| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Area.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/awt/geom/Arc2D.Float.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/geom/CubicCurve2D.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/geom/Area.html)    [**NO FRAMES**](http://docs.google.com/Area.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **java.awt.geom**

Class Area

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **java.awt.geom.Area**

**All Implemented Interfaces:** [Shape](http://docs.google.com/java/awt/Shape.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html)

public class **Area**extends [Object](http://docs.google.com/java/lang/Object.html)implements [Shape](http://docs.google.com/java/awt/Shape.html), [Cloneable](http://docs.google.com/java/lang/Cloneable.html)

An Area object stores and manipulates a resolution-independent description of an enclosed area of 2-dimensional space. Area objects can be transformed and can perform various Constructive Area Geometry (CAG) operations when combined with other Area objects. The CAG operations include area [addition](http://docs.google.com/java/awt/geom/Area.html#add(java.awt.geom.Area)), [subtraction](http://docs.google.com/java/awt/geom/Area.html#subtract(java.awt.geom.Area)), [intersection](http://docs.google.com/java/awt/geom/Area.html#intersect(java.awt.geom.Area)), and [exclusive or](http://docs.google.com/java/awt/geom/Area.html#exclusiveOr(java.awt.geom.Area)). See the linked method documentation for examples of the various operations.

The Area class implements the Shape interface and provides full support for all of its hit-testing and path iteration facilities, but an Area is more specific than a generalized path in a number of ways:

* Only closed paths and sub-paths are stored. Area objects constructed from unclosed paths are implicitly closed during construction as if those paths had been filled by the Graphics2D.fill method.
* The interiors of the individual stored sub-paths are all non-empty and non-overlapping. Paths are decomposed during construction into separate component non-overlapping parts, empty pieces of the path are discarded, and then these non-empty and non-overlapping properties are maintained through all subsequent CAG operations. Outlines of different component sub-paths may touch each other, as long as they do not cross so that their enclosed areas overlap.
* The geometry of the path describing the outline of the Area resembles the path from which it was constructed only in that it describes the same enclosed 2-dimensional area, but may use entirely different types and ordering of the path segments to do so.

Interesting issues which are not always obvious when using the Area include:

* Creating an Area from an unclosed (open) Shape results in a closed outline in the Area object.
* Creating an Area from a Shape which encloses no area (even when "closed") produces an empty Area. A common example of this issue is that producing an Area from a line will be empty since the line encloses no area. An empty Area will iterate no geometry in its PathIterator objects.
* A self-intersecting Shape may be split into two (or more) sub-paths each enclosing one of the non-intersecting portions of the original path.
* An Area may take more path segments to describe the same geometry even when the original outline is simple and obvious. The analysis that the Area class must perform on the path may not reflect the same concepts of "simple and obvious" as a human being perceives.

**Since:** 1.2

| **Constructor Summary** | |
| --- | --- |
| [**Area**](http://docs.google.com/java/awt/geom/Area.html#Area())()            Default constructor which creates an empty area. |
| [**Area**](http://docs.google.com/java/awt/geom/Area.html#Area(java.awt.Shape))([Shape](http://docs.google.com/java/awt/Shape.html) s)            The Area class creates an area geometry from the specified [Shape](http://docs.google.com/java/awt/Shape.html) object. |

| **Method Summary** | |
| --- | --- |
| void | [**add**](http://docs.google.com/java/awt/geom/Area.html#add(java.awt.geom.Area))([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)            Adds the shape of the specified Area to the shape of this Area. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**clone**](http://docs.google.com/java/awt/geom/Area.html#clone())()            Returns an exact copy of this Area object. |
| boolean | [**contains**](http://docs.google.com/java/awt/geom/Area.html#contains(double,%20double))(double x, double y)            Tests if the specified coordinates are inside the boundary of the Shape. |
| boolean | [**contains**](http://docs.google.com/java/awt/geom/Area.html#contains(double,%20double,%20double,%20double))(double x, double y, double w, double h)            Tests if the interior of the Shape entirely contains the specified rectangular area. |
| boolean | [**contains**](http://docs.google.com/java/awt/geom/Area.html#contains(java.awt.geom.Point2D))([Point2D](http://docs.google.com/java/awt/geom/Point2D.html) p)            Tests if a specified [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) is inside the boundary of the Shape. |
| boolean | [**contains**](http://docs.google.com/java/awt/geom/Area.html#contains(java.awt.geom.Rectangle2D))([Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) r)            Tests if the interior of the Shape entirely contains the specified Rectangle2D. |
| [Area](http://docs.google.com/java/awt/geom/Area.html) | [**createTransformedArea**](http://docs.google.com/java/awt/geom/Area.html#createTransformedArea(java.awt.geom.AffineTransform))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) t)            Creates a new Area object that contains the same geometry as this Area transformed by the specified AffineTransform. |
| boolean | [**equals**](http://docs.google.com/java/awt/geom/Area.html#equals(java.awt.geom.Area))([Area](http://docs.google.com/java/awt/geom/Area.html) other)            Tests whether the geometries of the two Area objects are equal. |
| void | [**exclusiveOr**](http://docs.google.com/java/awt/geom/Area.html#exclusiveOr(java.awt.geom.Area))([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)            Sets the shape of this Area to be the combined area of its current shape and the shape of the specified Area, minus their intersection. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getBounds**](http://docs.google.com/java/awt/geom/Area.html#getBounds())()            Returns a bounding [Rectangle](http://docs.google.com/java/awt/Rectangle.html) that completely encloses this Area. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/geom/Area.html#getBounds2D())()            Returns a high precision bounding [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) that completely encloses this Area. |
| [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) | [**getPathIterator**](http://docs.google.com/java/awt/geom/Area.html#getPathIterator(java.awt.geom.AffineTransform))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) at)            Creates a [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) for the outline of this Area object. |
| [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) | [**getPathIterator**](http://docs.google.com/java/awt/geom/Area.html#getPathIterator(java.awt.geom.AffineTransform,%20double))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) at, double flatness)            Creates a PathIterator for the flattened outline of this Area object. |
| void | [**intersect**](http://docs.google.com/java/awt/geom/Area.html#intersect(java.awt.geom.Area))([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)            Sets the shape of this Area to the intersection of its current shape and the shape of the specified Area. |
| boolean | [**intersects**](http://docs.google.com/java/awt/geom/Area.html#intersects(double,%20double,%20double,%20double))(double x, double y, double w, double h)            Tests if the interior of the Shape intersects the interior of a specified rectangular area. |
| boolean | [**intersects**](http://docs.google.com/java/awt/geom/Area.html#intersects(java.awt.geom.Rectangle2D))([Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) r)            Tests if the interior of the Shape intersects the interior of a specified Rectangle2D. |
| boolean | [**isEmpty**](http://docs.google.com/java/awt/geom/Area.html#isEmpty())()            Tests whether this Area object encloses any area. |
| boolean | [**isPolygonal**](http://docs.google.com/java/awt/geom/Area.html#isPolygonal())()            Tests whether this Area consists entirely of straight edged polygonal geometry. |
| boolean | [**isRectangular**](http://docs.google.com/java/awt/geom/Area.html#isRectangular())()            Tests whether this Area is rectangular in shape. |
| boolean | [**isSingular**](http://docs.google.com/java/awt/geom/Area.html#isSingular())()            Tests whether this Area is comprised of a single closed subpath. |
| void | [**reset**](http://docs.google.com/java/awt/geom/Area.html#reset())()            Removes all of the geometry from this Area and restores it to an empty area. |
| void | [**subtract**](http://docs.google.com/java/awt/geom/Area.html#subtract(java.awt.geom.Area))([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)            Subtracts the shape of the specified Area from the shape of this Area. |
| void | [**transform**](http://docs.google.com/java/awt/geom/Area.html#transform(java.awt.geom.AffineTransform))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) t)            Transforms the geometry of this Area using the specified [AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html). |

| **Methods inherited from class java.lang.**[**Object**](http://docs.google.com/java/lang/Object.html) |
| --- |
| [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)) |

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

### Area

public **Area**()

Default constructor which creates an empty area.

**Since:** 1.2

### Area

public **Area**([Shape](http://docs.google.com/java/awt/Shape.html) s)

The Area class creates an area geometry from the specified [Shape](http://docs.google.com/java/awt/Shape.html) object. The geometry is explicitly closed, if the Shape is not already closed. The fill rule (even-odd or winding) specified by the geometry of the Shape is used to determine the resulting enclosed area.

**Parameters:**s - the Shape from which the area is constructed **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if s is null**Since:** 1.2

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

### add

public void **add**([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)

Adds the shape of the specified Area to the shape of this Area. The resulting shape of this Area will include the union of both shapes, or all areas that were contained in either this or the specified Area.

// Example:  
 Area a1 = new Area([triangle 0,0 => 8,0 => 0,8]);  
 Area a2 = new Area([triangle 0,0 => 8,0 => 8,8]);  
 a1.add(a2);  
  
 a1(before) + a2 = a1(after)  
  
 ################ ################ ################  
 ############## ############## ################  
 ############ ############ ################  
 ########## ########## ################  
 ######## ######## ################  
 ###### ###### ###### ######  
 #### #### #### ####  
 ## ## ## ##

**Parameters:**rhs - the Area to be added to the current shape **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if rhs is null**Since:** 1.2

### subtract

public void **subtract**([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)

Subtracts the shape of the specified Area from the shape of this Area. The resulting shape of this Area will include areas that were contained only in this Area and not in the specified Area.

// Example:  
 Area a1 = new Area([triangle 0,0 => 8,0 => 0,8]);  
 Area a2 = new Area([triangle 0,0 => 8,0 => 8,8]);  
 a1.subtract(a2);  
  
 a1(before) - a2 = a1(after)  
  
 ################ ################  
 ############## ############## ##  
 ############ ############ ####  
 ########## ########## ######  
 ######## ######## ########  
 ###### ###### ######  
 #### #### ####  
 ## ## ##

**Parameters:**rhs - the Area to be subtracted from the current shape **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if rhs is null**Since:** 1.2

### intersect

public void **intersect**([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)

Sets the shape of this Area to the intersection of its current shape and the shape of the specified Area. The resulting shape of this Area will include only areas that were contained in both this Area and also in the specified Area.

// Example:  
 Area a1 = new Area([triangle 0,0 => 8,0 => 0,8]);  
 Area a2 = new Area([triangle 0,0 => 8,0 => 8,8]);  
 a1.intersect(a2);  
  
 a1(before) intersect a2 = a1(after)  
  
 ################ ################ ################  
 ############## ############## ############  
 ############ ############ ########  
 ########## ########## ####  
 ######## ########  
 ###### ######  
 #### ####  
 ## ##

**Parameters:**rhs - the Area to be intersected with this Area **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if rhs is null**Since:** 1.2

### exclusiveOr

public void **exclusiveOr**([Area](http://docs.google.com/java/awt/geom/Area.html) rhs)

Sets the shape of this Area to be the combined area of its current shape and the shape of the specified Area, minus their intersection. The resulting shape of this Area will include only areas that were contained in either this Area or in the specified Area, but not in both.

// Example:  
 Area a1 = new Area([triangle 0,0 => 8,0 => 0,8]);  
 Area a2 = new Area([triangle 0,0 => 8,0 => 8,8]);  
 a1.exclusiveOr(a2);  
  
 a1(before) xor a2 = a1(after)  
  
 ################ ################  
 ############## ############## ## ##  
 ############ ############ #### ####  
 ########## ########## ###### ######  
 ######## ######## ################  
 ###### ###### ###### ######  
 #### #### #### ####  
 ## ## ## ##

**Parameters:**rhs - the Area to be exclusive ORed with this Area. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if rhs is null**Since:** 1.2

### reset

public void **reset**()

Removes all of the geometry from this Area and restores it to an empty area.

**Since:** 1.2

### isEmpty

public boolean **isEmpty**()

Tests whether this Area object encloses any area.

**Returns:**true if this Area object represents an empty area; false otherwise.**Since:** 1.2

### isPolygonal

public boolean **isPolygonal**()

Tests whether this Area consists entirely of straight edged polygonal geometry.

**Returns:**true if the geometry of this Area consists entirely of line segments; false otherwise.**Since:** 1.2

### isRectangular

public boolean **isRectangular**()

Tests whether this Area is rectangular in shape.

**Returns:**true if the geometry of this Area is rectangular in shape; false otherwise.**Since:** 1.2

### isSingular

public boolean **isSingular**()

Tests whether this Area is comprised of a single closed subpath. This method returns true if the path contains 0 or 1 subpaths, or false if the path contains more than 1 subpath. The subpaths are counted by the number of [SEG\_MOVETO](http://docs.google.com/java/awt/geom/PathIterator.html#SEG_MOVETO) segments that appear in the path.

**Returns:**true if the Area is comprised of a single basic geometry; false otherwise.**Since:** 1.2

### getBounds2D

public [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) **getBounds2D**()

Returns a high precision bounding [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) that completely encloses this Area.

The Area class will attempt to return the tightest bounding box possible for the Shape. The bounding box will not be padded to include the control points of curves in the outline of the Shape, but should tightly fit the actual geometry of the outline itself.

**Specified by:**[getBounds2D](http://docs.google.com/java/awt/Shape.html#getBounds2D()) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Returns:**the bounding Rectangle2D for the Area.**Since:** 1.2 **See Also:**[Shape.getBounds()](http://docs.google.com/java/awt/Shape.html#getBounds())

### getBounds

public [Rectangle](http://docs.google.com/java/awt/Rectangle.html) **getBounds**()

Returns a bounding [Rectangle](http://docs.google.com/java/awt/Rectangle.html) that completely encloses this Area.

The Area class will attempt to return the tightest bounding box possible for the Shape. The bounding box will not be padded to include the control points of curves in the outline of the Shape, but should tightly fit the actual geometry of the outline itself. Since the returned object represents the bounding box with integers, the bounding box can only be as tight as the nearest integer coordinates that encompass the geometry of the Shape.

**Specified by:**[getBounds](http://docs.google.com/java/awt/Shape.html#getBounds()) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Returns:**the bounding Rectangle for the Area.**Since:** 1.2 **See Also:**[Shape.getBounds2D()](http://docs.google.com/java/awt/Shape.html#getBounds2D())

### clone

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

Returns an exact copy of this Area object.

**Overrides:**[clone](http://docs.google.com/java/lang/Object.html#clone()) in class [Object](http://docs.google.com/java/lang/Object.html) **Returns:**Created clone object**Since:** 1.2 **See Also:**[Cloneable](http://docs.google.com/java/lang/Cloneable.html)

### equals

public boolean **equals**([Area](http://docs.google.com/java/awt/geom/Area.html) other)

Tests whether the geometries of the two Area objects are equal. This method will return false if the argument is null.

**Parameters:**other - the Area to be compared to this Area **Returns:**true if the two geometries are equal; false otherwise.**Since:** 1.2

### transform

public void **transform**([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) t)

Transforms the geometry of this Area using the specified [AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html). The geometry is transformed in place, which permanently changes the enclosed area defined by this object.

**Parameters:**t - the transformation used to transform the area **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if t is null**Since:** 1.2

### createTransformedArea

public [Area](http://docs.google.com/java/awt/geom/Area.html) **createTransformedArea**([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) t)

Creates a new Area object that contains the same geometry as this Area transformed by the specified AffineTransform. This Area object is unchanged.

**Parameters:**t - the specified AffineTransform used to transform the new Area **Returns:**a new Area object representing the transformed geometry. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if t is null**Since:** 1.2

### contains

public boolean **contains**(double x,  
 double y)

Tests if the specified coordinates are inside the boundary of the Shape.

**Specified by:**[contains](http://docs.google.com/java/awt/Shape.html#contains(double,%20double)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**x - the specified X coordinate to be testedy - the specified Y coordinate to be tested **Returns:**true if the specified coordinates are inside the Shape boundary; false otherwise.**Since:** 1.2

### contains

public boolean **contains**([Point2D](http://docs.google.com/java/awt/geom/Point2D.html) p)

Tests if a specified [Point2D](http://docs.google.com/java/awt/geom/Point2D.html) is inside the boundary of the Shape.

**Specified by:**[contains](http://docs.google.com/java/awt/Shape.html#contains(java.awt.geom.Point2D)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**p - the specified Point2D to be tested **Returns:**true if the specified Point2D is inside the boundary of the Shape; false otherwise.**Since:** 1.2

### contains

public boolean **contains**(double x,  
 double y,  
 double w,  
 double h)

Tests if the interior of the Shape entirely contains the specified rectangular area. All coordinates that lie inside the rectangular area must lie within the Shape for the entire rectanglar area to be considered contained within the Shape.

The Shape.contains() method allows a Shape implementation to conservatively return false when:

* the intersect method returns true and
* the calculations to determine whether or not the Shape entirely contains the rectangular area are prohibitively expensive.

This means that for some Shapes this method might return false even though the Shape contains the rectangular area. The [Area](http://docs.google.com/java/awt/geom/Area.html) class performs more accurate geometric computations than most Shape objects and therefore can be used if a more precise answer is required.

**Specified by:**[contains](http://docs.google.com/java/awt/Shape.html#contains(double,%20double,%20double,%20double)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**x - the X coordinate of the upper-left corner of the specified rectangular areay - the Y coordinate of the upper-left corner of the specified rectangular areaw - the width of the specified rectangular areah - the height of the specified rectangular area **Returns:**true if the interior of the Shape entirely contains the specified rectangular area; false otherwise or, if the Shape contains the rectangular area and the intersects method returns true and the containment calculations would be too expensive to perform.**Since:** 1.2 **See Also:**[Area](http://docs.google.com/java/awt/geom/Area.html), [Shape.intersects(double, double, double, double)](http://docs.google.com/java/awt/Shape.html#intersects(double,%20double,%20double,%20double))

### contains

public boolean **contains**([Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) r)

Tests if the interior of the Shape entirely contains the specified Rectangle2D. The Shape.contains() method allows a Shape implementation to conservatively return false when:

* the intersect method returns true and
* the calculations to determine whether or not the Shape entirely contains the Rectangle2D are prohibitively expensive.

This means that for some Shapes this method might return false even though the Shape contains the Rectangle2D. The [Area](http://docs.google.com/java/awt/geom/Area.html) class performs more accurate geometric computations than most Shape objects and therefore can be used if a more precise answer is required.

**Specified by:**[contains](http://docs.google.com/java/awt/Shape.html#contains(java.awt.geom.Rectangle2D)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**r - The specified Rectangle2D **Returns:**true if the interior of the Shape entirely contains the Rectangle2D; false otherwise or, if the Shape contains the Rectangle2D and the intersects method returns true and the containment calculations would be too expensive to perform.**Since:** 1.2 **See Also:**[Shape.contains(double, double, double, double)](http://docs.google.com/java/awt/Shape.html#contains(double,%20double,%20double,%20double))

### intersects

public boolean **intersects**(double x,  
 double y,  
 double w,  
 double h)

Tests if the interior of the Shape intersects the interior of a specified rectangular area. The rectangular area is considered to intersect the Shape if any point is contained in both the interior of the Shape and the specified rectangular area.

The Shape.intersects() method allows a Shape implementation to conservatively return true when:

* there is a high probability that the rectangular area and the Shape intersect, but
* the calculations to accurately determine this intersection are prohibitively expensive.

This means that for some Shapes this method might return true even though the rectangular area does not intersect the Shape. The [Area](http://docs.google.com/java/awt/geom/Area.html) class performs more accurate computations of geometric intersection than most Shape objects and therefore can be used if a more precise answer is required.

**Specified by:**[intersects](http://docs.google.com/java/awt/Shape.html#intersects(double,%20double,%20double,%20double)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**x - the X coordinate of the upper-left corner of the specified rectangular areay - the Y coordinate of the upper-left corner of the specified rectangular areaw - the width of the specified rectangular areah - the height of the specified rectangular area **Returns:**true if the interior of the Shape and the interior of the rectangular area intersect, or are both highly likely to intersect and intersection calculations would be too expensive to perform; false otherwise.**Since:** 1.2 **See Also:**[Area](http://docs.google.com/java/awt/geom/Area.html)

### intersects

public boolean **intersects**([Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) r)

Tests if the interior of the Shape intersects the interior of a specified Rectangle2D. The Shape.intersects() method allows a Shape implementation to conservatively return true when:

* there is a high probability that the Rectangle2D and the Shape intersect, but
* the calculations to accurately determine this intersection are prohibitively expensive.

This means that for some Shapes this method might return true even though the Rectangle2D does not intersect the Shape. The [Area](http://docs.google.com/java/awt/geom/Area.html) class performs more accurate computations of geometric intersection than most Shape objects and therefore can be used if a more precise answer is required.

**Specified by:**[intersects](http://docs.google.com/java/awt/Shape.html#intersects(java.awt.geom.Rectangle2D)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**r - the specified Rectangle2D **Returns:**true if the interior of the Shape and the interior of the specified Rectangle2D intersect, or are both highly likely to intersect and intersection calculations would be too expensive to perform; false otherwise.**Since:** 1.2 **See Also:**[Shape.intersects(double, double, double, double)](http://docs.google.com/java/awt/Shape.html#intersects(double,%20double,%20double,%20double))

### getPathIterator

public [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) **getPathIterator**([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) at)

Creates a [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) for the outline of this Area object. This Area object is unchanged.

**Specified by:**[getPathIterator](http://docs.google.com/java/awt/Shape.html#getPathIterator(java.awt.geom.AffineTransform)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**at - an optional AffineTransform to be applied to the coordinates as they are returned in the iteration, or null if untransformed coordinates are desired **Returns:**the PathIterator object that returns the geometry of the outline of this Area, one segment at a time.**Since:** 1.2

### getPathIterator

public [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) **getPathIterator**([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) at,  
 double flatness)

Creates a PathIterator for the flattened outline of this Area object. Only uncurved path segments represented by the SEG\_MOVETO, SEG\_LINETO, and SEG\_CLOSE point types are returned by the iterator. This Area object is unchanged.

**Specified by:**[getPathIterator](http://docs.google.com/java/awt/Shape.html#getPathIterator(java.awt.geom.AffineTransform,%20double)) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Parameters:**at - an optional AffineTransform to be applied to the coordinates as they are returned in the iteration, or null if untransformed coordinates are desiredflatness - the maximum amount that the control points for a given curve can vary from colinear before a subdivided curve is replaced by a straight line connecting the end points **Returns:**the PathIterator object that returns the geometry of the outline of this Area, one segment at a time.**Since:** 1.2

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Area.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/awt/geom/Arc2D.Float.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/geom/CubicCurve2D.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/geom/Area.html)    [**NO FRAMES**](http://docs.google.com/Area.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

[Submit a bug or feature](http://bugs.sun.com/services/bugreport/index.jsp)

For further API reference and developer documentation, see [Java SE Developer Documentation](http://docs.google.com/webnotes/devdocs-vs-specs.html). That documentation contains more detailed, developer-targeted descriptions, with conceptual overviews, definitions of terms, workarounds, and working code examples.

Copyright 2006 Sun Microsystems, Inc. All rights reserved. Use is subject to [license terms](http://docs.google.com/legal/license.html). Also see the [documentation redistribution policy](http://java.sun.com/docs/redist.html).