| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/Polygon.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/PointerInfo.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/PopupMenu.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/Polygon.html)    [**NO FRAMES**](http://docs.google.com/Polygon.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#1ksv4uv) |

## **java.awt**

Class Polygon

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

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

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

The Polygon class encapsulates a description of a closed, two-dimensional region within a coordinate space. This region is bounded by an arbitrary number of line segments, each of which is one side of the polygon. Internally, a polygon comprises of a list of (x,y) coordinate pairs, where each pair defines a *vertex* of the polygon, and two successive pairs are the endpoints of a line that is a side of the polygon. The first and final pairs of (x,y) points are joined by a line segment that closes the polygon. This Polygon is defined with an even-odd winding rule. See [WIND\_EVEN\_ODD](http://docs.google.com/java/awt/geom/PathIterator.html#WIND_EVEN_ODD) for a definition of the even-odd winding rule. This class's hit-testing methods, which include the contains, intersects and inside methods, use the *insideness* definition described in the [Shape](http://docs.google.com/java/awt/Shape.html) class comments.

**Since:** 1.0 **See Also:**[Shape](http://docs.google.com/java/awt/Shape.html), [Serialized Form](http://docs.google.com/serialized-form.html#java.awt.Polygon)

| **Field Summary** | |
| --- | --- |
| protected  [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**bounds**](http://docs.google.com/java/awt/Polygon.html#bounds)            The bounds of this Polygon. |
| int | [**npoints**](http://docs.google.com/java/awt/Polygon.html#npoints)            The total number of points. |
| int[] | [**xpoints**](http://docs.google.com/java/awt/Polygon.html#xpoints)            The array of X coordinates. |
| int[] | [**ypoints**](http://docs.google.com/java/awt/Polygon.html#ypoints)            The array of Y coordinates. |

| **Constructor Summary** | |
| --- | --- |
| [**Polygon**](http://docs.google.com/java/awt/Polygon.html#Polygon())()            Creates an empty polygon. |
| [**Polygon**](http://docs.google.com/java/awt/Polygon.html#Polygon(int%5B%5D,%20int%5B%5D,%20int))(int[] xpoints, int[] ypoints, int npoints)            Constructs and initializes a Polygon from the specified parameters. |

| **Method Summary** | |
| --- | --- |
| void | [**addPoint**](http://docs.google.com/java/awt/Polygon.html#addPoint(int,%20int))(int x, int y)            Appends the specified coordinates to this Polygon. |
| boolean | [**contains**](http://docs.google.com/java/awt/Polygon.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/Polygon.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/Polygon.html#contains(int,%20int))(int x, int y)            Determines whether the specified coordinates are inside this Polygon. |
| boolean | [**contains**](http://docs.google.com/java/awt/Polygon.html#contains(java.awt.Point))([Point](http://docs.google.com/java/awt/Point.html) p)            Determines whether the specified [Point](http://docs.google.com/java/awt/Point.html) is inside this Polygon. |
| boolean | [**contains**](http://docs.google.com/java/awt/Polygon.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/Polygon.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. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getBoundingBox**](http://docs.google.com/java/awt/Polygon.html#getBoundingBox())()  **Deprecated.** *As of JDK version 1.1, replaced by getBounds().* |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getBounds**](http://docs.google.com/java/awt/Polygon.html#getBounds())()            Gets the bounding box of this Polygon. |
| [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) | [**getBounds2D**](http://docs.google.com/java/awt/Polygon.html#getBounds2D())()            Returns a high precision and more accurate bounding box of the Shape than the getBounds method. |
| [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) | [**getPathIterator**](http://docs.google.com/java/awt/Polygon.html#getPathIterator(java.awt.geom.AffineTransform))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) at)            Returns an iterator object that iterates along the boundary of this Polygon and provides access to the geometry of the outline of this Polygon. |
| [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) | [**getPathIterator**](http://docs.google.com/java/awt/Polygon.html#getPathIterator(java.awt.geom.AffineTransform,%20double))([AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) at, double flatness)            Returns an iterator object that iterates along the boundary of the Shape and provides access to the geometry of the outline of the Shape. |
| boolean | [**inside**](http://docs.google.com/java/awt/Polygon.html#inside(int,%20int))(int x, int y)  **Deprecated.** *As of JDK version 1.1, replaced by contains(int, int).* |
| boolean | [**intersects**](http://docs.google.com/java/awt/Polygon.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/Polygon.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. |
| void | [**invalidate**](http://docs.google.com/java/awt/Polygon.html#invalidate())()            Invalidates or flushes any internally-cached data that depends on the vertex coordinates of this Polygon. |
| void | [**reset**](http://docs.google.com/java/awt/Polygon.html#reset())()            Resets this Polygon object to an empty polygon. |
| void | [**translate**](http://docs.google.com/java/awt/Polygon.html#translate(int,%20int))(int deltaX, int deltaY)            Translates the vertices of the Polygon by deltaX along the x axis and by deltaY along the y axis. |

| **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** |
| --- |

### npoints

public int **npoints**

The total number of points. The value of npoints represents the number of valid points in this Polygon and might be less than the number of elements in [xpoints](http://docs.google.com/java/awt/Polygon.html#xpoints) or [ypoints](http://docs.google.com/java/awt/Polygon.html#ypoints). This value can be NULL.

**Since:** 1.0 **See Also:**[addPoint(int, int)](http://docs.google.com/java/awt/Polygon.html#addPoint(int,%20int))

### xpoints

public int[] **xpoints**

The array of X coordinates. The number of elements in this array might be more than the number of X coordinates in this Polygon. The extra elements allow new points to be added to this Polygon without re-creating this array. The value of [npoints](http://docs.google.com/java/awt/Polygon.html#npoints) is equal to the number of valid points in this Polygon.

**Since:** 1.0 **See Also:**[addPoint(int, int)](http://docs.google.com/java/awt/Polygon.html#addPoint(int,%20int))

### ypoints

public int[] **ypoints**

The array of Y coordinates. The number of elements in this array might be more than the number of Y coordinates in this Polygon. The extra elements allow new points to be added to this Polygon without re-creating this array. The value of npoints is equal to the number of valid points in this Polygon.

**Since:** 1.0 **See Also:**[addPoint(int, int)](http://docs.google.com/java/awt/Polygon.html#addPoint(int,%20int))

### bounds

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

The bounds of this Polygon. This value can be null.

**Since:** 1.0 **See Also:**[getBoundingBox()](http://docs.google.com/java/awt/Polygon.html#getBoundingBox()), [getBounds()](http://docs.google.com/java/awt/Polygon.html#getBounds())

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

### Polygon

public **Polygon**()

Creates an empty polygon.

**Since:** 1.0

### Polygon

public **Polygon**(int[] xpoints,  
 int[] ypoints,  
 int npoints)

Constructs and initializes a Polygon from the specified parameters.

**Parameters:**xpoints - an array of X coordinatesypoints - an array of Y coordinatesnpoints - the total number of points in the Polygon **Throws:** [NegativeArraySizeException](http://docs.google.com/java/lang/NegativeArraySizeException.html) - if the value of npoints is negative. [IndexOutOfBoundsException](http://docs.google.com/java/lang/IndexOutOfBoundsException.html) - if npoints is greater than the length of xpoints or the length of ypoints. [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if xpoints or ypoints is null.**Since:** 1.0

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

### reset

public void **reset**()

Resets this Polygon object to an empty polygon. The coordinate arrays and the data in them are left untouched but the number of points is reset to zero to mark the old vertex data as invalid and to start accumulating new vertex data at the beginning. All internally-cached data relating to the old vertices are discarded. Note that since the coordinate arrays from before the reset are reused, creating a new empty Polygon might be more memory efficient than resetting the current one if the number of vertices in the new polygon data is significantly smaller than the number of vertices in the data from before the reset.

**Since:** 1.4 **See Also:**[invalidate()](http://docs.google.com/java/awt/Polygon.html#invalidate())

### invalidate

public void **invalidate**()

Invalidates or flushes any internally-cached data that depends on the vertex coordinates of this Polygon. This method should be called after any direct manipulation of the coordinates in the xpoints or ypoints arrays to avoid inconsistent results from methods such as getBounds or contains that might cache data from earlier computations relating to the vertex coordinates.

**Since:** 1.4 **See Also:**[getBounds()](http://docs.google.com/java/awt/Polygon.html#getBounds())

### translate

public void **translate**(int deltaX,  
 int deltaY)

Translates the vertices of the Polygon by deltaX along the x axis and by deltaY along the y axis.

**Parameters:**deltaX - the amount to translate along the X axisdeltaY - the amount to translate along the Y axis**Since:** 1.1

### addPoint

public void **addPoint**(int x,  
 int y)

Appends the specified coordinates to this Polygon.

If an operation that calculates the bounding box of this Polygon has already been performed, such as getBounds or contains, then this method updates the bounding box.

**Parameters:**x - the specified X coordinatey - the specified Y coordinate**Since:** 1.0 **See Also:**[getBounds()](http://docs.google.com/java/awt/Polygon.html#getBounds()), [contains(java.awt.Point)](http://docs.google.com/java/awt/Polygon.html#contains(java.awt.Point))

### getBounds

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

Gets the bounding box of this Polygon. The bounding box is the smallest [Rectangle](http://docs.google.com/java/awt/Rectangle.html) whose sides are parallel to the x and y axes of the coordinate space, and can completely contain the Polygon.

**Specified by:**[getBounds](http://docs.google.com/java/awt/Shape.html#getBounds()) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Returns:**a Rectangle that defines the bounds of this Polygon.**Since:** 1.1 **See Also:**[Shape.getBounds2D()](http://docs.google.com/java/awt/Shape.html#getBounds2D())

### getBoundingBox

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public [Rectangle](http://docs.google.com/java/awt/Rectangle.html) **getBoundingBox**()

**Deprecated.** *As of JDK version 1.1, replaced by getBounds().*

Returns the bounds of this Polygon.

**Returns:**the bounds of this Polygon.**Since:** 1.0

### contains

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

Determines whether the specified [Point](http://docs.google.com/java/awt/Point.html) is inside this Polygon.

**Parameters:**p - the specified Point to be tested **Returns:**true if the Polygon contains the Point; false otherwise.**Since:** 1.0 **See Also:**[contains(double, double)](http://docs.google.com/java/awt/Polygon.html#contains(double,%20double))

### contains

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

Determines whether the specified coordinates are inside this Polygon.

**Parameters:**x - the specified X coordinate to be testedy - the specified Y coordinate to be tested **Returns:**true if this Polygon contains the specified coordinates (x,y); false otherwise.**Since:** 1.1 **See Also:**[contains(double, double)](http://docs.google.com/java/awt/Polygon.html#contains(double,%20double))

### inside

[@Deprecated](http://docs.google.com/java/lang/Deprecated.html)  
public boolean **inside**(int x,  
 int y)

**Deprecated.** *As of JDK version 1.1, replaced by contains(int, int).*

Determines whether the specified coordinates are contained in this Polygon.

**Parameters:**x - the specified X coordinate to be testedy - the specified Y coordinate to be tested **Returns:**true if this Polygon contains the specified coordinates (x,y); false otherwise.**Since:** 1.0 **See Also:**[contains(double, double)](http://docs.google.com/java/awt/Polygon.html#contains(double,%20double))

### getBounds2D

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

Returns a high precision and more accurate bounding box of the Shape than the getBounds method. Note that there is no guarantee that the returned [Rectangle2D](http://docs.google.com/java/awt/geom/Rectangle2D.html) is the smallest bounding box that encloses the Shape, only that the Shape lies entirely within the indicated Rectangle2D. The bounding box returned by this method is usually tighter than that returned by the getBounds method and never fails due to overflow problems since the return value can be an instance of the Rectangle2D that uses double precision values to store the dimensions.

**Specified by:**[getBounds2D](http://docs.google.com/java/awt/Shape.html#getBounds2D()) in interface [Shape](http://docs.google.com/java/awt/Shape.html) **Returns:**an instance of Rectangle2D that is a high-precision bounding box of the Shape.**Since:** 1.2 **See Also:**[Shape.getBounds()](http://docs.google.com/java/awt/Shape.html#getBounds())

### 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

### 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))

### 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))

### getPathIterator

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

Returns an iterator object that iterates along the boundary of this Polygon and provides access to the geometry of the outline of this Polygon. An optional [AffineTransform](http://docs.google.com/java/awt/geom/AffineTransform.html) can be specified so that the coordinates returned in the iteration are transformed accordingly.

**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:**a [PathIterator](http://docs.google.com/java/awt/geom/PathIterator.html) object that provides access to the geometry of this Polygon.**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)

Returns an iterator object that iterates along the boundary of the Shape and provides access to the geometry of the outline of the Shape. Only SEG\_MOVETO, SEG\_LINETO, and SEG\_CLOSE point types are returned by the iterator. Since polygons are already flat, the flatness parameter is ignored. An optional AffineTransform can be specified in which case the coordinates returned in the iteration are transformed accordingly.

**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 endpoints. Since polygons are already flat the flatness parameter is ignored. **Returns:**a PathIterator object that provides access to the Shape object's geometry.**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/Polygon.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/PointerInfo.html)   [**NEXT CLASS**](http://docs.google.com/java/awt/PopupMenu.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/Polygon.html)    [**NO FRAMES**](http://docs.google.com/Polygon.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | [FIELD](#3znysh7) | [CONSTR](#2et92p0) | [METHOD](#tyjcwt) | DETAIL: [FIELD](#1t3h5sf) | [CONSTR](#26in1rg) | [METHOD](#1ksv4uv) |

[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).