| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.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 PACKAGE**](http://docs.google.com/java/awt/print/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/java/beans/beancontext/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?java/beans/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package java.beans

Contains classes related to developing *beans* -- components based on the JavaBeansTM architecture.

**See:**

[**Description**](#3znysh7)

| **Interface Summary** | |
| --- | --- |
| [**AppletInitializer**](http://docs.google.com/java/beans/AppletInitializer.html) | This interface is designed to work in collusion with java.beans.Beans.instantiate. |
| [**BeanInfo**](http://docs.google.com/java/beans/BeanInfo.html) | A bean implementor who wishes to provide explicit information about their bean may provide a BeanInfo class that implements this BeanInfo interface and provides explicit information about the methods, properties, events, etc, of their bean. |
| [**Customizer**](http://docs.google.com/java/beans/Customizer.html) | A customizer class provides a complete custom GUI for customizing a target Java Bean. |
| [**DesignMode**](http://docs.google.com/java/beans/DesignMode.html) | This interface is intended to be implemented by, or delegated from, instances of java.beans.beancontext.BeanContext, in order to propagate to its nested hierarchy of java.beans.beancontext.BeanContextChild instances, the current "designTime" property. |
| [**ExceptionListener**](http://docs.google.com/java/beans/ExceptionListener.html) | An ExceptionListener is notified of internal exceptions. |
| [**PropertyChangeListener**](http://docs.google.com/java/beans/PropertyChangeListener.html) | A "PropertyChange" event gets fired whenever a bean changes a "bound" property. |
| [**PropertyEditor**](http://docs.google.com/java/beans/PropertyEditor.html) | A PropertyEditor class provides support for GUIs that want to allow users to edit a property value of a given type. |
| [**VetoableChangeListener**](http://docs.google.com/java/beans/VetoableChangeListener.html) | A VetoableChange event gets fired whenever a bean changes a "constrained" property. |
| [**Visibility**](http://docs.google.com/java/beans/Visibility.html) | Under some circumstances a bean may be run on servers where a GUI is not available. |

| **Class Summary** | |
| --- | --- |
| [**BeanDescriptor**](http://docs.google.com/java/beans/BeanDescriptor.html) | A BeanDescriptor provides global information about a "bean", including its Java class, its displayName, etc. |
| [**Beans**](http://docs.google.com/java/beans/Beans.html) | This class provides some general purpose beans control methods. |
| [**DefaultPersistenceDelegate**](http://docs.google.com/java/beans/DefaultPersistenceDelegate.html) | The DefaultPersistenceDelegate is a concrete implementation of the abstract PersistenceDelegate class and is the delegate used by default for classes about which no information is available. |
| [**Encoder**](http://docs.google.com/java/beans/Encoder.html) | An Encoder is a class which can be used to create files or streams that encode the state of a collection of JavaBeans in terms of their public APIs. |
| [**EventHandler**](http://docs.google.com/java/beans/EventHandler.html) | The EventHandler class provides support for dynamically generating event listeners whose methods execute a simple statement involving an incoming event object and a target object. |
| [**EventSetDescriptor**](http://docs.google.com/java/beans/EventSetDescriptor.html) | An EventSetDescriptor describes a group of events that a given Java bean fires. |
| [**Expression**](http://docs.google.com/java/beans/Expression.html) | An Expression object represents a primitive expression in which a single method is applied to a target and a set of arguments to return a result - as in "a.getFoo()". |
| [**FeatureDescriptor**](http://docs.google.com/java/beans/FeatureDescriptor.html) | The FeatureDescriptor class is the common baseclass for PropertyDescriptor, EventSetDescriptor, and MethodDescriptor, etc. |
| [**IndexedPropertyChangeEvent**](http://docs.google.com/java/beans/IndexedPropertyChangeEvent.html) | An "IndexedPropertyChange" event gets delivered whenever a component that conforms to the JavaBeans specification (a "bean") changes a bound indexed property. |
| [**IndexedPropertyDescriptor**](http://docs.google.com/java/beans/IndexedPropertyDescriptor.html) | An IndexedPropertyDescriptor describes a property that acts like an array and has an indexed read and/or indexed write method to access specific elements of the array. |
| [**Introspector**](http://docs.google.com/java/beans/Introspector.html) | The Introspector class provides a standard way for tools to learn about the properties, events, and methods supported by a target Java Bean. |
| [**MethodDescriptor**](http://docs.google.com/java/beans/MethodDescriptor.html) | A MethodDescriptor describes a particular method that a Java Bean supports for external access from other components. |
| [**ParameterDescriptor**](http://docs.google.com/java/beans/ParameterDescriptor.html) | The ParameterDescriptor class allows bean implementors to provide additional information on each of their parameters, beyond the low level type information provided by the java.lang.reflect.Method class. |
| [**PersistenceDelegate**](http://docs.google.com/java/beans/PersistenceDelegate.html) | The PersistenceDelegate class takes the responsibility for expressing the state of an instance of a given class in terms of the methods in the class's public API. |
| [**PropertyChangeEvent**](http://docs.google.com/java/beans/PropertyChangeEvent.html) | A "PropertyChange" event gets delivered whenever a bean changes a "bound" or "constrained" property. |
| [**PropertyChangeListenerProxy**](http://docs.google.com/java/beans/PropertyChangeListenerProxy.html) | A class which extends the EventListenerProxy specifically for adding a named PropertyChangeListener. |
| [**PropertyChangeSupport**](http://docs.google.com/java/beans/PropertyChangeSupport.html) | This is a utility class that can be used by beans that support bound properties. |
| [**PropertyDescriptor**](http://docs.google.com/java/beans/PropertyDescriptor.html) | A PropertyDescriptor describes one property that a Java Bean exports via a pair of accessor methods. |
| [**PropertyEditorManager**](http://docs.google.com/java/beans/PropertyEditorManager.html) | The PropertyEditorManager can be used to locate a property editor for any given type name. |
| [**PropertyEditorSupport**](http://docs.google.com/java/beans/PropertyEditorSupport.html) | This is a support class to help build property editors. |
| [**SimpleBeanInfo**](http://docs.google.com/java/beans/SimpleBeanInfo.html) | This is a support class to make it easier for people to provide BeanInfo classes. |
| [**Statement**](http://docs.google.com/java/beans/Statement.html) | A Statement object represents a primitive statement in which a single method is applied to a target and a set of arguments - as in "a.setFoo(b)". |
| [**VetoableChangeListenerProxy**](http://docs.google.com/java/beans/VetoableChangeListenerProxy.html) | A class which extends the EventListenerProxy specifically for associating a VetoableChangeListener with a "constrained" property. |
| [**VetoableChangeSupport**](http://docs.google.com/java/beans/VetoableChangeSupport.html) | This is a utility class that can be used by beans that support constrained properties. |
| [**XMLDecoder**](http://docs.google.com/java/beans/XMLDecoder.html) | The XMLDecoder class is used to read XML documents created using the XMLEncoder and is used just like the ObjectInputStream. |
| [**XMLEncoder**](http://docs.google.com/java/beans/XMLEncoder.html) | The XMLEncoder class is a complementary alternative to the ObjectOutputStream and can used to generate a textual representation of a *JavaBean* in the same way that the ObjectOutputStream can be used to create binary representation of Serializable objects. |

| **Exception Summary** | |
| --- | --- |
| [**IntrospectionException**](http://docs.google.com/java/beans/IntrospectionException.html) | Thrown when an exception happens during Introspection. |
| [**PropertyVetoException**](http://docs.google.com/java/beans/PropertyVetoException.html) | A PropertyVetoException is thrown when a proposed change to a property represents an unacceptable value. |

| **Annotation Types Summary** | |
| --- | --- |
| [**ConstructorProperties**](http://docs.google.com/java/beans/ConstructorProperties.html) | An annotation on a constructor that shows how the parameters of that constructor correspond to the constructed object's getter methods. |

## Package java.beans Description

Contains classes related to developing *beans* -- components based on the JavaBeansTM architecture. A few of the classes are used by beans while they run in an application. For example, the event classes are used by beans that fire property and vetoable change events (see [PropertyChangeEvent](http://docs.google.com/java/beans/PropertyChangeEvent.html)). However, most of the classes in this package are meant to be used by a bean editor (that is, a development environment for customizing and putting together beans to create an application). In particular, these classes help the bean editor create a user interface that the user can use to customize the bean. For example, a bean may contain a property of a special type that a bean editor may not know how to handle. By using the PropertyEditor interface, a bean developer can provide an editor for this special type.

To minimize the resources used by a bean, the classes used by bean editors are loaded only when the bean is being edited. They are not needed while the bean is running in an application and therefore not loaded. This information is kept in what's called a bean-info (see [BeanInfo](http://docs.google.com/java/beans/BeanInfo.html)).

Unless explicitly stated, null values or empty Strings are not valid parameters for the methods in this package. You may expect to see exceptions if these parameters are used.

## Long-Term Persistence

As of v1.4, the java.beans package provides support for *long-term persistence* -- reading and writing a bean as a textual representation of its property values. The property values are treated as beans, and are recursively read or written to capture their publicly available state. This approach is suitable for long-term storage because it relies only on public API, rather than the likely-to-change private implementation.**Note:** The persistence scheme cannot automatically instantiate custom inner classes, such as you might use for event handlers. By using the [EventHandler](http://docs.google.com/java/beans/EventHandler.html) class instead of inner classes for custom event handlers, you can avoid this problem.

You read and write beans in XML format using the [XMLDecoder](http://docs.google.com/java/beans/XMLDecoder.html) and [XMLEncoder](http://docs.google.com/java/beans/XMLEncoder.html) classes, respectively. One notable feature of the persistence scheme is that reading in a bean requires no special knowledge of the bean.

Writing out a bean, on the other hand, sometimes requires special knowledge of the bean's type. If the bean's state can be expressed using only the no-argument constructor and public getter and setter methods for properties, no special knowledge is required. Otherwise, the bean requires a custom *persistence delegate* -- an object that is in charge of writing out beans of a particular type. All classes provided in the JDK that descend from java.awt.Component, as well as all their properties, automatically have persistence delegates.

If you need (or choose) to provide a persistence delegate for a bean, you can do so either by using a [DefaultPersistenceDelegate](http://docs.google.com/java/beans/DefaultPersistenceDelegate.html) instance or by creating your own subclass of PersistenceDelegate. If the only reason a bean needs a persistence delegate is because you want to invoke the bean's constructor with property values as arguments, you can create the bean's persistence delegate with the one-argument DefaultPersistenceDelegate constructor. Otherwise, you need to implement your own persistence delegate, for which you're likely to need the following classes:

[PersistenceDelegate](http://docs.google.com/java/beans/PersistenceDelegate.html) The abstract class from which all persistence delegates descend. Your subclass should use its knowledge of the bean's type to provide whatever Statements and Expressions are necessary to create the bean and restore its state. [Statement](http://docs.google.com/java/beans/Statement.html) Represents the invocation of a single method on an object. Includes a set of arguments to the method. [Expression](http://docs.google.com/java/beans/Expression.html) A subclass of Statement used for methods that return a value.

Once you create a persistence delegate, you register it using the setPersistenceDelegate method of XMLEncoder.

## Related Documentation

For overview, architecture, and tutorial documentation, please see:

* [JavaBeans](http://java.sun.com/docs/books/tutorial/javabeans/), a trail in *The Java Tutorial*.
* [Long-Term Persistence](http://java.sun.com/products/jfc/tsc/articles/persistence2/), an article in *The Swing Connection*.

| | [**Overview**](http://docs.google.com/overview-summary.html) | **Package** | Class | [**Use**](http://docs.google.com/package-use.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 PACKAGE**](http://docs.google.com/java/awt/print/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/java/beans/beancontext/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?java/beans/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

[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).