| | [**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/javax/print/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/print/attribute/standard/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/print/attribute/package-summary.html)    [**NO FRAMES**](http://docs.google.com/package-summary.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |

## Package javax.print.attribute

Provides classes and interfaces that describe the types of JavaTM Print Service attributes and how they can be collected into attribute sets.

**See:**

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

| **Interface Summary** | |
| --- | --- |
| [**Attribute**](http://docs.google.com/javax/print/attribute/Attribute.html) | Interface Attribute is the base interface implemented by any and every printing attribute class to indicate that the class represents a printing attribute. |
| [**AttributeSet**](http://docs.google.com/javax/print/attribute/AttributeSet.html) | Interface AttributeSet specifies the interface for a set of printing attributes. |
| [**DocAttribute**](http://docs.google.com/javax/print/attribute/DocAttribute.html) | Interface DocAttribute is a tagging interface which a printing attribute class implements to indicate the attribute denotes a setting for a doc. |
| [**DocAttributeSet**](http://docs.google.com/javax/print/attribute/DocAttributeSet.html) | Interface DocAttributeSet specifies the interface for a set of doc attributes, i.e. |
| [**PrintJobAttribute**](http://docs.google.com/javax/print/attribute/PrintJobAttribute.html) | PrintJobAttribute is a tagging interface which a printing attribute class implements to indicate the attribute describes the status of a Print Job or some other characteristic of a Print Job. |
| [**PrintJobAttributeSet**](http://docs.google.com/javax/print/attribute/PrintJobAttributeSet.html) | Interface PrintJobAttributeSet specifies the interface for a set of print job attributes, i.e. |
| [**PrintRequestAttribute**](http://docs.google.com/javax/print/attribute/PrintRequestAttribute.html) | Interface PrintRequestAttribute is a tagging interface which a printing attribute class implements to indicate the attribute denotes a requested setting for a print job. |
| [**PrintRequestAttributeSet**](http://docs.google.com/javax/print/attribute/PrintRequestAttributeSet.html) | Interface PrintRequestAttributeSet specifies the interface for a set of print request attributes, i.e. |
| [**PrintServiceAttribute**](http://docs.google.com/javax/print/attribute/PrintServiceAttribute.html) | Interface PrintServiceAttribute is a tagging interface which a printing attribute class implements to indicate the attribute describes the status of a Print Service or some other characteristic of a Print Service. |
| [**PrintServiceAttributeSet**](http://docs.google.com/javax/print/attribute/PrintServiceAttributeSet.html) | Interface PrintServiceAttributeSet specifies the interface for a set of print job attributes, i.e. |
| [**SupportedValuesAttribute**](http://docs.google.com/javax/print/attribute/SupportedValuesAttribute.html) | Interface SupportedValuesAttribute is a tagging interface which a printing attribute class implements to indicate the attribute describes the supported values for another attribute. |

| **Class Summary** | |
| --- | --- |
| [**AttributeSetUtilities**](http://docs.google.com/javax/print/attribute/AttributeSetUtilities.html) | Class AttributeSetUtilities provides static methods for manipulating AttributeSets. |
| [**DateTimeSyntax**](http://docs.google.com/javax/print/attribute/DateTimeSyntax.html) | Class DateTimeSyntax is an abstract base class providing the common implementation of all attributes whose value is a date and time. |
| [**EnumSyntax**](http://docs.google.com/javax/print/attribute/EnumSyntax.html) | Class EnumSyntax is an abstract base class providing the common implementation of all "type safe enumeration" objects. |
| [**HashAttributeSet**](http://docs.google.com/javax/print/attribute/HashAttributeSet.html) | Class HashAttributeSet provides an AttributeSet implementation with characteristics of a hash map. |
| [**HashDocAttributeSet**](http://docs.google.com/javax/print/attribute/HashDocAttributeSet.html) | Class HashDocAttributeSet provides an attribute set which inherits its implementation from class [HashAttributeSet](http://docs.google.com/javax/print/attribute/HashAttributeSet.html) and enforces the semantic restrictions of interface [DocAttributeSet](http://docs.google.com/javax/print/attribute/DocAttributeSet.html). |
| [**HashPrintJobAttributeSet**](http://docs.google.com/javax/print/attribute/HashPrintJobAttributeSet.html) | Class HashPrintJobAttributeSet provides an attribute set which inherits its implementation from class [HashAttributeSet](http://docs.google.com/javax/print/attribute/HashAttributeSet.html) and enforces the semantic restrictions of interface [PrintJobAttributeSet](http://docs.google.com/javax/print/attribute/PrintJobAttributeSet.html). |
| [**HashPrintRequestAttributeSet**](http://docs.google.com/javax/print/attribute/HashPrintRequestAttributeSet.html) | Class HashPrintRequestAttributeSet inherits its implementation from class [HashAttributeSet](http://docs.google.com/javax/print/attribute/HashAttributeSet.html) and enforces the semantic restrictions of interface [PrintRequestAttributeSet](http://docs.google.com/javax/print/attribute/PrintRequestAttributeSet.html). |
| [**HashPrintServiceAttributeSet**](http://docs.google.com/javax/print/attribute/HashPrintServiceAttributeSet.html) | Class HashPrintServiceAttributeSet provides an attribute set which inherits its implementation from class [HashAttributeSet](http://docs.google.com/javax/print/attribute/HashAttributeSet.html) and enforces the semantic restrictions of interface [PrintServiceAttributeSet](http://docs.google.com/javax/print/attribute/PrintServiceAttributeSet.html). |
| [**IntegerSyntax**](http://docs.google.com/javax/print/attribute/IntegerSyntax.html) | Class IntegerSyntax is an abstract base class providing the common implementation of all attributes with integer values. |
| [**ResolutionSyntax**](http://docs.google.com/javax/print/attribute/ResolutionSyntax.html) | Class ResolutionSyntax is an abstract base class providing the common implementation of all attributes denoting a printer resolution. |
| [**SetOfIntegerSyntax**](http://docs.google.com/javax/print/attribute/SetOfIntegerSyntax.html) | Class SetOfIntegerSyntax is an abstract base class providing the common implementation of all attributes whose value is a set of nonnegative integers. |
| [**Size2DSyntax**](http://docs.google.com/javax/print/attribute/Size2DSyntax.html) | Class Size2DSyntax is an abstract base class providing the common implementation of all attributes denoting a size in two dimensions. |
| [**TextSyntax**](http://docs.google.com/javax/print/attribute/TextSyntax.html) | Class TextSyntax is an abstract base class providing the common implementation of all attributes whose value is a string. |
| [**URISyntax**](http://docs.google.com/javax/print/attribute/URISyntax.html) | Class URISyntax is an abstract base class providing the common implementation of all attributes whose value is a Uniform Resource Identifier (URI). |

| **Exception Summary** | |
| --- | --- |
| [**UnmodifiableSetException**](http://docs.google.com/javax/print/attribute/UnmodifiableSetException.html) | Thrown to indicate that the requested operation cannot be performed because the set is unmodifiable. |

## Package javax.print.attribute Description

Provides classes and interfaces that describe the types of JavaTM Print Service attributes and how they can be collected into attribute sets.

### What is an Attribute?

When setting up a print job, a client specifies two things: **print data** and **processing instructions.** The print data is the actual content to be printed. The processing instructions tell the printer how to print the print data, such as: what media to use, how many copies to print, and whether to print on one or both sides of a sheet. The client specifies these processing instructions with the attribute definitions of the Java Print Service API.

The print data and the processing instructions are separate entities. This means that:

* You can print the same print data at different times using different processing instructions.  
  For example, you can print a slide presentation on US letter-sized white paper, double-sided, stapled, 20 copies to make handouts for a talk; and you could print the same slide presentation on US letter-sized transparencies, single-sided, one copy to make the actual slides for the talk.
* You can use the same processing instructions at different times to print different data. For example, you could set your default processing instructions to: US letter-sized paper, double sided, stapled. Whenever you print a job, it prints with these settings, unless you explicitly override them.

The processing instruction does not specify how the print job processes the request; each processing instruction is only a description of the results of a print job. The print job determines the manner in which it achieves the results specified by the processing instructions. Representing processing instructions as descriptive items provides more flexibility for implementing print jobs.

#### Attribute Categories and Values

Each printer has a set of capabilities, such as the ability to print on different paper sizes or the ability to print more than one copy. Each of the capabilities has a range of values. For example, a printer's orientation capability might have this range of values: [landscape, portrait]. For each print request, the capability is set to one of these values. The Java Print Service API uses the term **attribute category** to refer to a printer capability and the term **attribute value** to refer to the value of the capability.

In the Java Print Service API, an attribute category is represented by a Java class implementing the [Attribute](http://docs.google.com/Attribute.html) interface. Attribute values are instances of such a class or one of its subclasses. For example, to specify the number of copies, an application constructs an instance of the [Copies](http://docs.google.com/standard/Copies.html) class with the number of desired copies and uses the Copies instance as part of the print request. In this case, the Copies class represents the attribute category, and the Copies instance represents the attribute value.

#### Attribute Roles

When submitting a print job to a printer, the client provides the attributes describing the characteristics of the print data, such as the document name, and how the print data should be printed, such as double-sided, five copies. If a print job consists of multiple pieces of print data, different pieces might have different processing instructions, such as 8 x 11 inch media for the first document, and 11 x 17 inch media for another document.

Once the printer starts processing the print job, additional information about the job becomes available, which might include: the job state (such as *completed* or *queued*) and the number of pages printed so far. These pieces of information are also attributes. Attributes can also describe the printer itself, such as: the printer name, the printer location, and the number of jobs queued.

The Java Print Service API defines these different kinds of attributes with five subinterfaces of Attribute:

* [DocAttribute](http://docs.google.com/DocAttribute.html) specifies a characteristic of an individual document and the print job settings to be applied to an individual document.
* [PrintRequestAttribute](http://docs.google.com/PrintRequestAttribute.html) specifies a setting applied to a whole print job and to all the documents in the print job.
* [PrintJobAttribute](http://docs.google.com/PrintJobAttribute.html) reports the status of a print job.
* [PrintServiceAttribute](http://docs.google.com/PrintServiceAttribute.html) reports the status of a print service.
* [SupportedValuesAttribute](http://docs.google.com/SupportedValuesAttribute.html) gives the supported values for another attribute.

Each attribute class implements one or more of these tagging subinterfaces to indicate where the attribute can be used in the API. If an attribute class implements multiple tagging subinterfaces, the attribute can be used in multiple contexts. For example, the media attribute can apply to one document in a print job as a DocAttribute or to an entire print job as a PrintRequestAttribute. Certain low-level attributes are never used on their own but are always aggregated into higher-level attributes. These low-level attribute classes only implement interface [Attribute](http://docs.google.com/Attribute.html), not any of the tagging subinterfaces.

The Java Print Service API defines a group of standard attribute classes modeled upon the attributes in the Internet Printing Protocol (IPP) version 1.1. The standard attribute classes are in the subpackage javax.print.attribute.standard to keep the actual attribute classes conceptually separate from the generic apparatus defined in package javax.print.attribute.

### Attribute Sets

A client usually needs to provide more than one processing instruction when submitting a print job. For example, the client might need to specify a media size of A4 and a landscape orientation. To send more than one processing instruction, the client collects the attributes into an attribute set, which the Java Print Service API represents with the [AttributeSet](http://docs.google.com/AttributeSet.html) interface.

The AttributeSet interface is similar to the [Map](http://docs.google.com/java/util/Map.html) interface: it provides a map of key to values, in which each key is unique and can contain no more than one value. However, the AttributeSet interface is designed to specifically support the needs of the Java Print Service API. An AttributeSet requires that:

1. Each key in an AttributeSet corresponds to a category, and the value of the key can only be one of the attribute values that belong to the category represented by the key. Thus, unlike a Map, an AttributeSet restricts the possible values of a key: an attribute category cannot be set to an attribute value that does not belong to that category.
2. No two attributes from the same category can exist in the same set. For example, an attribute collection must not contain both a "one-sided" attribute and a "two-sided" attribute because these two attributes give the printer conflicting instructions.
3. Only attributes implementing the Attribute interface can be added to the set.

The javax.print.attribute package includes [HashAttributeSet](http://docs.google.com/HashAttributeSet.html) as a concrete implementation of the attribute set interface. HashAttributeSet provides an attribute set based on a hash map. You can use this implementation or provide your own implementation of interface AttributeSet.

The Java Print Service API provides four specializations of an attribute set that are restricted to contain just one of the four kinds of attributes, as discussed in the [Attribute Roles](#_2et92p0) section:

* [DocAttributeSet](http://docs.google.com/DocAttributeSet.html)
* [PrintRequestAttributeSet](http://docs.google.com/PrintRequestAttributeSet.html)
* [PrintJobAttributeSet](http://docs.google.com/PrintJobAttributeSet.html)
* [PrintServiceAttributeSet](http://docs.google.com/PrintServiceAttributeSet.html)

Notice that only four kinds of attribute sets are listed here, but there are five kinds of attributes. Interface [SupportedValuesAttribute](http://docs.google.com/SupportedValuesAttribute.html) denotes an attribute that gives the supported values for another attribute. Supported-values attributes are never aggregated into attribute sets, so there is no attribute set subinterface defined for them.

In some contexts, an attribute set is read-only, which means that the client is only allowed to examine an attribute set's contents but not change them. In other contexts, the attribute set is read-write, which means that the client is allowed both to examine and to change an attribute set's contents. For a read-only attribute set, calling a mutating operation throws an UnmodifiableSetException.

Package javax.print.attribute includes one concrete implementation of each of the attribute set subinterfaces:

* [HashDocAttributeSet](http://docs.google.com/HashDocAttributeSet.html)
* [HashPrintRequestAttributeSet](http://docs.google.com/HashPrintRequestAttributeSet.html),
* [HashPrintJobAttributeSet](http://docs.google.com/HashPrintJobAttributeSet.html),
* [HashPrintServiceAttributeSet](http://docs.google.com/HashPrintServiceAttributeSet.html).

All of these classes extend [HashAttributeSet](http://docs.google.com/HashAttributeSet.html) and enforce the restriction that the attribute set is only allowed to contain the corresponding kind of attribute.

### Attribute Class Design

An attribute value is a small, atomic data item, such as an integer or an enumerated value. The Java Print Service API does not use primitive data types, such as int, to represent attribute values for these reasons:

* Primitive data types are not type-safe. For example, a compiler should not allow a "copies" attribute value to be used for a "sides" attribute.
* Some attributes must be represented as a record of several values. One example is printer resolution, which requires two numbers, such as 600 and 300 representing 600 x 300 dpi.

For type-safety and to represent all attributes uniformly, the Java Print Service API defines each attribute category as a class, such as class Copies, class [Sides](http://docs.google.com/standard/Sides.html), and class [PrinterResolution](http://docs.google.com/standard/PrinterResolution.html). Each attribute class wraps one or more primitive data items containing the attribute's value. Attribute set operations perform frequent comparisons between attribute category objects when adding attributes, finding existing attributes in the same category, and looking up an attribute given its category. Because an attribute category is represented by a class, fast attribute-value comparisons can be performed with the Class.equals method.

Even though the Java Print Service API includes a large number of different attribute categories, there are only a few different types of attribute values. Most attributes can be represented by a small number of data types, such as: integer values, integer ranges, text, or an enumeration of integer values. The type of the attribute value that a category accepts is called the attribute's abstract syntax. To provide consistency and reduce code duplication, the Java Print Service API defines abstract syntax classes to represent each abstract syntax, and these classes are used as the parent of standard attributes whenever possible. The abstract syntax classes are:

* [EnumSyntax](http://docs.google.com/EnumSyntax.html) provides a type-safe enumeration in which enumerated values are represented as singleton objects. Each enumeration singleton is an instance of the enumeration class that wraps a hidden int value.
* [IntegerSyntax](http://docs.google.com/IntegerSyntax.html) is the abstract syntax for integer-valued attributes.
* [TextSyntax](http://docs.google.com/TextSyntax.html) is the abstract syntax for text-valued attributes, and includes a locale giving the text string's natural language.
* [SetOfIntegerSyntax](http://docs.google.com/SetOfIntegerSyntax.html) is the abstract syntax for attributes representing a range or set of integers
* [ResolutionSyntax](http://docs.google.com/ResolutionSyntax.html) is the abstract syntax for attributes representing resolution values, such as 600x300 dpi.
* [Size2DSyntax](http://docs.google.com/Size2DSyntax.html) is the abstract syntax for attributes representing a two-dimensional size, such as a paper size of 8.5 x 11 inches.
* [DateTimeSyntax](http://docs.google.com/DateTimeSyntax.html) is the abstract syntax for attributes whose value is a date and time.
* [URISyntax](http://docs.google.com/URISyntax.html) is the abstract syntax for attributes whose value is a Uniform Resource Indicator.

The abstract syntax classes are independent of the attributes that use them. In fact, applications that have nothing to do with printing can use the abstract syntax classes. Although most of the standard attribute classes extend one of the abstract syntax classes, no attribute class is required to extend one of these classes. The abstract syntax classes merely provide a convenient implementation that can be shared by many attribute classes.

Each attribute class implements the Attribute interface, either directly or indirectly, to mark it as a printing attribute. An attribute class that can appear in restricted attribute sets in certain contexts also implements one or more subinterfaces of Attribute. Most attribute classes also extend the appropriate abstract syntax class to get the implementation. Consider the Sides attribute class:

public class Sides  
 extends EnumSyntax  
 implements DocAttribute, PrintRequestAttribute, PrintJobAttribute  
 {  
 public final Object getCategory()  
 {  
 return Sides.class;  
 }  
 ...  
 }

Since every attribute class implements Attribute, every attribute class must provide an implementation for the [getCategory](http://docs.google.com/javax/print/attribute/Attribute.html#getCategory()) method, which returns the attribute category. In the case of Sides, the getCategory method returns Sides.class. The getCategory method is final to ensure that any vendor-defined subclasses of a standard attribute class appear in the same category. Every attribute object is immutable once constructed so that attribute object references can be passed around freely. To get a different attribute value, construct a different attribute object.

### Attribute Vendors

The Java Print Service API is designed so that vendors can:

* define new vendor-specific values for any standard attribute defined in  [javax.print.attribute.standard](http://docs.google.com/standard/package-summary.html).
* define new attribute categories representing the vendor printer's proprietary capabilities not already supported by the standard attributes.

To define a new value for an attribute, a client can construct instances of such attributes with arbitrary values at runtime. However, an enumerated attribute using an abstract syntax class of EnumSyntax specifies all the possible attribute values at compile time as singleton instances of the attribute class. This means that new enumerated values cannot be constructed at run time. To define new vendor-specific values for a standard enumerated attribute, the vendor must define a new attribute class specifying the new singleton instances. To ensure that the new attribute values fall in the same category as the standard attribute values, the new attribute class must be a subclass of the standard attribute class.

To define a new attribute category, a vendor defines a new attribute class. This attribute class, like the standard attribute classes, implements Attribute or one of its subinterfaces and extends an abstract syntax class. The vendor can either use an existing abstract syntax class or define a new one. The new vendor-defined attribute can be used wherever an Attribute is used, such as in an AttributeSet.

### Using Attributes

A typical printing application uses the PrintRequestAttributeSet because print-request attributes are the types of attributes that client usually specifies. This example demonstrates creating an attribute set of print-request attributes and locating a printer that can print the document according to the specified attributes:

FileInputStream psStream;  
try {  
 psstream = new FileInputStream("file.ps");  
} catch (FileNotFoundException ffne) {  
}  
if (psstream == null) {  
 return;  
}  
//Set the document type. See the DocFlavor documentation for  
//more information.  
DocFlavor psInFormat = DocFlavor.INPUT\_STREAM.POSTSCRIPT;  
Doc myDoc = new SimpleDoc(pstream, psInFormat, null);   
PrintRequestAttributeSet aset = new HashPrintRequestAttributeSet();  
aset.add(new Copies(5));  
aset.add(MediaSize.A4);  
aset.add(Sides.DUPLEX);  
  
PrintService[] services =   
 PrintServiceLookup.lookupPrintServices(psInFormat, aset);  
if (services.length > 0) {  
 DocPrintJob job = services[0].createPrintJob();  
 try {  
 job.print(myDoc, aset);  
 } catch (PrintException pe) {}  
}

Please note: In the javax.print APIs, a null reference parameter to methods is incorrect unless explicitly documented on the method as having a meaningful interpretation. Usage to the contrary is incorrect coding and may result in a run time exception either immediately or at some later time. IllegalArgumentException and NullPointerException are examples of typical and acceptable run time exceptions for such cases.

**Since:** 1.4

| | [**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/javax/print/package-summary.html)   [**NEXT PACKAGE**](http://docs.google.com/javax/print/attribute/standard/package-summary.html) | [**FRAMES**](http://docs.google.com/index.html?javax/print/attribute/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).