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

## **java.awt.im.spi**

Interface InputMethod

public interface **InputMethod**

Defines the interface for an input method that supports complex text input. Input methods traditionally support text input for languages that have more characters than can be represented on a standard-size keyboard, such as Chinese, Japanese, and Korean. However, they may also be used to support phonetic text input for English or character reordering for Thai.

Subclasses of InputMethod can be loaded by the input method framework; they can then be selected either through the API ([InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale))) or the user interface (the input method selection menu).

**Since:** 1.3

| **Method Summary** | |
| --- | --- |
| void | [**activate**](http://docs.google.com/java/awt/im/spi/InputMethod.html#activate())()            Activates the input method for immediate input processing. |
| void | [**deactivate**](http://docs.google.com/java/awt/im/spi/InputMethod.html#deactivate(boolean))(boolean isTemporary)            Deactivates the input method. |
| void | [**dispatchEvent**](http://docs.google.com/java/awt/im/spi/InputMethod.html#dispatchEvent(java.awt.AWTEvent))([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) event)            Dispatches the event to the input method. |
| void | [**dispose**](http://docs.google.com/java/awt/im/spi/InputMethod.html#dispose())()            Releases the resources used by this input method. |
| void | [**endComposition**](http://docs.google.com/java/awt/im/spi/InputMethod.html#endComposition())()            Ends any input composition that may currently be going on in this context. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getControlObject**](http://docs.google.com/java/awt/im/spi/InputMethod.html#getControlObject())()            Returns a control object from this input method, or null. |
| [Locale](http://docs.google.com/java/util/Locale.html) | [**getLocale**](http://docs.google.com/java/awt/im/spi/InputMethod.html#getLocale())()            Returns the current input locale. |
| void | [**hideWindows**](http://docs.google.com/java/awt/im/spi/InputMethod.html#hideWindows())()            Closes or hides all windows opened by this input method instance or its class. |
| boolean | [**isCompositionEnabled**](http://docs.google.com/java/awt/im/spi/InputMethod.html#isCompositionEnabled())()            Determines whether this input method is enabled. |
| void | [**notifyClientWindowChange**](http://docs.google.com/java/awt/im/spi/InputMethod.html#notifyClientWindowChange(java.awt.Rectangle))([Rectangle](http://docs.google.com/java/awt/Rectangle.html) bounds)            Notifies this input method of changes in the client window location or state. |
| void | [**reconvert**](http://docs.google.com/java/awt/im/spi/InputMethod.html#reconvert())()            Starts the reconversion operation. |
| void | [**removeNotify**](http://docs.google.com/java/awt/im/spi/InputMethod.html#removeNotify())()            Notifies the input method that a client component has been removed from its containment hierarchy, or that input method support has been disabled for the component. |
| void | [**setCharacterSubsets**](http://docs.google.com/java/awt/im/spi/InputMethod.html#setCharacterSubsets(java.lang.Character.Subset%5B%5D))([Character.Subset](http://docs.google.com/java/lang/Character.Subset.html)[] subsets)            Sets the subsets of the Unicode character set that this input method is allowed to input. |
| void | [**setCompositionEnabled**](http://docs.google.com/java/awt/im/spi/InputMethod.html#setCompositionEnabled(boolean))(boolean enable)            Enables or disables this input method for composition, depending on the value of the parameter enable. |
| void | [**setInputMethodContext**](http://docs.google.com/java/awt/im/spi/InputMethod.html#setInputMethodContext(java.awt.im.spi.InputMethodContext))([InputMethodContext](http://docs.google.com/java/awt/im/spi/InputMethodContext.html) context)            Sets the input method context, which is used to dispatch input method events to the client component and to request information from the client component. |
| boolean | [**setLocale**](http://docs.google.com/java/awt/im/spi/InputMethod.html#setLocale(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) locale)            Attempts to set the input locale. |

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

### setInputMethodContext

void **setInputMethodContext**([InputMethodContext](http://docs.google.com/java/awt/im/spi/InputMethodContext.html) context)

Sets the input method context, which is used to dispatch input method events to the client component and to request information from the client component.

This method is called once immediately after instantiating this input method.

**Parameters:**context - the input method context for this input method **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if context is null

### setLocale

boolean **setLocale**([Locale](http://docs.google.com/java/util/Locale.html) locale)

Attempts to set the input locale. If the input method supports the desired locale, it changes its behavior to support input for the locale and returns true. Otherwise, it returns false and does not change its behavior.

This method is called

* by [InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale)),
* when switching to this input method through the user interface if the user specified a locale or if the previously selected input method's [getLocale](http://docs.google.com/java/awt/im/spi/InputMethod.html#getLocale()) method returns a non-null value.

**Parameters:**locale - locale to input **Returns:**whether the specified locale is supported **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if locale is null

### getLocale

[Locale](http://docs.google.com/java/util/Locale.html) **getLocale**()

Returns the current input locale. Might return null in exceptional cases.

This method is called

* by [InputContext.getLocale](http://docs.google.com/java/awt/im/InputContext.html#getLocale()) and
* when switching from this input method to a different one through the user interface.

**Returns:**the current input locale, or null

### setCharacterSubsets

void **setCharacterSubsets**([Character.Subset](http://docs.google.com/java/lang/Character.Subset.html)[] subsets)

Sets the subsets of the Unicode character set that this input method is allowed to input. Null may be passed in to indicate that all characters are allowed.

This method is called

* immediately after instantiating this input method,
* when switching to this input method from a different one, and
* by [InputContext.setCharacterSubsets](http://docs.google.com/java/awt/im/InputContext.html#setCharacterSubsets(java.lang.Character.Subset%5B%5D)).

**Parameters:**subsets - the subsets of the Unicode character set from which characters may be input

### setCompositionEnabled

void **setCompositionEnabled**(boolean enable)

Enables or disables this input method for composition, depending on the value of the parameter enable.

An input method that is enabled for composition interprets incoming events for both composition and control purposes, while a disabled input method does not interpret events for composition. Note however that events are passed on to the input method regardless whether it is enabled or not, and that an input method that is disabled for composition may still interpret events for control purposes, including to enable or disable itself for composition.

For input methods provided by host operating systems, it is not always possible to determine whether this operation is supported. For example, an input method may enable composition only for some locales, and do nothing for other locales. For such input methods, it is possible that this method does not throw [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html), but also does not affect whether composition is enabled.

This method is called

* by [InputContext.setCompositionEnabled](http://docs.google.com/java/awt/im/InputContext.html#setCompositionEnabled(boolean)),
* when switching to this input method from a different one using the user interface or [InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale)), if the previously selected input method's [isCompositionEnabled](http://docs.google.com/java/awt/im/spi/InputMethod.html#isCompositionEnabled()) method returns without throwing an exception.

**Parameters:**enable - whether to enable the input method for composition **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this input method does not support the enabling/disabling operation**See Also:**[isCompositionEnabled()](http://docs.google.com/java/awt/im/spi/InputMethod.html#isCompositionEnabled())

### isCompositionEnabled

boolean **isCompositionEnabled**()

Determines whether this input method is enabled. An input method that is enabled for composition interprets incoming events for both composition and control purposes, while a disabled input method does not interpret events for composition.

This method is called

* by [InputContext.isCompositionEnabled](http://docs.google.com/java/awt/im/InputContext.html#isCompositionEnabled()) and
* when switching from this input method to a different one using the user interface or [InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale)).

**Returns:**true if this input method is enabled for composition; false otherwise. **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if this input method does not support checking whether it is enabled for composition**See Also:**[setCompositionEnabled(boolean)](http://docs.google.com/java/awt/im/spi/InputMethod.html#setCompositionEnabled(boolean))

### reconvert

void **reconvert**()

Starts the reconversion operation. The input method obtains the text to be reconverted from the current client component using the [InputMethodRequests.getSelectedText](http://docs.google.com/java/awt/im/InputMethodRequests.html#getSelectedText(java.text.AttributedCharacterIterator.Attribute%5B%5D)) method. It can use other InputMethodRequests methods to request additional information required for the reconversion operation. The composed and committed text produced by the operation is sent to the client component as a sequence of InputMethodEvents. If the given text cannot be reconverted, the same text should be sent to the client component as committed text.

This method is called by [InputContext.reconvert](http://docs.google.com/java/awt/im/InputContext.html#reconvert()).

**Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if the input method does not support the reconversion operation.

### dispatchEvent

void **dispatchEvent**([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) event)

Dispatches the event to the input method. If input method support is enabled for the focussed component, incoming events of certain types are dispatched to the current input method for this component before they are dispatched to the component's methods or event listeners. The input method decides whether it needs to handle the event. If it does, it also calls the event's consume method; this causes the event to not get dispatched to the component's event processing methods or event listeners.

Events are dispatched if they are instances of InputEvent or its subclasses. This includes instances of the AWT classes KeyEvent and MouseEvent.

This method is called by [InputContext.dispatchEvent](http://docs.google.com/java/awt/im/InputContext.html#dispatchEvent(java.awt.AWTEvent)).

**Parameters:**event - the event being dispatched to the input method **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if event is null

### notifyClientWindowChange

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

Notifies this input method of changes in the client window location or state. This method is called while this input method is the current input method of its input context and notifications for it are enabled (see [InputMethodContext.enableClientWindowNotification](http://docs.google.com/java/awt/im/spi/InputMethodContext.html#enableClientWindowNotification(java.awt.im.spi.InputMethod,%20boolean))). Calls to this method are temporarily suspended if the input context's [removeNotify](http://docs.google.com/java/awt/im/InputContext.html#removeNotify(java.awt.Component)) method is called, and resume when the input method is activated for a new client component. It is called in the following situations:

* when the window containing the current client component changes in location, size, visibility, iconification state, or when the window is closed.
* from enableClientWindowNotification(inputMethod, true) if the current client component exists,
* when activating the input method for the first time after it called enableClientWindowNotification(inputMethod, true) if during the call no current client component was available,
* when activating the input method for a new client component after the input context's removeNotify method has been called.

**Parameters:**bounds - client window's [bounds](http://docs.google.com/java/awt/Component.html#getBounds()) on the screen; or null if the client window is iconified or invisible

### activate

void **activate**()

Activates the input method for immediate input processing.

If an input method provides its own windows, it should make sure at this point that all necessary windows are open and visible.

This method is called

* by [InputContext.dispatchEvent](http://docs.google.com/java/awt/im/InputContext.html#dispatchEvent(java.awt.AWTEvent)) when a client component receives a FOCUS\_GAINED event,
* when switching to this input method from a different one using the user interface or [InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale)).

The method is only called when the input method is inactive. A newly instantiated input method is assumed to be inactive.

### deactivate

void **deactivate**(boolean isTemporary)

Deactivates the input method. The isTemporary argument has the same meaning as in [FocusEvent.isTemporary](http://docs.google.com/java/awt/event/FocusEvent.html#isTemporary()).

If an input method provides its own windows, only windows that relate to the current composition (such as a lookup choice window) should be closed at this point. It is possible that the input method will be immediately activated again for a different client component, and closing and reopening more persistent windows (such as a control panel) would create unnecessary screen flicker. Before an instance of a different input method class is activated, [hideWindows()](http://docs.google.com/java/awt/im/spi/InputMethod.html#hideWindows()) is called on the current input method.

This method is called

* by [InputContext.dispatchEvent](http://docs.google.com/java/awt/im/InputContext.html#dispatchEvent(java.awt.AWTEvent)) when a client component receives a FOCUS\_LOST event,
* when switching from this input method to a different one using the user interface or [InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale)),
* before [removeNotify](http://docs.google.com/java/awt/im/spi/InputMethod.html#removeNotify()) if the current client component is removed.

The method is only called when the input method is active.

**Parameters:**isTemporary - whether the focus change is temporary

### hideWindows

void **hideWindows**()

Closes or hides all windows opened by this input method instance or its class.

This method is called

* before calling [activate](http://docs.google.com/java/awt/im/spi/InputMethod.html#activate()) on an instance of a different input method class,
* before calling [dispose](http://docs.google.com/java/awt/im/spi/InputMethod.html#dispose()) on this input method.

The method is only called when the input method is inactive.

### removeNotify

void **removeNotify**()

Notifies the input method that a client component has been removed from its containment hierarchy, or that input method support has been disabled for the component.

This method is called by [InputContext.removeNotify](http://docs.google.com/java/awt/im/InputContext.html#removeNotify(java.awt.Component)).

The method is only called when the input method is inactive.

### endComposition

void **endComposition**()

Ends any input composition that may currently be going on in this context. Depending on the platform and possibly user preferences, this may commit or delete uncommitted text. Any changes to the text are communicated to the active component using an input method event.

A text editing component may call this in a variety of situations, for example, when the user moves the insertion point within the text (but outside the composed text), or when the component's text is saved to a file or copied to the clipboard.

This method is called

* by [InputContext.endComposition](http://docs.google.com/java/awt/im/InputContext.html#endComposition()),
* by [InputContext.dispatchEvent](http://docs.google.com/java/awt/im/InputContext.html#dispatchEvent(java.awt.AWTEvent)) when switching to a different client component
* when switching from this input method to a different one using the user interface or [InputContext.selectInputMethod](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale)).

### dispose

void **dispose**()

Releases the resources used by this input method. In particular, the input method should dispose windows and close files that are no longer needed.

This method is called by [InputContext.dispose](http://docs.google.com/java/awt/im/InputContext.html#dispose()).

The method is only called when the input method is inactive. No method of this interface is called on this instance after dispose.

### getControlObject

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

Returns a control object from this input method, or null. A control object provides methods that control the behavior of the input method or obtain information from the input method. The type of the object is an input method specific class. Clients have to compare the result against known input method control object classes and cast to the appropriate class to invoke the methods provided.

This method is called by [InputContext.getInputMethodControlObject](http://docs.google.com/java/awt/im/InputContext.html#getInputMethodControlObject()).

**Returns:**a control object from this input method, or null

| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/InputMethod.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   [**NEXT CLASS**](http://docs.google.com/java/awt/im/spi/InputMethodContext.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/im/spi/InputMethod.html)    [**NO FRAMES**](http://docs.google.com/InputMethod.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).