

ResourcesUserGuideImplementations

[Home](#) [Wiki](#) [Guide](#) [Getting Started](#) [Messages](#) [Standard](#) [Creating](#) [API](#) [Source](#)

3. Standard Resources / [ResourceFactory](#) Implementations

Commons Resources ships with the following implementations of [Resources](#) and [ResourceFactory](#) provided:

Resources	ResourceFactory	Description
PropertyResources	PropertyResourcesFactory	retrieves values from properties files
XMLResources	XMLResourcesFactory	retrieves values from XML documents
JDBCResources	PropertyResourcesFactory	retrieves values from a database using JDBC
WebappPropertyResources	WebappPropertyResourcesFactory	Web App: retrieves values from properties files .
WebappXMLResources	WebappXMLResourcesFactory	Web App: retrieves values from XML documents
ResourceBundleResources	ResourceBundleResourcesFactory	wraps java.util.ResourceBundle

The following sections describe how to use these implementations:

- [3.1 Property Resources](#)
- [3.2 XML Resources](#)
- [3.3 JDBC Resources](#)
- [3.4 Web Application Property Resources](#)
- [3.5 Web Application XML Resources](#)
- [3.6 java.util.ResourceBundle Resources](#)

3.1 Property Resources

PropertyResources wrap a family (one per Locale) of properties files that share a base URL and have name suffixes reflecting the Locale.

Resources are looked up in a hierarchy of properties files identical to that performed by *java.util.ResourceBundle*. For example, if the configuration URL is passed as `file:c:/myapp/foo/Bar`, then...

File	Locale Suffix	for
<code>c:/myapp/foo/Bar.properties</code>		default resources
<code>c:/myapp/foo/Bar_en.properties</code>	en	English resources
<code>c:/myapp/foo/Bar_en_US.properties</code>	en_US	US English resources
<code>c:/myapp/foo/Bar_en_GB.properties</code>	en_GB	UK English resources

N.B. **PropertyResources** assumes all files will end with **.properties**

In these properties files, you specify key/value pairs in the normal way, for example...

```
error.date=Date is invalid.  
error.number=Number is invalid.
```

To use **PropertyResources**, you create it using it's factory and then use in the normal way...

```
// Create the ResourceFactory  
ResourceFactory factory = new PropertyResourcesFactory();  
  
// Create the Resources  
Resources resources = factory.getResources("Bar", "file:c:/myapp/foo/Bar");  
  
// Retrieve an i18n String value  
String msg = resources.getString("some.key", locale, null);
```

See **PropertyResources** [JavaDoc](#) / [source](#) and **PropertyResourcesFactory** [JavaDoc](#) / [source](#).

3.2 XML Resources

XMLResources works in pretty much the same way as **PropertyResources**, except the files are in XML format and **XMLResources** assumes a **.xml** file extension. Using the same example, if the configuration URL is passed as **file:c:/myapp/foo/Bar** then... ..

File	Locale Suffix	for
c:/myapp/foo/Bar.xml		default resources
c:/myapp/foo/Bar_en.xml	en	English resources
c:/myapp/foo/Bar_en_US.xml	en_US	US English resources
c:/myapp/foo/Bar_en_GB.xml	en_GB	UK English resources

In these XML files, you specify resources in the following way...

```
<resources>
  <resource id="error.date">
    Date is invalid.
  </resource>

  <resource id="error.number">
    Number is invalid.
  </resource>
</resources>
```

To use **XMLResources**, you create it using it's factory and then use in the normal way...

```
// Create the ResourcesFactory
ResourcesFactory factory = new XMLResourcesFactory();

// Create the Resources
Resources resources = factory.getResources("Bar", "file:c:/myapp/foo/Bar");

// Retrieve an i18n String value
String msg = resources.getString("some.key", locale, null);
```

See **XMLResources** [JavaDoc](#) / [source](#) and **XMLResourcesFactory** [JavaDoc](#) / [source](#).

3.3 JDBC Resources

JDBCResources retrieves messages from a database using JDBC. The table you use to store the messages needs three columns containing:

- the locale value (e.g. "en_US" for US English)
- the message key
- the message text

The names of these columns and the table is configured, along with the Driver information, in a JDBC Properties file. For example in **c:/myapp/foo/Bar.properties**...

```
jdbc.connect.driver    = org.gjt.mm.mysql.Driver
jdbc.connect.url       = jdbc:mysql://localhost/MyDatabase
jdbc.connect.login     = MyUserName
jdbc.connect.password  = MyPassword

jdbc.sql.table         = MyMessages
jdbc.sql.locale.column = locale_key
jdbc.sql.key.column    = message_key
jdbc.sql.val.column    = message_text
```

N.B. **JDBCResources** appends **.properties** to the URL you supply for this file.

To use `JDBCResources`, you create it using its factory, with the URL for the JDBC Properties file and then use in the normal way...

```
// Create the ResourcesFactory
ResourcesFactory factory = new JDBCResourcesFactory();

// Create the Resources
Resources resources = factory.getResources("Bar", "file:c:myapp/foo/Bar");

// Retrieve an i18n String value
String msg = resources.getString("some.key", locale, null);
```

See `JDBCResources` [JavaDoc](#) / [source](#) and `JDBCResourcesFactory` [JavaDoc](#) / [source](#).

3.4 Web Application Property Resources

`WebappPropertyResources` works the same way as `PropertyResources`, but in a Web Application environment. The only difference is you specify a context relative URL and you need to initialize the `ServletContext`....

```
// Create the ResourcesFactory
ResourcesFactory factory = new WebappPropertyResourcesFactory();
factory.setServletContext(servletContext);

// Create the Resources
Resources resources = factory.getResources("Bar", "/org/apache/struts/Bar");

// Retrieve an i18n String value
String msg = resources.getString("some.key", locale, null);
```

See `WebappPropertyResources` [JavaDoc](#) / [source](#) and `WebappPropertyResourcesFactory` [JavaDoc](#) / [source](#).

3.5 Web Application XML Resources

`WebappXMLResources` works the same way as `XMLResources`, but in a Web Application environment. The only difference is you specify a context relative URL and you need to initialize the `ServletContext`....

```
// Create the ResourcesFactory
ResourcesFactory factory = new WebappXMLResourcesFactory();
factory.setServletContext(servletContext);

// Create the Resources
Resources resources = factory.getResources("Bar", "/org/apache/struts/Bar");

// Retrieve an i18n String value
String msg = resources.getString("some.key", locale, null);
```

See `WebappXMLResources` [JavaDoc](#) / [source](#) and `WebappXMLResourcesFactory` [JavaDoc](#) / [source](#).

3.6 java.util.ResourceBundle Resources

`ResourceBundleResources` is a `Resources` implementation that wraps a `java.util.ResourceBundle` instance.

```
// Create the ResourcesFactory
ResourcesFactory factory = new ResourceBundleResourcesFactory();

// Create the Resources
Resources resources = factory.getResources("Bar", "BarResources");

// Retrieve an i18n String value
String msg = resources.getString("some.key", locale, null);
```

See **ResourceBundleResources** [JavaDoc](#) / [source](#) and **ResourceBundleResourcesFactory** [JavaDoc](#) / [source](#).

Home	Wiki	Guide	Getting Started	Messages	Standard	Creating	API	Source
----------------------	----------------------	-----------------------	---------------------------------	--------------------------	--------------------------	--------------------------	---------------------	------------------------