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

## **java.awt.im**

Interface InputMethodRequests

**All Known Subinterfaces:** [InputMethodContext](http://docs.google.com/java/awt/im/spi/InputMethodContext.html)

public interface **InputMethodRequests**

InputMethodRequests defines the requests that a text editing component has to handle in order to work with input methods. The component can implement this interface itself or use a separate object that implements it. The object implementing this interface must be returned from the component's getInputMethodRequests method.

The text editing component also has to provide an input method event listener.

The interface is designed to support one of two input user interfaces:

* *on-the-spot* input, where the composed text is displayed as part of the text component's text body.
* *below-the-spot* input, where the composed text is displayed in a separate composition window just below the insertion point where the text will be inserted when it is committed. Note that, if text is selected within the component's text body, this text will be replaced by the committed text upon commitment; therefore it is not considered part of the context that the text is input into.

**Since:** 1.2 **See Also:**[Component.getInputMethodRequests()](http://docs.google.com/java/awt/Component.html#getInputMethodRequests()), [InputMethodListener](http://docs.google.com/java/awt/event/InputMethodListener.html)

| **Method Summary** | |
| --- | --- |
| [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) | [**cancelLatestCommittedText**](http://docs.google.com/java/awt/im/InputMethodRequests.html#cancelLatestCommittedText(java.text.AttributedCharacterIterator.Attribute%5B%5D))([AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)[] attributes)            Gets the latest committed text from the text editing component and removes it from the component's text body. |
| [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) | [**getCommittedText**](http://docs.google.com/java/awt/im/InputMethodRequests.html#getCommittedText(int,%20int,%20java.text.AttributedCharacterIterator.Attribute%5B%5D))(int beginIndex, int endIndex, [AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)[] attributes)            Gets an iterator providing access to the entire text and attributes contained in the text editing component except for uncommitted text. |
| int | [**getCommittedTextLength**](http://docs.google.com/java/awt/im/InputMethodRequests.html#getCommittedTextLength())()            Gets the length of the entire text contained in the text editing component except for uncommitted (composed) text. |
| int | [**getInsertPositionOffset**](http://docs.google.com/java/awt/im/InputMethodRequests.html#getInsertPositionOffset())()            Gets the offset of the insert position in the committed text contained in the text editing component. |
| [TextHitInfo](http://docs.google.com/java/awt/font/TextHitInfo.html) | [**getLocationOffset**](http://docs.google.com/java/awt/im/InputMethodRequests.html#getLocationOffset(int,%20int))(int x, int y)            Gets the offset within the composed text for the specified absolute x and y coordinates on the screen. |
| [AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) | [**getSelectedText**](http://docs.google.com/java/awt/im/InputMethodRequests.html#getSelectedText(java.text.AttributedCharacterIterator.Attribute%5B%5D))([AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)[] attributes)            Gets the currently selected text from the text editing component. |
| [Rectangle](http://docs.google.com/java/awt/Rectangle.html) | [**getTextLocation**](http://docs.google.com/java/awt/im/InputMethodRequests.html#getTextLocation(java.awt.font.TextHitInfo))([TextHitInfo](http://docs.google.com/java/awt/font/TextHitInfo.html) offset)            Gets the location of a specified offset in the current composed text, or of the selection in committed text. |

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

### getTextLocation

[Rectangle](http://docs.google.com/java/awt/Rectangle.html) **getTextLocation**([TextHitInfo](http://docs.google.com/java/awt/font/TextHitInfo.html) offset)

Gets the location of a specified offset in the current composed text, or of the selection in committed text. This information is, for example, used to position the candidate window near the composed text, or a composition window near the location where committed text will be inserted.

If the component has composed text (because the most recent InputMethodEvent sent to it contained composed text), then the offset is relative to the composed text - offset 0 indicates the first character in the composed text. The location returned should be for this character.

If the component doesn't have composed text, the offset should be ignored, and the location returned should reflect the beginning (in line direction) of the highlight in the last line containing selected text. For example, for horizontal left-to-right text (such as English), the location to the left of the left-most character on the last line containing selected text is returned. For vertical top-to-bottom text, with lines proceding from right to left, the location to the top of the left-most line containing selected text is returned.

The location is represented as a 0-thickness caret, that is, it has 0 width if the text is drawn horizontally, and 0 height if the text is drawn vertically. Other text orientations need to be mapped to horizontal or vertical orientation. The rectangle uses absolute screen coordinates.

**Parameters:**offset - the offset within the composed text, if there is composed text; null otherwise **Returns:**a rectangle representing the screen location of the offset

### getLocationOffset

[TextHitInfo](http://docs.google.com/java/awt/font/TextHitInfo.html) **getLocationOffset**(int x,  
 int y)

Gets the offset within the composed text for the specified absolute x and y coordinates on the screen. This information is used, for example to handle mouse clicks and the mouse cursor. The offset is relative to the composed text, so offset 0 indicates the beginning of the composed text.

Return null if the location is outside the area occupied by the composed text.

**Parameters:**x - the absolute x coordinate on screeny - the absolute y coordinate on screen **Returns:**a text hit info describing the offset in the composed text.

### getInsertPositionOffset

int **getInsertPositionOffset**()

Gets the offset of the insert position in the committed text contained in the text editing component. This is the offset at which characters entered through an input method are inserted. This information is used by an input method, for example, to examine the text surrounding the insert position.

**Returns:**the offset of the insert position

### getCommittedText

[AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) **getCommittedText**(int beginIndex,  
 int endIndex,  
 [AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)[] attributes)

Gets an iterator providing access to the entire text and attributes contained in the text editing component except for uncommitted text. Uncommitted (composed) text should be ignored for index calculations and should not be made accessible through the iterator.

The input method may provide a list of attributes that it is interested in. In that case, information about other attributes that the implementor may have need not be made accessible through the iterator. If the list is null, all available attribute information should be made accessible.

**Parameters:**beginIndex - the index of the first characterendIndex - the index of the character following the last characterattributes - a list of attributes that the input method is interested in **Returns:**an iterator providing access to the text and its attributes

### getCommittedTextLength

int **getCommittedTextLength**()

Gets the length of the entire text contained in the text editing component except for uncommitted (composed) text.

**Returns:**the length of the text except for uncommitted text

### cancelLatestCommittedText

[AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) **cancelLatestCommittedText**([AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)[] attributes)

Gets the latest committed text from the text editing component and removes it from the component's text body. This is used for the "Undo Commit" feature in some input methods, where the committed text reverts to its previous composed state. The composed text will be sent to the component using an InputMethodEvent.

Generally, this feature should only be supported immediately after the text was committed, not after the user performed other operations on the text. When the feature is not supported, return null.

The input method may provide a list of attributes that it is interested in. In that case, information about other attributes that the implementor may have need not be made accessible through the iterator. If the list is null, all available attribute information should be made accessible.

**Parameters:**attributes - a list of attributes that the input method is interested in **Returns:**the latest committed text, or null when the "Undo Commit" feature is not supported

### getSelectedText

[AttributedCharacterIterator](http://docs.google.com/java/text/AttributedCharacterIterator.html) **getSelectedText**([AttributedCharacterIterator.Attribute](http://docs.google.com/java/text/AttributedCharacterIterator.Attribute.html)[] attributes)

Gets the currently selected text from the text editing component. This may be used for a variety of purposes. One of them is the "Reconvert" feature in some input methods. In this case, the input method will typically send an input method event to replace the selected text with composed text. Depending on the input method's capabilities, this may be the original composed text for the selected text, the latest composed text entered anywhere in the text, or a version of the text that's converted back from the selected text.

The input method may provide a list of attributes that it is interested in. In that case, information about other attributes that the implementor may have need not be made accessible through the iterator. If the list is null, all available attribute information should be made accessible.

**Parameters:**attributes - a list of attributes that the input method is interested in **Returns:**the currently selected text

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

[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).