| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/LdapName.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/naming/ldap/LdapContext.html)   [**NEXT CLASS**](http://docs.google.com/javax/naming/ldap/LdapReferralException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/naming/ldap/LdapName.html)    [**NO FRAMES**](http://docs.google.com/LdapName.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **javax.naming.ldap**

Class LdapName

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.naming.ldap.LdapName**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html), [Comparable](http://docs.google.com/java/lang/Comparable.html)<[Object](http://docs.google.com/java/lang/Object.html)>, [Name](http://docs.google.com/javax/naming/Name.html)

public class **LdapName**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Name](http://docs.google.com/javax/naming/Name.html)

This class represents a distinguished name as specified by [RFC 2253](http://ietf.org//rfc/rfc2253.txt). A distinguished name, or DN, is composed of an ordered list of components called *relative distinguished name*s, or RDNs. Details of a DN's syntax are described in RFC 2253.

This class resolves a few ambiguities found in RFC 2253 as follows:

* RFC 2253 leaves the term "whitespace" undefined. The ASCII space character 0x20 (" ") is used in its place.
* Whitespace is allowed on either side of ',', ';', '=', and '+'. Such whitespace is accepted but not generated by this code, and is ignored when comparing names.
* AttributeValue strings containing '=' or non-leading '#' characters (unescaped) are accepted.

String names passed to LdapName or returned by it use the full Unicode character set. They may also contain characters encoded into UTF-8 with each octet represented by a three-character substring such as "\\B4". They may not, however, contain characters encoded into UTF-8 with each octet represented by a single character in the string: the meaning would be ambiguous.

LdapName will properly parse all valid names, but does not attempt to detect all possible violations when parsing invalid names. It is "generous" in accepting invalid names. The "validity" of a name is determined ultimately when it is supplied to an LDAP server, which may accept or reject the name based on factors such as its schema information and interoperability considerations.

When names are tested for equality, attribute types, both binary and string values, are case-insensitive. String values with different but equivalent usage of quoting, escaping, or UTF8-hex-encoding are considered equal. The order of components in multi-valued RDNs (such as "ou=Sales+cn=Bob") is not significant.

The components of a LDAP name, that is, RDNs, are numbered. The indexes of a LDAP name with n RDNs range from 0 to n-1. This range may be written as [0,n). The right most RDN is at index 0, and the left most RDN is at index n-1. For example, the distinguished name: "CN=Steve Kille, O=Isode Limited, C=GB" is numbered in the following sequence ranging from 0 to 2: {C=GB, O=Isode Limited, CN=Steve Kille}. An empty LDAP name is represented by an empty RDN list.

Concurrent multithreaded read-only access of an instance of LdapName need not be synchronized.

Unless otherwise noted, the behavior of passing a null argument to a constructor or method in this class will cause a NullPointerException to be thrown.

**Since:** 1.5 **See Also:**[Serialized Form](http://docs.google.com/serialized-form.html#javax.naming.ldap.LdapName)

| **Constructor Summary** | |
| --- | --- |
| [**LdapName**](http://docs.google.com/javax/naming/ldap/LdapName.html#LdapName(java.util.List))([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> rdns)            Constructs an LDAP name given its parsed RDN components. |
| [**LdapName**](http://docs.google.com/javax/naming/ldap/LdapName.html#LdapName(java.lang.String))([String](http://docs.google.com/java/lang/String.html) name)            Constructs an LDAP name from the given distinguished name. |

| **Method Summary** | |
| --- | --- |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**add**](http://docs.google.com/javax/naming/ldap/LdapName.html#add(int,%20javax.naming.ldap.Rdn))(int posn, [Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html) comp)            Adds a single RDN at a specified position within this LDAP name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**add**](http://docs.google.com/javax/naming/ldap/LdapName.html#add(int,%20java.lang.String))(int posn, [String](http://docs.google.com/java/lang/String.html) comp)            Adds a single component at a specified position within this LDAP name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**add**](http://docs.google.com/javax/naming/ldap/LdapName.html#add(javax.naming.ldap.Rdn))([Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html) comp)            Adds a single RDN to the end of this LDAP name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**add**](http://docs.google.com/javax/naming/ldap/LdapName.html#add(java.lang.String))([String](http://docs.google.com/java/lang/String.html) comp)            Adds a single component to the end of this LDAP name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**addAll**](http://docs.google.com/javax/naming/ldap/LdapName.html#addAll(int,%20java.util.List))(int posn, [List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> suffixRdns)            Adds the RDNs of a name -- in order -- at a specified position within this name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**addAll**](http://docs.google.com/javax/naming/ldap/LdapName.html#addAll(int,%20javax.naming.Name))(int posn, [Name](http://docs.google.com/javax/naming/Name.html) suffix)            Adds the components of a name -- in order -- at a specified position within this name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**addAll**](http://docs.google.com/javax/naming/ldap/LdapName.html#addAll(java.util.List))([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> suffixRdns)            Adds the RDNs of a name -- in order -- to the end of this name. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**addAll**](http://docs.google.com/javax/naming/ldap/LdapName.html#addAll(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) suffix)            Adds the components of a name -- in order -- to the end of this name. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/javax/naming/ldap/LdapName.html#clone())()            Generates a new copy of this name. |
| int | [**compareTo**](http://docs.google.com/javax/naming/ldap/LdapName.html#compareTo(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Compares this LdapName with the specified Object for order. |
| boolean | [**endsWith**](http://docs.google.com/javax/naming/ldap/LdapName.html#endsWith(java.util.List))([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> rdns)            Determines whether the specified RDN sequence forms a suffix of this LDAP name. |
| boolean | [**endsWith**](http://docs.google.com/javax/naming/ldap/LdapName.html#endsWith(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) n)            Determines whether this LDAP name ends with a specified LDAP name suffix. |
| boolean | [**equals**](http://docs.google.com/javax/naming/ldap/LdapName.html#equals(java.lang.Object))([Object](http://docs.google.com/java/lang/Object.html) obj)            Determines whether two LDAP names are equal. |
| [String](http://docs.google.com/java/lang/String.html) | [**get**](http://docs.google.com/javax/naming/ldap/LdapName.html#get(int))(int posn)            Retrieves a component of this LDAP name as a string. |
| [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[String](http://docs.google.com/java/lang/String.html)> | [**getAll**](http://docs.google.com/javax/naming/ldap/LdapName.html#getAll())()            Retrieves the components of this name as an enumeration of strings. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**getPrefix**](http://docs.google.com/javax/naming/ldap/LdapName.html#getPrefix(int))(int posn)            Creates a name whose components consist of a prefix of the components of this LDAP name. |
| [Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html) | [**getRdn**](http://docs.google.com/javax/naming/ldap/LdapName.html#getRdn(int))(int posn)            Retrieves an RDN of this LDAP name as an Rdn. |
| [List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> | [**getRdns**](http://docs.google.com/javax/naming/ldap/LdapName.html#getRdns())()            Retrieves the list of relative distinguished names. |
| [Name](http://docs.google.com/javax/naming/Name.html) | [**getSuffix**](http://docs.google.com/javax/naming/ldap/LdapName.html#getSuffix(int))(int posn)            Creates a name whose components consist of a suffix of the components in this LDAP name. |
| int | [**hashCode**](http://docs.google.com/javax/naming/ldap/LdapName.html#hashCode())()            Computes the hash code of this LDAP name. |
| boolean | [**isEmpty**](http://docs.google.com/javax/naming/ldap/LdapName.html#isEmpty())()            Determines whether this LDAP name is empty. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**remove**](http://docs.google.com/javax/naming/ldap/LdapName.html#remove(int))(int posn)            Removes a component from this LDAP name. |
| int | [**size**](http://docs.google.com/javax/naming/ldap/LdapName.html#size())()            Retrieves the number of components in this LDAP name. |
| boolean | [**startsWith**](http://docs.google.com/javax/naming/ldap/LdapName.html#startsWith(java.util.List))([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> rdns)            Determines whether the specified RDN sequence forms a prefix of this LDAP name. |
| boolean | [**startsWith**](http://docs.google.com/javax/naming/ldap/LdapName.html#startsWith(javax.naming.Name))([Name](http://docs.google.com/javax/naming/Name.html) n)            Determines whether this LDAP name starts with a specified LDAP name prefix. |
| [String](http://docs.google.com/java/lang/String.html) | [**toString**](http://docs.google.com/javax/naming/ldap/LdapName.html#toString())()            Returns a string representation of this LDAP name in a format defined by [RFC 2253](http://ietf.org/rfc/rfc2253.txt) and described in the class description. |

| **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)) |

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

### LdapName

public **LdapName**([String](http://docs.google.com/java/lang/String.html) name)  
 throws [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html)

Constructs an LDAP name from the given distinguished name.

**Parameters:**name - This is a non-null distinguished name formatted according to the rules defined in [RFC 2253](http://ietf.org/rfc/rfc2253.txt). **Throws:** [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html) - if a syntax violation is detected.**See Also:**[Rdn.escapeValue(Object value)](http://docs.google.com/javax/naming/ldap/Rdn.html#escapeValue(java.lang.Object))

### LdapName

public **LdapName**([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> rdns)

Constructs an LDAP name given its parsed RDN components.

The indexing of RDNs in the list follows the numbering of RDNs described in the class description.

**Parameters:**rdns - The non-null list of Rdns forming this LDAP name.

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

### size

public int **size**()

Retrieves the number of components in this LDAP name.

**Specified by:**[size](http://docs.google.com/javax/naming/Name.html#size()) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Returns:**The non-negative number of components in this LDAP name.

### isEmpty

public boolean **isEmpty**()

Determines whether this LDAP name is empty. An empty name is one with zero components.

**Specified by:**[isEmpty](http://docs.google.com/javax/naming/Name.html#isEmpty()) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Returns:**true if this LDAP name is empty, false otherwise.

### getAll

public [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[String](http://docs.google.com/java/lang/String.html)> **getAll**()

Retrieves the components of this name as an enumeration of strings. The effect of updates to this name on this enumeration is undefined. If the name has zero components, an empty (non-null) enumeration is returned. The order of the components returned by the enumeration is same as the order in which the components are numbered as described in the class description.

**Specified by:**[getAll](http://docs.google.com/javax/naming/Name.html#getAll()) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Returns:**A non-null enumeration of the components of this LDAP name. Each element of the enumeration is of class String.

### get

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

Retrieves a component of this LDAP name as a string.

**Specified by:**[get](http://docs.google.com/javax/naming/Name.html#get(int)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**posn - The 0-based index of the component to retrieve. Must be in the range [0,size()). **Returns:**The non-null component at index posn. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if posn is outside the specified range.

### getRdn

public [Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html) **getRdn**(int posn)

Retrieves an RDN of this LDAP name as an Rdn.

**Parameters:**posn - The 0-based index of the RDN to retrieve. Must be in the range [0,size()). **Returns:**The non-null RDN at index posn. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if posn is outside the specified range.

### getPrefix

public [Name](http://docs.google.com/javax/naming/Name.html) **getPrefix**(int posn)

Creates a name whose components consist of a prefix of the components of this LDAP name. Subsequent changes to this name will not affect the name that is returned and vice versa.

**Specified by:**[getPrefix](http://docs.google.com/javax/naming/Name.html#getPrefix(int)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**posn - The 0-based index of the component at which to stop. Must be in the range [0,size()]. **Returns:**An instance of LdapName consisting of the components at indexes in the range [0,posn). If posn is zero, an empty LDAP name is returned. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If posn is outside the specified range.

### getSuffix

public [Name](http://docs.google.com/javax/naming/Name.html) **getSuffix**(int posn)

Creates a name whose components consist of a suffix of the components in this LDAP name. Subsequent changes to this name do not affect the name that is returned and vice versa.

**Specified by:**[getSuffix](http://docs.google.com/javax/naming/Name.html#getSuffix(int)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**posn - The 0-based index of the component at which to start. Must be in the range [0,size()]. **Returns:**An instance of LdapName consisting of the components at indexes in the range [posn,size()). If posn is equal to size(), an empty LDAP name is returned. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If posn is outside the specified range.

### startsWith

public boolean **startsWith**([Name](http://docs.google.com/javax/naming/Name.html) n)

Determines whether this LDAP name starts with a specified LDAP name prefix. A name n is a prefix if it is equal to getPrefix(n.size())--in other words this LDAP name starts with 'n'. If n is null or not a RFC2253 formatted name as described in the class description, false is returned.

**Specified by:**[startsWith](http://docs.google.com/javax/naming/Name.html#startsWith(javax.naming.Name)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**n - The LDAP name to check. **Returns:**true if n is a prefix of this LDAP name, false otherwise.**See Also:**[getPrefix(int posn)](http://docs.google.com/javax/naming/ldap/LdapName.html#getPrefix(int))

### startsWith

public boolean **startsWith**([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> rdns)

Determines whether the specified RDN sequence forms a prefix of this LDAP name. Returns true if this LdapName is at least as long as rdns, and for every position p in the range [0, rdns.size()) the component getRdn(p) matches rdns.get(p). Returns false otherwise. If rdns is null, false is returned.

**Parameters:**rdns - The sequence of Rdns to check. **Returns:**true if rdns form a prefix of this LDAP name, false otherwise.

### endsWith

public boolean **endsWith**([Name](http://docs.google.com/javax/naming/Name.html) n)

Determines whether this LDAP name ends with a specified LDAP name suffix. A name n is a suffix if it is equal to getSuffix(size()-n.size())--in other words this LDAP name ends with 'n'. If n is null or not a RFC2253 formatted name as described in the class description, false is returned.

**Specified by:**[endsWith](http://docs.google.com/javax/naming/Name.html#endsWith(javax.naming.Name)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**n - The LDAP name to check. **Returns:**true if n is a suffix of this name, false otherwise.**See Also:**[getSuffix(int posn)](http://docs.google.com/javax/naming/ldap/LdapName.html#getSuffix(int))

### endsWith

public boolean **endsWith**([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> rdns)

Determines whether the specified RDN sequence forms a suffix of this LDAP name. Returns true if this LdapName is at least as long as rdns, and for every position p in the range [size() - rdns.size(), size()) the component getRdn(p) matches rdns.get(p). Returns false otherwise. If rdns is null, false is returned.

**Parameters:**rdns - The sequence of Rdns to check. **Returns:**true if rdns form a suffix of this LDAP name, false otherwise.

### addAll

public [Name](http://docs.google.com/javax/naming/Name.html) **addAll**([Name](http://docs.google.com/javax/naming/Name.html) suffix)  
 throws [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html)

Adds the components of a name -- in order -- to the end of this name.

**Specified by:**[addAll](http://docs.google.com/javax/naming/Name.html#addAll(javax.naming.Name)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**suffix - The non-null components to add. **Returns:**The updated name (not a new instance). **Throws:** [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html) - if suffix is not a valid LDAP name, or if the addition of the components would violate the syntax rules of this LDAP name.

### addAll

public [Name](http://docs.google.com/javax/naming/Name.html) **addAll**([List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> suffixRdns)

Adds the RDNs of a name -- in order -- to the end of this name.

**Parameters:**suffixRdns - The non-null suffix Rdns to add. **Returns:**The updated name (not a new instance).

### addAll

public [Name](http://docs.google.com/javax/naming/Name.html) **addAll**(int posn,  
 [Name](http://docs.google.com/javax/naming/Name.html) suffix)  
 throws [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html)

Adds the components of a name -- in order -- at a specified position within this name. Components of this LDAP name at or after the index (if any) of the first new component are shifted up (away from index 0) to accomodate the new components.

**Specified by:**[addAll](http://docs.google.com/javax/naming/Name.html#addAll(int,%20javax.naming.Name)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**suffix - The non-null components to add.posn - The index at which to add the new component. Must be in the range [0,size()]. **Returns:**The updated name (not a new instance). **Throws:** [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html) - if suffix is not a valid LDAP name, or if the addition of the components would violate the syntax rules of this LDAP name. IndexOutOfBoundsException. - If posn is outside the specified range.

### addAll

public [Name](http://docs.google.com/javax/naming/Name.html) **addAll**(int posn,  
 [List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> suffixRdns)

Adds the RDNs of a name -- in order -- at a specified position within this name. RDNs of this LDAP name at or after the index (if any) of the first new RDN are shifted up (away from index 0) to accomodate the new RDNs.

**Parameters:**suffixRdns - The non-null suffix Rdns to add.posn - The index at which to add the suffix RDNs. Must be in the range [0,size()]. **Returns:**The updated name (not a new instance). **Throws:** IndexOutOfBoundsException. - If posn is outside the specified range.

### add

public [Name](http://docs.google.com/javax/naming/Name.html) **add**([String](http://docs.google.com/java/lang/String.html) comp)  
 throws [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html)

Adds a single component to the end of this LDAP name.

**Specified by:**[add](http://docs.google.com/javax/naming/Name.html#add(java.lang.String)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**comp - The non-null component to add. **Returns:**The updated LdapName, not a new instance. Cannot be null. **Throws:** [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html) - If adding comp at end of the name would violate the name's syntax.

### add

public [Name](http://docs.google.com/javax/naming/Name.html) **add**([Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html) comp)

Adds a single RDN to the end of this LDAP name.

**Parameters:**comp - The non-null RDN to add. **Returns:**The updated LdapName, not a new instance. Cannot be null.

### add

public [Name](http://docs.google.com/javax/naming/Name.html) **add**(int posn,  
 [String](http://docs.google.com/java/lang/String.html) comp)  
 throws [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html)

Adds a single component at a specified position within this LDAP name. Components of this LDAP name at or after the index (if any) of the new component are shifted up by one (away from index 0) to accommodate the new component.

**Specified by:**[add](http://docs.google.com/javax/naming/Name.html#add(int,%20java.lang.String)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**comp - The non-null component to add.posn - The index at which to add the new component. Must be in the range [0,size()]. **Returns:**The updated LdapName, not a new instance. Cannot be null. **Throws:** IndexOutOfBoundsException. - If posn is outside the specified range. [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html) - If adding comp at the specified position would violate the name's syntax.

### add

public [Name](http://docs.google.com/javax/naming/Name.html) **add**(int posn,  
 [Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html) comp)

Adds a single RDN at a specified position within this LDAP name. RDNs of this LDAP name at or after the index (if any) of the new RDN are shifted up by one (away from index 0) to accommodate the new RDN.

**Parameters:**comp - The non-null RDN to add.posn - The index at which to add the new RDN. Must be in the range [0,size()]. **Returns:**The updated LdapName, not a new instance. Cannot be null. **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - If posn is outside the specified range.

### remove

public [Object](http://docs.google.com/java/lang/Object.html) **remove**(int posn)  
 throws [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html)

Removes a component from this LDAP name. The component of this name at the specified position is removed. Components with indexes greater than this position (if any) are shifted down (toward index 0) by one.

**Specified by:**[remove](http://docs.google.com/javax/naming/Name.html#remove(int)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**posn - The index of the component to remove. Must be in the range [0,size()). **Returns:**The component removed (a String). **Throws:** [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if posn is outside the specified range. [InvalidNameException](http://docs.google.com/javax/naming/InvalidNameException.html) - if deleting the component would violate the syntax rules of the name.

### getRdns

public [List](http://docs.google.com/java/util/List.html)<[Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html)> **getRdns**()

Retrieves the list of relative distinguished names. The contents of the list are unmodifiable. The indexing of RDNs in the returned list follows the numbering of RDNs as described in the class description. If the name has zero components, an empty list is returned.

**Returns:**The name as a list of RDNs which are instances of the class [Rdn](http://docs.google.com/javax/naming/ldap/Rdn.html).

### clone

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

Generates a new copy of this name. Subsequent changes to the components of this name will not affect the new copy, and vice versa.

**Specified by:**[clone](http://docs.google.com/javax/naming/Name.html#clone()) in interface [Name](http://docs.google.com/javax/naming/Name.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 copy of the this LDAP name.**See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### toString

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

Returns a string representation of this LDAP name in a format defined by [RFC 2253](http://ietf.org/rfc/rfc2253.txt) and described in the class description. If the name has zero components an empty string is returned.

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

### equals

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

Determines whether two LDAP names are equal. If obj is null or not an LDAP name, false is returned.

Two LDAP names are equal if each RDN in one is equal to the corresponding RDN in the other. This implies both have the same number of RDNs, and each RDN's equals() test against the corresponding RDN in the other name returns true. See [Rdn.equals(Object obj)](http://docs.google.com/javax/naming/ldap/Rdn.html#equals(java.lang.Object)) for a definition of RDN equality.

**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 possibly null object to compare against. **Returns:**true if obj is equal to this LDAP name, false otherwise.**See Also:**[hashCode()](http://docs.google.com/javax/naming/ldap/LdapName.html#hashCode())

### compareTo

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

Compares this LdapName with the specified Object for order. Returns a negative integer, zero, or a positive integer as this Name is less than, equal to, or greater than the given Object.

If obj is null or not an instance of LdapName, ClassCastException is thrown.

Ordering of LDAP names follows the lexicographical rules for string comparison, with the extension that this applies to all the RDNs in the LDAP name. All the RDNs are lined up in their specified order and compared lexicographically. See [Rdn.compareTo(Object obj)](http://docs.google.com/javax/naming/ldap/Rdn.html#compareTo(java.lang.Object)) for RDN comparison rules.

If this LDAP name is lexicographically lesser than obj, a negative number is returned. If this LDAP name is lexicographically greater than obj, a positive number is returned.

**Specified by:**[compareTo](http://docs.google.com/java/lang/Comparable.html#compareTo(T)) in interface [Comparable](http://docs.google.com/java/lang/Comparable.html)<[Object](http://docs.google.com/java/lang/Object.html)>**Specified by:**[compareTo](http://docs.google.com/javax/naming/Name.html#compareTo(java.lang.Object)) in interface [Name](http://docs.google.com/javax/naming/Name.html) **Parameters:**obj - The non-null LdapName instance to compare against. **Returns:**A negative integer, zero, or a positive integer as this Name is less than, equal to, or greater than the given obj. **Throws:** [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if obj is null or not a LdapName.**See Also:**[Comparable.compareTo(Object)](http://docs.google.com/java/lang/Comparable.html#compareTo(T))

### hashCode

public int **hashCode**()

Computes the hash code of this LDAP name. The hash code is the sum of the hash codes of individual RDNs of this name.

**Overrides:**[hashCode](http://docs.google.com/java/lang/Object.html#hashCode()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**An int representing the hash code of this name.**See Also:**[equals(java.lang.Object)](http://docs.google.com/javax/naming/ldap/LdapName.html#equals(java.lang.Object))

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/LdapName.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/naming/ldap/LdapContext.html)   [**NEXT CLASS**](http://docs.google.com/javax/naming/ldap/LdapReferralException.html) | [**FRAMES**](http://docs.google.com/index.html?javax/naming/ldap/LdapName.html)    [**NO FRAMES**](http://docs.google.com/LdapName.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

[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).