| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/GapContent.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/javax/swing/text/FlowView.FlowStrategy.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/GlyphView.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/GapContent.html)    [**NO FRAMES**](http://docs.google.com/GapContent.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#2s8eyo1) |

## **javax.swing.text**

Class GapContent

[java.lang.Object](http://docs.google.com/java/lang/Object.html)  
 **javax.swing.text.GapContent**

**All Implemented Interfaces:** [Serializable](http://docs.google.com/java/io/Serializable.html), [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html)

public class **GapContent**extends [Object](http://docs.google.com/java/lang/Object.html)implements [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html), [Serializable](http://docs.google.com/java/io/Serializable.html)

An implementation of the AbstractDocument.Content interface implemented using a gapped buffer similar to that used by emacs. The underlying storage is a array of unicode characters with a gap somewhere. The gap is moved to the location of changes to take advantage of common behavior where most changes are in the same location. Changes that occur at a gap boundary are generally cheap and moving the gap is generally cheaper than moving the array contents directly to accomodate the change.

The positions tracking change are also generally cheap to maintain. The Position implementations (marks) store the array index and can easily calculate the sequential position from the current gap location. Changes only require update to the the marks between the old and new gap boundaries when the gap is moved, so generally updating the marks is pretty cheap. The marks are stored sorted so they can be located quickly with a binary search. This increases the cost of adding a mark, and decreases the cost of keeping the mark updated.

| **Constructor Summary** | |
| --- | --- |
| [**GapContent**](http://docs.google.com/javax/swing/text/GapContent.html#GapContent())()            Creates a new GapContent object. |
| [**GapContent**](http://docs.google.com/javax/swing/text/GapContent.html#GapContent(int))(int initialLength)            Creates a new GapContent object, with the initial size specified. |

| **Method Summary** | |
| --- | --- |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**allocateArray**](http://docs.google.com/javax/swing/text/GapContent.html#allocateArray(int))(int len)            Allocate an array to store items of the type appropriate (which is determined by the subclass). |
| [Position](http://docs.google.com/javax/swing/text/Position.html) | [**createPosition**](http://docs.google.com/javax/swing/text/GapContent.html#createPosition(int))(int offset)            Creates a position within the content that will track change as the content is mutated. |
| protected  [Object](http://docs.google.com/java/lang/Object.html) | [**getArray**](http://docs.google.com/javax/swing/text/GapContent.html#getArray())()            Access to the array. |
| protected  int | [**getArrayLength**](http://docs.google.com/javax/swing/text/GapContent.html#getArrayLength())()            Get the length of the allocated array. |
| void | [**getChars**](http://docs.google.com/javax/swing/text/GapContent.html#getChars(int,%20int,%20javax.swing.text.Segment))(int where, int len, [Segment](http://docs.google.com/javax/swing/text/Segment.html) chars)            Retrieves a portion of the content. |
| protected  int | [**getGapEnd**](http://docs.google.com/javax/swing/text/GapContent.html#getGapEnd())()            Access to the end of the gap. |
| protected  int | [**getGapStart**](http://docs.google.com/javax/swing/text/GapContent.html#getGapStart())()            Access to the start of the gap. |
| protected  [Vector](http://docs.google.com/java/util/Vector.html) | [**getPositionsInRange**](http://docs.google.com/javax/swing/text/GapContent.html#getPositionsInRange(java.util.Vector,%20int,%20int))([Vector](http://docs.google.com/java/util/Vector.html) v, int offset, int length)            Returns a Vector containing instances of UndoPosRef for the Positions in the range offset to offset + length. |
| [String](http://docs.google.com/java/lang/String.html) | [**getString**](http://docs.google.com/javax/swing/text/GapContent.html#getString(int,%20int))(int where, int len)            Retrieves a portion of the content. |
| [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) | [**insertString**](http://docs.google.com/javax/swing/text/GapContent.html#insertString(int,%20java.lang.String))(int where, [String](http://docs.google.com/java/lang/String.html) str)            Inserts a string into the content. |
| int | [**length**](http://docs.google.com/javax/swing/text/GapContent.html#length())()            Returns the length of the content. |
| [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) | [**remove**](http://docs.google.com/javax/swing/text/GapContent.html#remove(int,%20int))(int where, int nitems)            Removes part of the content. |
| protected  void | [**replace**](http://docs.google.com/javax/swing/text/GapContent.html#replace(int,%20int,%20java.lang.Object,%20int))(int position, int rmSize, [Object](http://docs.google.com/java/lang/Object.html) addItems, int addSize)            Replace the given logical position in the storage with the given new items. |
| protected  void | [**resetMarksAtZero**](http://docs.google.com/javax/swing/text/GapContent.html#resetMarksAtZero())()            Resets all the marks that have an offset of 0 to have an index of zero as well. |
| protected  void | [**shiftEnd**](http://docs.google.com/javax/swing/text/GapContent.html#shiftEnd(int))(int newSize)            Make the gap bigger, moving any necessary data and updating the appropriate marks |
| protected  void | [**shiftGap**](http://docs.google.com/javax/swing/text/GapContent.html#shiftGap(int))(int newGapStart)            Move the start of the gap to a new location, without changing the size of the gap. |
| protected  void | [**shiftGapEndUp**](http://docs.google.com/javax/swing/text/GapContent.html#shiftGapEndUp(int))(int newGapEnd)            Adjust the gap end upward. |
| protected  void | [**shiftGapStartDown**](http://docs.google.com/javax/swing/text/GapContent.html#shiftGapStartDown(int))(int newGapStart)            Adjust the gap end downward. |
| protected  void | [**updateUndoPositions**](http://docs.google.com/javax/swing/text/GapContent.html#updateUndoPositions(java.util.Vector,%20int,%20int))([Vector](http://docs.google.com/java/util/Vector.html) positions, int offset, int length)            Resets the location for all the UndoPosRef instances in positions. |

| **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)) |

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

### GapContent

public **GapContent**()

Creates a new GapContent object. Initial size defaults to 10.

### GapContent

public **GapContent**(int initialLength)

Creates a new GapContent object, with the initial size specified. The initial size will not be allowed to go below 2, to give room for the implied break and the gap.

**Parameters:**initialLength - the initial size

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

### allocateArray

protected [Object](http://docs.google.com/java/lang/Object.html) **allocateArray**(int len)

Allocate an array to store items of the type appropriate (which is determined by the subclass).

### getArrayLength

protected int **getArrayLength**()

Get the length of the allocated array.

### length

public int **length**()

Returns the length of the content.

**Specified by:**[length](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#length()) in interface [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html) **Returns:**the length >= 1**See Also:**[AbstractDocument.Content.length()](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#length())

### insertString

public [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) **insertString**(int where,  
 [String](http://docs.google.com/java/lang/String.html) str)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Inserts a string into the content.

**Specified by:**[insertString](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#insertString(int,%20java.lang.String)) in interface [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html) **Parameters:**where - the starting position >= 0, < length()str - the non-null string to insert **Returns:**an UndoableEdit object for undoing **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) - if the specified position is invalid**See Also:**[AbstractDocument.Content.insertString(int, java.lang.String)](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#insertString(int,%20java.lang.String))

### remove

public [UndoableEdit](http://docs.google.com/javax/swing/undo/UndoableEdit.html) **remove**(int where,  
 int nitems)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Removes part of the content.

**Specified by:**[remove](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#remove(int,%20int)) in interface [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html) **Parameters:**where - the starting position >= 0, where + nitems < length()nitems - the number of characters to remove >= 0 **Returns:**an UndoableEdit object for undoing **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) - if the specified position is invalid**See Also:**[AbstractDocument.Content.remove(int, int)](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#remove(int,%20int))

### getString

public [String](http://docs.google.com/java/lang/String.html) **getString**(int where,  
 int len)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Retrieves a portion of the content.

**Specified by:**[getString](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#getString(int,%20int)) in interface [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html) **Parameters:**where - the starting position >= 0len - the length to retrieve >= 0 **Returns:**a string representing the content **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) - if the specified position is invalid**See Also:**[AbstractDocument.Content.getString(int, int)](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#getString(int,%20int))

### getChars

public void **getChars**(int where,  
 int len,  
 [Segment](http://docs.google.com/javax/swing/text/Segment.html) chars)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Retrieves a portion of the content. If the desired content spans the gap, we copy the content. If the desired content does not span the gap, the actual store is returned to avoid the copy since it is contiguous.

**Specified by:**[getChars](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#getChars(int,%20int,%20javax.swing.text.Segment)) in interface [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html) **Parameters:**where - the starting position >= 0, where + len <= length()len - the number of characters to retrieve >= 0chars - the Segment object to return the characters in **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) - if the specified position is invalid**See Also:**[AbstractDocument.Content.getChars(int, int, javax.swing.text.Segment)](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#getChars(int,%20int,%20javax.swing.text.Segment))

### createPosition

public [Position](http://docs.google.com/javax/swing/text/Position.html) **createPosition**(int offset)  
 throws [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html)

Creates a position within the content that will track change as the content is mutated.

**Specified by:**[createPosition](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html#createPosition(int)) in interface [AbstractDocument.Content](http://docs.google.com/javax/swing/text/AbstractDocument.Content.html) **Parameters:**offset - the offset to track >= 0 **Returns:**the position **Throws:** [BadLocationException](http://docs.google.com/javax/swing/text/BadLocationException.html) - if the specified position is invalid

### shiftEnd

protected void **shiftEnd**(int newSize)

Make the gap bigger, moving any necessary data and updating the appropriate marks

### shiftGap

protected void **shiftGap**(int newGapStart)

Move the start of the gap to a new location, without changing the size of the gap. This moves the data in the array and updates the marks accordingly.

### resetMarksAtZero

protected void **resetMarksAtZero**()

Resets all the marks that have an offset of 0 to have an index of zero as well.

### shiftGapStartDown

protected void **shiftGapStartDown**(int newGapStart)

Adjust the gap end downward. This doesn't move any data, but it does update any marks affected by the boundary change. All marks from the old gap start down to the new gap start are squeezed to the end of the gap (their location has been removed).

### shiftGapEndUp

protected void **shiftGapEndUp**(int newGapEnd)

Adjust the gap end upward. This doesn't move any data, but it does update any marks affected by the boundary change. All marks from the old gap end up to the new gap end are squeezed to the end of the gap (their location has been removed).

### getPositionsInRange

protected [Vector](http://docs.google.com/java/util/Vector.html) **getPositionsInRange**([Vector](http://docs.google.com/java/util/Vector.html) v,  
 int offset,  
 int length)

Returns a Vector containing instances of UndoPosRef for the Positions in the range offset to offset + length. If v is not null the matching Positions are placed in there. The vector with the resulting Positions are returned.

**Parameters:**v - the Vector to use, with a new one created on nulloffset - the starting offset >= 0length - the length >= 0 **Returns:**the set of instances

### updateUndoPositions

protected void **updateUndoPositions**([Vector](http://docs.google.com/java/util/Vector.html) positions,  
 int offset,  
 int length)

Resets the location for all the UndoPosRef instances in positions.

This is meant for internal usage, and is generally not of interest to subclasses.

**Parameters:**positions - the UndoPosRef instances to reset

### getArray

protected final [Object](http://docs.google.com/java/lang/Object.html) **getArray**()

Access to the array. The actual type of the array is known only by the subclass.

### getGapStart

protected final int **getGapStart**()

Access to the start of the gap.

### getGapEnd

protected final int **getGapEnd**()

Access to the end of the gap.

### replace

protected void **replace**(int position,  
 int rmSize,  
 [Object](http://docs.google.com/java/lang/Object.html) addItems,  
 int addSize)

Replace the given logical position in the storage with the given new items. This will move the gap to the area being changed if the gap is not currently located at the change location.

**Parameters:**position - the location to make the replacement. This is not the location in the underlying storage array, but the location in the contiguous space being modeled.rmSize - the number of items to removeaddItems - the new items to place in storage.

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/GapContent.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/javax/swing/text/FlowView.FlowStrategy.html)   [**NEXT CLASS**](http://docs.google.com/javax/swing/text/GlyphView.html) | [**FRAMES**](http://docs.google.com/index.html?javax/swing/text/GapContent.html)    [**NO FRAMES**](http://docs.google.com/GapContent.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).