| | [**Overview**](http://docs.google.com/overview-summary.html) | [**Package**](http://docs.google.com/package-summary.html) | **Class** | [**Use**](http://docs.google.com/class-use/InputContext.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/InputMethodHighlight.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/im/InputContext.html)    [**NO FRAMES**](http://docs.google.com/InputContext.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

## **java.awt.im**

Class InputContext

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

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

Provides methods to control text input facilities such as input methods and keyboard layouts. Two methods handle both input methods and keyboard layouts: selectInputMethod lets a client component select an input method or keyboard layout by locale, getLocale lets a client component obtain the locale of the current input method or keyboard layout. The other methods more specifically support interaction with input methods: They let client components control the behavior of input methods, and dispatch events from the client component to the input method.

By default, one InputContext instance is created per Window instance, and this input context is shared by all components within the window's container hierarchy. However, this means that only one text input operation is possible at any one time within a window, and that the text needs to be committed when moving the focus from one text component to another. If this is not desired, text components can create their own input context instances.

The Java Platform supports input methods that have been developed in the Java programming language, using the interfaces in the [java.awt.im.spi](http://docs.google.com/java/awt/im/spi/package-summary.html) package, and installed into a Java SE Runtime Environment as extensions. Implementations may also support using the native input methods of the platforms they run on; however, not all platforms and locales provide input methods. Keyboard layouts are provided by the host platform.

Input methods are *unavailable* if (a) no input method written in the Java programming language has been installed and (b) the Java Platform implementation or the underlying platform does not support native input methods. In this case, input contexts can still be created and used; their behavior is specified with the individual methods below.

**Since:** 1.2 **See Also:**[Component.getInputContext()](http://docs.google.com/java/awt/Component.html#getInputContext()), [Component.enableInputMethods(boolean)](http://docs.google.com/java/awt/Component.html#enableInputMethods(boolean))

| **Constructor Summary** | |
| --- | --- |
| protected | [**InputContext**](http://docs.google.com/java/awt/im/InputContext.html#InputContext())()            Constructs an InputContext. |

| **Method Summary** | |
| --- | --- |
| void | [**dispatchEvent**](http://docs.google.com/java/awt/im/InputContext.html#dispatchEvent(java.awt.AWTEvent))([AWTEvent](http://docs.google.com/java/awt/AWTEvent.html) event)            Dispatches an event to the active input method. |
| void | [**dispose**](http://docs.google.com/java/awt/im/InputContext.html#dispose())()            Releases the resources used by this input context. |
| void | [**endComposition**](http://docs.google.com/java/awt/im/InputContext.html#endComposition())()            Ends any input composition that may currently be going on in this context. |
| [Object](http://docs.google.com/java/lang/Object.html) | [**getInputMethodControlObject**](http://docs.google.com/java/awt/im/InputContext.html#getInputMethodControlObject())()            Returns a control object from the current input method, or null. |
| static [InputContext](http://docs.google.com/java/awt/im/InputContext.html) | [**getInstance**](http://docs.google.com/java/awt/im/InputContext.html#getInstance())()            Returns a new InputContext instance. |
| [Locale](http://docs.google.com/java/util/Locale.html) | [**getLocale**](http://docs.google.com/java/awt/im/InputContext.html#getLocale())()            Returns the current locale of the current input method or keyboard layout. |
| boolean | [**isCompositionEnabled**](http://docs.google.com/java/awt/im/InputContext.html#isCompositionEnabled())()            Determines whether the current input method is enabled for composition. |
| void | [**reconvert**](http://docs.google.com/java/awt/im/InputContext.html#reconvert())()            Asks the current input method to reconvert text from the current client component. |
| void | [**removeNotify**](http://docs.google.com/java/awt/im/InputContext.html#removeNotify(java.awt.Component))([Component](http://docs.google.com/java/awt/Component.html) client)            Notifies the input context that a client component has been removed from its containment hierarchy, or that input method support has been disabled for the component. |
| boolean | [**selectInputMethod**](http://docs.google.com/java/awt/im/InputContext.html#selectInputMethod(java.util.Locale))([Locale](http://docs.google.com/java/util/Locale.html) locale)            Attempts to select an input method or keyboard layout that supports the given locale, and returns a value indicating whether such an input method or keyboard layout has been successfully selected. |
| void | [**setCharacterSubsets**](http://docs.google.com/java/awt/im/InputContext.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 input methods of this input context should be allowed to input. |
| void | [**setCompositionEnabled**](http://docs.google.com/java/awt/im/InputContext.html#setCompositionEnabled(boolean))(boolean enable)            Enables or disables the current input method for composition, depending on the value of the parameter enable. |

| **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** |
| --- |

### InputContext

protected **InputContext**()

Constructs an InputContext. This method is protected so clients cannot instantiate InputContext directly. Input contexts are obtained by calling [getInstance()](http://docs.google.com/java/awt/im/InputContext.html#getInstance()).

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

### getInstance

public static [InputContext](http://docs.google.com/java/awt/im/InputContext.html) **getInstance**()

Returns a new InputContext instance.

### selectInputMethod

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

Attempts to select an input method or keyboard layout that supports the given locale, and returns a value indicating whether such an input method or keyboard layout has been successfully selected. The following steps are taken until an input method has been selected:

* If the currently selected input method or keyboard layout supports the requested locale, it remains selected.
* If there is no input method or keyboard layout available that supports the requested locale, the current input method or keyboard layout remains selected.
* If the user has previously selected an input method or keyboard layout for the requested locale from the user interface, then the most recently selected such input method or keyboard layout is reselected.
* Otherwise, an input method or keyboard layout that supports the requested locale is selected in an implementation dependent way.

Before switching away from an input method, any currently uncommitted text is committed. If no input method or keyboard layout supporting the requested locale is available, then false is returned.

Not all host operating systems provide API to determine the locale of the currently selected native input method or keyboard layout, and to select a native input method or keyboard layout by locale. For host operating systems that don't provide such API, selectInputMethod assumes that native input methods or keyboard layouts provided by the host operating system support only the system's default locale.

A text editing component may call this method, for example, when the user changes the insertion point, so that the user can immediately continue typing in the language of the surrounding text.

**Parameters:**locale - The desired new locale. **Returns:**true if the input method or keyboard layout that's active after this call supports the desired locale. **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if locale is null

### getLocale

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

Returns the current locale of the current input method or keyboard layout. Returns null if the input context does not have a current input method or keyboard layout or if the current input method's [InputMethod.getLocale()](http://docs.google.com/java/awt/im/spi/InputMethod.html#getLocale()) method returns null.

Not all host operating systems provide API to determine the locale of the currently selected native input method or keyboard layout. For host operating systems that don't provide such API, getLocale assumes that the current locale of all native input methods or keyboard layouts provided by the host operating system is the system's default locale.

**Returns:**the current locale of the current input method or keyboard layout**Since:** 1.3

### setCharacterSubsets

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

Sets the subsets of the Unicode character set that input methods of this input context should be allowed to input. Null may be passed in to indicate that all characters are allowed. The initial value is null. The setting applies to the current input method as well as input methods selected after this call is made. However, applications cannot rely on this call having the desired effect, since this setting cannot be passed on to all host input methods - applications still need to apply their own character validation. If no input methods are available, then this method has no effect.

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

### setCompositionEnabled

public void **setCompositionEnabled**(boolean enable)

Enables or disables the current 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.

**Parameters:**enable - whether to enable the current input method for composition **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if there is no current input method available or the current input method does not support the enabling/disabling operation**Since:** 1.3 **See Also:**[isCompositionEnabled()](http://docs.google.com/java/awt/im/InputContext.html#isCompositionEnabled())

### isCompositionEnabled

public boolean **isCompositionEnabled**()

Determines whether the current input method is enabled for composition. 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.

**Returns:**true if the current input method is enabled for composition; false otherwise **Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if there is no current input method available or the current input method does not support checking whether it is enabled for composition**Since:** 1.3 **See Also:**[setCompositionEnabled(boolean)](http://docs.google.com/java/awt/im/InputContext.html#setCompositionEnabled(boolean))

### reconvert

public void **reconvert**()

Asks the current input method to reconvert text from the current client component. The input method obtains the text to be reconverted from the client component using the [InputMethodRequests.getSelectedText](http://docs.google.com/java/awt/im/InputMethodRequests.html#getSelectedText(java.text.AttributedCharacterIterator.Attribute%5B%5D)) method. The other InputMethodRequests methods must be prepared to deal with further information requests by the input method. The composed and/or committed text will be sent to the client component as a sequence of InputMethodEvents. If the input method cannot reconvert the given text, the text is returned as committed text in an InputMethodEvent.

**Throws:** [UnsupportedOperationException](http://docs.google.com/java/lang/UnsupportedOperationException.html) - if there is no current input method available or the current input method does not support the reconversion operation.**Since:** 1.3

### dispatchEvent

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

Dispatches an event to the active input method. Called by AWT. If no input method is available, then the event will never be consumed.

**Parameters:**event - The event **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if event is null

### removeNotify

public void **removeNotify**([Component](http://docs.google.com/java/awt/Component.html) client)

Notifies the input context 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 usually called from the client component's [Component.removeNotify](http://docs.google.com/java/awt/Component.html#removeNotify()) method. Potentially pending input from input methods for this component is discarded. If no input methods are available, then this method has no effect.

**Parameters:**client - Client component **Throws:** [NullPointerException](http://docs.google.com/java/lang/NullPointerException.html) - if client is null

### endComposition

public 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. If no input methods are available, then this method has no effect.

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.

### dispose

public void **dispose**()

Releases the resources used by this input context. Called by AWT for the default input context of each Window. If no input methods are available, then this method has no effect.

### getInputMethodControlObject

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

Returns a control object from the current 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.

If no input methods are available or the current input method does not provide an input method control object, then null is returned.

**Returns:**A control object from the current 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/InputContext.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/InputMethodHighlight.html) | [**FRAMES**](http://docs.google.com/index.html?java/awt/im/InputContext.html)    [**NO FRAMES**](http://docs.google.com/InputContext.html)     [**All Classes**](http://docs.google.com/allclasses-noframe.html) |
| SUMMARY: NESTED | FIELD | [CONSTR](#3znysh7) | [METHOD](#2et92p0) | DETAIL: FIELD | [CONSTR](#3dy6vkm) | [METHOD](#4d34og8) |

[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).