| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ImmutableDescriptor.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/management/DynamicMBean.html)   [**NEXT CLASS**](http://docs.google.com/javax/management/InstanceAlreadyExistsException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/ImmutableDescriptor.html)    [**NO FRAMES**](http://docs.google.com/ImmutableDescriptor.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#2s8eyo1) | [METHOD](#lnxbz9) |

## **javax.management**

Class ImmutableDescriptor

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.management.ImmutableDescriptor**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Descriptor](http://docs.google.com/javax/management/Descriptor.html)

public class **ImmutableDescriptor**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Descriptor](http://docs.google.com/javax/management/Descriptor.html)

An immutable descriptor.

**Since:** 1.6 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#javax.management.ImmutableDescriptor)

| **Field Summary** | |
| --- | --- |
| static [ImmutableDescriptor](http://docs.google.com/javax/management/ImmutableDescriptor.html) | [**EMPTY\_DESCRIPTOR**](http://docs.google.com/javax/management/ImmutableDescriptor.html#EMPTY_DESCRIPTOR)            An empty descriptor. |

| **Constructor Summary** | |
| --- | --- |
| [**ImmutableDescriptor**](http://docs.google.com/javax/management/ImmutableDescriptor.html#ImmutableDescriptor(java.util.Map))([Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> fields)            Construct a descriptor where the names and values of the fields are the keys and values of the given Map. |
| [**ImmutableDescriptor**](http://docs.google.com/javax/management/ImmutableDescriptor.html#ImmutableDescriptor(java.lang.String...))([String](http://docs.google.com/java/lang/String.html)... fields)            Construct a descriptor containing the given fields. |
| [**ImmutableDescriptor**](http://docs.google.com/javax/management/ImmutableDescriptor.html#ImmutableDescriptor(java.lang.String%5B%5D,%20java.lang.Object%5B%5D))([String](http://docs.google.com/java/lang/String.html)[] fieldNames, [Object](http://docs.google.com/java/lang/Object.html)[] fieldValues)            Construct a descriptor containing the given fields and values. |

| **Method Summary** | |
| --- | --- |
| [Descriptor](http://docs.google.com/javax/management/Descriptor.html) | [**clone**](http://docs.google.com/javax/management/ImmutableDescriptor.html#clone())()            Returns a descriptor which is equal to this descriptor. |
| boolean | [**equals**](http://docs.google.com/javax/management/ImmutableDescriptor.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) o)            Compares this descriptor to the given object. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getFieldNames**](http://docs.google.com/javax/management/ImmutableDescriptor.html#getFieldNames())()            Returns all the field names in the descriptor. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getFields**](http://docs.google.com/javax/management/ImmutableDescriptor.html#getFields())()            Returns all of the fields contained in this descriptor as a string array. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getFieldValue**](http://docs.google.com/javax/management/ImmutableDescriptor.html#getFieldValue(java.lang.String))([String](http://docs.google.com/java/lang/String.html) fieldName)            Returns the value for a specific field name, or null if no value is present for that name. |
| [Object](http://docs.google.com/java/lang/Object.html)[] | [**getFieldValues**](http://docs.google.com/javax/management/ImmutableDescriptor.html#getFieldValues(java.lang.String...))([String](http://docs.google.com/java/lang/String.html)... fieldNames)            Returns all the field values in the descriptor as an array of Objects. |
| int | [**hashCode**](http://docs.google.com/javax/management/ImmutableDescriptor.html#hashCode())()            Returns the hash code value for this descriptor. |
| boolean | [**isValid**](http://docs.google.com/javax/management/ImmutableDescriptor.html#isValid())()            Returns true if all of the fields have legal values given their names. |
| void | [**removeField**](http://docs.google.com/javax/management/ImmutableDescriptor.html#removeField(java.lang.String))([String](http://docs.google.com/java/lang/String.html) fieldName)            Removes a field from the descriptor. |
| void | [**setField**](http://docs.google.com/javax/management/ImmutableDescriptor.html#setField(java.lang.String,%20java.lang.Object))([String](http://docs.google.com/java/lang/String.html) fieldName, [Object](http://docs.google.com/java/lang/Object.html) fieldValue)            This operation is unsupported since this class is immutable. |
| void | [**setFields**](http://docs.google.com/javax/management/ImmutableDescriptor.html#setFields(java.lang.String%5B%5D,%20java.lang.Object%5B%5D))([String](http://docs.google.com/java/lang/String.html)[] fieldNames, [Object](http://docs.google.com/java/lang/Object.html)[] fieldValues)            This operation is unsupported since this class is immutable. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/javax/management/ImmutableDescriptor.html#toString())()            Returns a string representation of the object. |
| static [ImmutableDescriptor](http://docs.google.com/javax/management/ImmutableDescriptor.html) | [**union**](http://docs.google.com/javax/management/ImmutableDescriptor.html#union(javax.management.Descriptor...))([Descriptor](http://docs.google.com/javax/management/Descriptor.html)... descriptors)            Return an ImmutableDescriptor whose contents are the union of the given descriptors. |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [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)) |

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

### EMPTY\_DESCRIPTOR

public static final [ImmutableDescriptor](http://docs.google.com/javax/management/ImmutableDescriptor.html) **EMPTY\_DESCRIPTOR**

An empty descriptor.

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

### ImmutableDescriptor

public **ImmutableDescriptor**([String](http://docs.google.com/java/lang/String.html)[] fieldNames,  
 [Object](http://docs.google.com/java/lang/Object.html)[] fieldValues)

Construct a descriptor containing the given fields and values.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if either array is null, or if the arrays have different sizes, or if a field name is null or empty, or if the same field name appears more than once.

### ImmutableDescriptor

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

Construct a descriptor containing the given fields. Each String must be of the form fieldName=fieldValue. The field name ends at the first = character; for example if the String is a=b=c then the field name is a and its value is b=c.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the parameter is null, or if a field name is empty, or if the same field name appears more than once, or if one of the strings does not contain an = character.

### ImmutableDescriptor

public **ImmutableDescriptor**([Map](http://docs.google.com/java/util/Map.html)<[String](http://docs.google.com/java/lang/String.html),?> fields)

Construct a descriptor where the names and values of the fields are the keys and values of the given Map.

**Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the parameter is null, or if a field name is null or empty, or if the same field name appears more than once (which can happen because field names are not case sensitive).

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

### union

public static [ImmutableDescriptor](http://docs.google.com/javax/management/ImmutableDescriptor.html) **union**([Descriptor](http://docs.google.com/javax/management/Descriptor.html)... descriptors)

Return an ImmutableDescriptor whose contents are the union of the given descriptors. Every field name that appears in any of the descriptors will appear in the result with the value that it has when the method is called. Subsequent changes to any of the descriptors do not affect the ImmutableDescriptor returned here.

In the simplest case, there is only one descriptor and the returned ImmutableDescriptor is a copy of its fields at the time this method is called:

Descriptor d = something();  
 ImmutableDescriptor copy = ImmutableDescriptor.union(d);

**Parameters:**descriptors - the descriptors to be combined. Any of the descriptors can be null, in which case it is skipped. **Returns:**an ImmutableDescriptor that is the union of the given descriptors. The returned object may be identical to one of the input descriptors if it is an ImmutableDescriptor that contains all of the required fields. **Throws:** [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if two Descriptors contain the same field name with different associated values. Primitive array values are considered the same if they are of the same type with the same elements. Object array values are considered the same if [Arrays.deepEquals(Object[],Object[])](http://docs.google.com/java/util/Arrays.html#deepEquals(java.lang.Object%5B%5D,%20java.lang.Object%5B%5D)) returns true.

### getFieldValue

public final [Object](http://docs.google.com/java/lang/Object.html) **getFieldValue**([String](http://docs.google.com/java/lang/String.html) fieldName)

**Description copied from interface:** [**Descriptor**](http://docs.google.com/javax/management/Descriptor.html#getFieldValue(java.lang.String)) Returns the value for a specific field name, or null if no value is present for that name.

**Specified by:**[getFieldValue](http://docs.google.com/javax/management/Descriptor.html#getFieldValue(java.lang.String)) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Parameters:**fieldName - the field name. **Returns:**the corresponding value, or null if the field is not present.

### getFields

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

**Description copied from interface:** [**Descriptor**](http://docs.google.com/javax/management/Descriptor.html#getFields()) Returns all of the fields contained in this descriptor as a string array.

**Specified by:**[getFields](http://docs.google.com/javax/management/Descriptor.html#getFields()) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Returns:**String array of fields in the format *fieldName=fieldValue*

If the value of a field is not a String, then the toString() method will be called on it and the returned value, enclosed in parentheses, used as the value for the field in the returned array. If the value of a field is null, then the value of the field in the returned array will be empty. If the descriptor is empty, you will get an empty array.**See Also:**[Descriptor.setFields(java.lang.String[], java.lang.Object[])](http://docs.google.com/javax/management/Descriptor.html#setFields(java.lang.String%5B%5D,%20java.lang.Object%5B%5D))

### getFieldValues

public final [Object](http://docs.google.com/java/lang/Object.html)[] **getFieldValues**([String](http://docs.google.com/java/lang/String.html)... fieldNames)

**Description copied from interface:** [**Descriptor**](http://docs.google.com/javax/management/Descriptor.html#getFieldValues(java.lang.String...)) Returns all the field values in the descriptor as an array of Objects. The returned values are in the same order as the fieldNames String array parameter.

**Specified by:**[getFieldValues](http://docs.google.com/javax/management/Descriptor.html#getFieldValues(java.lang.String...)) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Parameters:**fieldNames - String array of the names of the fields that the values should be returned for. If the array is empty then an empty array will be returned. If the array is null then all values will be returned, as if the parameter were the array returned by [Descriptor.getFieldNames()](http://docs.google.com/javax/management/Descriptor.html#getFieldNames()). If a field name in the array does not exist, including the case where it is null or the empty string, then null is returned for the matching array element being returned. **Returns:**Object array of field values. If the list of fieldNames is empty, you will get an empty array.

### getFieldNames

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

**Description copied from interface:** [**Descriptor**](http://docs.google.com/javax/management/Descriptor.html#getFieldNames()) Returns all the field names in the descriptor.

**Specified by:**[getFieldNames](http://docs.google.com/javax/management/Descriptor.html#getFieldNames()) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Returns:**String array of field names. If the descriptor is empty, you will get an empty array.

### equals

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

Compares this descriptor to the given object. The objects are equal if the given object is also a Descriptor, and if the two Descriptors have the same field names (possibly differing in case) and the same associated values. The respective values for a field in the two Descriptors are equal if the following conditions hold:

* If one value is null then the other must be too.
* If one value is a primitive array then the other must be a primitive array of the same type with the same elements.
* If one value is an object array then the other must be too and [Arrays.deepEquals(Object[],Object[])](http://docs.google.com/java/util/Arrays.html#deepEquals(java.lang.Object%5B%5D,%20java.lang.Object%5B%5D)) must return true.
* Otherwise [Object.equals(Object)](http://docs.google.com/java/lang/Object.html#equals(java.lang.Object)) must return true.

**Specified by:**[equals](http://docs.google.com/javax/management/Descriptor.html#equals(java.lang.Object)) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html)**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:**o - the object to compare with. **Returns:**true if the objects are the same; 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 the hash code value for this descriptor. The hash code is computed as the sum of the hash codes for each field in the descriptor. The hash code of a field with name n and value v is n.toLowerCase().hashCode() ^ h. Here h is the hash code of v, computed as follows:

* If v is null then h is 0.
* If v is a primitive array then h is computed using the appropriate overloading of java.util.Arrays.hashCode.
* If v is an object array then h is computed using [Arrays.deepHashCode(Object[])](http://docs.google.com/java/util/Arrays.html#deepHashCode(java.lang.Object%5B%5D)).
* Otherwise h is v.hashCode().

**Specified by:**[hashCode](http://docs.google.com/javax/management/Descriptor.html#hashCode()) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html)**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**()

**Description copied from class:** [**Object**](http://docs.google.com/java/lang/Object.html#toString()) Returns a string representation of the object. In general, the toString method returns a string that "textually represents" this object. The result should be a concise but informative representation that is easy for a person to read. It is recommended that all subclasses override this method.

The toString method for class Object returns a string consisting of the name of the class of which the object is an instance, the at-sign character `@', and the unsigned hexadecimal representation of the hash code of the object. In other words, this method returns a string equal to the value of:

getClass().getName() + '@' + Integer.toHexString(hashCode())

**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.

### isValid

public boolean **isValid**()

Returns true if all of the fields have legal values given their names. This method always returns true, but a subclass can override it to return false when appropriate.

**Specified by:**[isValid](http://docs.google.com/javax/management/Descriptor.html#isValid()) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Returns:**true if the values are legal. **Throws:** [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) - if the validity checking fails. The method returns false if the descriptor is not valid, but throws this exception if the attempt to determine validity fails.

### clone

public [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **clone**()

Returns a descriptor which is equal to this descriptor. Changes to the returned descriptor will have no effect on this descriptor, and vice versa.

This method returns the object on which it is called. A subclass can override it to return another object provided the contract is respected.

**Specified by:**[clone](http://docs.google.com/javax/management/Descriptor.html#clone()) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html)**Overrides:**[clone](http://docs.google.com/java/lang/Object.html#clone()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**a clone of this instance. **Throws:** [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) - for illegal value for field Names or field Values. If the descriptor construction fails for any reason, this exception will be thrown.**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### setFields

public final void **setFields**([String](http://docs.google.com/java/lang/String.html)[] fieldNames,  
 [Object](http://docs.google.com/java/lang/Object.html)[] fieldValues)  
 throws [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html)

This operation is unsupported since this class is immutable. If this call would change a mutable descriptor with the same contents, then a [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) wrapping an [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) is thrown. Otherwise, the behavior is the same as it would be for a mutable descriptor: either an exception is thrown because of illegal parameters, or there is no effect.

**Specified by:**[setFields](http://docs.google.com/javax/management/Descriptor.html#setFields(java.lang.String%5B%5D,%20java.lang.Object%5B%5D)) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Parameters:**fieldNames - String array of field names. The array and array elements cannot be null.fieldValues - Object array of the corresponding field values. The array cannot be null. Elements of the array can be null. **Throws:** [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) - if the change fails for any reason. Wrapped exception is [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) if fieldNames or fieldValues is null, or if the arrays are of different lengths, or if there is an illegal value in one of them. Wrapped exception is [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) if the descriptor is immutable, and the call would change its contents.**See Also:**[Descriptor.getFields()](http://docs.google.com/javax/management/Descriptor.html#getFields())

### setField

public final void **setField**([String](http://docs.google.com/java/lang/String.html) fieldName,  
 [Object](http://docs.google.com/java/lang/Object.html) fieldValue)  
 throws [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html)

This operation is unsupported since this class is immutable. If this call would change a mutable descriptor with the same contents, then a [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) wrapping an [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) is thrown. Otherwise, the behavior is the same as it would be for a mutable descriptor: either an exception is thrown because of illegal parameters, or there is no effect.

**Specified by:**[setField](http://docs.google.com/javax/management/Descriptor.html#setField(java.lang.String,%20java.lang.Object)) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Parameters:**fieldName - The field name to be set. Cannot be null or empty.fieldValue - The field value to be set for the field name. Can be null if that is a valid value for the field. **Throws:** [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) - if the field name or field value is illegal (wrapped exception is [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html)); or if the descriptor is immutable (wrapped exception is [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html)).

### removeField

public final void **removeField**([String](http://docs.google.com/java/lang/String.html) fieldName)

Removes a field from the descriptor.

**Specified by:**[removeField](http://docs.google.com/javax/management/Descriptor.html#removeField(java.lang.String)) in interface [Descriptor](http://docs.google.com/javax/management/Descriptor.html) **Parameters:**fieldName - String name of the field to be removed. If the field name is illegal or the field is not found, no exception is thrown. **Throws:** [RuntimeOperationsException](http://docs.google.com/javax/management/RuntimeOperationsException.html) - if a field of the given name exists and the descriptor is immutable. The wrapped exception will be an [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html).

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ImmutableDescriptor.html) | [**Tree**](http://docs.google.com/package-tree.html) | [**Deprecated**](http://docs.google.com/deprecated-list.html) | [**Index**](http://docs.google.com/index-files/index-1.html) | [**Help**](http://docs.google.com/help-doc.html) | | --- | --- | --- | --- | --- | --- | --- | --- | | | ***Java™ Platform***  ***Standard Ed. 6*** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| [**PREV CLASS**](http://docs.google.com/javax/management/DynamicMBean.html)   [**NEXT CLASS**](http://docs.google.com/javax/management/InstanceAlreadyExistsException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/management/ImmutableDescriptor.html)    [**NO FRAMES**](http://docs.google.com/ImmutableDescriptor.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#2s8eyo1) | [METHOD](#lnxbz9) |

[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).