Color State List Resource

See also

- Color (simple value)

A `ColorStateList` is an object you can define in XML that you can apply as a color, but will actually change colors, depending on the state of the `View` object to which it is applied. For example, a `Button` widget can exist in one of several different states (pressed, focused, or neither) and, using a color state list, you can provide a different color during each state.

You can describe the state list in an XML file. Each color is defined in an `<item>` element inside a single `<selector>` element. Each `<item>` uses various attributes to describe the state in which it should be used.

During each state change, the state list is traversed top to bottom and the first item that matches the current state will be used—the selection is *not* based on the "best match," but simply the first item that meets the minimum criteria of the state.

> **Note:** If you want to provide a static color resource, use a simple Color value.

FILE LOCATION:

`res/color/filename.xml`
The filename will be used as the resource ID.

COMPILED RESOURCE DATATYPE:

Resource pointer to a `ColorStateList`.

RESOURCE REFERENCE:

In Java: `R.color.filename`
In XML: `@[package:]color/filename`

SYNTAX:

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android" >
    <item
        android:color="hex_color"
        android:state_pressed=["true" | "false"]
        android:state_focused=["true" | "false"]
        android:state_selected=["true" | "false"]
        android:state_checkable=["true" | "false"]
        android:state_checked=["true" | "false"]
        android:state_enabled=["true" | "false"]
        android:state_window_focused=["true" | "false"] />
</selector>
```

ELEMENTS:

`<selector>`

Required. This must be the root element. Contains one or more `<item>` elements.

attributes:

`xmlns:android`

String. **Required.** Defines the XML namespace, which must be `"http://schemas.android.com/apk/res/android"`.

`<item>`

Defines a color to use during certain states, as described by its attributes. Must be a child of a `<selector>` element.

attributes:

`android:color`

Hexadeximal color. **Required**. The color is specified with an RGB value and optional alpha channel.

The value always begins with a pound (#) character and then followed by the Alpha-Red-Green-Blue information in one of the following formats:

- *#RGB*
- *#ARGB*
- *#RRGGBB*
- *#AARRGGBB*

`android:state_pressed`

Boolean. "true" if this item should be used when the object is pressed (such as when a button is touched/clicked); "false" if this item should be used in the default, non-pressed state.

`android:state_focused`

Boolean. "true" if this item should be used when the object is focused (such as when a button is highlighted using the trackball/d-pad); "false" if this item should be used in the default, non-focused state.

`android:state_selected`

Boolean. "true" if this item should be used when the object is selected (such as when a tab is opened); "false" if this item should be used when the object is not selected.

`android:state_checkable`

Boolean. "true" if this item should be used when the object is checkable; "false" if this item should be used when the object is not checkable. (Only useful if the object can transition between a checkable and non-checkable widget.)

`android:state_checked`

Boolean. "true" if this item should be used when the object is checked; "false" if it should be used when the object is un-checked.

`android:state_enabled`

Boolean. "true" if this item should be used when the object is enabled (capable of receiving touch/click events); "false" if it should be used when the object is disabled.

`android:state_window_focused`

Boolean. "true" if this item should be used when the application window has focus (the application is in the foreground), "false" if this item should be used when the application window does not have focus (for example, if the notification shade is pulled down or a dialog appears).

> **Note:** Remember that the first item in the state list that matches the current state of the object will be applied. So if the first item in the list contains none of the state attributes above, then it will be applied every time, which is why your default value should always be last, as demonstrated in the following example.

EXAMPLE:

XML file saved at `res/color/button_text.xml`:

```
<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">
    <item android:state_pressed="true"
          android:color="#ffff0000"/> <!-- pressed -->
    <item android:state_focused="true"
          android:color="#ff0000ff"/> <!-- focused -->
    <item android:color="#ff000000"/> <!-- default -->
</selector>
```

This layout XML will apply the color list to a View:

```
<Button
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:text="@string/button_text"
    android:textColor="@color/button_text" />
```

SEE ALSO:

- Color (simple value)
- `ColorStateList`
- State List Drawable