| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ResourceBundle.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/util/RandomAccess.html)   [**NEXT CLASS**](http://docs.google.com/java/util/ResourceBundle.Control.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/ResourceBundle.html)    [**NO FRAMES**](http://docs.google.com/ResourceBundle.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#4d34og8) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

## **java.util**

Class ResourceBundle

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

**Direct Known Subclasses:** [ListResourceBundle](http://docs.google.com/java/util/ListResourceBundle.html), [PropertyResourceBundle](http://docs.google.com/java/util/PropertyResourceBundle.html)

public abstract class **ResourceBundle**extends [Object](http://docs.google.com/java/lang/Object.html)

Resource bundles contain locale-specific objects. When your program needs a locale-specific resource, a String for example, your program can load it from the resource bundle that is appropriate for the current user's locale. In this way, you can write program code that is largely independent of the user's locale isolating most, if not all, of the locale-specific information in resource bundles.

This allows you to write programs that can:

* be easily localized, or translated, into different languages
* handle multiple locales at once
* be easily modified later to support even more locales

Resource bundles belong to families whose members share a common base name, but whose names also have additional components that identify their locales. For example, the base name of a family of resource bundles might be "MyResources". The family should have a default resource bundle which simply has the same name as its family - "MyResources" - and will be used as the bundle of last resort if a specific locale is not supported. The family can then provide as many locale-specific members as needed, for example a German one named "MyResources\_de".

Each resource bundle in a family contains the same items, but the items have been translated for the locale represented by that resource bundle. For example, both "MyResources" and "MyResources\_de" may have a String that's used on a button for canceling operations. In "MyResources" the String may contain "Cancel" and in "MyResources\_de" it may contain "Abbrechen".

If there are different resources for different countries, you can make specializations: for example, "MyResources\_de\_CH" contains objects for the German language (de) in Switzerland (CH). If you want to only modify some of the resources in the specialization, you can do so.

When your program needs a locale-specific object, it loads the ResourceBundle class using the [getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale)) method:

ResourceBundle myResources =  
 ResourceBundle.getBundle("MyResources", currentLocale);

Resource bundles contain key/value pairs. The keys uniquely identify a locale-specific object in the bundle. Here's an example of a ListResourceBundle that contains two key/value pairs:

public class MyResources extends ListResourceBundle {  
 protected Object[][] getContents() {  
 return new Object[][] {  
 // LOCALIZE THE SECOND STRING OF EACH ARRAY (e.g., "OK")  
 {"OkKey", "OK"},  
 {"CancelKey", "Cancel"},  
 // END OF MATERIAL TO LOCALIZE  
 };  
 }  
 }

Keys are always Strings. In this example, the keys are "OkKey" and "CancelKey". In the above example, the values are also Strings--"OK" and "Cancel"--but they don't have to be. The values can be any type of object.

You retrieve an object from resource bundle using the appropriate getter method. Because "OkKey" and "CancelKey" are both strings, you would use getString to retrieve them:

button1 = new Button(myResources.getString("OkKey"));  
 button2 = new Button(myResources.getString("CancelKey"));

The getter methods all require the key as an argument and return the object if found. If the object is not found, the getter method throws a MissingResourceException.

Besides getString, ResourceBundle also provides a method for getting string arrays, getStringArray, as well as a generic getObject method for any other type of object. When using getObject, you'll have to cast the result to the appropriate type. For example:

int[] myIntegers = (int[]) myResources.getObject("intList");

The Java Platform provides two subclasses of ResourceBundle, ListResourceBundle and PropertyResourceBundle, that provide a fairly simple way to create resources. As you saw briefly in a previous example, ListResourceBundle manages its resource as a list of key/value pairs. PropertyResourceBundle uses a properties file to manage its resources.

If ListResourceBundle or PropertyResourceBundle do not suit your needs, you can write your own ResourceBundle subclass. Your subclasses must override two methods: handleGetObject and getKeys().

#### ResourceBundle.Control

The [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) class provides information necessary to perform the bundle loading process by the getBundle factory methods that take a ResourceBundle.Control instance. You can implement your own subclass in order to enable non-standard resource bundle formats, change the search strategy, or define caching parameters. Refer to the descriptions of the class and the [getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader,%20java.util.ResourceBundle.Control)) factory method for details.

#### Cache Management

Resource bundle instances created by the getBundle factory methods are cached by default, and the factory methods return the same resource bundle instance multiple times if it has been cached. getBundle clients may clear the cache, manage the lifetime of cached resource bundle instances using time-to-live values, or specify not to cache resource bundle instances. Refer to the descriptions of the [getBundle factory method](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader,%20java.util.ResourceBundle.Control)), [clearCache](http://docs.google.com/java/util/ResourceBundle.html#clearCache(java.lang.ClassLoader)), [ResourceBundle.Control.getTimeToLive](http://docs.google.com/java/util/ResourceBundle.Control.html#getTimeToLive(java.lang.String,%20java.util.Locale)), and [ResourceBundle.Control.needsReload](http://docs.google.com/java/util/ResourceBundle.Control.html#needsReload(java.lang.String,%20java.util.Locale,%20java.lang.String,%20java.lang.ClassLoader,%20java.util.ResourceBundle,%20long)) for details.

#### Example

The following is a very simple example of a ResourceBundle subclass, MyResources, that manages two resources (for a larger number of resources you would probably use a Map). Notice that you don't need to supply a value if a "parent-level" ResourceBundle handles the same key with the same value (as for the okKey below).

// default (English language, United States)  
 public class MyResources extends ResourceBundle {  
 public Object handleGetObject(String key) {  
 if (key.equals("okKey")) return "Ok";  
 if (key.equals("cancelKey")) return "Cancel";  
 return null;  
 }  
   
 public Enumeration<String> getKeys() {  
 return Collections.enumeration(keySet());  
 }  
   
 // Overrides handleKeySet() so that the getKeys() implementation  
 // can rely on the keySet() value.  
 protected Set<String> handleKeySet() {  
 return new HashSet<String>(Arrays.asList("okKey", "cancelKey"));  
 }  
 }  
  
 // German language  
 public class MyResources\_de extends MyResources {  
 public Object handleGetObject(String key) {  
 // don't need okKey, since parent level handles it.  
 if (key.equals("cancelKey")) return "Abbrechen";  
 return null;  
 }  
   
 protected Set<String> handleKeySet() {  
 return new HashSet<String>(Arrays.asList("cancelKey"));  
 }  
 }

You do not have to restrict yourself to using a single family of ResourceBundles. For example, you could have a set of bundles for exception messages, ExceptionResources (ExceptionResources\_fr, ExceptionResources\_de, ...), and one for widgets, WidgetResource (WidgetResources\_fr, WidgetResources\_de, ...); breaking up the resources however you like.

**Since:** JDK1.1 **See Also:**[ListResourceBundle](http://docs.google.com/java/util/ListResourceBundle.html), [PropertyResourceBundle](http://docs.google.com/java/util/PropertyResourceBundle.html), [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html)

| **Nested Class Summary** | |
| --- | --- |
| static class | [**ResourceBundle.Control**](http://docs.google.com/java/util/ResourceBundle.Control.html)            ResourceBundle.Control defines a set of callback methods that are invoked by the [ResourceBundle.getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader,%20java.util.ResourceBundle.Control)) factory methods during the bundle loading process. |

| **Field Summary** | |
| --- | --- |
| protected  [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**parent**](http://docs.google.com/java/util/ResourceBundle.html#parent)            The parent bundle of this bundle. |

| **Constructor Summary** | |
| --- | --- |
| [**ResourceBundle**](http://docs.google.com/java/util/ResourceBundle.html#ResourceBundle())()            Sole constructor. |

| **Method Summary** | |
| --- | --- |
| static void | [**clearCache**](http://docs.google.com/java/util/ResourceBundle.html#clearCache())()            Removes all resource bundles from the cache that have been loaded using the caller's class loader. |
| static void | [**clearCache**](http://docs.google.com/java/util/ResourceBundle.html#clearCache(java.lang.ClassLoader))([ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)            Removes all resource bundles from the cache that have been loaded using the given class loader. |
| boolean | [**containsKey**](http://docs.google.com/java/util/ResourceBundle.html#containsKey(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Determines whether the given key is contained in this ResourceBundle or its parent bundles. |
| static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**getBundle**](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String))([String](http://docs.google.com/java/lang/String.html) baseName)            Gets a resource bundle using the specified base name, the default locale, and the caller's class loader. |
| static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**getBundle**](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale))([String](http://docs.google.com/java/lang/String.html) baseName, [Locale](http://docs.google.com/java/util/Locale.html) locale)            Gets a resource bundle using the specified base name and locale, and the caller's class loader. |
| static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**getBundle**](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader))([String](http://docs.google.com/java/lang/String.html) baseName, [Locale](http://docs.google.com/java/util/Locale.html) locale, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)            Gets a resource bundle using the specified base name, locale, and class loader. |
| static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**getBundle**](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader,%20java.util.ResourceBundle.Control))([String](http://docs.google.com/java/lang/String.html) baseName, [Locale](http://docs.google.com/java/util/Locale.html) targetLocale, [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader, [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) control)            Returns a resource bundle using the specified base name, target locale, class loader and control. |
| static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**getBundle**](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.util.ResourceBundle.Control))([String](http://docs.google.com/java/lang/String.html) baseName, [Locale](http://docs.google.com/java/util/Locale.html) targetLocale, [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) control)            Returns a resource bundle using the specified base name, target locale and control, and the caller's class loader. |
| static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) | [**getBundle**](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.ResourceBundle.Control))([String](http://docs.google.com/java/lang/String.html) baseName, [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) control)            Returns a resource bundle using the specified base name, the default locale and the specified control. |
| abstract  [Enumeration](http://docs.google.com/java/util/Enumeration.html)<[String](http://docs.google.com/java/lang/String.html)> | [**getKeys**](http://docs.google.com/java/util/ResourceBundle.html#getKeys())()            Returns an enumeration of the keys. |
| [Locale](http://docs.google.com/java/util/Locale.html) | [**getLocale**](http://docs.google.com/java/util/ResourceBundle.html#getLocale())()            Returns the locale of this resource bundle. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getObject**](http://docs.google.com/java/util/ResourceBundle.html#getObject(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Gets an object for the given key from this resource bundle or one of its parents. |
| [String](http://docs.google.com/java/lang/String.html) | [**getString**](http://docs.google.com/java/util/ResourceBundle.html#getString(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Gets a string for the given key from this resource bundle or one of its parents. |
| [String](http://docs.google.com/java/lang/String.html)[] | [**getStringArray**](http://docs.google.com/java/util/ResourceBundle.html#getStringArray(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Gets a string array for the given key from this resource bundle or one of its parents. |
| protected abstract  [Object](http://docs.google.com/java/lang/Object.html) | [**handleGetObject**](http://docs.google.com/java/util/ResourceBundle.html#handleGetObject(java.lang.String))([String](http://docs.google.com/java/lang/String.html) key)            Gets an object for the given key from this resource bundle. |
| protected  [Set](http://docs.google.com/java/util/Set.html)<[String](http://docs.google.com/java/lang/String.html)> | [**handleKeySet**](http://docs.google.com/java/util/ResourceBundle.html#handleKeySet())()            Returns a Set of the keys contained *only* in this ResourceBundle. |
| [Set](http://docs.google.com/java/util/Set.html)<[String](http://docs.google.com/java/lang/String.html)> | [**keySet**](http://docs.google.com/java/util/ResourceBundle.html#keySet())()            Returns a Set of all keys contained in this ResourceBundle and its parent bundles. |
| protected  void | [**setParent**](http://docs.google.com/java/util/ResourceBundle.html#setParent(java.util.ResourceBundle))([ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) parent)            Sets the parent bundle of this bundle. |

| **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)) |

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

### parent

protected [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **parent**

The parent bundle of this bundle. The parent bundle is searched by [getObject](http://docs.google.com/java/util/ResourceBundle.html#getObject(java.lang.String)) when this bundle does not contain a particular resource.

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

### ResourceBundle

public **ResourceBundle**()

Sole constructor. (For invocation by subclass constructors, typically implicit.)

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

### getString

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

Gets a string for the given key from this resource bundle or one of its parents. Calling this method is equivalent to calling(String) [getObject](http://docs.google.com/java/util/ResourceBundle.html#getObject(java.lang.String))(key).

**Parameters:**key - the key for the desired string **Returns:**the string for the given key **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no object for the given key can be found [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the object found for the given key is not a string

### getStringArray

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

Gets a string array for the given key from this resource bundle or one of its parents. Calling this method is equivalent to calling(String[]) [getObject](http://docs.google.com/java/util/ResourceBundle.html#getObject(java.lang.String))(key).

**Parameters:**key - the key for the desired string array **Returns:**the string array for the given key **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no object for the given key can be found [ClassCastException](http://docs.google.com/java/lang/ClassCastException.html) - if the object found for the given key is not a string array

### getObject

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

Gets an object for the given key from this resource bundle or one of its parents. This method first tries to obtain the object from this resource bundle using [handleGetObject](http://docs.google.com/java/util/ResourceBundle.html#handleGetObject(java.lang.String)). If not successful, and the parent resource bundle is not null, it calls the parent's getObject method. If still not successful, it throws a MissingResourceException.

**Parameters:**key - the key for the desired object **Returns:**the object for the given key **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no object for the given key can be found

### getLocale

public [Locale](http://docs.google.com/java/util/Locale.html) **getLocale**()

Returns the locale of this resource bundle. This method can be used after a call to getBundle() to determine whether the resource bundle returned really corresponds to the requested locale or is a fallback.

**Returns:**the locale of this resource bundle

### setParent

protected void **setParent**([ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) parent)

Sets the parent bundle of this bundle. The parent bundle is searched by [getObject](http://docs.google.com/java/util/ResourceBundle.html#getObject(java.lang.String)) when this bundle does not contain a particular resource.

**Parameters:**parent - this bundle's parent bundle.

### getBundle

public static final [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **getBundle**([String](http://docs.google.com/java/lang/String.html) baseName)

Gets a resource bundle using the specified base name, the default locale, and the caller's class loader. Calling this method is equivalent to callinggetBundle(baseName, Locale.getDefault(), this.getClass().getClassLoader()),except that getClassLoader() is run with the security privileges of ResourceBundle. See [getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader)) for a complete description of the search and instantiation strategy.

**Parameters:**baseName - the base name of the resource bundle, a fully qualified class name **Returns:**a resource bundle for the given base name and the default locale **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if baseName is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no resource bundle for the specified base name can be found

### getBundle

public static final [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **getBundle**([String](http://docs.google.com/java/lang/String.html) baseName,  
 [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) control)

Returns a resource bundle using the specified base name, the default locale and the specified control. Calling this method is equivalent to calling

getBundle(baseName, Locale.getDefault(),  
 this.getClass().getClassLoader(), control),

except that getClassLoader() is run with the security privileges of ResourceBundle. See [getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader,%20java.util.ResourceBundle.Control)) for the complete description of the resource bundle loading process with a ResourceBundle.Control.

**Parameters:**baseName - the base name of the resource bundle, a fully qualified class namecontrol - the control which gives information for the resource bundle loading process **Returns:**a resource bundle for the given base name and the default locale **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if baseName or control is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no resource bundle for the specified base name can be found [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given control doesn't perform properly (e.g., control.getCandidateLocales returns null.) Note that validation of control is performed as needed.**Since:** 1.6

### getBundle

public static final [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **getBundle**([String](http://docs.google.com/java/lang/String.html) baseName,  
 [Locale](http://docs.google.com/java/util/Locale.html) locale)

Gets a resource bundle using the specified base name and locale, and the caller's class loader. Calling this method is equivalent to callinggetBundle(baseName, locale, this.getClass().getClassLoader()),except that getClassLoader() is run with the security privileges of ResourceBundle. See [getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader)) for a complete description of the search and instantiation strategy.

**Parameters:**baseName - the base name of the resource bundle, a fully qualified class namelocale - the locale for which a resource bundle is desired **Returns:**a resource bundle for the given base name and locale **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if baseName or locale is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no resource bundle for the specified base name can be found

### getBundle

public static final [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **getBundle**([String](http://docs.google.com/java/lang/String.html) baseName,  
 [Locale](http://docs.google.com/java/util/Locale.html) targetLocale,  
 [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) control)

Returns a resource bundle using the specified base name, target locale and control, and the caller's class loader. Calling this method is equivalent to calling

getBundle(baseName, targetLocale, this.getClass().getClassLoader(),  
 control),

except that getClassLoader() is run with the security privileges of ResourceBundle. See [getBundle](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader,%20java.util.ResourceBundle.Control)) for the complete description of the resource bundle loading process with a ResourceBundle.Control.

**Parameters:**baseName - the base name of the resource bundle, a fully qualified class nametargetLocale - the locale for which a resource bundle is desiredcontrol - the control which gives information for the resource bundle loading process **Returns:**a resource bundle for the given base name and a Locale in locales **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if baseName, locales or control is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no resource bundle for the specified base name in any of the locales can be found. [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given control doesn't perform properly (e.g., control.getCandidateLocales returns null.) Note that validation of control is performed as needed.**Since:** 1.6

### getBundle

public static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **getBundle**([String](http://docs.google.com/java/lang/String.html) baseName,  
 [Locale](http://docs.google.com/java/util/Locale.html) locale,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)

Gets a resource bundle using the specified base name, locale, and class loader.

Conceptually, getBundle uses the following strategy for locating and instantiating resource bundles:

getBundle uses the base name, the specified locale, and the default locale (obtained from [Locale.getDefault](http://docs.google.com/java/util/Locale.html#getDefault())) to generate a sequence of *candidate bundle names*. If the specified locale's language, country, and variant are all empty strings, then the base name is the only candidate bundle name. Otherwise, the following sequence is generated from the attribute values of the specified locale (language1, country1, and variant1) and of the default locale (language2, country2, and variant2):

* baseName + "\_" + language1 + "\_" + country1 + "\_" + variant1
* baseName + "\_" + language1 + "\_" + country1
* baseName + "\_" + language1
* baseName + "\_" + language2 + "\_" + country2 + "\_" + variant2
* baseName + "\_" + language2 + "\_" + country2
* baseName + "\_" + language2
* baseName

Candidate bundle names where the final component is an empty string are omitted. For example, if country1 is an empty string, the second candidate bundle name is omitted.

getBundle then iterates over the candidate bundle names to find the first one for which it can *instantiate* an actual resource bundle. For each candidate bundle name, it attempts to create a resource bundle:

* First, it attempts to load a class using the candidate bundle name. If such a class can be found and loaded using the specified class loader, is assignment compatible with ResourceBundle, is accessible from ResourceBundle, and can be instantiated, getBundle creates a new instance of this class and uses it as the *result resource bundle*.
* Otherwise, getBundle attempts to locate a property resource file. It generates a path name from the candidate bundle name by replacing all "." characters with "/" and appending the string ".properties". It attempts to find a "resource" with this name using [ClassLoader.getResource](http://docs.google.com/java/lang/ClassLoader.html#getResource(java.lang.String)). (Note that a "resource" in the sense of getResource has nothing to do with the contents of a resource bundle, it is just a container of data, such as a file.) If it finds a "resource", it attempts to create a new [PropertyResourceBundle](http://docs.google.com/java/util/PropertyResourceBundle.html) instance from its contents. If successful, this instance becomes the *result resource bundle*.

If no result resource bundle has been found, a MissingResourceException is thrown.

Once a result resource bundle has been found, its *parent chain* is instantiated. getBundle iterates over the candidate bundle names that can be obtained by successively removing variant, country, and language (each time with the preceding "\_") from the bundle name of the result resource bundle. As above, candidate bundle names where the final component is an empty string are omitted. With each of the candidate bundle names it attempts to instantiate a resource bundle, as described above. Whenever it succeeds, it calls the previously instantiated resource bundle's [setParent](http://docs.google.com/java/util/ResourceBundle.html#setParent(java.util.ResourceBundle)) method with the new resource bundle, unless the previously instantiated resource bundle already has a non-null parent.

getBundle caches instantiated resource bundles and may return the same resource bundle instance multiple times.

The baseName argument should be a fully qualified class name. However, for compatibility with earlier versions, Sun's Java SE Runtime Environments do not verify this, and so it is possible to access PropertyResourceBundles by specifying a path name (using "/") instead of a fully qualified class name (using ".").

**Example:**

The following class and property files are provided:

MyResources.class  
 MyResources.properties  
 MyResources\_fr.properties  
 MyResources\_fr\_CH.class  
 MyResources\_fr\_CH.properties  
 MyResources\_en.properties  
 MyResources\_es\_ES.class

The contents of all files are valid (that is, public non-abstract subclasses of ResourceBundle for the ".class" files, syntactically correct ".properties" files). The default locale is Locale("en", "GB").

Calling getBundle with the shown locale argument values instantiates resource bundles from the following sources:

* Locale("fr", "CH"): result MyResources\_fr\_CH.class, parent MyResources\_fr.properties, parent MyResources.class
* Locale("fr", "FR"): result MyResources\_fr.properties, parent MyResources.class
* Locale("de", "DE"): result MyResources\_en.properties, parent MyResources.class
* Locale("en", "US"): result MyResources\_en.properties, parent MyResources.class
* Locale("es", "ES"): result MyResources\_es\_ES.class, parent MyResources.class

The file MyResources\_fr\_CH.properties is never used because it is hidden by MyResources\_fr\_CH.class. Likewise, MyResources.properties is also hidden by MyResources.class.

**Parameters:**baseName - the base name of the resource bundle, a fully qualified class namelocale - the locale for which a resource bundle is desiredloader - the class loader from which to load the resource bundle **Returns:**a resource bundle for the given base name and locale **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if baseName, locale, or loader is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no resource bundle for the specified base name can be found**Since:** 1.2

### getBundle

public static [ResourceBundle](http://docs.google.com/java/util/ResourceBundle.html) **getBundle**([String](http://docs.google.com/java/lang/String.html) baseName,  
 [Locale](http://docs.google.com/java/util/Locale.html) targetLocale,  
 [ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader,  
 [ResourceBundle.Control](http://docs.google.com/java/util/ResourceBundle.Control.html) control)

Returns a resource bundle using the specified base name, target locale, class loader and control. Unlike the [getBundle factory methods with no control argument](http://docs.google.com/java/util/ResourceBundle.html#getBundle(java.lang.String,%20java.util.Locale,%20java.lang.ClassLoader)), the given control specifies how to locate and instantiate resource bundles. Conceptually, the bundle loading process with the given control is performed in the following steps.

1. This factory method looks up the resource bundle in the cache for the specified baseName, targetLocale and loader. If the requested resource bundle instance is found in the cache and the time-to-live periods of the instance and all of its parent instances have not expired, the instance is returned to the caller. Otherwise, this factory method proceeds with the loading process below.
2. The [control.getFormats](http://docs.google.com/java/util/ResourceBundle.Control.html#getFormats(java.lang.String)) method is called to get resource bundle formats to produce bundle or resource names. The strings "java.class" and "java.properties" designate class-based and [property](http://docs.google.com/java/util/PropertyResourceBundle.html)-based resource bundles, respectively. Other strings starting with "java." are reserved for future extensions and must not be used for application-defined formats. Other strings designate application-defined formats.
3. The [control.getCandidateLocales](http://docs.google.com/java/util/ResourceBundle.Control.html#getCandidateLocales(java.lang.String,%20java.util.Locale)) method is called with the target locale to get a list of *candidate Locales* for which resource bundles are searched.
4. The [control.newBundle](http://docs.google.com/java/util/ResourceBundle.Control.html#newBundle(java.lang.String,%20java.util.Locale,%20java.lang.String,%20java.lang.ClassLoader,%20boolean)) method is called to instantiate a ResourceBundle for the base bundle name, a candidate locale, and a format. (Refer to the note on the cache lookup below.) This step is iterated over all combinations of the candidate locales and formats until the newBundle method returns a ResourceBundle instance or the iteration has used up all the combinations. For example, if the candidate locales are Locale("de", "DE"), Locale("de") and Locale("") and the formats are "java.class" and "java.properties", then the following is the sequence of locale-format combinations to be used to call control.newBundle.

| **Locale** | **format** |
| --- | --- |
| Locale("de", "DE") | java.class |
| Locale("de", "DE") | java.properties |
| Locale("de") | java.class |
| Locale("de") | java.properties |
| Locale("") | java.class |
| Locale("") | java.properties |

1. If the previous step has found no resource bundle, proceed to Step 6. If a bundle has been found that is a base bundle (a bundle for Locale("")), and the candidate locale list only contained Locale(""), return the bundle to the caller. If a bundle has been found that is a base bundle, but the candidate locale list contained locales other than Locale(""), put the bundle on hold and proceed to Step 6. If a bundle has been found that is not a base bundle, proceed to Step 7.
2. The [control.getFallbackLocale](http://docs.google.com/java/util/ResourceBundle.Control.html#getFallbackLocale(java.lang.String,%20java.util.Locale)) method is called to get a fallback locale (alternative to the current target locale) to try further finding a resource bundle. If the method returns a non-null locale, it becomes the next target locale and the loading process starts over from Step 3. Otherwise, if a base bundle was found and put on hold in a previous Step 5, it is returned to the caller now. Otherwise, a MissingResourceException is thrown.
3. At this point, we have found a resource bundle that's not the base bundle. If this bundle set its parent during its instantiation, it is returned to the caller. Otherwise, its [parent chain](http://docs.google.com/ResourceBundle.html#parent_chain) is instantiated based on the list of candidate locales from which it was found. Finally, the bundle is returned to the caller.

During the resource bundle loading process above, this factory method looks up the cache before calling the [control.newBundle](http://docs.google.com/java/util/ResourceBundle.Control.html#newBundle(java.lang.String,%20java.util.Locale,%20java.lang.String,%20java.lang.ClassLoader,%20boolean)) method. If the time-to-live period of the resource bundle found in the cache has expired, the factory method calls the [control.needsReload](http://docs.google.com/java/util/ResourceBundle.Control.html#needsReload(java.lang.String,%20java.util.Locale,%20java.lang.String,%20java.lang.ClassLoader,%20java.util.ResourceBundle,%20long)) method to determine whether the resource bundle needs to be reloaded. If reloading is required, the factory method calls control.newBundle to reload the resource bundle. If control.newBundle returns null, the factory method puts a dummy resource bundle in the cache as a mark of nonexistent resource bundles in order to avoid lookup overhead for subsequent requests. Such dummy resource bundles are under the same expiration control as specified by control.

All resource bundles loaded are cached by default. Refer to [control.getTimeToLive](http://docs.google.com/java/util/ResourceBundle.Control.html#getTimeToLive(java.lang.String,%20java.util.Locale)) for details.

The following is an example of the bundle loading process with the default ResourceBundle.Control implementation.

Conditions:

* Base bundle name: foo.bar.Messages
* Requested Locale: [Locale.ITALY](http://docs.google.com/java/util/Locale.html#ITALY)
* Default Locale: [Locale.FRENCH](http://docs.google.com/java/util/Locale.html#FRENCH)
* Available resource bundles: foo/bar/Messages\_fr.properties and foo/bar/Messages.properties

First, getBundle tries loading a resource bundle in the following sequence.

* class foo.bar.Messages\_it\_IT
* file foo/bar/Messages\_it\_IT.properties
* class foo.bar.Messages\_it
* file foo/bar/Messages\_it.properties
* class foo.bar.Messages
* file foo/bar/Messages.properties

At this point, getBundle finds foo/bar/Messages.properties, which is put on hold because it's the base bundle. getBundle calls [control.getFallbackLocale("foo.bar.Messages", Locale.ITALY)](http://docs.google.com/java/util/ResourceBundle.Control.html#getFallbackLocale(java.lang.String,%20java.util.Locale)) which returns Locale.FRENCH. Next, getBundle tries loading a bundle in the following sequence.

* class foo.bar.Messages\_fr
* file foo/bar/Messages\_fr.properties
* class foo.bar.Messages
* file foo/bar/Messages.properties

getBundle finds foo/bar/Messages\_fr.properties and creates a ResourceBundle instance. Then, getBundle sets up its parent chain from the list of the candiate locales. Only foo/bar/Messages.properties is found in the list and getBundle creates a ResourceBundle instance that becomes the parent of the instance for foo/bar/Messages\_fr.properties.

**Parameters:**baseName - the base name of the resource bundle, a fully qualified class nametargetLocale - the locale for which a resource bundle is desiredloader - the class loader from which to load the resource bundlecontrol - the control which gives information for the resource bundle loading process **Returns:**a resource bundle for the given base name and locale **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if baseName, targetLocale, loader, or control is null [MissingResourceException](http://docs.google.com/java/util/MissingResourceException.html) - if no resource bundle for the specified base name can be found [IllegalArgumentException](http://docs.google.com/java/lang/IllegalArgumentException.html) - if the given control doesn't perform properly (e.g., control.getCandidateLocales returns null.) Note that validation of control is performed as needed.**Since:** 1.6

### clearCache

public static final void **clearCache**()

Removes all resource bundles from the cache that have been loaded using the caller's class loader.

**Since:** 1.6 **See Also:**[ResourceBundle.Control.getTimeToLive(String,Locale)](http://docs.google.com/java/util/ResourceBundle.Control.html#getTimeToLive(java.lang.String,%20java.util.Locale))

### clearCache

public static final void **clearCache**([ClassLoader](http://docs.google.com/java/lang/ClassLoader.html) loader)

Removes all resource bundles from the cache that have been loaded using the given class loader.

**Parameters:**loader - the class loader **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if loader is null**Since:** 1.6 **See Also:**[ResourceBundle.Control.getTimeToLive(String,Locale)](http://docs.google.com/java/util/ResourceBundle.Control.html#getTimeToLive(java.lang.String,%20java.util.Locale))

### handleGetObject

protected abstract [Object](http://docs.google.com/java/lang/Object.html) **handleGetObject**([String](http://docs.google.com/java/lang/String.html) key)

Gets an object for the given key from this resource bundle. Returns null if this resource bundle does not contain an object for the given key.

**Parameters:**key - the key for the desired object **Returns:**the object for the given key, or null **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null

### getKeys

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

Returns an enumeration of the keys.

**Returns:**an Enumeration of the keys contained in this ResourceBundle and its parent bundles.

### containsKey

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

Determines whether the given key is contained in this ResourceBundle or its parent bundles.

**Parameters:**key - the resource key **Returns:**true if the given key is contained in this ResourceBundle or its parent bundles; false otherwise. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if key is null**Since:** 1.6

### keySet

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

Returns a Set of all keys contained in this ResourceBundle and its parent bundles.

**Returns:**a Set of all keys contained in this ResourceBundle and its parent bundles.**Since:** 1.6

### handleKeySet

protected [Set](http://docs.google.com/java/util/Set.html)<[String](http://docs.google.com/java/lang/String.html)> **handleKeySet**()

Returns a Set of the keys contained *only* in this ResourceBundle.

The default implementation returns a Set of the keys returned by the [getKeys](http://docs.google.com/java/util/ResourceBundle.html#getKeys()) method except for the ones for which the [handleGetObject](http://docs.google.com/java/util/ResourceBundle.html#handleGetObject(java.lang.String)) method returns null. Once the Set has been created, the value is kept in this ResourceBundle in order to avoid producing the same Set in the next calls. Override this method in subclass implementations for faster handling.

**Returns:**a Set of the keys contained only in this ResourceBundle**Since:** 1.6

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/ResourceBundle.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/util/RandomAccess.html)   [**NEXT CLASS**](http://docs.google.com/java/util/ResourceBundle.Control.html) | [**FRAMES**](http://docs.google.com/index.html?java/util/ResourceBundle.html)    [**NO FRAMES**](http://docs.google.com/ResourceBundle.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: [NESTED](#3znysh7) | [FIELD](#2et92p0) | [CONSTR](#tyjcwt) | [METHOD](#3dy6vkm) | DETAIL: [FIELD](#4d34og8) | [CONSTR](#17dp8vu) | [METHOD](#26in1rg) |

[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).