calling the constructor this object represents **Throws:** [IllegalAccessException](http://docs.google.com/java/lang/IllegalAccessException.html) - if this Constructor object enforces Java language access control and the underlying constructor is inaccessible. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the number of actual and formal parameters differ; if an unwrapping conversion for primitive arguments fails; or if, after possible unwrapping, a parameter value cannot be converted to the corresponding formal parameter type by a method invocation conversion; if this constructor pertains to an enum type. [InstantiationException](http://docs.google.com/java/lang/InstantiationException.html) - if the class that declares the underlying constructor represents an abstract class. [InvocationTargetException](http://docs.google.com/java/lang/reflect/InvocationTargetException.html) - if the underlying constructor throws an exception. [ExceptionInInitializerError](http://docs.google.com/java/lang/ExceptionInInitializerError.html) - if the initialization provoked by this method fails.

### isVarArgs

public boolean **isVarArgs**()

Returns true if this constructor was declared to take a variable number of arguments; returns false otherwise.

**Returns:**true if an only if this constructor was declared to take a variable number of arguments.**Since:** 1.5

### isSynthetic

public boolean **isSynthetic**()

Returns true if this constructor is a synthetic constructor; returns false otherwise.

**Specified by:**[isSynthetic](http://docs.google.com/java/lang/reflect/Member.html#isSynthetic()) in interface [Member](http://docs.google.com/java/lang/reflect/Member.html) **Returns:**true if and only if this constructor is a synthetic constructor as defined by the Java Language Specification.**Since:** 1.5

### getAnnotation

public <T extends [Annotation](http://docs.google.com/java/lang/annotation/Annotation.html)> T **getAnnotation**([Class](http://docs.google.com/java/lang/Class.html)<T> annotationClass)

**Description copied from interface:** [**AnnotatedElement**](http://docs.google.com/java/lang/reflect/AnnotatedElement.html#getAnnotation(java.lang.Class)) Returns this element's annotation for the specified type if such an annotation is present, else null.

**Specified by:**[getAnnotation](http://docs.google.com/java/lang/reflect/AnnotatedElement.html#getAnnotation(java.lang.Class)) in interface [AnnotatedElement](http://docs.google.com/java/lang/reflect/AnnotatedElement.html)**Overrides:**[getAnnotation](http://docs.google.com/java/lang/reflect/AccessibleObject.html#getAnnotation(java.lang.Class)) in class [AccessibleObject](http://docs.google.com/java/lang/reflect/AccessibleObject.html) **Parameters:**annotationClass - the Class object corresponding to the annotation type **Returns:**this element's annotation for the specified annotation type if present on this element, else null **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if the given annotation class is null**Since:** 1.5

### getDeclaredAnnotations

public [Annotation](http://docs.google.com/java/lang/annotation/Annotation.html)[] **getDeclaredAnnotations**()

**Description copied from interface:** [**AnnotatedElement**](http://docs.google.com/java/lang/reflect/AnnotatedElement.html#getDeclaredAnnotations()) Returns all annotations that are directly present on this element. Unlike the other methods in this interface, this method ignores inherited annotations. (Returns an array of length zero if no annotations are directly present on this element.) The caller of this method is free to modify the returned array; it will have no effect on the arrays returned to other callers.

**Specified by:**[getDeclaredAnnotations](http://docs.google.com/java/lang/reflect/AnnotatedElement.html#getDeclaredAnnotations()) in interface [AnnotatedElement](http://docs.google.com/java/lang/reflect/AnnotatedElement.html)**Overrides:**[getDeclaredAnnotations](http://docs.google.com/java/lang/reflect/AccessibleObject.html#getDeclaredAnnotations()) in class [AccessibleObject](http://docs.google.com/java/lang/reflect/AccessibleObject.html) **Returns:**All annotations directly present on this element**Since:** 1.5

### getParameterAnnotations

public [Annotation](http://docs.google.com/java/lang/annotation/Annotation.html)[][] **getParameterAnnotations**()

Returns an array of arrays that represent the annotations on the formal parameters, in declaration order, of the method represented by this Constructor object. (Returns an array of length zero if the underlying method is parameterless. If the method has one or more parameters, a nested array of length zero is returned for each parameter with no annotations.) The annotation objects contained in the returned arrays are serializable. The caller of this method is free to modify the returned arrays; it will have no effect on the arrays returned to other callers.

**Returns:**an array of arrays that represent the annotations on the formal parameters, in declaration order, of the method represented by this Constructor object**Since:** 1.5

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Constructor.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/java/lang/reflect/Array.html)   [**NEXT CLASS**](http://docs.google.com/java/lang/reflect/Field.html) | [**FRAMES**](http://docs.google.com/index.html?java/lang/reflect/Constructor.html)    [**NO FRAMES**](http://docs.google.com/Constructor.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | CONSTR | [METHOD](#tyjcwt) | DETAIL: FIELD | CONSTR | [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).