public class

KeyEvent

extends InputEvent
implements Parcelable

Summary: Nested Classes | Constants | Inherited Constants | Fields | Inherited Fields | Ctors | Methods | Inherited Methods | [Expand All]

Added in API level 1

java.lang.Object
  ↳android.view.InputEvent
    ↳android.view.KeyEvent

Class Overview

Object used to report key and button events.

Each key press is described by a sequence of key events. A key press starts with a key event with `ACTION_DOWN (/reference/android/view/KeyEvent.html#ACTION_DOWN)`. If the key is held sufficiently long that it repeats, then the initial down is followed additional key events with `ACTION_DOWN (/reference/android/view/KeyEvent.html#ACTION_DOWN)` and a non-zero value for `getRepeatCount() (/reference/android/view/KeyEvent.html#getRepeatCount())`. The last key event is a `ACTION_UP (/reference/android/view/KeyEvent.html#ACTION_UP)` for the key up. If the key press is canceled, the key up event will have the `FLAG_CANCELED (/reference/android/view/KeyEvent.html#FLAG_CANCELED)` flag set.

Key events are generally accompanied by a key code (`getKeyCode() (/reference/android/view/KeyEvent.html#getKeyCode())`), scan code (`getScanCode() (/reference/android/view/KeyEvent.html#getScanCode())`) and meta state (`getMetaState() (/reference/android/view/KeyEvent.html#getMetaState())`). Key code constants are defined in this class. Scan code constants are raw device-specific codes obtained from the OS and so are not generally meaningful to applications unless interpreted using the `KeyCharacterMap (/reference/android/view/KeyCharacterMap.html)`. Meta states describe the pressed state of key modifiers such as `META_SHIFT_ON (/reference/android/view/KeyEvent.html#META_SHIFT_ON)` or `META_ALT_ON (/reference/android/view/KeyEvent.html#META_ALT_ON)`.

Key codes typically correspond one-to-one with individual keys on an input device. Many keys and key combinations serve quite different functions on different input devices so care must be taken when interpreting them. Always use the `KeyCharacterMap (/reference/android/view/KeyCharacterMap.html)` associated with the input device when mapping keys to characters. Be aware that there may be multiple key input devices active at the same time and each will have its own key character map.

As soft input methods can use multiple and inventive ways of inputting text, there is no guarantee that any key press on a soft keyboard will generate a key event: this is left to the IME's discretion, and in fact sending such events is discouraged. You should never rely on receiving KeyEvents for any key on a soft input method. In particular, the default software keyboard will never send any key event to any application targetting Jelly Bean or later, and will only send events for some presses of the delete and return keys to applications targetting Ice Cream Sandwich or earlier. Be aware that other software input methods may never send key events regardless of the version. Consider using editor actions like `IME_ACTION_DONE (/reference/android/view/inputmethod/EditorInfo.html#IME_ACTION_DONE)` if you need specific interaction with the software keyboard, as it gives more visibility to the user as to how your application will react to key presses.

When interacting with an IME, the framework may deliver key events with the special action `ACTION_MULTIPLE` (/reference/android/view/KeyEvent.html#ACTION_MULTIPLE) that either specifies that single repeated key code or a sequence of characters to insert.

In general, the framework cannot guarantee that the key events it delivers to a view always constitute complete key sequences since some events may be dropped or modified by containing views before they are delivered. The view implementation should be prepared to handle `FLAG_CANCELED` (/reference/android/view/KeyEvent.html#FLAG_CANCELED) and should tolerate anomalous situations such as receiving a new `ACTION_DOWN` (/reference/android/view/KeyEvent.html#ACTION_DOWN) without first having received an `ACTION_UP` (/reference/android/view/KeyEvent.html#ACTION_UP) for the prior key press.

Refer to `InputDevice` (/reference/android/view/InputDevice.html) for more information about how different kinds of input devices and sources represent keys and buttons.

Summary

Nested Classes

interface KeyEvent.Callback	
class KeyEvent.DispatcherState	Use with `dispatch(Callback, DispatcherState, Object)` for more advanced key dispatching, such as long presses.

Constants

int ACTION_DOWN	`getAction()` value: the key has been pressed down.
int ACTION_MULTIPLE	`getAction()` value: multiple duplicate key events have occurred in a row, or a complex string is being delivered.
int ACTION_UP	`getAction()` value: the key has been released.
int FLAG_CANCELED	When associated with up key events, this indicates that the key press has been canceled.
int FLAG_CANCELED_LONG_PRESS	Set when a key event has `FLAG_CANCELED` set because a long press action was executed while it was down.
int FLAG_EDITOR_ACTION	This mask is used for compatibility, to identify enter keys that are coming from an IME whose enter key has been auto-labelled "next" or "done".
int FLAG_FALLBACK	Set when a key event has been synthesized to implement default behavior for an event that the application did not handle.
int FLAG_FROM_SYSTEM	This mask is set if an event was known to come from a trusted part of the system.

int FLAG_KEEP_TOUCH_MODE	This mask is set if we don't want the key event to cause us to leave touch mode.
int FLAG_LONG_PRESS	This flag is set for the first key repeat that occurs after the long press timeout.
int FLAG_SOFT_KEYBOARD	This mask is set if the key event was generated by a software keyboard.
int FLAG_TRACKING	Set for `ACTION_UP` when this event's key code is still being tracked from its initial down.
int FLAG_VIRTUAL_HARD_KEY	This key event was generated by a virtual (on-screen) hard key area.
int FLAG_WOKE_HERE	This mask is set if the device woke because of this key event.
int KEYCODE_0	Key code constant: '0' key.
int KEYCODE_1	Key code constant: '1' key.
int KEYCODE_2	Key code constant: '2' key.
int KEYCODE_3	Key code constant: '3' key.
int KEYCODE_3D_MODE	Key code constant: 3D Mode key.
int KEYCODE_4	Key code constant: '4' key.
int KEYCODE_5	Key code constant: '5' key.
int KEYCODE_6	Key code constant: '6' key.
int KEYCODE_7	Key code constant: '7' key.
int KEYCODE_8	Key code constant: '8' key.
int KEYCODE_9	Key code constant: '9' key.
int KEYCODE_A	Key code constant: 'A' key.
int KEYCODE_ALT_LEFT	Key code constant: Left Alt modifier key.
int KEYCODE_ALT_RIGHT	Key code constant: Right Alt modifier key.
int KEYCODE_APOSTROPHE	Key code constant: ''' (apostrophe) key.
int KEYCODE_APP_SWITCH	Key code constant: App switch key.
int KEYCODE_ASSIST	Key code constant: Assist key.
int KEYCODE_AT	Key code constant: '@' key.
int KEYCODE_AVR_INPUT	Key code constant: A/V Receiver input key.
int KEYCODE_AVR_POWER	Key code constant: A/V Receiver power key.
int KEYCODE_B	Key code constant: 'B' key.
int KEYCODE_BACK	Key code constant: Back key.
int KEYCODE_BACKSLASH	Key code constant: '\' key.
int KEYCODE_BOOKMARK	Key code constant: Bookmark key.
int KEYCODE_BREAK	Key code constant: Break / Pause key.
int KEYCODE_BRIGHTNESS_DOWN	Key code constant: Brightness Down key.
int KEYCODE_BRIGHTNESS_UP	Key code constant: Brightness Up key.
int KEYCODE_BUTTON_1	Key code constant: Generic Game Pad Button #1.
int KEYCODE_BUTTON_10	Key code constant: Generic Game Pad Button #10.
int KEYCODE_BUTTON_11	Key code constant: Generic Game Pad Button #11.
int KEYCODE_BUTTON_12	Key code constant: Generic Game Pad Button #12.

int KEYCODE_BUTTON_13	Key code constant: Generic Game Pad Button #13.
int KEYCODE_BUTTON_14	Key code constant: Generic Game Pad Button #14.
int KEYCODE_BUTTON_15	Key code constant: Generic Game Pad Button #15.
int KEYCODE_BUTTON_16	Key code constant: Generic Game Pad Button #16.
int KEYCODE_BUTTON_2	Key code constant: Generic Game Pad Button #2.
int KEYCODE_BUTTON_3	Key code constant: Generic Game Pad Button #3.
int KEYCODE_BUTTON_4	Key code constant: Generic Game Pad Button #4.
int KEYCODE_BUTTON_5	Key code constant: Generic Game Pad Button #5.
int KEYCODE_BUTTON_6	Key code constant: Generic Game Pad Button #6.
int KEYCODE_BUTTON_7	Key code constant: Generic Game Pad Button #7.
int KEYCODE_BUTTON_8	Key code constant: Generic Game Pad Button #8.
int KEYCODE_BUTTON_9	Key code constant: Generic Game Pad Button #9.
int KEYCODE_BUTTON_A	Key code constant: A Button key.
int KEYCODE_BUTTON_B	Key code constant: B Button key.
int KEYCODE_BUTTON_C	Key code constant: C Button key.
int KEYCODE_BUTTON_L1	Key code constant: L1 Button key.
int KEYCODE_BUTTON_L2	Key code constant: L2 Button key.
int KEYCODE_BUTTON_MODE	Key code constant: Mode Button key.
int KEYCODE_BUTTON_R1	Key code constant: R1 Button key.
int KEYCODE_BUTTON_R2	Key code constant: R2 Button key.
int KEYCODE_BUTTON_SELECT	Key code constant: Select Button key.
int KEYCODE_BUTTON_START	Key code constant: Start Button key.
int KEYCODE_BUTTON_THUMBL	Key code constant: Left Thumb Button key.
int KEYCODE_BUTTON_THUMBR	Key code constant: Right Thumb Button key.
int KEYCODE_BUTTON_X	Key code constant: X Button key.
int KEYCODE_BUTTON_Y	Key code constant: Y Button key.
int KEYCODE_BUTTON_Z	Key code constant: Z Button key.
int KEYCODE_C	Key code constant: 'C' key.
int KEYCODE_CALCULATOR	Key code constant: Calculator special function key.
int KEYCODE_CALENDAR	Key code constant: Calendar special function key.
int KEYCODE_CALL	Key code constant: Call key.
int KEYCODE_CAMERA	Key code constant: Camera key.
int KEYCODE_CAPS_LOCK	Key code constant: Caps Lock key.
int KEYCODE_CAPTIONS	Key code constant: Toggle captions key.
int KEYCODE_CHANNEL_DOWN	Key code constant: Channel down key.

int KEYCODE_CHANNEL_UP	Key code constant: Channel up key.
int KEYCODE_CLEAR	Key code constant: Clear key.
int KEYCODE_COMMA	Key code constant: ',' key.
int KEYCODE_CONTACTS	Key code constant: Contacts special function key.
int KEYCODE_CTRL_LEFT	Key code constant: Left Control modifier key.
int KEYCODE_CTRL_RIGHT	Key code constant: Right Control modifier key.
int KEYCODE_D	Key code constant: 'D' key.
int KEYCODE_DEL	Key code constant: Backspace key.
int KEYCODE_DPAD_CENTER	Key code constant: Directional Pad Center key.
int KEYCODE_DPAD_DOWN	Key code constant: Directional Pad Down key.
int KEYCODE_DPAD_LEFT	Key code constant: Directional Pad Left key.
int KEYCODE_DPAD_RIGHT	Key code constant: Directional Pad Right key.
int KEYCODE_DPAD_UP	Key code constant: Directional Pad Up key.
int KEYCODE_DVR	Key code constant: DVR key.
int KEYCODE_E	Key code constant: 'E' key.
int KEYCODE_EISU	Key code constant: Japanese alphanumeric key.
int KEYCODE_ENDCALL	Key code constant: End Call key.
int KEYCODE_ENTER	Key code constant: Enter key.
int KEYCODE_ENVELOPE	Key code constant: Envelope special function key.
int KEYCODE_EQUALS	Key code constant: '=' key.
int KEYCODE_ESCAPE	Key code constant: Escape key.
int KEYCODE_EXPLORER	Key code constant: Explorer special function key.
int KEYCODE_F	Key code constant: 'F' key.
int KEYCODE_F1	Key code constant: F1 key.
int KEYCODE_F10	Key code constant: F10 key.
int KEYCODE_F11	Key code constant: F11 key.
int KEYCODE_F12	Key code constant: F12 key.
int KEYCODE_F2	Key code constant: F2 key.
int KEYCODE_F3	Key code constant: F3 key.
int KEYCODE_F4	Key code constant: F4 key.
int KEYCODE_F5	Key code constant: F5 key.
int KEYCODE_F6	Key code constant: F6 key.
int KEYCODE_F7	Key code constant: F7 key.
int KEYCODE_F8	Key code constant: F8 key.
int KEYCODE_F9	Key code constant: F9 key.
int KEYCODE_FOCUS	Key code constant: Camera Focus key.
int KEYCODE_FORWARD	Key code constant: Forward key.
int KEYCODE_FORWARD_DEL	Key code constant: Forward Delete key.
int KEYCODE_FUNCTION	Key code constant: Function modifier key.
int KEYCODE_G	Key code constant: 'G' key.
int KEYCODE_GRAVE	Key code constant: '`' (backtick) key.

int KEYCODE_GUIDE	Key code constant: Guide key.
int KEYCODE_H	Key code constant: 'H' key.
int KEYCODE_HEADSETHOOK	Key code constant: Headset Hook key.
int KEYCODE_HENKAN	Key code constant: Japanese conversion key.
int KEYCODE_HOME	Key code constant: Home key.
int KEYCODE_I	Key code constant: 'I' key.
int KEYCODE_INFO	Key code constant: Info key.
int KEYCODE_INSERT	Key code constant: Insert key.
int KEYCODE_J	Key code constant: 'J' key.
int KEYCODE_K	Key code constant: 'K' key.
int KEYCODE_KANA	Key code constant: Japanese kana key.
int KEYCODE_KATAKANA_HIRAGANA	Key code constant: Japanese katakana / hiragana key.
int KEYCODE_L	Key code constant: 'L' key.
int KEYCODE_LANGUAGE_SWITCH	Key code constant: Language Switch key.
int KEYCODE_LEFT_BRACKET	Key code constant: '[' key.
int KEYCODE_M	Key code constant: 'M' key.
int KEYCODE_MANNER_MODE	Key code constant: Manner Mode key.
int KEYCODE_MEDIA_AUDIO_TRACK	Key code constant: Audio Track key Switches the audio tracks.
int KEYCODE_MEDIA_CLOSE	Key code constant: Close media key.
int KEYCODE_MEDIA_EJECT	Key code constant: Eject media key.
int KEYCODE_MEDIA_FAST_FORWARD	Key code constant: Fast Forward media key.
int KEYCODE_MEDIA_NEXT	Key code constant: Play Next media key.
int KEYCODE_MEDIA_PAUSE	Key code constant: Pause media key.
int KEYCODE_MEDIA_PLAY	Key code constant: Play media key.
int KEYCODE_MEDIA_PLAY_PAUSE	Key code constant: Play/Pause media key.
int KEYCODE_MEDIA_PREVIOUS	Key code constant: Play Previous media key.
int KEYCODE_MEDIA_RECORD	Key code constant: Record media key.
int KEYCODE_MEDIA_REWIND	Key code constant: Rewind media key.
int KEYCODE_MEDIA_STOP	Key code constant: Stop media key.
int KEYCODE_MENU	Key code constant: Menu key.
int KEYCODE_META_LEFT	Key code constant: Left Meta modifier key.
int KEYCODE_META_RIGHT	Key code constant: Right Meta modifier key.
int KEYCODE_MINUS	Key code constant: '-'.
int KEYCODE_MOVE_END	Key code constant: End Movement key.
int KEYCODE_MOVE_HOME	Key code constant: Home Movement key.
int KEYCODE_MUHENKAN	Key code constant: Japanese non-conversion key.
int KEYCODE_MUSIC	Key code constant: Music special function key.
int KEYCODE_MUTE	Key code constant: Mute key.
int KEYCODE_N	Key code constant: 'N' key.
int KEYCODE_NOTIFICATION	Key code constant: Notification key.
int KEYCODE_NUM	Key code constant: Number modifier key.
int KEYCODE_NUMPAD_0	Key code constant: Numeric keypad '0' key.
int KEYCODE_NUMPAD_1	Key code constant: Numeric keypad '1' key.

int KEYCODE_NUMPAD_2	Key code constant: Numeric keypad '2' key.
int KEYCODE_NUMPAD_3	Key code constant: Numeric keypad '3' key.
int KEYCODE_NUMPAD_4	Key code constant: Numeric keypad '4' key.
int KEYCODE_NUMPAD_5	Key code constant: Numeric keypad '5' key.
int KEYCODE_NUMPAD_6	Key code constant: Numeric keypad '6' key.
int KEYCODE_NUMPAD_7	Key code constant: Numeric keypad '7' key.
int KEYCODE_NUMPAD_8	Key code constant: Numeric keypad '8' key.
int KEYCODE_NUMPAD_9	Key code constant: Numeric keypad '9' key.
int KEYCODE_NUMPAD_ADD	Key code constant: Numeric keypad '+' key (for addition).
int KEYCODE_NUMPAD_COMMA	Key code constant: Numeric keypad ',' key (for decimals or digit grouping).
int KEYCODE_NUMPAD_DIVIDE	Key code constant: Numeric keypad '/' key (for division).
int KEYCODE_NUMPAD_DOT	Key code constant: Numeric keypad '.' key (for decimals or digit grouping).
int KEYCODE_NUMPAD_ENTER	Key code constant: Numeric keypad Enter key.
int KEYCODE_NUMPAD_EQUALS	Key code constant: Numeric keypad '=' key.
int KEYCODE_NUMPAD_LEFT_PAREN	Key code constant: Numeric keypad '(' key.
int KEYCODE_NUMPAD_MULTIPLY	Key code constant: Numeric keypad '*' key (for multiplication).
int KEYCODE_NUMPAD_RIGHT_PAREN	Key code constant: Numeric keypad ')' key.
int KEYCODE_NUMPAD_SUBTRACT	Key code constant: Numeric keypad '-' key (for subtraction).
int KEYCODE_NUM_LOCK	Key code constant: Num Lock key.
int KEYCODE_O	Key code constant: 'O' key.
int KEYCODE_P	Key code constant: 'P' key.
int KEYCODE_PAGE_DOWN	Key code constant: Page Down key.
int KEYCODE_PAGE_UP	Key code constant: Page Up key.
int KEYCODE_PERIOD	Key code constant: '.' key.
int KEYCODE_PICTSYMBOLS	Key code constant: Picture Symbols modifier key.
int KEYCODE_PLUS	Key code constant: '+' key.
int KEYCODE_POUND	Key code constant: '#' key.
int KEYCODE_POWER	Key code constant: Power key.
int KEYCODE_PROG_BLUE	Key code constant: Blue "programmable" key.
int KEYCODE_PROG_GREEN	Key code constant: Green "programmable" key.
int KEYCODE_PROG_RED	Key code constant: Red "programmable" key.
int KEYCODE_PROG_YELLOW	Key code constant: Yellow "programmable" key.
int KEYCODE_Q	Key code constant: 'Q' key.
int KEYCODE_R	Key code constant: 'R' key.
int KEYCODE_RIGHT_BRACKET	Key code constant: ']' key.
int KEYCODE_RO	Key code constant: Japanese Ro key.
int KEYCODE_S	Key code constant: 'S' key.
int KEYCODE_SCROLL_LOCK	Key code constant: Scroll Lock key.

int KEYCODE_SEARCH	Key code constant: Search key.
int KEYCODE_SEMICOLON	Key code constant: ';' key.
int KEYCODE_SETTINGS	Key code constant: Settings key.
int KEYCODE_SHIFT_LEFT	Key code constant: Left Shift modifier key.
int KEYCODE_SHIFT_RIGHT	Key code constant: Right Shift modifier key.
int KEYCODE_SLASH	Key code constant: '/' key.
int KEYCODE_SOFT_LEFT	Key code constant: Soft Left key.
int KEYCODE_SOFT_RIGHT	Key code constant: Soft Right key.
int KEYCODE_SPACE	Key code constant: Space key.
int KEYCODE_STAR	Key code constant: '*' key.
int KEYCODE_STB_INPUT	Key code constant: Set-top-box input key.
int KEYCODE_STB_POWER	Key code constant: Set-top-box power key.
int KEYCODE_SWITCH_CHARSET	Key code constant: Switch Charset modifier key.
int KEYCODE_SYM	Key code constant: Symbol modifier key.
int KEYCODE_SYSRQ	Key code constant: System Request / Print Screen key.
int KEYCODE_T	Key code constant: 'T' key.
int KEYCODE_TAB	Key code constant: Tab key.
int KEYCODE_TV	Key code constant: TV key.
int KEYCODE_TV_INPUT	Key code constant: TV input key.
int KEYCODE_TV_POWER	Key code constant: TV power key.
int KEYCODE_U	Key code constant: 'U' key.
int KEYCODE_UNKNOWN	Key code constant: Unknown key code.
int KEYCODE_V	Key code constant: 'V' key.
int KEYCODE_VOLUME_DOWN	Key code constant: Volume Down key.
int KEYCODE_VOLUME_MUTE	Key code constant: Volume Mute key.
int KEYCODE_VOLUME_UP	Key code constant: Volume Up key.
int KEYCODE_W	Key code constant: 'W' key.
int KEYCODE_WINDOW	Key code constant: Window key.
int KEYCODE_X	Key code constant: 'X' key.
int KEYCODE_Y	Key code constant: 'Y' key.
int KEYCODE_YEN	Key code constant: Japanese Yen key.
int KEYCODE_Z	Key code constant: 'Z' key.
int KEYCODE_ZENKAKU_HANKAKU	Key code constant: Japanese full-width / half-width key.
int KEYCODE_ZOOM_IN	Key code constant: Zoom in key.
int KEYCODE_ZOOM_OUT	Key code constant: Zoom out key.
int MAX_KEYCODE	*This constant was deprecated in API level 3. There are now more than MAX_KEYCODE keycodes. Use `getMaxKeyCode()` instead.*
int META_ALT_LEFT_ON	This mask is used to check whether the left ALT meta key is pressed.
int META_ALT_MASK	This mask is a combination of `META_ALT_ON`, `META_ALT_LEFT_ON` and `META_ALT_RIGHT_ON`.

int META_ALT_ON	This mask is used to check whether one of the ALT meta keys is pressed.
int META_ALT_RIGHT_ON	This mask is used to check whether the right the ALT meta key is pressed.
int META_CAPS_LOCK_ON	This mask is used to check whether the CAPS LOCK meta key is on.
int META_CTRL_LEFT_ON	This mask is used to check whether the left CTRL meta key is pressed.
int META_CTRL_MASK	This mask is a combination of `META_CTRL_ON`, `META_CTRL_LEFT_ON` and `META_CTRL_RIGHT_ON`.
int META_CTRL_ON	This mask is used to check whether one of the CTRL meta keys is pressed.
int META_CTRL_RIGHT_ON	This mask is used to check whether the right CTRL meta key is pressed.
int META_FUNCTION_ON	This mask is used to check whether the FUNCTION meta key is pressed.
int META_META_LEFT_ON	This mask is used to check whether the left META meta key is pressed.
int META_META_MASK	This mask is a combination of `META_META_ON`, `META_META_LEFT_ON` and `META_META_RIGHT_ON`.
int META_META_ON	This mask is used to check whether one of the META meta keys is pressed.
int META_META_RIGHT_ON	This mask is used to check whether the right META meta key is pressed.
int META_NUM_LOCK_ON	This mask is used to check whether the NUM LOCK meta key is on.
int META_SCROLL_LOCK_ON	This mask is used to check whether the SCROLL LOCK meta key is on.
int META_SHIFT_LEFT_ON	This mask is used to check whether the left SHIFT meta key is pressed.

int META_SHIFT_MASK	This mask is a combination of `META_SHIFT_ON`, `META_SHIFT_LEFT_ON` and `META_SHIFT_RIGHT_ON`.
int META_SHIFT_ON	This mask is used to check whether one of the SHIFT meta keys is pressed.
int META_SHIFT_RIGHT_ON	This mask is used to check whether the right SHIFT meta key is pressed.
int META_SYM_ON	This mask is used to check whether the SYM meta key is pressed.

Inherited Constants [Expand]

▸ From interface android.os.Parcelable

Fields

public static final Creator<KeyEvent> CREATOR

Inherited Fields [Expand]

▸ From class android.view.InputEvent

Public Constructors

KeyEvent (int action, int code)
Create a new key event.

KeyEvent (long downTime, long eventTime, int action, int code, int repeat)
Create a new key event.

KeyEvent (long downTime, long eventTime, int action, int code, int repeat, int metaState)
Create a new key event.

KeyEvent (long downTime, long eventTime, int action, int code, int repeat, int metaState, int deviceId, int scan
Create a new key event.

KeyEvent (long downTime, long eventTime, int action, int code, int repeat, int metaState, int deviceId, int scan
Create a new key event.

KeyEvent (long downTime, long eventTime, int action, int code, int repeat, int metaState, int deviceId, int scan
Create a new key event.

KeyEvent (long time, String characters, int deviceId, int flags)
Create a new key event for a string of characters.

KeyEvent (KeyEvent origEvent)
Make an exact copy of an existing key event.

KeyEvent (KeyEvent origEvent, long eventTime, int newRepeat)
This constructor was deprecated in API level 5. Use `changeTimeRepeat(KeyEvent, long, int)` instea

Public Methods

static KeyEvent	changeAction (KeyEvent event, int action) Create a new key event that is the same as the given one, but whose action is replaced with the given value.
static KeyEvent	changeFlags (KeyEvent event, int flags) Create a new key event that is the same as the given one, but whose flags are replaced with the given value.
static KeyEvent	changeTimeRepeat (KeyEvent event, long eventTime, int newRepeat) Create a new key event that is the same as the given one, but whose event time and repeat count are replaced with the given value.

static KeyEvent	changeTimeRepeat (KeyEvent event, long eventTime, int newRepeat, int newFlags) Create a new key event that is the same as the given one, but whose event time and repeat count are replaced with the given value.
final boolean	dispatch (KeyEvent.Callback receiver, KeyEvent.DispatcherState state, Object target) Deliver this key event to a `KeyEvent.Callback` interface.
final boolean	dispatch (KeyEvent.Callback receiver) *This method was deprecated in API level 5. Use `dispatch(Callback, DispatcherState, Object)` instead.*
final int	getAction () Retrieve the action of this key event.
final String	getCharacters () For the special case of a `ACTION_MULTIPLE` event with key code of `KEYCODE_UNKNOWN`, this is a raw string of characters associated with the event.
static int	getDeadChar (int accent, int c) Get the character that is produced by putting accent on the character c.
final int	getDeviceId () Gets the id for the device that this event came from.
char	getDisplayLabel () Gets the primary character for this key.
final long	getDownTime () Retrieve the time of the most recent key down event, in the `uptimeMillis()` time base.
final long	getEventTime () Retrieve the time this event occurred, in the `uptimeMillis()` time base.
final int	getFlags () Returns the flags for this key event.
final KeyCharacterMap	getKeyCharacterMap () Gets the `KeyCharacterMap` associated with the keyboard device.
final int	getKeyCode () Retrieve the key code of the key event.
boolean	getKeyData (KeyCharacterMap.KeyData results) *This method was deprecated in API level 11. instead use `getDisplayLabel()`, `getNumber()` or `getUnicodeChar(int)`.*
char	getMatch (char[] chars) Gets the first character in the character array that can be generated by the specified key code.
char	getMatch (char[] chars, int metaState) Gets the first character in the character array that can be generated by the specified key code.
static int	getMaxKeyCode () Returns the maximum keycode.
final int	getMetaState () Returns the state of the meta keys.
static int	getModifierMetaStateMask () Gets a mask that includes all valid modifier key meta state bits.

final int	getModifiers () Returns the state of the modifier keys.
char	getNumber () Gets the number or symbol associated with the key.
final int	getRepeatCount () Retrieve the repeat count of the event.
final int	getScanCode () Retrieve the hardware key id of this key event. These values are not reliable and vary from device to device.
final int	getSource () Gets the source of the event.
int	getUnicodeChar () Gets the Unicode character generated by the specified key and meta key state combination.
int	getUnicodeChar (int metaState) Gets the Unicode character generated by the specified key and meta key state combination.
final boolean	hasModifiers (int modifiers) Returns true if only the specified modifiers keys are pressed.
final boolean	hasNoModifiers () Returns true if no modifier keys are pressed.
final boolean	isAltPressed () Returns the pressed state of the ALT meta key.
final boolean	isCanceled () For `ACTION_UP` events, indicates that the event has been canceled as per `FLAG_CANCELED`.
final boolean	isCapsLockOn () Returns the locked state of the CAPS LOCK meta key.
final boolean	isCtrlPressed () Returns the pressed state of the CTRL meta key.
final boolean	isFunctionPressed () Returns the pressed state of the FUNCTION meta key.
final static boolean	isGamepadButton (int keyCode) Returns true if the specified keycode is a gamepad button.
final boolean	isLongPress () For `ACTION_DOWN` events, indicates that the event has been canceled as per `FLAG_LONG_PRESS`.
final boolean	isMetaPressed () Returns the pressed state of the META meta key.
static boolean	isModifierKey (int keyCode) Returns true if this key code is a modifier key.

final boolean	isNumLockOn () Returns the locked state of the NUM LOCK meta key.
boolean	isPrintingKey () Returns true if this key produces a glyph.
final boolean	isScrollLockOn () Returns the locked state of the SCROLL LOCK meta key.
final boolean	isShiftPressed () Returns the pressed state of the SHIFT meta key.
final boolean	isSymPressed () Returns the pressed state of the SYM meta key.
final boolean	isSystem () Is this a system key? System keys can not be used for menu shortcuts.
final boolean	isTracking () For `ACTION_UP` events, indicates that the event is still being tracked from its initial down event as per `FLAG_TRACKING`.
static int	keyCodeFromString (String symbolicName) Gets a keycode by its symbolic name such as "KEYCODE_A" or an equivalent numeric constant such as "1001".
static String	keyCodeToString (int keyCode) Returns a string that represents the symbolic name of the specified keycode such as "KEYCODE_A", "KEYCODE_DPAD_UP", or an equivalent numeric constant such as "1001" if unknown.
static boolean	metaStateHasModifiers (int metaState, int modifiers) Returns true if only the specified modifier keys are pressed according to the specified meta state.
static boolean	metaStateHasNoModifiers (int metaState) Returns true if no modifiers keys are pressed according to the specified meta state.
static int	normalizeMetaState (int metaState) Normalizes the specified meta state.
final void	setSource (int source) Modifies the source of the event.
final void	startTracking () Call this during `onKeyDown(int, KeyEvent)` to have the system track the key through its final up (possibly including a long press).
String	toString () Returns a string containing a concise, human-readable description of this object.
void	writeToParcel (Parcel out, int flags) Flatten this object in to a Parcel.

Inherited Methods [Expand]

▶ From class android.view.InputEvent

▶ From class java.lang.Object

▶ From interface android.os.Parcelable

Constants

public static final int **ACTION_DOWN**

Added in API level 1

`getAction()` (/reference/android/view/KeyEvent.html#getAction()) value: the key has been pressed down.

Constant Value: 0 (0x00000000)

public static final int **ACTION_MULTIPLE**

Added in API level 1

`getAction()` (/reference/android/view/KeyEvent.html#getAction()) value: multiple duplicate key events have occurred in a row, or a complex string is being delivered. If the key code is not {#link `KEYCODE_UNKNOWN` (/reference/android/view/KeyEvent.html#KEYCODE_UNKNOWN) then the {#link `getRepeatCount()` (/reference/android/view/KeyEvent.html#getRepeatCount()) method returns the number of times the given key code should be executed. Otherwise, if the key code is `KEYCODE_UNKNOWN` (/reference/android/view/KeyEvent.html#KEYCODE_UNKNOWN), then this is a sequence of characters as returned by `getCharacters()` (/reference/android/view/KeyEvent.html#getCharacters()).

Constant Value: 2 (0x00000002)

public static final int **ACTION_UP**

Added in API level 1

`getAction()` (/reference/android/view/KeyEvent.html#getAction()) value: the key has been released.

Constant Value: 1 (0x00000001)

public static final int **FLAG_CANCELED**

Added in API level 5

When associated with up key events, this indicates that the key press has been canceled. Typically this is used with virtual touch screen keys, where the user can slide from the virtual key area on to the display: in that case, the application will receive a canceled up event and should not perform the action normally associated with the key. Note that for this to work, the application can not perform an action for a key until it receives an up or the long press timeout has expired.

Constant Value: 32 (0x00000020)

public static final int **FLAG_CANCELED_LONG_PRESS**

Added in API level 5

Set when a key event has `FLAG_CANCELED` (/reference/android/view/KeyEvent.html#FLAG_CANCELED) set because a long press action was executed while it was down.

Constant Value: 256 (0x00000100)

public static final int **FLAG_EDITOR_ACTION**

Added in API level 3

This mask is used for compatibility, to identify enter keys that are coming from an IME whose enter key has been auto-labelled "next" or "done". This allows TextView to dispatch these as normal enter keys for old applications, but still do the appropriate action when receiving them.

Constant Value: 16 (0x00000010)

public static final int **FLAG_FALLBACK**

Added in API level 11

Set when a key event has been synthesized to implement default behavior for an event that the application did not handle. Fallback key events are generated by unhandled trackball motions (to emulate a directional keypad) and by certain unhandled key presses that are declared in the key map (such as special function numeric keypad keys when numlock is off).

Constant Value: 1024 (0x00000400)

public static final int **FLAG_FROM_SYSTEM**

Added in API level 3

This mask is set if an event was known to come from a trusted part of the system. That is, the event is known to come from the user, and could not have been spoofed by a third party component.

Constant Value: 8 (0x00000008)

public static final int **FLAG_KEEP_TOUCH_MODE**

Added in API level 3

This mask is set if we don't want the key event to cause us to leave touch mode.

Constant Value: 4 (0x00000004)

public static final int **FLAG_LONG_PRESS**

Added in API level 5

This flag is set for the first key repeat that occurs after the long press timeout.

Constant Value: 128 (0x00000080)

public static final int **FLAG_SOFT_KEYBOARD**

Added in API level 3

This mask is set if the key event was generated by a software keyboard.

Constant Value: 2 (0x00000002)

public static final int **FLAG_TRACKING**

Added in API level 5

Set for `ACTION_UP` (/reference/android/view/KeyEvent.html#ACTION_UP) when this event's key code is still being tracked from its initial down. That is, somebody requested that tracking started on the key down and a long press has not caused the tracking to be canceled.

Constant Value: 512 (0x00000200)

public static final int **FLAG_VIRTUAL_HARD_KEY**

Added in API level 5

This key event was generated by a virtual (on-screen) hard key area. Typically this is an area of the touchscreen, outside of the regular display, dedicated to "hardware" buttons.

Constant Value: 64 (0x00000040)

public static final int **FLAG_WOKE_HERE**

Added in API level 1

This mask is set if the device woke because of this key event.

Constant Value: 1 (0x00000001)

public static final int **KEYCODE_0**

Added in API level 1

Key code constant: '0' key.

Constant Value: 7 (0x00000007)

public static final int **KEYCODE_1**

Added in API level 1

Key code constant: '1' key.

Constant Value: 8 (0x00000008)

public static final int **KEYCODE_2**

Added in API level 1

Key code constant: '2' key.

Constant Value: 9 (0x00000009)

public static final int **KEYCODE_3**

Added in API level 1

Key code constant: '3' key.

Constant Value: 10 (0x0000000a)

public static final int **KEYCODE_3D_MODE**

Added in API level 14

Key code constant: 3D Mode key. Toggles the display between 2D and 3D mode.

Constant Value: 206 (0x000000ce)

public static final int **KEYCODE_4**

Added in API level 1

Key code constant: '4' key.

Constant Value: 11 (0x0000000b)

public static final int **KEYCODE_5**

Added in API level 1

Key code constant: '5' key.

Constant Value: 12 (0x0000000c)

public static final int **KEYCODE_6**

Added in API level 1

Key code constant: '6' key.

Constant Value: 13 (0x0000000d)

public static final int **KEYCODE_7**

Added in API level 1

Key code constant: '7' key.

Constant Value: 14 (0x0000000e)

public static final int **KEYCODE_8**

Added in API level 1

Key code constant: '8' key.

Constant Value: 15 (0x0000000f)

public static final int **KEYCODE_9** Added in API level 1

Key code constant: '9' key.

Constant Value: 16 (0x00000010)

public static final int **KEYCODE_A** Added in API level 1

Key code constant: 'A' key.

Constant Value: 29 (0x0000001d)

public static final int **KEYCODE_ALT_LEFT** Added in API level 1

Key code constant: Left Alt modifier key.

Constant Value: 57 (0x00000039)

public static final int **KEYCODE_ALT_RIGHT** Added in API level 1

Key code constant: Right Alt modifier key.

Constant Value: 58 (0x0000003a)

public static final int **KEYCODE_APOSTROPHE** Added in API level 1

Key code constant: ''' (apostrophe) key.

Constant Value: 75 (0x0000004b)

public static final int **KEYCODE_APP_SWITCH** Added in API level 11

Key code constant: App switch key. Should bring up the application switcher dialog.

Constant Value: 187 (0x000000bb)

public static final int **KEYCODE_ASSIST** Added in API level 16

Key code constant: Assist key. Launches the global assist activity. Not delivered to applications.

Constant Value: 219 (0x000000db)

public static final int **KEYCODE_AT** Added in API level 1

Key code constant: '@' key.

Constant Value: 77 (0x0000004d)

public static final int **KEYCODE_AVR_INPUT** Added in API level 11

Key code constant: A/V Receiver input key. On TV remotes, switches the input mode on an external A/V Receiver.

Constant Value: 182 (0x000000b6)

public static final int **KEYCODE_AVR_POWER** Added in API level 11

Key code constant: A/V Receiver power key. On TV remotes, toggles the power on an external A/V Receiver.

Constant Value: 181 (0x000000b5)

public static final int **KEYCODE_B** Added in API level 1

Key code constant: 'B' key.

Constant Value: 30 (0x0000001e)

public static final int **KEYCODE_BACK** Added in API level 1

Key code constant: Back key.

Constant Value: 4 (0x00000004)

public static final int **KEYCODE_BACKSLASH** Added in API level 1

Key code constant: '\' key.

Constant Value: 73 (0x00000049)

public static final int **KEYCODE_BOOKMARK** Added in API level 11

Key code constant: Bookmark key. On some TV remotes, bookmarks content or web pages.

Constant Value: 174 (0x000000ae)

public static final int **KEYCODE_BREAK** Added in API level 11

Key code constant: Break / Pause key.

Constant Value: 121 (0x00000079)

public static final int **KEYCODE_BRIGHTNESS_DOWN** Added in API level 18

Key code constant: Brightness Down key. Adjusts the screen brightness down.

Constant Value: 220 (0x000000dc)

public static final int **KEYCODE_BRIGHTNESS_UP** Added in API level 18

Key code constant: Brightness Up key. Adjusts the screen brightness up.

Constant Value: 221 (0x000000dd)

public static final int **KEYCODE_BUTTON_1** Added in API level 12

Key code constant: Generic Game Pad Button #1.

Constant Value: 188 (0x000000bc)

public static final int **KEYCODE_BUTTON_10**

Added in API level 12

Key code constant: Generic Game Pad Button #10.

Constant Value: 197 (0x000000c5)

public static final int **KEYCODE_BUTTON_11**

Added in API level 12

Key code constant: Generic Game Pad Button #11.

Constant Value: 198 (0x000000c6)

public static final int **KEYCODE_BUTTON_12**

Added in API level 12

Key code constant: Generic Game Pad Button #12.

Constant Value: 199 (0x000000c7)

public static final int **KEYCODE_BUTTON_13**

Added in API level 12

Key code constant: Generic Game Pad Button #13.

Constant Value: 200 (0x000000c8)

public static final int **KEYCODE_BUTTON_14**

Added in API level 12

Key code constant: Generic Game Pad Button #14.

Constant Value: 201 (0x000000c9)

public static final int **KEYCODE_BUTTON_15**

Added in API level 12

Key code constant: Generic Game Pad Button #15.

Constant Value: 202 (0x000000ca)

public static final int **KEYCODE_BUTTON_16**

Added in API level 12

Key code constant: Generic Game Pad Button #16.

Constant Value: 203 (0x000000cb)

public static final int **KEYCODE_BUTTON_2**

Added in API level 12

Key code constant: Generic Game Pad Button #2.

Constant Value: 189 (0x000000bd)

public static final int **KEYCODE_BUTTON_3**

Added in API level 12

Key code constant: Generic Game Pad Button #3.

Constant Value: 190 (0x000000be)

public static final int **KEYCODE_BUTTON_4**

Added in API level 12

Key code constant: Generic Game Pad Button #4.

Constant Value: 191 (0x000000bf)

public static final int **KEYCODE_BUTTON_5**

Added in API level 12

Key code constant: Generic Game Pad Button #5.

Constant Value: 192 (0x000000c0)

public static final int **KEYCODE_BUTTON_6**

Added in API level 12

Key code constant: Generic Game Pad Button #6.

Constant Value: 193 (0x000000c1)

public static final int **KEYCODE_BUTTON_7**

Added in API level 12

Key code constant: Generic Game Pad Button #7.

Constant Value: 194 (0x000000c2)

public static final int **KEYCODE_BUTTON_8**

Added in API level 12

Key code constant: Generic Game Pad Button #8.

Constant Value: 195 (0x000000c3)

public static final int **KEYCODE_BUTTON_9**

Added in API level 12

Key code constant: Generic Game Pad Button #9.

Constant Value: 196 (0x000000c4)

public static final int **KEYCODE_BUTTON_A**

Added in API level 9

Key code constant: A Button key. On a game controller, the A button should be either the button labeled A or the first button on the bottom row of controller buttons.

Constant Value: 96 (0x00000060)

public static final int **KEYCODE_BUTTON_B**

Added in API level 9

Key code constant: B Button key. On a game controller, the B button should be either the button labeled B or the second button on the bottom row of controller buttons.

Constant Value: 97 (0x00000061)

public static final int **KEYCODE_BUTTON_C**

Added in API level 9

Key code constant: C Button key. On a game controller, the C button should be either the button labeled C or the third button on the bottom row of controller buttons.

Constant Value: 98 (0x00000062)

public static final int **KEYCODE_BUTTON_L1**

Added in API level 9

Key code constant: L1 Button key. On a game controller, the L1 button should be either the button labeled L1 (or L) or the top left trigger button.

Constant Value: 102 (0x00000066)

public static final int **KEYCODE_BUTTON_L2**

Added in API level 9

Key code constant: L2 Button key. On a game controller, the L2 button should be either the button labeled L2 or the bottom left trigger button.

Constant Value: 104 (0x00000068)

public static final int **KEYCODE_BUTTON_MODE**

Added in API level 9

Key code constant: Mode Button key. On a game controller, the button labeled Mode.

Constant Value: 110 (0x0000006e)

public static final int **KEYCODE_BUTTON_R1**

Added in API level 9

Key code constant: R1 Button key. On a game controller, the R1 button should be either the button labeled R1 (or R) or the top right trigger button.

Constant Value: 103 (0x00000067)

public static final int **KEYCODE_BUTTON_R2**

Added in API level 9

Key code constant: R2 Button key. On a game controller, the R2 button should be either the button labeled R2 or the bottom right trigger button.

Constant Value: 105 (0x00000069)

public static final int **KEYCODE_BUTTON_SELECT**

Added in API level 9

Key code constant: Select Button key. On a game controller, the button labeled Select.

Constant Value: 109 (0x0000006d)

public static final int **KEYCODE_BUTTON_START**

Added in API level 9

Key code constant: Start Button key. On a game controller, the button labeled Start.

Constant Value: 108 (0x0000006c)

public static final int **KEYCODE_BUTTON_THUMBL**

Added in API level 9

Key code constant: Left Thumb Button key. On a game controller, the left thumb button indicates that the left (or only) joystick is pressed.

Constant Value: 106 (0x0000006a)

public static final int **KEYCODE_BUTTON_THUMBR**

Added in API level 9

Key code constant: Right Thumb Button key. On a game controller, the right thumb button indicates that the right joystick is pressed.

Constant Value: 107 (0x0000006b)

public static final int **KEYCODE_BUTTON_X**

Added in API level 9

Key code constant: X Button key. On a game controller, the X button should be either the button labeled X or the first button on the upper row of controller buttons.

Constant Value: 99 (0x00000063)

public static final int **KEYCODE_BUTTON_Y**

Added in API level 9

Key code constant: Y Button key. On a game controller, the Y button should be either the button labeled Y or the second button on the upper row of controller buttons.

Constant Value: 100 (0x00000064)

public static final int **KEYCODE_BUTTON_Z**

Added in API level 9

Key code constant: Z Button key. On a game controller, the Z button should be either the button labeled Z or the third button on the upper row of controller buttons.

Constant Value: 101 (0x00000065)

public static final int **KEYCODE_C**

Added in API level 1

Key code constant: 'C' key.

Constant Value: 31 (0x0000001f)

public static final int **KEYCODE_CALCULATOR**

Added in API level 15

Key code constant: Calculator special function key. Used to launch a calculator application.

Constant Value: 210 (0x000000d2)

public static final int **KEYCODE_CALENDAR**

Added in API level 15

Key code constant: Calendar special function key. Used to launch a calendar application.

Constant Value: 208 (0x000000d0)

public static final int **KEYCODE_CALL**

Added in API level 1

Key code constant: Call key.

Constant Value: 5 (0x00000005)

public static final int **KEYCODE_CAMERA**

Added in API level 1

Key code constant: Camera key. Used to launch a camera application or take pictures.

Constant Value: 27 (0x0000001b)

public static final int **KEYCODE_CAPS_LOCK**

Key code constant: Caps Lock key.

Constant Value: 115 (0x00000073)

public static final int **KEYCODE_CAPTIONS**

Added in API level 11

Key code constant: Toggle captions key. Switches the mode for closed-captioning text, for example during television shows.

Constant Value: 175 (0x000000af)

public static final int **KEYCODE_CHANNEL_DOWN**

Added in API level 11

Key code constant: Channel down key. On TV remotes, decrements the television channel.

Constant Value: 167 (0x000000a7)

public static final int **KEYCODE_CHANNEL_UP**

Added in API level 11

Key code constant: Channel up key. On TV remotes, increments the television channel.

Constant Value: 166 (0x000000a6)

public static final int **KEYCODE_CLEAR**

Added in API level 1

Key code constant: Clear key.

Constant Value: 28 (0x0000001c)

public static final int **KEYCODE_COMMA**

Added in API level 1

Key code constant: ',' key.

Constant Value: 55 (0x00000037)

public static final int **KEYCODE_CONTACTS**

Added in API level 15

Key code constant: Contacts special function key. Used to launch an address book application.

Constant Value: 207 (0x000000cf)

public static final int **KEYCODE_CTRL_LEFT**

Added in API level 11

Key code constant: Left Control modifier key.

Constant Value: 113 (0x00000071)

public static final int **KEYCODE_CTRL_RIGHT**

Added in API level 11

Key code constant: Right Control modifier key.

Constant Value: 114 (0x00000072)

public static final int **KEYCODE_D**

Added in API level 1

Key code constant: 'D' key.

Constant Value: 32 (0x00000020)

public static final int **KEYCODE_DEL**

Added in API level 1

Key code constant: Backspace key. Deletes characters before the insertion point, unlike `KEYCODE_FORWARD_DEL` (/reference/android/view/KeyEvent.html#KEYCODE_FORWARD_DEL).

Constant Value: 67 (0x00000043)

public static final int **KEYCODE_DPAD_CENTER**

Added in API level 1

Key code constant: Directional Pad Center key. May also be synthesized from trackball motions.

Constant Value: 23 (0x00000017)

public static final int **KEYCODE_DPAD_DOWN**

Added in API level 1

Key code constant: Directional Pad Down key. May also be synthesized from trackball motions.

Constant Value: 20 (0x00000014)

public static final int **KEYCODE_DPAD_LEFT**

Added in API level 1

Key code constant: Directional Pad Left key. May also be synthesized from trackball motions.

Constant Value: 21 (0x00000015)

public static final int **KEYCODE_DPAD_RIGHT**

Added in API level 1

Key code constant: Directional Pad Right key. May also be synthesized from trackball motions.

Constant Value: 22 (0x00000016)

public static final int **KEYCODE_DPAD_UP**

Added in API level 1

Key code constant: Directional Pad Up key. May also be synthesized from trackball motions.

Constant Value: 19 (0x00000013)

public static final int **KEYCODE_DVR**

Added in API level 11

Key code constant: DVR key. On some TV remotes, switches to a DVR mode for recorded shows.

Constant Value: 173 (0x000000ad)

public static final int **KEYCODE_E**

Added in API level 1

Key code constant: 'E' key.

Constant Value: 33 (0x00000021)

public static final int **KEYCODE_EISU**

Added in API level 16

Key code constant: Japanese alphanumeric key.

Constant Value: 212 (0x000000d4)

public static final int **KEYCODE_ENDCALL**

Added in API level 1

Key code constant: End Call key.

Constant Value: 6 (0x00000006)

public static final int **KEYCODE_ENTER**

Added in API level 1

Key code constant: Enter key.

Constant Value: 66 (0x00000042)

public static final int **KEYCODE_ENVELOPE**

Added in API level 1

Key code constant: Envelope special function key. Used to launch a mail application.

Constant Value: 65 (0x00000041)

public static final int **KEYCODE_EQUALS**

Added in API level 1

Key code constant: '=' key.

Constant Value: 70 (0x00000046)

public static final int **KEYCODE_ESCAPE**

Added in API level 11

Key code constant: Escape key.

Constant Value: 111 (0x0000006f)

public static final int **KEYCODE_EXPLORER**

Added in API level 1

Key code constant: Explorer special function key. Used to launch a browser application.

Constant Value: 64 (0x00000040)

public static final int **KEYCODE_F**

Added in API level 1

Key code constant: 'F' key.

Constant Value: 34 (0x00000022)

public static final int **KEYCODE_F1**

Added in API level 11

Key code constant: F1 key.

Constant Value: 131 (0x00000083)

public static final int **KEYCODE_F10**

Added in API level 11

Key code constant: F10 key.

Constant Value: 140 (0x0000008c)

public static final int **KEYCODE_F11**

Added in API level 11

Key code constant: F11 key.

Constant Value: 141 (0x0000008d)

public static final int **KEYCODE_F12**

Added in API level 11

Key code constant: F12 key.

Constant Value: 142 (0x0000008e)

public static final int **KEYCODE_F2**

Added in API level 11

Key code constant: F2 key.

Constant Value: 132 (0x00000084)

public static final int **KEYCODE_F3**

Added in API level 11

Key code constant: F3 key.

Constant Value: 133 (0x00000085)

public static final int **KEYCODE_F4**

Added in API level 11

Key code constant: F4 key.

Constant Value: 134 (0x00000086)

public static final int **KEYCODE_F5**

Added in API level 11

Key code constant: F5 key.

Constant Value: 135 (0x00000087)

public static final int **KEYCODE_F6**

Added in API level 11

Key code constant: F6 key.

Constant Value: 136 (0x00000088)

public static final int **KEYCODE_F7**

Added in API level 11

Key code constant: F7 key.

Constant Value: 137 (0x00000089)

public static final int **KEYCODE_F8**

Added in API level 11

Key code constant: F8 key.

Constant Value: 138 (0x0000008a)

public static final int **KEYCODE_F9**

Added in API level 11

Key code constant: F9 key.

Constant Value: 139 (0x0000008b)

public static final int **KEYCODE_FOCUS**

Added in API level 1

Key code constant: Camera Focus key. Used to focus the camera.

Constant Value: 80 (0x00000050)

public static final int **KEYCODE_FORWARD**

Added in API level 11

Key code constant: Forward key. Navigates forward in the history stack. Complement of `KEYCODE_BACK` (/reference/android/view/KeyEvent.html#KEYCODE_BACK).

Constant Value: 125 (0x0000007d)

public static final int **KEYCODE_FORWARD_DEL**

Added in API level 11

Key code constant: Forward Delete key. Deletes characters ahead of the insertion point, unlike `KEYCODE_DEL` (/reference/android/view/KeyEvent.html#KEYCODE_DEL).

Constant Value: 112 (0x00000070)

public static final int **KEYCODE_FUNCTION**

Added in API level 11

Key code constant: Function modifier key.

Constant Value: 119 (0x00000077)

public static final int **KEYCODE_G**

Added in API level 1

Key code constant: 'G' key.

Constant Value: 35 (0x00000023)

public static final int **KEYCODE_GRAVE**

Added in API level 1

Key code constant: '`' (backtick) key.

Constant Value: 68 (0x00000044)

public static final int **KEYCODE_GUIDE**

Added in API level 11

Key code constant: Guide key. On TV remotes, shows a programming guide.

Constant Value: 172 (0x000000ac)

public static final int **KEYCODE_H**

Added in API level 1

Key code constant: 'H' key.

Constant Value: 36 (0x00000024)

public static final int **KEYCODE_HEADSETHOOK**

Added in API level 1

Key code constant: Headset Hook key. Used to hang up calls and stop media.

Constant Value: 79 (0x0000004f)

public static final int **KEYCODE_HENKAN**

Added in API level 16

Key code constant: Japanese conversion key.

Constant Value: 214 (0x000000d6)

public static final int **KEYCODE_HOME**

Added in API level 1

Key code constant: Home key. This key is handled by the framework and is never delivered to applications.

Constant Value: 3 (0x00000003)

public static final int **KEYCODE_I**

Added in API level 1

Key code constant: 'I' key.

Constant Value: 37 (0x00000025)

public static final int **KEYCODE_INFO**

Added in API level 11

Key code constant: Info key. Common on TV remotes to show additional information related to what is currently being viewed.

Constant Value: 165 (0x000000a5)

public static final int **KEYCODE_INSERT**

Added in API level 11

Key code constant: Insert key. Toggles insert / overwrite edit mode.

Constant Value: 124 (0x0000007c)

public static final int **KEYCODE_J**

Added in API level 1

Key code constant: 'J' key.

Constant Value: 38 (0x00000026)

public static final int **KEYCODE_K**

Added in API level 1

Key code constant: 'K' key.

Constant Value: 39 (0x00000027)

public static final int **KEYCODE_KANA**

Added in API level 16

Key code constant: Japanese kana key.

Constant Value: 218 (0x000000da)

public static final int **KEYCODE_KATAKANA_HIRAGANA**

Added in API level 16

Key code constant: Japanese katakana / hiragana key.

Constant Value: 215 (0x000000d7)

public static final int **KEYCODE_L**

Added in API level 1

Key code constant: 'L' key.

Constant Value: 40 (0x00000028)

public static final int **KEYCODE_LANGUAGE_SWITCH**

Added in API level 14

Key code constant: Language Switch key. Toggles the current input language such as switching between English and Japanese on a QWERTY keyboard. On some devices, the same function may be performed by pressing Shift+Spacebar.

Constant Value: 204 (0x000000cc)

public static final int **KEYCODE_LEFT_BRACKET**

Added in API level 1

Key code constant: '[' key.

Constant Value: 71 (0x00000047)

public static final int **KEYCODE_M**

Added in API level 1

Key code constant: 'M' key.

Constant Value: 41 (0x00000029)

public static final int **KEYCODE_MANNER_MODE**

Added in API level 14

Key code constant: Manner Mode key. Toggles silent or vibrate mode on and off to make the device behave more politely in certain settings such as on a crowded train. On some devices, the key may only operate when long-pressed.

Constant Value: 205 (0x000000cd)

public static final int **KEYCODE_MEDIA_AUDIO_TRACK**

Added in API level 19

Key code constant: Audio Track key Switches the audio tracks.

Constant Value: 222 (0x000000de)

public static final int **KEYCODE_MEDIA_CLOSE**

Added in API level 11

Key code constant: Close media key. May be used to close a CD tray, for example.

Constant Value: 128 (0x00000080)

public static final int **KEYCODE_MEDIA_EJECT**

Added in API level 11

Key code constant: Eject media key. May be used to eject a CD tray, for example.

Constant Value: 129 (0x00000081)

public static final int **KEYCODE_MEDIA_FAST_FORWARD**

Added in API level 3

Key code constant: Fast Forward media key.

Constant Value: 90 (0x0000005a)

public static final int **KEYCODE_MEDIA_NEXT** Added in API level 3

Key code constant: Play Next media key.

Constant Value: 87 (0x00000057)

public static final int **KEYCODE_MEDIA_PAUSE** Added in API level 11

Key code constant: Pause media key.

Constant Value: 127 (0x0000007f)

public static final int **KEYCODE_MEDIA_PLAY** Added in API level 11

Key code constant: Play media key.

Constant Value: 126 (0x0000007e)

public static final int **KEYCODE_MEDIA_PLAY_PAUSE** Added in API level 3

Key code constant: Play/Pause media key.

Constant Value: 85 (0x00000055)

public static final int **KEYCODE_MEDIA_PREVIOUS** Added in API level 3

Key code constant: Play Previous media key.

Constant Value: 88 (0x00000058)

public static final int **KEYCODE_MEDIA_RECORD** Added in API level 11

Key code constant: Record media key.

Constant Value: 130 (0x00000082)

public static final int **KEYCODE_MEDIA_REWIND** Added in API level 3

Key code constant: Rewind media key.

Constant Value: 89 (0x00000059)

public static final int **KEYCODE_MEDIA_STOP** Added in API level 3

Key code constant: Stop media key.

Constant Value: 86 (0x00000056)

public static final int **KEYCODE_MENU** Added in API level 1

Key code constant: Menu key.

Constant Value: 82 (0x00000052)

public static final int **KEYCODE_META_LEFT**

Added in API level 11

Key code constant: Left Meta modifier key.

Constant Value: 117 (0x00000075)

public static final int **KEYCODE_META_RIGHT**

Added in API level 11

Key code constant: Right Meta modifier key.

Constant Value: 118 (0x00000076)

public static final int **KEYCODE_MINUS**

Added in API level 1

Key code constant: '-'.

Constant Value: 69 (0x00000045)

public static final int **KEYCODE_MOVE_END**

Added in API level 11

Key code constant: End Movement key. Used for scrolling or moving the cursor around to the end of a line or to the bottom of a list.

Constant Value: 123 (0x0000007b)

public static final int **KEYCODE_MOVE_HOME**

Added in API level 11

Key code constant: Home Movement key. Used for scrolling or moving the cursor around to the start of a line or to the top of a list.

Constant Value: 122 (0x0000007a)

public static final int **KEYCODE_MUHENKAN**

Added in API level 16

Key code constant: Japanese non-conversion key.

Constant Value: 213 (0x000000d5)

public static final int **KEYCODE_MUSIC**

Added in API level 15

Key code constant: Music special function key. Used to launch a music player application.

Constant Value: 209 (0x000000d1)

public static final int **KEYCODE_MUTE**

Added in API level 3

Key code constant: Mute key. Mutes the microphone, unlike `KEYCODE_VOLUME_MUTE` (/reference/android/view/KeyEvent.html#KEYCODE_VOLUME_MUTE).

Constant Value: 91 (0x0000005b)

public static final int **KEYCODE_N**

Added in API level 1

Key code constant: 'N' key.

Constant Value: 42 (0x0000002a)

public static final int **KEYCODE_NOTIFICATION**

Added in API level 1

Key code constant: Notification key.

Constant Value: 83 (0x00000053)

public static final int **KEYCODE_NUM**

Added in API level 1

Key code constant: Number modifier key. Used to enter numeric symbols. This key is not Num Lock; it is more like `KEYCODE_ALT_LEFT` (/reference/android/view/KeyEvent.html#KEYCODE_ALT_LEFT) and is interpreted as an ALT key by `MetaKeyKeyListener` (/reference/android/text/method/MetaKeyKeyListener.html).

Constant Value: 78 (0x0000004e)

public static final int **KEYCODE_NUMPAD_0**

Added in API level 11

Key code constant: Numeric keypad '0' key.

Constant Value: 144 (0x00000090)

public static final int **KEYCODE_NUMPAD_1**

Added in API level 11

Key code constant: Numeric keypad '1' key.

Constant Value: 145 (0x00000091)

public static final int **KEYCODE_NUMPAD_2**

Added in API level 11

Key code constant: Numeric keypad '2' key.

Constant Value: 146 (0x00000092)

public static final int **KEYCODE_NUMPAD_3**

Added in API level 11

Key code constant: Numeric keypad '3' key.

Constant Value: 147 (0x00000093)

public static final int **KEYCODE_NUMPAD_4**

Added in API level 11

Key code constant: Numeric keypad '4' key.

Constant Value: 148 (0x00000094)

public static final int **KEYCODE_NUMPAD_5**

Added in API level 11

Key code constant: Numeric keypad '5' key.

Constant Value: 149 (0x00000095)

public static final int **KEYCODE_NUMPAD_6**

Added in API level 11

Key code constant: Numeric keypad '6' key.

Constant Value: 150 (0x00000096)

public static final int **KEYCODE_NUMPAD_7**

Added in API level 11

Key code constant: Numeric keypad '7' key.

Constant Value: 151 (0x00000097)

public static final int **KEYCODE_NUMPAD_8**

Added in API level 11

Key code constant: Numeric keypad '8' key.

Constant Value: 152 (0x00000098)

public static final int **KEYCODE_NUMPAD_9**

Added in API level 11

Key code constant: Numeric keypad '9' key.

Constant Value: 153 (0x00000099)

public static final int **KEYCODE_NUMPAD_ADD**

Added in API level 11

Key code constant: Numeric keypad '+' key (for addition).

Constant Value: 157 (0x0000009d)

public static final int **KEYCODE_NUMPAD_COMMA**

Added in API level 11

Key code constant: Numeric keypad ',' key (for decimals or digit grouping).

Constant Value: 159 (0x0000009f)

public static final int **KEYCODE_NUMPAD_DIVIDE**

Added in API level 11

Key code constant: Numeric keypad '/' key (for division).

Constant Value: 154 (0x0000009a)

public static final int **KEYCODE_NUMPAD_DOT**

Added in API level 11

Key code constant: Numeric keypad '.' key (for decimals or digit grouping).

Constant Value: 158 (0x0000009e)

public static final int **KEYCODE_NUMPAD_ENTER**

Added in API level 11

Key code constant: Numeric keypad Enter key.

Constant Value: 160 (0x000000a0)

public static final int **KEYCODE_NUMPAD_EQUALS**

Added in API level 11

Key code constant: Numeric keypad '=' key.

Constant Value: 161 (0x000000a1)

public static final int **KEYCODE_NUMPAD_LEFT_PAREN**

Added in API level 11

Key code constant: Numeric keypad '(' key.

Constant Value: 162 (0x000000a2)

public static final int **KEYCODE_NUMPAD_MULTIPLY**

Added in API level 11

Key code constant: Numeric keypad '*' key (for multiplication).

Constant Value: 155 (0x0000009b)

public static final int **KEYCODE_NUMPAD_RIGHT_PAREN**

Added in API level 11

Key code constant: Numeric keypad ')' key.

Constant Value: 163 (0x000000a3)

public static final int **KEYCODE_NUMPAD_SUBTRACT**

Added in API level 11

Key code constant: Numeric keypad '-' key (for subtraction).

Constant Value: 156 (0x0000009c)

public static final int **KEYCODE_NUM_LOCK**

Added in API level 11

Key code constant: Num Lock key. This is the Num Lock key; it is different from `KEYCODE_NUM` (/reference/android/view/KeyEvent.html#KEYCODE_NUM). This key alters the behavior of other keys on the numeric keypad.

Constant Value: 143 (0x0000008f)

public static final int **KEYCODE_O**

Added in API level 1

Key code constant: 'O' key.

Constant Value: 43 (0x0000002b)

public static final int **KEYCODE_P**

Added in API level 1

Key code constant: 'P' key.

Constant Value: 44 (0x0000002c)

public static final int **KEYCODE_PAGE_DOWN**

Added in API level 9

Key code constant: Page Down key.

Constant Value: 93 (0x0000005d)

public static final int **KEYCODE_PAGE_UP**

Added in API level 9

Key code constant: Page Up key.

Constant Value: 92 (0x0000005c)

public static final int **KEYCODE_PERIOD**

Added in API level 1

Key code constant: '.' key.

Constant Value: 56 (0x00000038)

public static final int **KEYCODE_PICTSYMBOLS**

Added in API level 9

Key code constant: Picture Symbols modifier key. Used to switch symbol sets (Emoji, Kao-moji).

Constant Value: 94 (0x0000005e)

public static final int **KEYCODE_PLUS**

Added in API level 1

Key code constant: '+' key.

Constant Value: 81 (0x00000051)

public static final int **KEYCODE_POUND**

Added in API level 1

Key code constant: '#' key.

Constant Value: 18 (0x00000012)

public static final int **KEYCODE_POWER**

Added in API level 1

Key code constant: Power key.

Constant Value: 26 (0x0000001a)

public static final int **KEYCODE_PROG_BLUE**

Added in API level 11

Key code constant: Blue "programmable" key. On TV remotes, acts as a contextual/programmable key.

Constant Value: 186 (0x000000ba)

public static final int **KEYCODE_PROG_GREEN**

Added in API level 11

Key code constant: Green "programmable" key. On TV remotes, actsas a contextual/programmable key.

Constant Value: 184 (0x000000b8)

public static final int **KEYCODE_PROG_RED**

Added in API level 11

Key code constant: Red "programmable" key. On TV remotes, acts as a contextual/programmable key.

Constant Value: 183 (0x000000b7)

public static final int **KEYCODE_PROG_YELLOW**

Added in API level 11

Key code constant: Yellow "programmable" key. On TV remotes, acts as a contextual/programmable key.

Constant Value: 185 (0x000000b9)

public static final int **KEYCODE_Q**

Added in API level 1

Key code constant: 'Q' key.

Constant Value: 45 (0x0000002d)

public static final int **KEYCODE_R**

Key code constant: 'R' key.

Constant Value: 46 (0x0000002e)

public static final int **KEYCODE_RIGHT_BRACKET**

Added in API level 1

Key code constant: ']' key.

Constant Value: 72 (0x00000048)

public static final int **KEYCODE_RO**

Added in API level 16

Key code constant: Japanese Ro key.

Constant Value: 217 (0x000000d9)

public static final int **KEYCODE_S**

Added in API level 1

Key code constant: 'S' key.

Constant Value: 47 (0x0000002f)

public static final int **KEYCODE_SCROLL_LOCK**

Added in API level 11

Key code constant: Scroll Lock key.

Constant Value: 116 (0x00000074)

public static final int **KEYCODE_SEARCH**

Added in API level 1

Key code constant: Search key.

Constant Value: 84 (0x00000054)

public static final int **KEYCODE_SEMICOLON**

Added in API level 1

Key code constant: ';' key.

Constant Value: 74 (0x0000004a)

public static final int **KEYCODE_SETTINGS**

Added in API level 11

Key code constant: Settings key. Starts the system settings activity.

Constant Value: 176 (0x000000b0)

public static final int **KEYCODE_SHIFT_LEFT**

Added in API level 1

Key code constant: Left Shift modifier key.

Constant Value: 59 (0x0000003b)

public static final int **KEYCODE_SHIFT_RIGHT**

Added in API level 1

Key code constant: Right Shift modifier key.

Constant Value: 60 (0x0000003c)

public static final int **KEYCODE_SLASH**

Added in API level 1

Key code constant: '/' key.

Constant Value: 76 (0x0000004c)

public static final int **KEYCODE_SOFT_LEFT**

Added in API level 1

Key code constant: Soft Left key. Usually situated below the display on phones and used as a multi-function feature key for selecting a software defined function shown on the bottom left of the display.

Constant Value: 1 (0x00000001)

public static final int **KEYCODE_SOFT_RIGHT**

Added in API level 1

Key code constant: Soft Right key. Usually situated below the display on phones and used as a multi-function feature key for selecting a software defined function shown on the bottom right of the display.

Constant Value: 2 (0x00000002)

public static final int **KEYCODE_SPACE**

Added in API level 1

Key code constant: Space key.

Constant Value: 62 (0x0000003e)

public static final int **KEYCODE_STAR**

Added in API level 1

Key code constant: '*' key.

Constant Value: 17 (0x00000011)

public static final int **KEYCODE_STB_INPUT**

Added in API level 11

Key code constant: Set-top-box input key. On TV remotes, switches the input mode on an external Set-top-box.

Constant Value: 180 (0x000000b4)

public static final int **KEYCODE_STB_POWER**

Added in API level 11

Key code constant: Set-top-box power key. On TV remotes, toggles the power on an external Set-top-box.

Constant Value: 179 (0x000000b3)

public static final int **KEYCODE_SWITCH_CHARSET**

Added in API level 9

Key code constant: Switch Charset modifier key. Used to switch character sets (Kanji, Katakana).

Constant Value: 95 (0x0000005f)

public static final int **KEYCODE_SYM**

Added in API level 1

Key code constant: Symbol modifier key. Used to enter alternate symbols.

Constant Value: 63 (0x0000003f)

public static final int **KEYCODE_SYSRQ**

Added in API level 11

Key code constant: System Request / Print Screen key.

Constant Value: 120 (0x00000078)

public static final int **KEYCODE_T**

Added in API level 1

Key code constant: 'T' key.

Constant Value: 48 (0x00000030)

public static final int **KEYCODE_TAB**

Added in API level 1

Key code constant: Tab key.

Constant Value: 61 (0x0000003d)

public static final int **KEYCODE_TV**

Added in API level 11

Key code constant: TV key. On TV remotes, switches to viewing live TV.

Constant Value: 170 (0x000000aa)

public static final int **KEYCODE_TV_INPUT**

Added in API level 11

Key code constant: TV input key. On TV remotes, switches the input on a television screen.

Constant Value: 178 (0x000000b2)

public static final int **KEYCODE_TV_POWER**

Added in API level 11

Key code constant: TV power key. On TV remotes, toggles the power on a television screen.

Constant Value: 177 (0x000000b1)

public static final int **KEYCODE_U**

Added in API level 1

Key code constant: 'U' key.

Constant Value: 49 (0x00000031)

public static final int **KEYCODE_UNKNOWN**

Added in API level 1

Key code constant: Unknown key code.

Constant Value: 0 (0x00000000)

public static final int **KEYCODE_V**

Added in API level 1

Key code constant: 'V' key.

Constant Value: 50 (0x00000032)

public static final int **KEYCODE_VOLUME_DOWN**

Added in API level 1

Key code constant: Volume Down key. Adjusts the speaker volume down.

Constant Value: 25 (0x00000019)

public static final int **KEYCODE_VOLUME_MUTE**

Added in API level 11

Key code constant: Volume Mute key. Mutes the speaker, unlike `KEYCODE_MUTE` (/reference/android/view/KeyEvent.html#KEYCODE_MUTE). This key should normally be implemented as a toggle such that the first press mutes the speaker and the second press restores the original volume.

Constant Value: 164 (0x000000a4)

public static final int **KEYCODE_VOLUME_UP**

Added in API level 1

Key code constant: Volume Up key. Adjusts the speaker volume up.

Constant Value: 24 (0x00000018)

public static final int **KEYCODE_W**

Added in API level 1

Key code constant: 'W' key.

Constant Value: 51 (0x00000033)

public static final int **KEYCODE_WINDOW**

Added in API level 11

Key code constant: Window key. On TV remotes, toggles picture-in-picture mode or other windowing functions.

Constant Value: 171 (0x000000ab)

public static final int **KEYCODE_X**

Added in API level 1

Key code constant: 'X' key.

Constant Value: 52 (0x00000034)

public static final int **KEYCODE_Y**

Added in API level 1

Key code constant: 'Y' key.

Constant Value: 53 (0x00000035)

public static final int **KEYCODE_YEN**

Added in API level 16

Key code constant: Japanese Yen key.

Constant Value: 216 (0x000000d8)

public static final int **KEYCODE_Z**

Added in API level 1

Key code constant: 'Z' key.

Constant Value: 54 (0x00000036)

public static final int **KEYCODE_ZENKAKU_HANKAKU**

Added in API level 16

Key code constant: Japanese full-width / half-width key.

Constant Value: 211 (0x000000d3)

public static final int **KEYCODE_ZOOM_IN**

Added in API level 11

Key code constant: Zoom in key.

Constant Value: 168 (0x000000a8)

public static final int **KEYCODE_ZOOM_OUT**

Added in API level 11

Key code constant: Zoom out key.

Constant Value: 169 (0x000000a9)

public static final int **MAX_KEYCODE**

Added in API level 1

> **This constant was deprecated in API level 3.**
> There are now more than MAX_KEYCODE keycodes. Use `getMaxKeyCode()` `(/reference/android/view/KeyEvent.html#getMaxKeyCode())` instead.

Constant Value: 84 (0x00000054)

public static final int **META_ALT_LEFT_ON**

Added in API level 1

This mask is used to check whether the left ALT meta key is pressed.

See Also
`isAltPressed()`
`getMetaState()`
`KEYCODE_ALT_LEFT`

Constant Value: 16 (0x00000010)

public static final int **META_ALT_MASK**

Added in API level 11

This mask is a combination of `META_ALT_ON (/reference/android/view/KeyEvent.html#META_ALT_ON)`, `META_ALT_LEFT_ON (/reference/android/view/KeyEvent.html#META_ALT_LEFT_ON)` and `META_ALT_RIGHT_ON (/reference/android/view/KeyEvent.html#META_ALT_RIGHT_ON)`.

Constant Value: 50 (0x00000032)

public static final int **META_ALT_ON**

Added in API level 1

This mask is used to check whether one of the ALT meta keys is pressed.

See Also
`isAltPressed()`
`getMetaState()`
`KEYCODE_ALT_LEFT`
`KEYCODE_ALT_RIGHT`

Constant Value: 2 (0x00000002)

public static final int **META_ALT_RIGHT_ON**

Added in API level 1

This mask is used to check whether the right the ALT meta key is pressed.

See Also

`isAltPressed()`
`getMetaState()`
`KEYCODE_ALT_RIGHT`

Constant Value: 32 (0x00000020)

public static final int **META_CAPS_LOCK_ON**

Added in API level 11

This mask is used to check whether the CAPS LOCK meta key is on.

See Also

`isCapsLockOn()`
`getMetaState()`
`KEYCODE_CAPS_LOCK`

Constant Value: 1048576 (0x00100000)

public static final int **META_CTRL_LEFT_ON**

Added in API level 11

This mask is used to check whether the left CTRL meta key is pressed.

See Also

`isCtrlPressed()`
`getMetaState()`
`KEYCODE_CTRL_LEFT`

Constant Value: 8192 (0x00002000)

public static final int **META_CTRL_MASK**

Added in API level 11

This mask is a combination of `META_CTRL_ON` (/reference/android/view/KeyEvent.html#META_CTRL_ON), `META_CTRL_LEFT_ON` (/reference/android/view/KeyEvent.html#META_CTRL_LEFT_ON) and `META_CTRL_RIGHT_ON` (/reference/android/view/KeyEvent.html#META_CTRL_RIGHT_ON).

Constant Value: 28672 (0x00007000)

public static final int **META_CTRL_ON**

Added in API level 11

This mask is used to check whether one of the CTRL meta keys is pressed.

See Also

`isCtrlPressed()`
`getMetaState()`
`KEYCODE_CTRL_LEFT`
`KEYCODE_CTRL_RIGHT`

Constant Value: 4096 (0x00001000)

public static final int **META_CTRL_RIGHT_ON**

Added in API level 11

This mask is used to check whether the right CTRL meta key is pressed.

See Also

`isCtrlPressed()`
`getMetaState()`
`KEYCODE_CTRL_RIGHT`

Constant Value: 16384 (0x00004000)

public static final int **META_FUNCTION_ON**

Added in API level 11

This mask is used to check whether the FUNCTION meta key is pressed.

See Also

`isFunctionPressed()`
`getMetaState()`

Constant Value: 8 (0x00000008)

public static final int **META_META_LEFT_ON**

Added in API level 11

This mask is used to check whether the left META meta key is pressed.

See Also

`isMetaPressed()`
`getMetaState()`
`KEYCODE_META_LEFT`

Constant Value: 131072 (0x00020000)

public static final int **META_META_MASK**

Added in API level 11

This mask is a combination of `META_META_ON` (/reference/android/view/KeyEvent.html#META_META_ON), `META_META_LEFT_ON` (/reference/android/view/KeyEvent.html#META_META_LEFT_ON) and `META_META_RIGHT_ON` (/reference/android/view/KeyEvent.html#META_META_RIGHT_ON).

Constant Value: 458752 (0x00070000)

public static final int **META_META_ON**

Added in API level 11

This mask is used to check whether one of the META meta keys is pressed.

See Also

`isMetaPressed()`
`getMetaState()`
`KEYCODE_META_LEFT`
`KEYCODE_META_RIGHT`

Constant Value: 65536 (0x00010000)

public static final int **META_META_RIGHT_ON**

Added in API level 11

This mask is used to check whether the right META meta key is pressed.

See Also

`isMetaPressed()`
`getMetaState()`
`KEYCODE_META_RIGHT`

Constant Value: 262144 (0x00040000)

public static final int **META_NUM_LOCK_ON**

Added in API level 11

This mask is used to check whether the NUM LOCK meta key is on.

See Also

`isNumLockOn()`
`getMetaState()`
`KEYCODE_NUM_LOCK`

Constant Value: 2097152 (0x00200000)

public static final int **META_SCROLL_LOCK_ON**

Added in API level 11

This mask is used to check whether the SCROLL LOCK meta key is on.

See Also

`isScrollLockOn()`
`getMetaState()`
`KEYCODE_SCROLL_LOCK`

Constant Value: 4194304 (0x00400000)

public static final int **META_SHIFT_LEFT_ON**

Added in API level 1

This mask is used to check whether the left SHIFT meta key is pressed.

See Also

`isShiftPressed()`
`getMetaState()`
`KEYCODE_SHIFT_LEFT`

Constant Value: 64 (0x00000040)

public static final int **META_SHIFT_MASK**

Added in API level 11

This mask is a combination of `META_SHIFT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_ON), `META_SHIFT_LEFT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_LEFT_ON) and `META_SHIFT_RIGHT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_RIGHT_ON).

Constant Value: 193 (0x000000c1)

public static final int **META_SHIFT_ON**

Added in API level 1

This mask is used to check whether one of the SHIFT meta keys is pressed.

See Also

`isShiftPressed()`
`getMetaState()`
`KEYCODE_SHIFT_LEFT`
`KEYCODE_SHIFT_RIGHT`

Constant Value: 1 (0x00000001)

public static final int **META_SHIFT_RIGHT_ON**

Added in API level 1

This mask is used to check whether the right SHIFT meta key is pressed.

See Also

`isShiftPressed()`
`getMetaState()`
`KEYCODE_SHIFT_RIGHT`

Constant Value: 128 (0x00000080)

public static final int **META_SYM_ON**

Added in API level 1

This mask is used to check whether the SYM meta key is pressed.

See Also

`isSymPressed()`
`getMetaState()`

Constant Value: 4 (0x00000004)

Fields

public static final Creator<KeyEvent> **CREATOR**

Added in API level 1

Public Constructors

public **KeyEvent** (int action, int code)

Added in API level 1

Create a new key event.

Parameters

action Action code: either `ACTION_DOWN`, `ACTION_UP`, or `ACTION_MULTIPLE`.

code The key code.

public **KeyEvent** (long downTime, long eventTime, int action, int code, int repeat)

Added in API level 1

Create a new key event.

Parameters

downTime The time (in `uptimeMillis()`) at which this key code originally went down.

eventTime The time (in `uptimeMillis()`) at which this event happened.

action Action code: either `ACTION_DOWN`, `ACTION_UP`, or `ACTION_MULTIPLE`.

code The key code.

repeat A repeat count for down events (> 0 if this is after the initial down) or event count for multiple events.

public **KeyEvent** (long downTime, long eventTime, int action, int code, int repeat, int metaState)

Added in API level 1

Create a new key event.

Parameters

downTime	The time (in `uptimeMillis()`) at which this key code originally went down.
eventTime	The time (in `uptimeMillis()`) at which this event happened.
action	Action code: either `ACTION_DOWN`, `ACTION_UP`, or `ACTION_MULTIPLE`.
code	The key code.
repeat	A repeat count for down events (> 0 if this is after the initial down) or event count for multiple events.
metaState	Flags indicating which meta keys are currently pressed.

public **KeyEvent** (long downTime, long eventTime, int action, int code, int repeat, int metaState, int deviceId, int scancode)

Added in API level 1

Create a new key event.

Parameters

downTime	The time (in `uptimeMillis()`) at which this key code originally went down.
eventTime	The time (in `uptimeMillis()`) at which this event happened.
action	Action code: either `ACTION_DOWN`, `ACTION_UP`, or `ACTION_MULTIPLE`.
code	The key code.
repeat	A repeat count for down events (> 0 if this is after the initial down) or event count for multiple events.
metaState	Flags indicating which meta keys are currently pressed.
deviceId	The device ID that generated the key event.
scancode	Raw device scan code of the event.

public **KeyEvent** (long downTime, long eventTime, int action, int code, int repeat, int metaState, int deviceId, int scancode, int flags)

Added in API level 1

Create a new key event.

Parameters

downTime	The time (in `uptimeMillis()`) at which this key code originally went down.
eventTime	The time (in `uptimeMillis()`) at which this event happened.
action	Action code: either `ACTION_DOWN`, `ACTION_UP`, or `ACTION_MULTIPLE`.
code	The key code.
repeat	A repeat count for down events (> 0 if this is after the initial down) or event count for multiple events.

metaState	Flags indicating which meta keys are currently pressed.
deviceId	The device ID that generated the key event.
scancode	Raw device scan code of the event.
flags	The flags for this key event

public **KeyEvent** (long downTime, long eventTime, int action, int code, int repeat, int metaState, int deviceId, int scancode, int flags, int source)

Added in API level 9

Create a new key event.

Parameters

downTime	The time (in `uptimeMillis()`) at which this key code originally went down.
eventTime	The time (in `uptimeMillis()`) at which this event happened.
action	Action code: either `ACTION_DOWN`, `ACTION_UP`, or `ACTION_MULTIPLE`.
code	The key code.
repeat	A repeat count for down events (> 0 if this is after the initial down) or event count for multiple events.
metaState	Flags indicating which meta keys are currently pressed.
deviceId	The device ID that generated the key event.
scancode	Raw device scan code of the event.
flags	The flags for this key event
source	The input source such as `SOURCE_KEYBOARD`.

public **KeyEvent** (long time, String characters, int deviceId, int flags)

Added in API level 3

Create a new key event for a string of characters. The key code, action, repeat count and source will automatically be set to `KEYCODE_UNKNOWN` (/reference/android/view/KeyEvent.html#KEYCODE_UNKNOWN), `ACTION_MULTIPLE` (/reference/android/view/KeyEvent.html#ACTION_MULTIPLE), 0, and `SOURCE_KEYBOARD` (/reference/android/view/InputDevice.html#SOURCE_KEYBOARD) for you.

Parameters

time	The time (in `uptimeMillis()`) at which this event occured.
characters	The string of characters.
deviceId	The device ID that generated the key event.
flags	The flags for this key event

public **KeyEvent** (KeyEvent origEvent)

Added in API level 3

Make an exact copy of an existing key event.

public **KeyEvent** (KeyEvent origEvent, long eventTime, int newRepeat)

Added in API level 1

> **This constructor was deprecated in API level 5.**

> Use `changeTimeRepeat(KeyEvent, long, int)` (/reference/android/view/KeyEvent.html#changeTimeRepeat(android.view.KeyEvent, long, int)) instead.

Copy an existing key event, modifying its time and repeat count.

Parameters

origEvent	The existing event to be copied.
eventTime	The new event time (in `uptimeMillis()`) of the event.
newRepeat	The new repeat count of the event.

Public Methods

public static KeyEvent **changeAction** (KeyEvent event, int action) Added in API level 3

Create a new key event that is the same as the given one, but whose action is replaced with the given value.

Parameters

event	The existing event to be copied. This is not modified.
action	The new action code of the event.

public static KeyEvent **changeFlags** (KeyEvent event, int flags) Added in API level 3

Create a new key event that is the same as the given one, but whose flags are replaced with the given value.

Parameters

event	The existing event to be copied. This is not modified.
flags	The new flags constant.

public static KeyEvent **changeTimeRepeat** (KeyEvent event, long eventTime, int newRepeat) Added in API level 3

Create a new key event that is the same as the given one, but whose event time and repeat count are replaced with the given value.

Parameters

event	The existing event to be copied. This is not modified.
eventTime	The new event time (in `uptimeMillis()`) of the event.
newRepeat	The new repeat count of the event.

public static KeyEvent **changeTimeRepeat** (KeyEvent event, long eventTime, int newRepeat, int newFlags) Added in API level 5

Create a new key event that is the same as the given one, but whose event time and repeat count are replaced with the given value.

Parameters

event	The existing event to be copied. This is not modified.
eventTime	The new event time (in `uptimeMillis()`) of the event.

newRepeat	The new repeat count of the event.
newFlags	New flags for the event, replacing the entire value in the original event.

public final boolean **dispatch** (KeyEvent.Callback receiver, KeyEvent.DispatcherState state, Object target)

Added in API level 5

Deliver this key event to a `KeyEvent.Callback` (/reference/android/view/KeyEvent.Callback.html) interface. If this is an ACTION_MULTIPLE event and it is not handled, then an attempt will be made to deliver a single normal event.

Parameters

receiver	The Callback that will be given the event.
state	State information retained across events.
target	The target of the dispatch, for use in tracking.

Returns

The return value from the Callback method that was called.

public final boolean **dispatch** (KeyEvent.Callback receiver)

Added in API level 1

This method was deprecated in API level 5.
Use `dispatch(Callback, DispatcherState, Object)` (/reference/android/view/KeyEvent.html#dispatch(android.view.KeyEvent.Callback, android.view.KeyEvent.DispatcherState, java.lang.Object)) instead.

public final int **getAction** ()

Added in API level 1

Retrieve the action of this key event. May be either `ACTION_DOWN` (/reference/android/view/KeyEvent.html#ACTION_DOWN), `ACTION_UP` (/reference/android/view/KeyEvent.html#ACTION_UP), or `ACTION_MULTIPLE` (/reference/android/view/KeyEvent.html#ACTION_MULTIPLE).

Returns

The event action: ACTION_DOWN, ACTION_UP, or ACTION_MULTIPLE.

public final String **getCharacters** ()

Added in API level 3

For the special case of a `ACTION_MULTIPLE` (/reference/android/view/KeyEvent.html#ACTION_MULTIPLE) event with key code of `KEYCODE_UNKNOWN` (/reference/android/view/KeyEvent.html#KEYCODE_UNKNOWN), this is a raw string of characters associated with the event. In all other cases it is null.

Returns

Returns a String of 1 or more characters associated with the event.

public static int **getDeadChar** (int accent, int c)

Added in API level 1

Get the character that is produced by putting accent on the character c. For example, getDeadChar('`', 'e') returns è.

public final int **getDeviceId** ()

Added in API level 1

Gets the id for the device that this event came from. An id of zero indicates that the event didn't come from a physical device and maps to the default keymap. The other numbers are arbitrary and you shouldn't depend on the values.

Returns

The device id.

public char **getDisplayLabel** () Added in API level 1

Gets the primary character for this key. In other words, the label that is physically printed on it.

Returns

The display label character, or 0 if none (eg. for non-printing keys).

public final long **getDownTime** () Added in API level 1

Retrieve the time of the most recent key down event, in the `uptimeMillis()` (/reference/android/os/SystemClock.html#uptimeMillis()) time base. If this is a down event, this will be the same as `getEventTime()` (/reference/android/view/KeyEvent.html#getEventTime()). Note that when chording keys, this value is the down time of the most recently pressed key, which may *not* be the same physical key of this event.

Returns

Returns the most recent key down time, in the `uptimeMillis()` time base

public final long **getEventTime** () Added in API level 1

Retrieve the time this event occurred, in the `uptimeMillis()` (/reference/android/os/SystemClock.html#uptimeMillis()) time base.

Returns

Returns the time this event occurred, in the `uptimeMillis()` time base.

public final int **getFlags** () Added in API level 1

Returns the flags for this key event.

See Also

`FLAG_WOKE_HERE`

public final KeyCharacterMap **getKeyCharacterMap** () Added in API level 11

Gets the `KeyCharacterMap` (/reference/android/view/KeyCharacterMap.html) associated with the keyboard device.

Returns

The associated key character map.

Throws

KeyCharacterMap.UnavailableException} if the key character map could not be loaded because it was malformed or the default key character map is missing from the system.

See Also

`load(int)`

public final int **getKeyCode** () Added in API level 1

Retrieve the key code of the key event. This is the physical key that was pressed, *not* the Unicode character.

Returns

The key code of the event.

public boolean **getKeyData** (KeyCharacterMap.KeyData results) Added in API level 1

> **This method was deprecated in API level 11.**
> instead use `getDisplayLabel()` (/reference/android/view/KeyEvent.html#getDisplayLabel()), `getNumber()` (/reference/android/view/KeyEvent.html#getNumber()) or `getUnicodeChar(int)` (/reference/android/view/KeyEvent.html#getUnicodeChar(int)).

Get the character conversion data for a given key code.

Parameters

results	A `KeyCharacterMap.KeyData` instance that will be filled with the results.

Returns

True if the key was mapped. If the key was not mapped, results is not modified.

public char **getMatch** (char[] chars) Added in API level 1

Gets the first character in the character array that can be generated by the specified key code.

This is a convenience function that returns the same value as `getMatch(chars, 0)` (/reference/android/view/KeyEvent.html#getMatch(char[], int)).

Parameters

chars	The array of matching characters to consider.

Returns

The matching associated character, or 0 if none.

public char **getMatch** (char[] chars, int metaState) Added in API level 1

Gets the first character in the character array that can be generated by the specified key code. If there are multiple choices, prefers the one that would be generated with the specified meta key modifier state.

Parameters

chars	The array of matching characters to consider.
metaState	The preferred meta key modifier state.

Returns

The matching associated character, or 0 if none.

public static int **getMaxKeyCode** () Added in API level 3

Returns the maximum keycode.

public final int **getMetaState** ()

Added in API level 1

Returns the state of the meta keys.

Returns

an integer in which each bit set to 1 represents a pressed meta key

See Also

isAltPressed()
isShiftPressed()
isSymPressed()
isCtrlPressed()
isMetaPressed()
isFunctionPressed()
isCapsLockOn()
isNumLockOn()
isScrollLockOn()
META_ALT_ON
META_ALT_LEFT_ON
META_ALT_RIGHT_ON
META_SHIFT_ON
META_SHIFT_LEFT_ON
META_SHIFT_RIGHT_ON
META_SYM_ON
META_FUNCTION_ON
META_CTRL_ON
META_CTRL_LEFT_ON
META_CTRL_RIGHT_ON
META_META_ON
META_META_LEFT_ON
META_META_RIGHT_ON
META_CAPS_LOCK_ON
META_NUM_LOCK_ON
META_SCROLL_LOCK_ON
getModifiers()

public static int **getModifierMetaStateMask** ()

Added in API level 11

Gets a mask that includes all valid modifier key meta state bits.

For the purposes of this function, KEYCODE_CAPS_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), KEYCODE_SCROLL_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and KEYCODE_NUM_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, the mask specifically excludes META_CAPS_LOCK_ON (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON), META_SCROLL_LOCK_ON (/reference/android/view/KeyEvent.html#META_SCROLL_LOCK_ON) and META_NUM_LOCK_ON (/reference/android/view/KeyEvent.html#META_NUM_LOCK_ON).

Returns

The modifier meta state mask which is a combination of META_SHIFT_ON, `META_SHIFT_LEFT_ON`, `META_SHIFT_RIGHT_ON`, `META_ALT_ON`, `META_ALT_LEFT_ON`, `META_ALT_RIGHT_ON`, `META_CTRL_ON`, `META_CTRL_LEFT_ON`, `META_CTRL_RIGHT_ON`, `META_META_ON`, `META_META_LEFT_ON`, `META_META_RIGHT_ON`, `META_SYM_ON`, `META_FUNCTION_ON`.

public final int **getModifiers** ()

Added in API level 13

Returns the state of the modifier keys.

For the purposes of this function, `KEYCODE_CAPS_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), `KEYCODE_SCROLL_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and `KEYCODE_NUM_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, this function specifically masks out `META_CAPS_LOCK_ON` (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON), `META_SCROLL_LOCK_ON` (/reference/android/view/KeyEvent.html#META_SCROLL_LOCK_ON) and `META_NUM_LOCK_ON` (/reference/android/view/KeyEvent.html#META_NUM_LOCK_ON).

The value returned consists of the meta state (from `getMetaState()` (/reference/android/view/KeyEvent.html#getMetaState())) normalized using `normalizeMetaState(int)` (/reference/android/view/KeyEvent.html#normalizeMetaState(int)) and then masked with `getModifierMetaStateMask()` (/reference/android/view/KeyEvent.html#getModifierMetaStateMask()) so that only valid modifier bits are retained.

Returns
An integer in which each bit set to 1 represents a pressed modifier key.

See Also
`getMetaState()`

public char **getNumber** ()

Added in API level 1

Gets the number or symbol associated with the key.

The character value is returned, not the numeric value. If the key is not a number, but is a symbol, the symbol is retuned.

This method is intended to to support dial pads and other numeric or symbolic entry on keyboards where certain keys serve dual function as alphabetic and symbolic keys. This method returns the number or symbol associated with the key independent of whether the user has pressed the required modifier.

For example, on one particular keyboard the keys on the top QWERTY row generate numbers when ALT is pressed such that ALT-Q maps to '1'. So for that keyboard when `getNumber()` (/reference/android/view/KeyEvent.html#getNumber()) is called with `KEYCODE_Q` (/reference/android/view/KeyEvent.html#KEYCODE_Q) it returns '1' so that the user can type numbers without pressing ALT when it makes sense.

Returns
The associated numeric or symbolic character, or 0 if none.

public final int **getRepeatCount** ()

Added in API level 1

Retrieve the repeat count of the event. For both key up and key down events, this is the number of times the key has repeated with the first down starting at 0 and counting up from there. For multiple key events, this is the number of down/up pairs that have occurred.

Returns

The number of times the key has repeated.

public final int **getScanCode** ()

Added in API level 1

Retrieve the hardware key id of this key event. These values are not reliable and vary from device to device.

Mostly this is here for debugging purposes.

public final int **getSource** ()

Added in API level 9

Gets the source of the event.

Returns

The event source or `SOURCE_UNKNOWN` if unknown.

public int **getUnicodeChar** ()

Added in API level 1

Gets the Unicode character generated by the specified key and meta key state combination.

Returns the Unicode character that the specified key would produce when the specified meta bits (see `MetaKeyKeyListener` (/reference/android/text/method/MetaKeyKeyListener.html)) were active.

Returns 0 if the key is not one that is used to type Unicode characters.

If the return value has bit `COMBINING_ACCENT` (/reference/android/view/KeyCharacterMap.html#COMBINING_ACCENT) set, the key is a "dead key" that should be combined with another to actually produce a character -- see `getDeadChar(int, int)` (/reference/android/view/KeyCharacterMap.html#getDeadChar(int, int)) -- after masking with `COMBINING_ACCENT_MASK` (/reference/android/view/KeyCharacterMap.html#COMBINING_ACCENT_MASK).

Returns

The associated character or combining accent, or 0 if none.

public int **getUnicodeChar** (int metaState)

Added in API level 1

Gets the Unicode character generated by the specified key and meta key state combination.

Returns the Unicode character that the specified key would produce when the specified meta bits (see `MetaKeyKeyListener` (/reference/android/text/method/MetaKeyKeyListener.html)) were active.

Returns 0 if the key is not one that is used to type Unicode characters.

If the return value has bit `COMBINING_ACCENT` (/reference/android/view/KeyCharacterMap.html#COMBINING_ACCENT) set, the key is a "dead key" that should be combined with another to actually produce a character -- see `getDeadChar(int, int)` (/reference/android/view/KeyCharacterMap.html#getDeadChar(int, int)) -- after masking with `COMBINING_ACCENT_MASK` (/reference/android/view/KeyCharacterMap.html#COMBINING_ACCENT_MASK).

Parameters

metaState	The meta key modifier state.

Returns

The associated character or combining accent, or 0 if none.

public final boolean **hasModifiers** (int modifiers)

Added in API level 11

Returns true if only the specified modifiers keys are pressed. Returns false if a different combination of modifier keys are pressed.

For the purposes of this function, `KEYCODE_CAPS_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), `KEYCODE_SCROLL_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and `KEYCODE_NUM_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, this function ignores `META_CAPS_LOCK_ON` (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON), `META_SCROLL_LOCK_ON` (/reference/android/view/KeyEvent.html#META_SCROLL_LOCK_ON) and `META_NUM_LOCK_ON` (/reference/android/view/KeyEvent.html#META_NUM_LOCK_ON).

If the specified modifier mask includes directional modifiers, such as `META_SHIFT_LEFT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_LEFT_ON), then this method ensures that the modifier is pressed on that side. If the specified modifier mask includes non-directional modifiers, such as `META_SHIFT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_ON), then this method ensures that the modifier is pressed on either side. If the specified modifier mask includes both directional and non-directional modifiers for the same type of key, such as `META_SHIFT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_ON) and `META_SHIFT_LEFT_ON` (/reference/android/view/KeyEvent.html#META_SHIFT_LEFT_ON), then this method throws an illegal argument exception.

Parameters

modifiers	The meta state of the modifier keys to check. May be a combination of modifier meta states as defined by `getModifierMetaStateMask()`. May be 0 to ensure that no modifier keys are pressed.

Returns

True if only the specified modifier keys are pressed.

Throws

IllegalArgumentException	if the modifiers parameter contains invalid modifiers

See Also

`metaStateHasModifiers(int, int)`

public final boolean **hasNoModifiers** ()

Added in API level 11

Returns true if no modifier keys are pressed.

For the purposes of this function, `KEYCODE_CAPS_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), `KEYCODE_SCROLL_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and `KEYCODE_NUM_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, this function ignores `META_CAPS_LOCK_ON` (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON), `META_SCROLL_LOCK_ON` (/reference/android/view/KeyEvent.html#META_SCROLL_LOCK_ON) and `META_NUM_LOCK_ON` (/reference/android/view/KeyEvent.html#META_NUM_LOCK_ON).

The meta state is normalized prior to comparison using `normalizeMetaState(int)` (/reference/android/view/KeyEvent.html#normalizeMetaState(int)).

Returns

True if no modifier keys are pressed.

See Also

`metaStateHasNoModifiers(int)`

public final boolean **isAltPressed** ()

Added in API level 1

Returns the pressed state of the ALT meta key.

Returns

true if the ALT key is pressed, false otherwise

See Also

`KEYCODE_ALT_LEFT`
`KEYCODE_ALT_RIGHT`
`META_ALT_ON`

public final boolean **isCanceled** ()

Added in API level 5

For `ACTION_UP` (/reference/android/view/KeyEvent.html#ACTION_UP) events, indicates that the event has been canceled as per `FLAG_CANCELED` (/reference/android/view/KeyEvent.html#FLAG_CANCELED).

public final boolean **isCapsLockOn** ()

Added in API level 11

Returns the locked state of the CAPS LOCK meta key.

Returns

true if the CAPS LOCK key is on, false otherwise

See Also

`KEYCODE_CAPS_LOCK`
`META_CAPS_LOCK_ON`

public final boolean **isCtrlPressed** ()

Added in API level 11

Returns the pressed state of the CTRL meta key.

Returns

true if the CTRL key is pressed, false otherwise

See Also

KEYCODE_CTRL_LEFT
KEYCODE_CTRL_RIGHT
META_CTRL_ON

public final boolean **isFunctionPressed** ()

Added in API level 11

Returns the pressed state of the FUNCTION meta key.

Returns

true if the FUNCTION key is pressed, false otherwise

See Also

KEYCODE_FUNCTION
META_FUNCTION_ON

public static final boolean **isGamepadButton** (int keyCode)

Added in API level 12

Returns true if the specified keycode is a gamepad button.

Returns

True if the keycode is a gamepad button, such as KEYCODE_BUTTON_A.

public final boolean **isLongPress** ()

Added in API level 5

For ACTION_DOWN (/reference/android/view/KeyEvent.html#ACTION_DOWN) events, indicates that the event has been canceled as per FLAG_LONG_PRESS (/reference/android/view/KeyEvent.html#FLAG_LONG_PRESS).

public final boolean **isMetaPressed** ()

Added in API level 11

Returns the pressed state of the META meta key.

Returns

true if the META key is pressed, false otherwise

See Also

KEYCODE_META_LEFT
KEYCODE_META_RIGHT
META_META_ON

public static boolean **isModifierKey** (int keyCode)

Added in API level 1

Returns true if this key code is a modifier key.

For the purposes of this function, KEYCODE_CAPS_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), KEYCODE_SCROLL_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and KEYCODE_NUM_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, this function return false for those keys.

Returns

True if the key code is one of KEYCODE_SHIFT_LEFT KEYCODE_SHIFT_RIGHT, `KEYCODE_ALT_LEFT`, `KEYCODE_ALT_RIGHT`, `KEYCODE_CTRL_LEFT`, `KEYCODE_CTRL_RIGHT`, `KEYCODE_META_LEFT`, or `KEYCODE_META_RIGHT`, `KEYCODE_SYM`, `KEYCODE_NUM`, `KEYCODE_FUNCTION`.

public final boolean **isNumLockOn** ()

Added in API level 11

Returns the locked state of the NUM LOCK meta key.

Returns

true if the NUM LOCK key is on, false otherwise

See Also

`KEYCODE_NUM_LOCK`
`META_NUM_LOCK_ON`

public boolean **isPrintingKey** ()

Added in API level 1

Returns true if this key produces a glyph.

Returns

True if the key is a printing key.

public final boolean **isScrollLockOn** ()

Added in API level 11

Returns the locked state of the SCROLL LOCK meta key.

Returns

true if the SCROLL LOCK key is on, false otherwise

See Also

`KEYCODE_SCROLL_LOCK`
`META_SCROLL_LOCK_ON`

public final boolean **isShiftPressed** ()

Added in API level 1

Returns the pressed state of the SHIFT meta key.

Returns

true if the SHIFT key is pressed, false otherwise

See Also

`KEYCODE_SHIFT_LEFT`
`KEYCODE_SHIFT_RIGHT`
`META_SHIFT_ON`

public final boolean **isSymPressed** ()

Added in API level 1

Returns the pressed state of the SYM meta key.

Returns

true if the SYM key is pressed, false otherwise

See Also

`KEYCODE_SYM`
`META_SYM_ON`

public final boolean **isSystem** ()

Added in API level 1

Is this a system key? System keys can not be used for menu shortcuts. TODO: this information should come from a table somewhere. TODO: should the dpad keys be here? arguably, because they also shouldn't be menu shortcuts

public final boolean **isTracking** ()

Added in API level 5

For `ACTION_UP` (/reference/android/view/KeyEvent.html#ACTION_UP) events, indicates that the event is still being tracked from its initial down event as per `FLAG_TRACKING` (/reference/android/view/KeyEvent.html#FLAG_TRACKING).

public static int **keyCodeFromString** (String symbolicName)

Added in API level 12

Gets a keycode by its symbolic name such as "KEYCODE_A" or an equivalent numeric constant such as "1001".

Parameters

symbolicName The symbolic name of the keycode.

Returns

The keycode or `KEYCODE_UNKNOWN` if not found.

See Also

`ERROR(/#keycodeToString(int))`

public static String **keyCodeToString** (int keyCode)

Added in API level 12

Returns a string that represents the symbolic name of the specified keycode such as "KEYCODE_A", "KEYCODE_DPAD_UP", or an equivalent numeric constant such as "1001" if unknown.

Parameters

keyCode The key code.

Returns

The symbolic name of the specified keycode.

See Also

`getDisplayLabel(int)`

public static boolean **metaStateHasModifiers** (int metaState, int modifiers)

Added in API level 11

Returns true if only the specified modifier keys are pressed according to the specified meta state. Returns false if a different combination of modifier keys are pressed.

For the purposes of this function, `KEYCODE_CAPS_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), `KEYCODE_SCROLL_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and `KEYCODE_NUM_LOCK` (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, this function ignores `META_CAPS_LOCK_ON` (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON), `META_SCROLL_LOCK_ON` (/reference/android/view/KeyEvent.html#META_SCROLL_LOCK_ON) and `META_NUM_LOCK_ON` (/reference/android/view/KeyEvent.html#META_NUM_LOCK_ON).

If the specified modifier mask includes directional modifiers, such as META_SHIFT_LEFT_ON (/reference/android/view/KeyEvent.html#META_SHIFT_LEFT_ON), then this method ensures that the modifier is pressed on that side. If the specified modifier mask includes non-directional modifiers, such as META_SHIFT_ON (/reference/android/view/KeyEvent.html#META_SHIFT_ON), then this method ensures that the modifier is pressed on either side. If the specified modifier mask includes both directional and non-directional modifiers for the same type of key, such as META_SHIFT_ON (/reference/android/view/KeyEvent.html#META_SHIFT_ON) and META_SHIFT_LEFT_ON (/reference/android/view/KeyEvent.html#META_SHIFT_LEFT_ON), then this method throws an illegal argument exception.

Parameters

metaState	The meta state to consider.
modifiers	The meta state of the modifier keys to check. May be a combination of modifier meta states as defined by getModifierMetaStateMask(). May be 0 to ensure that no modifier keys are pressed.

Returns

True if only the specified modifier keys are pressed.

Throws

IllegalArgumentException	if the modifiers parameter contains invalid modifiers

See Also

hasModifiers(int)

public static boolean **metaStateHasNoModifiers** (int metaState)

Added in API level 11

Returns true if no modifiers keys are pressed according to the specified meta state.

For the purposes of this function, KEYCODE_CAPS_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_CAPS_LOCK), KEYCODE_SCROLL_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_SCROLL_LOCK), and KEYCODE_NUM_LOCK (/reference/android/view/KeyEvent.html#KEYCODE_NUM_LOCK) are not considered modifier keys. Consequently, this function ignores META_CAPS_LOCK_ON (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON), META_SCROLL_LOCK_ON (/reference/android/view/KeyEvent.html#META_SCROLL_LOCK_ON) and META_NUM_LOCK_ON (/reference/android/view/KeyEvent.html#META_NUM_LOCK_ON).

The meta state is normalized prior to comparison using normalizeMetaState(int) (/reference/android/view/KeyEvent.html#normalizeMetaState(int)).

Parameters

metaState	The meta state to consider.

Returns

True if no modifier keys are pressed.

See Also

hasNoModifiers()

public static int **normalizeMetaState** (int metaState)

Added in API level 11

Normalizes the specified meta state.

The meta state is normalized such that if either the left or right modifier meta state bits are set then the result will also include the universal bit for that modifier.

If the specified meta state contains `META_ALT_LEFT_ON` (/reference/android/view/KeyEvent.html#META_ALT_LEFT_ON) then the result will also contain `META_ALT_ON` (/reference/android/view/KeyEvent.html#META_ALT_ON) in addition to `META_ALT_LEFT_ON` (/reference/android/view/KeyEvent.html#META_ALT_LEFT_ON) and the other bits that were specified in the input. The same is process is performed for shift, control and meta.

If the specified meta state contains synthetic meta states defined by `MetaKeyKeyListener` (/reference/android/text/method/MetaKeyKeyListener.html), then those states are translated here and the original synthetic meta states are removed from the result. `META_CAP_LOCKED` (/reference/android/text/method/MetaKeyKeyListener.html#META_CAP_LOCKED) is translated to `META_CAPS_LOCK_ON` (/reference/android/view/KeyEvent.html#META_CAPS_LOCK_ON). `META_ALT_LOCKED` (/reference/android/text/method/MetaKeyKeyListener.html#META_ALT_LOCKED) is translated to `META_ALT_ON` (/reference/android/view/KeyEvent.html#META_ALT_ON). `META_SYM_LOCKED` (/reference/android/text/method/MetaKeyKeyListener.html#META_SYM_LOCKED) is translated to `META_SYM_ON` (/reference/android/view/KeyEvent.html#META_SYM_ON).

Undefined meta state bits are removed.

Parameters

metaState	The meta state.

Returns

The normalized meta state.

public final void **setSource** (int source)

Added in API level 12

Modifies the source of the event.

Parameters

source	The new source.

public final void **startTracking** ()

Added in API level 5

Call this during `onKeyDown(int, KeyEvent)` (/reference/android/view/KeyEvent.Callback.html#onKeyDown(int, android.view.KeyEvent)) to have the system track the key through its final up (possibly including a long press). Note that only one key can be tracked at a time -- if another key down event is received while a previous one is being tracked, tracking is stopped on the previous event.

public String **toString** ()

Added in API level 1

Returns a string containing a concise, human-readable description of this object. Subclasses are encouraged to override this method and provide an implementation that takes into account the object's type and data. The default implementation is equivalent to the following expression:

```
getClass().getName() + '@' + Integer.toHexString(hashCode(
```

See Writing a useful `toString` method (/reference/java/lang/Object.html#writing_toString) if you intend implementing your own `toString` method.

Returns

a printable representation of this object.

public void **writeToParcel** (Parcel out, int flags)

Added in API level 1

Flatten this object in to a Parcel.

Parameters

out	The Parcel in which the object should be written.
flags	Additional flags about how the object should be written. May be 0 or `PARCELABLE_WRITE_RETURN_VALUE`.