public class

Added in API level 1

AutoCompleteTextView

extends EditText

implements Filter.FilterListener

java.lang.Object
  ↳android.view.View
    ↳android.widget.TextView
      ↳android.widget.EditText
        ↳android.widget.AutoCompleteTextView

▶ Known Direct Subclasses
MultiAutoCompleteTextView

Class Overview

An editable text view that shows completion suggestions automatically while the user is typing. The list of suggestions is displayed in a drop down menu from which the user can choose an item to replace the content of the edit box with.

The drop down can be dismissed at any time by pressing the back key or, if no item is selected in the drop down, by pressing the enter/dpad center key.

The list of suggestions is obtained from a data adapter and appears only after a given number of characters defined by `the threshold (/reference/android/widget/AutoCompleteTextView.html#getThreshold())`.

The following code snippet shows how to create a text view which suggests various countries names while the user is typing:

```
public class CountriesActivity extends Activity {
    protected void onCreate(Bundle icicle) {
        super.onCreate(icicle);
        setContentView(R.layout.countries);

        ArrayAdapter<String> adapter = new ArrayAdapter<String>(this,
                android.R.layout.simple_dropdown_item_1line, COUNTRIES);
        AutoCompleteTextView textView = (AutoCompleteTextView)
                findViewById(R.id.countries_list);
        textView.setAdapter(adapter);
    }

    private static final String[] COUNTRIES = new String[] {
        "Belgium", "France", "Italy", "Germany", "Spain"
    };
}
```

See the Text Fields (/guide/topics/ui/controls/text.html) guide.

Summary

Nested Classes

interface AutoCompleteTextView.OnDismissListener	Listener to respond to the AutoCompleteTextView's completion list being dismissed.
interface AutoCompleteTextView.Validator	This interface is used to make sure that the text entered in this TextView complies to a certain format.

XML Attributes

Attribute Name	*Related Method*	*Description*

android:completionHint	setCompletionHint(CharSequence)	Defines the hint displayed in the drop down menu.
android:completionHintView		Defines the hint view displayed in the drop down menu.
android:completionThreshold	setThreshold(int)	Defines the number of characters that the user must type before completion suggestions are displayed in a drop down menu.
android:dropDownAnchor	setDropDownAnchor(int)	View to anchor the auto-complete dropdown to.
android:dropDownHeight	setDropDownHeight(int)	Specifies the basic height of the dropdown.
android:dropDownHorizontalOffset		Amount of pixels by which the drop down should be offset horizontally.
android:dropDownSelector		Selector in a drop down list.
android:dropDownVerticalOffset		Amount of pixels by which the drop down should be offset vertically.
android:dropDownWidth	setDropDownWidth(int)	Specifies the basic width of the dropdown.
android:popupBackground	setDropDownBackgroundResource(int)	

Inherited XML Attributes [Expand]

- From class android.widget.TextView
- From class android.view.View

Inherited Constants [Expand]

- From class android.view.View

Inherited Fields [Expand]

- From class android.view.View

Public Constructors

AutoCompleteTextView (Context context)
AutoCompleteTextView (Context context, AttributeSet attrs)
AutoCompleteTextView (Context context, AttributeSet attrs, int defStyle)

Public Methods

void clearListSelection ()
Clear the list selection.

void dismissDropDown ()
Closes the drop down if present on screen.

boolean enoughToFilter ()
Returns `true` if the amount of text in the field meets or exceeds the `getThreshold()` requirement.

ListAdapter getAdapter ()
Returns a filterable list adapter used for auto completion.

CharSequence getCompletionHint ()
Gets the optional hint text displayed at the bottom of the the matching list.

getDropDownAnchor ()

int

Returns the id for the view that the auto-complete drop down list is anchored to.

getDropDownBackground ()

Drawable

Gets the background of the auto-complete drop-down list.

getDropDownHeight ()

int

Returns the current height for the auto-complete drop down list.

getDropDownHorizontalOffset ()

int

Gets the horizontal offset used for the auto-complete drop-down list.

getDropDownVerticalOffset ()

int

Gets the vertical offset used for the auto-complete drop-down list.

getDropDownWidth ()

int

Returns the current width for the auto-complete drop down list.

getItemClickListener ()

AdapterView.OnItemClickListener

This method was deprecated in API level 3. Use `getOnItemClickListener()` intead

getItemSelectedListener ()

AdapterView.OnItemSelectedListener

This method was deprecated in API level 3. Use `getOnItemSelectedListener()` intead

getListSelection ()

int

Get the position of the dropdown view selection, if there is one.

getOnItemClickListener ()

AdapterView.OnItemClickListener

Returns the listener that is notified whenever the user clicks an item in the drop down list.

getOnItemSelectedListener ()

AdapterView.OnItemSelectedListener

Returns the listener that is notified whenever the user selects an item in the drop down list.

getThreshold ()

int

Returns the number of characters the user must type before the drop down list is shown.

getValidator ()

AutoCompleteTextView.Validator

Returns the Validator set with `setValidator(AutoCompleteTextView.Validator)`, or `null` if it was not set.

isPerformingCompletion ()

boolean

Identifies whether the view is currently performing a text completion, so subclasses can decide whether to respond to text changed events.

isPopupShowing ()

boolean

Indicates whether the popup menu is showing.

onCommitCompletion (CompletionInfo completion)

void

Called by the framework in response to a text completion from the current input method, provided by it calling `InputConnection.commitCompletion()`.

onFilterComplete (int count)

void

Notifies the end of a filtering operation.

onKeyDown (int keyCode, KeyEvent event)

boolean	Default implementation of `KeyEvent.Callback.onKeyDown()`: perform press of the view when `KEYCODE_DPAD_CENTER` or `KEYCODE_ENTER` is released, if the view is enabled and clickable.
	onKeyPreIme (int keyCode, KeyEvent event)
boolean	Handle a key event before it is processed by any input method associated with the view hierarchy.
	onKeyUp (int keyCode, KeyEvent event)
boolean	Default implementation of `KeyEvent.Callback.onKeyUp()`: perform clicking of the view when `KEYCODE_DPAD_CENTER` or `KEYCODE_ENTER` is released.
	onWindowFocusChanged (boolean hasWindowFocus)
void	Called when the window containing this view gains or loses focus.
	performCompletion ()
void	Performs the text completion by converting the selected item from the drop down list into a string, replacing the text box's content with this string and finally dismissing the drop down menu.
	performValidation ()
void	If a validator was set on this view and the current string is not valid, ask the validator to fix it.
	setAdapter (T adapter)
<T extends ListAdapter & Filterable> void	Changes the list of data used for auto completion.
	setCompletionHint (CharSequence hint)
void	Sets the optional hint text that is displayed at the bottom of the the matching list.
	setDropDownAnchor (int id)
void	Sets the view to which the auto-complete drop down list should anchor.
	setDropDownBackgroundDrawable (Drawable d)
void	Sets the background of the auto-complete drop-down list.
	setDropDownBackgroundResource (int id)
void	Sets the background of the auto-complete drop-down list.
	setDropDownHeight (int height)
void	Sets the current height for the auto-complete drop down list.
	setDropDownHorizontalOffset (int offset)
void	Sets the horizontal offset used for the auto-complete drop-down list.
	setDropDownVerticalOffset (int offset)
void	Sets the vertical offset used for the auto-complete drop-down list.
	setDropDownWidth (int width)
void	Sets the current width for the auto-complete drop down list.
void	setListSelection (int position) Set the position of the dropdown view selection.
void	setOnClickListener (View.OnClickListener listener) Register a callback to be invoked when this view is clicked.
	setOnDismissListener (AutoCompleteTextView.OnDismissListener dismissListener)
void	Set a listener that will be invoked whenever the AutoCompleteTextView's list of completions is dismissed.
	setOnItemClickListener (AdapterView.OnItemClickListener l)

void	Sets the listener that will be notified when the user clicks an item in the drop down list. setOnItemSelectedListener (AdapterView.OnItemSelectedListener l)
void	Sets the listener that will be notified when the user selects an item in the drop down list. setText (CharSequence text, boolean filter)
void	Like `setText(CharSequence)`, except that it can disable filtering. setThreshold (int threshold)
void	Specifies the minimum number of characters the user has to type in the edit box before the drop down list is shown. setValidator (AutoCompleteTextView.Validator validator)
void	Sets the validator used to perform text validation. showDropDown ()
void	Displays the drop down on screen.

Protected Methods

CharSequence	convertSelectionToString (Object selectedItem) Converts the selected item from the drop down list into a sequence of character that can be used in the edit box.
Filter	getFilter () Returns the Filter obtained from `getFilter()`, or `null` if `setAdapter(T)` was not called with a Filterable.
void	onAttachedToWindow () This is called when the view is attached to a window.
void	onDetachedFromWindow () This is called when the view is detached from a window.
void	onDisplayHint (int hint) Gives this view a hint about whether is displayed or not.
void	onFocusChanged (boolean focused, int direction, Rect previouslyFocusedRect) Called by the view system when the focus state of this view changes.
void	performFiltering (CharSequence text, int keyCode) Starts filtering the content of the drop down list.
void	replaceText (CharSequence text) Performs the text completion by replacing the current text by the selected item.
boolean	setFrame (int l, int t, int r, int b) Assign a size and position to this view.

Inherited Methods [Expand]

- From class android.widget.EditText
- From class android.widget.TextView
- From class android.view.View
- From class java.lang.Object
- From interface android.graphics.drawable.Drawable.Callback
- From interface android.view.KeyEvent.Callback
- From interface android.view.ViewTreeObserver.OnPreDrawListener
- From interface android.view.accessibility.AccessibilityEventSource
- From interface android.widget.Filter.FilterListener

XML Attributes

android:completionHint

Defines the hint displayed in the drop down menu.

Must be a string value, using '\\;' to escape characters such as '\\n' or '\\uxxxx' for a unicode character.

This may also be a reference to a resource (in the form "@[*package*:]*type*:*name*") or theme attribute (in the form "?[*package*:][*type*:]*name*") containing a value of this type.

This corresponds to the global attribute resource symbol completionHint (/reference/android/R.attr.html#completionHint).

Related Methods
setCompletionHint(CharSequence)

android:completionHintView

Defines the hint view displayed in the drop down menu.

Must be a reference to another resource, in the form "@[+][*package*:]*type*:*name*" or to a theme attribute in the form "?[*package*:][*type*:]*name*".

This corresponds to the global attribute resource symbol completionHintView (/reference/android/R.attr.html#completionHintView).

Related Methods

android:completionThreshold

Defines the number of characters that the user must type before completion suggestions are displayed in a drop down menu.

Must be an integer value, such as "100".

This may also be a reference to a resource (in the form "@[*package*:]*type*:*name*") or theme attribute (in the form "?[*package*:][*type*:]*name*") containing a value of this type.

This corresponds to the global attribute resource symbol completionThreshold (/reference/android/R.attr.html#completionThreshold).

Related Methods
setThreshold(int)

android:dropDownAnchor

View to anchor the auto-complete dropdown to. If not specified, the text view itself is used.

Must be a reference to another resource, in the form "@[+][*package*:]*type*:*name*" or to a theme attribute in the form "?[*package*:][*type*:]*name*".

This corresponds to the global attribute resource symbol dropDownAnchor (/reference/android/R.attr.html#dropDownAnchor).

Related Methods
setDropDownAnchor(int)

android:dropDownHeight

Specifies the basic height of the dropdown. Its value may be a dimension (such as "12dip") for a constant height, fill_parent or match_parent to fill the height of the screen, or wrap_content to match the height of the content of the drop down.

May be a dimension value, which is a floating point number appended with a unit such as "14.5sp". Available units are: px (pixels), dp (density-independent pixels), sp (scaled pixels based on preferred font size), in (inches), mm (millimeters).

This may also be a reference to a resource (in the form "@[*package*:]*type*:*name*") or theme attribute (in the form "?[*package*:][*type*:]*name*") containing a value of this type.

May be one of the following constant values.

Constant	Value	Description
`fill_parent`	-1	The dropdown should fit the height of the screen. This constant is deprecated starting from API Level 8 and is replaced by `match_parent`.
`match_parent`	-1	The dropdown should fit the height of the screen. Introduced in API Level 8.
`wrap_content`	-2	The dropdown should fit the height of the content.

This corresponds to the global attribute resource symbol `dropDownHeight` (/reference/android/R.attr.html#dropDownHeight).

Related Methods

setDropDownHeight(int)

android:dropDownHorizontalOffset

Amount of pixels by which the drop down should be offset horizontally.

Must be a dimension value, which is a floating point number appended with a unit such as "`14.5sp`". Available units are: px (pixels), dp (density-independent pixels), sp (scaled pixels based on preferred font size), in (inches), mm (millimeters).

This may also be a reference to a resource (in the form "`@[`*`package`*`:]`*`type`*`:`*`name`*") or theme attribute (in the form "`?[`*`package`*`:][`*`type`*`:]`*`name`*") containing a value of this type.

This corresponds to the global attribute resource symbol `dropDownHorizontalOffset` (/reference/android/R.attr.html#dropDownHorizontalOffset).

Related Methods

android:dropDownSelector

Selector in a drop down list.

May be a reference to another resource, in the form "`@[+][`*`package`*`:]`*`type`*`:`*`name`*" or to a theme attribute in the form "`?[`*`package`*`:][`*`type`*`:]`*`name`*".

May be a color value, in the form of "`#`*`rgb`*", "`#`*`argb`*", "`#`*`rrggbb`*", or "`#`*`aarrggbb`*".

This corresponds to the global attribute resource symbol `dropDownSelector` (/reference/android/R.attr.html#dropDownSelector).

Related Methods

android:dropDownVerticalOffset

Amount of pixels by which the drop down should be offset vertically.

Must be a dimension value, which is a floating point number appended with a unit such as "`14.5sp`". Available units are: px (pixels), dp (density-independent pixels), sp (scaled pixels based on preferred font size), in (inches), mm (millimeters).

This may also be a reference to a resource (in the form "`@[`*`package`*`:]`*`type`*`:`*`name`*") or theme attribute (in the form "`?[`*`package`*`:][`*`type`*`:]`*`name`*") containing a value of this type.

This corresponds to the global attribute resource symbol `dropDownVerticalOffset` (/reference/android/R.attr.html#dropDownVerticalOffset).

Related Methods

android:dropDownWidth

Specifies the basic width of the dropdown. Its value may be a dimension (such as "12dip") for a constant width, fill_parent or match_parent to match the width of the screen, or wrap_content to match the width of the anchored view.

May be a dimension value, which is a floating point number appended with a unit such as "`14.5sp`". Available units are: px (pixels), dp (density-independent pixels), sp (scaled pixels based on preferred font size), in (inches), mm (millimeters).

This may also be a reference to a resource (in the form "`@[package:]type:name`") or theme attribute (in the form "`?[package:][type:]name`") containing a value of this type.

May be one of the following constant values.

Constant	Value	Description
`fill_parent`	-1	The dropdown should fill the width of the screen. This constant is deprecated starting from API Level 8 and is replaced by `match_parent`.
`match_parent`	-1	The dropdown should fit the width of the screen. Introduced in API Level 8.
`wrap_content`	-2	The dropdown should fit the width of its anchor.

This corresponds to the global attribute resource symbol `dropDownWidth` (/reference/android/R.attr.html#dropDownWidth).

Related Methods
setDropDownWidth(int)

android:popupBackground

Related Methods
setDropDownBackgroundResource(int)

Public Constructors

public **AutoCompleteTextView** (Context context) Added in API level 1

public **AutoCompleteTextView** (Context context, AttributeSet attrs) Added in API level 1

public **AutoCompleteTextView** (Context context, AttributeSet attrs, int defStyle) Added in API level 1

Public Methods

public void **clearListSelection** () Added in API level 3

Clear the list selection. This may only be temporary, as user input will often bring it back.

public void **dismissDropDown** () Added in API level 1

Closes the drop down if present on screen.

public boolean **enoughToFilter** () Added in API level 1

Returns `true` if the amount of text in the field meets or exceeds the `getThreshold()` (/reference/android/widget/AutoCompleteTextView.html#getThreshold()) requirement. You can override this to impose a different standard for when filtering will be triggered.

public ListAdapter **getAdapter** () Added in API level 1

Returns a filterable list adapter used for auto completion.

Returns
a data adapter used for auto completion

public CharSequence **getCompletionHint** () Added in API level 16

Gets the optional hint text displayed at the bottom of the the matching list.

Related XML Attributes
android:completionHint

Returns
The hint text, if any

See Also
`setCompletionHint(CharSequence)`

public int **getDropDownAnchor** () Added in API level 3

Returns the id for the view that the auto-complete drop down list is anchored to.

Related XML Attributes
android:dropDownAnchor

Returns
the view's id, or `NO_ID` if none specified

public Drawable **getDropDownBackground** () Added in API level 5

Gets the background of the auto-complete drop-down list.

Related XML Attributes
android:popupBackground

Returns
the background drawable

public int **getDropDownHeight** () Added in API level 4

Returns the current height for the auto-complete drop down list. This can be a fixed height, or `MATCH_PARENT (/reference/android/view/ViewGroup.LayoutParams.html#MATCH_PARENT)` to fill the screen, or `WRAP_CONTENT (/reference/android/view/ViewGroup.LayoutParams.html#WRAP_CONTENT)` to fit the height of the drop down's content.

Related XML Attributes
android:dropDownHeight

Returns
the height for the drop down list

public int **getDropDownHorizontalOffset** () Added in API level 5

Gets the horizontal offset used for the auto-complete drop-down list.

Returns
the horizontal offset

public int **getDropDownVerticalOffset** () Added in API level 5

Gets the vertical offset used for the auto-complete drop-down list.

Returns
the vertical offset

public int **getDropDownWidth** () Added in API level 3

Returns the current width for the auto-complete drop down list. This can be a fixed width, or `MATCH_PARENT (/reference/android/view/ViewGroup.LayoutParams.html#MATCH_PARENT)` to fill the screen, or `WRAP_CONTENT (/reference/android/view/ViewGroup.LayoutParams.html#WRAP_CONTENT)` to fit the width of its anchor view.

Related XML Attributes
android:dropDownWidth

Returns
the width for the drop down list

public AdapterView.OnItemClickListener **getItemClickListener** () Added in API level 1

This method was deprecated in API level 3.
Use `getOnItemClickListener()` (/reference/android/widget/AutoCompleteTextView.html#getOnItemClickListener()) intead

Returns the listener that is notified whenever the user clicks an item in the drop down list.

Returns
the item click listener

public AdapterView.OnItemSelectedListener **getItemSelectedListener** ()Added in API level 1

This method was deprecated in API level 3.
Use `getOnItemSelectedListener()` (/reference/android/widget/AutoCompleteTextView.html#getOnItemSelectedListener()) intead

Returns the listener that is notified whenever the user selects an item in the drop down list.

Returns
the item selected listener

public int **getListSelection** () Added in API level 3

Get the position of the dropdown view selection, if there is one. Returns `ListView.INVALID_POSITION` (/reference/android/widget/AdapterView.html#INVALID_POSITION) if there is no dropdown or if there is no selection.

Returns
the position of the current selection, if there is one, or `ListView.INVALID_POSITION` if not.

See Also
`getSelectedItemPosition()`

public AdapterView.OnItemClickListener **getOnItemClickListener** () Added in API level 3

Returns the listener that is notified whenever the user clicks an item in the drop down list.

Returns
the item click listener

public AdapterView.OnItemSelectedListener **getOnItemSelectedListener** () Added in API level 3

Returns the listener that is notified whenever the user selects an item in the drop down list.

Returns
the item selected listener

public int **getThreshold** () Added in API level 1

Returns the number of characters the user must type before the drop down list is shown.

Related XML Attributes
android:completionThreshold

Returns
the minimum number of characters to type to show the drop down

See Also
`setThreshold(int)`

public AutoCompleteTextView.Validator **getValidator** () Added in API level 1

Returns the Validator set with `setValidator(AutoCompleteTextView.Validator)` (/reference/android/widget/AutoCompleteTextView.html#setValidator(android.widget.AutoCompleteTextView.Validator)), or `null` if it was not set.

See Also

`setValidator(android.widget.AutoCompleteTextView.Validator)`
`performValidation()`

public boolean **isPerformingCompletion** ()

Added in API level 3

Identifies whether the view is currently performing a text completion, so subclasses can decide whether to respond to text changed events.

public boolean **isPopupShowing** ()

Added in API level 1

Indicates whether the popup menu is showing.

Returns

true if the popup menu is showing, false otherwise

public void **onCommitCompletion** (CompletionInfo completion)

Added in API level 3

Called by the framework in response to a text completion from the current input method, provided by it calling `InputConnection.commitCompletion()` (/reference/android/view/inputmethod/InputConnection.html#commitCompletion(android.view.inputmethod.CompletionInfo)). The default implementation does nothing; text views that are supporting auto-completion should override this to do their desired behavior.

Parameters

completion The auto complete text the user has selected.

public void **onFilterComplete** (int count)

Added in API level 1

Notifies the end of a filtering operation.

Parameters

count the number of values computed by the filter

public boolean **onKeyDown** (int keyCode, KeyEvent event)

Added in API level 1

Default implementation of `KeyEvent.Callback.onKeyDown()` (/reference/android/view/KeyEvent.Callback.html#onKeyDown(int, android.view.KeyEvent)): perform press of the view when `KEYCODE_DPAD_CENTER` (/reference/android/view/KeyEvent.html#KEYCODE_DPAD_CENTER) or `KEYCODE_ENTER` (/reference/android/view/KeyEvent.html#KEYCODE_ENTER) is released, if the view is enabled and clickable.

Key presses in software keyboards will generally NOT trigger this listener, although some may elect to do so in some situations. Do not rely on this to catch software key presses.

Parameters

keyCode A key code that represents the button pressed, from `KeyEvent`.

event The KeyEvent object that defines the button action.

Returns

If you handled the event, return true. If you want to allow the event to be handled by the next receiver, return false.

public boolean **onKeyPreIme** (int keyCode, KeyEvent event)

Added in API level 3

Handle a key event before it is processed by any input method associated with the view hierarchy. This can be used to intercept key events in special situations before the IME consumes them; a typical example would be handling the BACK key to update the application's UI instead of allowing the IME to see it and close itself.

Parameters

keyCode The value in event.getKeyCode().

event Description of the key event.

Returns

If you handled the event, return true. If you want to allow the event to be handled by the next receiver, return false.

public boolean **onKeyUp** (int keyCode, KeyEvent event) Added in API level 1

Default implementation of KeyEvent.Callback.onKeyUp() (/reference/android/view/KeyEvent.Callback.html#onKeyUp(int, android.view.KeyEvent)): perform clicking of the view when KEYCODE_DPAD_CENTER (/reference/android/view/KeyEvent.html#KEYCODE_DPAD_CENTER) or KEYCODE_ENTER (/reference/android/view/KeyEvent.html#KEYCODE_ENTER) is released.

Key presses in software keyboards will generally NOT trigger this listener, although some may elect to do so in some situations. Do not rely on this to catch software key presses.

Parameters

keyCode A key code that represents the button pressed, from KeyEvent.

event The KeyEvent object that defines the button action.

Returns

If you handled the event, return true. If you want to allow the event to be handled by the next receiver, return false.

public void **onWindowFocusChanged** (boolean hasWindowFocus) Added in API level 1

Called when the window containing this view gains or loses focus. Note that this is separate from view focus: to receive key events, both your view and its window must have focus. If a window is displayed on top of yours that takes input focus, then your own window will lose focus but the view focus will remain unchanged.

Parameters

hasWindowFocus True if the window containing this view now has focus, false otherwise.

public void **performCompletion** () Added in API level 1

Performs the text completion by converting the selected item from the drop down list into a string, replacing the text box's content with this string and finally dismissing the drop down menu.

public void **performValidation** () Added in API level 1

If a validator was set on this view and the current string is not valid, ask the validator to fix it.

See Also

getValidator()
setValidator(android.widget.AutoCompleteTextView.Validator)

public void **setAdapter** (T adapter) Added in API level 1

Changes the list of data used for auto completion. The provided list must be a filterable list adapter.

The caller is still responsible for managing any resources used by the adapter. Notably, when the AutoCompleteTextView is closed or released, the adapter is not notified. A common case is the use of CursorAdapter (/reference/android/widget/CursorAdapter.html), which contains a Cursor (/reference/android/database/Cursor.html) that must be closed. This can be done automatically (see startManagingCursor() (/reference/android/app/Activity.html#startManagingCursor(android.database.Cursor)), or by manually closing the cursor when the AutoCompleteTextView is dismissed.

Parameters

adapter the adapter holding the auto completion data

See Also

getAdapter()
Filterable
ListAdapter

public void **setCompletionHint** (CharSequence hint) Added in API level 1

Sets the optional hint text that is displayed at the bottom of the the matching list. This can be used as a cue to the user on how to best use the list, or to provide extra information.

Related XML Attributes
android:completionHint

Parameters

hint the text to be displayed to the user

See Also
`getCompletionHint()`

public void **setDropDownAnchor** (int id) Added in API level 3

Sets the view to which the auto-complete drop down list should anchor. The view corresponding to this id will not be loaded until the next time it is needed to avoid loading a view which is not yet instantiated.

Related XML Attributes
android:dropDownAnchor

Parameters

id the id to anchor the drop down list view to

public void **setDropDownBackgroundDrawable** (Drawable d) Added in API level 5

Sets the background of the auto-complete drop-down list.

Related XML Attributes
android:popupBackground

Parameters

d the drawable to set as the background

public void **setDropDownBackgroundResource** (int id) Added in API level 5

Sets the background of the auto-complete drop-down list.

Related XML Attributes
android:popupBackground

Parameters

id the id of the drawable to set as the background

public void **setDropDownHeight** (int height) Added in API level 4

Sets the current height for the auto-complete drop down list. This can be a fixed height, or `MATCH_PARENT (/reference/android/view/ViewGroup.LayoutParams.html#MATCH_PARENT)` to fill the screen, or `WRAP_CONTENT (/reference/android/view/ViewGroup.LayoutParams.html#WRAP_CONTENT)` to fit the height of the drop down's content.

Related XML Attributes
android:dropDownHeight

Parameters

height the height to use

public void **setDropDownHorizontalOffset** (int offset) Added in API level 5

Sets the horizontal offset used for the auto-complete drop-down list.

Parameters

offset the horizontal offset

public void **setDropDownVerticalOffset** (int offset) Added in API level 5

Sets the vertical offset used for the auto-complete drop-down list.

Parameters

offset the vertical offset

public void **setDropDownWidth** (int width) Added in API level 3

Sets the current width for the auto-complete drop down list. This can be a fixed width, or MATCH_PARENT (/reference/android/view/ViewGroup.LayoutParams.html#MATCH_PARENT) to fill the screen, or WRAP_CONTENT (/reference/android/view/ViewGroup.LayoutParams.html#WRAP_CONTENT) to fit the width of its anchor view.

Related XML Attributes

android:dropDownWidth

Parameters

width the width to use

public void **setListSelection** (int position) Added in API level 3

Set the position of the dropdown view selection.

Parameters

position The position to move the selector to.

public void **setOnClickListener** (View.OnClickListener listener) Added in API level 1

Register a callback to be invoked when this view is clicked. If this view is not clickable, it becomes clickable.

Parameters

listener The callback that will run

public void **setOnDismissListener** (AutoCompleteTextView.OnDismissListener dismissListener) Added in API level 17

Set a listener that will be invoked whenever the AutoCompleteTextView's list of completions is dismissed.

Parameters

dismissListener Listener to invoke when completions are dismissed

public void **setOnItemClickListener** (AdapterView.OnItemClickListener l) Added in API level 1

Sets the listener that will be notified when the user clicks an item in the drop down list.

Parameters

l the item click listener

public void **setOnItemSelectedListener** (AdapterView.OnItemSelectedListener l) Added in API level 1

Sets the listener that will be notified when the user selects an item in the drop down list.

Parameters

l the item selected listener

public void **setText** (CharSequence text, boolean filter) Added in API level 17

Like setText(CharSequence) (/reference/android/widget/TextView.html#setText(java.lang.CharSequence)), except that it can disable filtering.

Parameters

filter If `false`, no filtering will be performed as a result of this call.

public void **setThreshold** (int threshold) Added in API level 1

Specifies the minimum number of characters the user has to type in the edit box before the drop down list is shown.

When `threshold` is less than or equals 0, a threshold of 1 is applied.

Related XML Attributes

android:completionThreshold

Parameters

threshold the number of characters to type before the drop down is shown

See Also

`getThreshold()`

public void **setValidator** (AutoCompleteTextView.Validator validator) Added in API level 1

Sets the validator used to perform text validation.

Parameters

validator The validator used to validate the text entered in this widget.

See Also

`getValidator()`
`performValidation()`

public void **showDropDown** () Added in API level 1

Displays the drop down on screen.

Protected Methods

protected CharSequence **convertSelectionToString** (Object selectedItem) Added in API level 1

Converts the selected item from the drop down list into a sequence of character that can be used in the edit box.

Parameters

selectedItem the item selected by the user for completion

Returns

a sequence of characters representing the selected suggestion

protected Filter **getFilter** () Added in API level 1

Returns the Filter obtained from `getFilter()` (/reference/android/widget/Filterable.html#getFilter()), or `null` if `setAdapter(T)` (/reference/android/widget/AutoCompleteTextView.html#setAdapter(T)) was not called with a Filterable.

protected void **onAttachedToWindow** () Added in API level 1

This is called when the view is attached to a window. At this point it has a Surface and will start drawing. Note that this function is guaranteed to be called before `onDraw(android.graphics.Canvas)` (/reference/android/view/View.html#onDraw(android.graphics.Canvas)), however it may be called any time before the first onDraw -- including before or after `onMeasure(int, int)` (/reference/android/view/View.html#onMeasure(int, int)).

protected void **onDetachedFromWindow** () Added in API level 1

This is called when the view is detached from a window. At this point it no longer has a surface for drawing.

protected void **onDisplayHint** (int hint) Added in API level 8

Gives this view a hint about whether is displayed or not. For instance, when a View moves out of the screen, it might receives a display hint indicating the view is not displayed. Applications should not *rely* on this hint as there is no guarantee that they will receive one.

Parameters

hint	A hint about whether or not this view is displayed: `VISIBLE` or `INVISIBLE`.

protected void **onFocusChanged** (boolean focused, int direction, Rect previouslyFocusedRect) Added in API level 1

Called by the view system when the focus state of this view changes. When the focus change event is caused by directional navigation, direction and previouslyFocusedRect provide insight into where the focus is coming from. When overriding, be sure to call up through to the super class so that the standard focus handling will occur.

Parameters

focused	True if the View has focus; false otherwise.
direction	The direction focus has moved when requestFocus() is called to give this view focus. Values are `FOCUS_UP`, `FOCUS_DOWN`, `FOCUS_LEFT`, `FOCUS_RIGHT`, `FOCUS_FORWARD`, or `FOCUS_BACKWARD`. It may not always apply, in which case use the default.
previouslyFocusedRect	The rectangle, in this view's coordinate system, of the previously focused view. If applicable, this will be passed in as finer grained information about where the focus is coming from (in addition to direction). Will be `null` otherwise.

protected void **performFiltering** (CharSequence text, int keyCode) Added in API level 1

Starts filtering the content of the drop down list. The filtering pattern is the content of the edit box. Subclasses should override this method to filter with a different pattern, for instance a substring of `text`.

Parameters

text	the filtering pattern
keyCode	the last character inserted in the edit box; beware that this will be null when text is being added through a soft input method.

protected void **replaceText** (CharSequence text) Added in API level 1

Performs the text completion by replacing the current text by the selected item. Subclasses should override this method to avoid replacing the whole content of the edit box.

Parameters

text	the selected suggestion in the drop down list

protected boolean **setFrame** (int l, int t, int r, int b) Added in API level 1

Assign a size and position to this view. This is called from layout.

Parameters

l	Left position, relative to parent
t	Top position, relative to parent
r	Right position, relative to parent
b	Bottom position, relative to parent

Returns

true if the new size and position are different than the previous ones