Android toast default position

The toasts are displayed at the bottom of the activity screen on Android devices and are aligned in the center. This is the default behavior of a toast. If you want to change the position of the Toast message, it is possible to use the setGravity method. toast.setGravity (Gravity.TOP, 0, 0); This shows the Toast at the top of the screen above the ActionBar if it is implemented. If you want to display the toast at the top and left, toast.setGravity (Gravity.TOP Gravity.LEFT, 0.0); The same goes for Bottom and Right toast.setGravity Gravity.RIGHT, 0.0); To view the Toast at the Centre, toast.setGravity (Gravity.CENTER, 0, 0); The second is the third parameter can be used to set shifts for direction X and Y direction respectively. More Android-related messages, More messages: Quick Notifications provide easy-to-understand comments on operations in small pop-ups. This message only takes the amount of space needed to view notification content. You can always see and interact with the current activity. Quick notifications automatically disappear after the viewing time has expired. For example, when you click Send an email, the quick message Send a message... will appear, as shown in the screenshot below: If user feedback is needed for status notifications, consider using the notifications. Basic Knowledge First, create a Toast object using one of the makeText methods. This method requires three settings: application context, text message, and fast notification duration. This returns a correctly initiated Fast Message object. You can view quick notifications using the show settings(), as shown in the following example: text val - Hello toast! is worth a long time - Toast.LENGTH_SHORT val toast - Toast.makeText (applicationContext, text, duration) toast.show() Background - getApplicationContext(); Text by CharSequence - Hello toast!; Int-Toast.LENGTH_SHORT duration; Toast toast - Toast.makeText (context, text, duration); toast.show(); This example shows all the settings you need for most quick notifications. You rarely need other settings. However, you can set quick notifications in another location, or even want to use your own layout instead of a regular text message. You can do these things by following the following sections. You can also follow methods and avoid keeping the object Quick Notifications, such as Toast.makeText (context, text, duration).show() text, duration).show(); Set the location of quick notifications A regular quick notification appears horizontally in the center near the bottom of the screen. You can change this position by setGravity (int, int, int). The above method accepts 3 parameters: gravity constant, position compensation value x and y-position clearing value. For example, if you decide that a quick notification will appear in the top left corner, you can set the gravity constant as: toast.setGravity (Gravity.TOP or Gravity.LEFT, 0, 0) toast.setGravity (Gravity.TOP) Gravity.LEFT, 0.0); If you want to move the position to the right, increase the value of the second parameters. To go down, increase the value of the end settings. Create a personalized quick notification view If a regular text message isn't enough, you can create a custom layout for your quick notifications. To create a custom layout, specify the View layout in XML or in your app code and move the original View object to SetView(View). The following snout contains a custom layout for 'linearLayout xmlns:android' android:id-id/custom_toast_container android:orientation-horizontal android:layout_width-fill_parent android:layout_height-fill_parent android:padding #DAAA-8dp android:background ImageView android:src-@drawable/droid android:layout_width-wrap_content android:layout_height-wrap_content android:layout_marginright-8dp/ImageView TextView android:id/text android:layout_width-wrap_content android:layout_height-wrap_content android:textcolor#FFF/TextView A quick notification (register as a layout/custom_toast.xml): Note that the LINEarLayout component ID is custom_toast_container. You should use this ID and the XML custom_toast layout ID to improve the layout, as shown below: val inflater layoutInflater val container: ViewGroup - findViewById(R.id.custom_toast_container) val layout: ViewGroup - inflater.infflated (R.layout.custom_toast, container) text val: TextView - layout.findViewById (R.id.text) text.text - This is a custom toast with (Toast (applicationContext) - setGravity (Gravity.CENTER_VERTICAL, 0, 0) duration - Toast.LENGTH_LONG view layout - infflator - getLayout Inflater View the layout - inflater.inflating (R.layout.custom_toast, (ViewGroup) findViewById (R.id.custom_toast_container)); TextView text - (TextView) layout.findViewById (R.id.text); text.setText (This is a custom toast); Toast toast - new toast (getApplicationContext)); 0, 0); toast.setDuration (Toast.LENGTH_LONG); toast.setView (layout); toast.show(); First, retrieve LayoutInflater using getLayoutInflater(or getSystemService)) and then improve XML's layout with inflating (int, ViewGroup). The first is the layout resource ID, and the second is Native View. You can use this enhanced layout to find other View objects in your composition, save and now set content for ImageView and TextView. Finally, create a quick message using Toast (Context) and set some properties for this quick notification, such as gravity and duration properties. Then call setView and move it to an improved layout. At this point, you can view a quick notification with a custom layout by calling the show.) Note: Don't use the public creation function for quick notifications unless you plan to set a layout using setView(View). If you don't have a custom layout to use, you should use makeText (Context, int, int) to create a quick message. A toast provides simple feedback about an operation in a small popup. It fills only the amount of space needed for the message and the current activity remains visible and interactive. The toasts automatically disappear after an expiration period. Simple Toast First, insert a Toast object with one of the makeText methods. This method takes three parameters: the context of the application, the text message and the duration of the toast. also supports Toast.LENGTH_LONG Toast.makeText (getApplicationContext(), ertains messag Toast.LENGTH_SHORT).show(); You can set up a toast; toast.show(); Custom Toast You can also create a Toast that uses a custom XML layout rather than just displaying plain text. First of all, just set the line 'LinearLayout xmlns:android' android:id/toast_layout_root ImageView android:src-@drawable/droid android:layout_width-wrap_content android:layout_height-wrap_content android:layout_marginright-8dp/ImageView TextView android:id/text android:layout_width-wrap_content android:layout_height-wrap_content android:textcolor#FFF/TextView Note that the LinearLayout element ID is You must use this ID to inflate the XML layout: empty private displayToast (Message thong) // Inflate the XML toast layout Display layout - getLayoutInflater(R.layout.toast_layout, (ViewGroup) findViewById (R.id.toast_layout_root); // Fill out the message in textview TextView - (TextView) layout.findViewById (R.id.text); text.setText (message); // Build toast, set the view and display Toast toast - new Toast (getApplicationContext)); toast.setView(); layout toast.show And then you can view the custom toast using displayToast (Message) ;). References, I'm having the exact opposite problem of this post - in particular, I want to display a toast in the default location (centered, just above the state bar at the bottom), but it's still appearing horizontally and vertically centered. Here is the code and call I use to display the toast (toastNavigation method is in a separate class from the call): static public empty toastNavigation (Context context, CharSequence message, int duration, int gravity, int gravity_xOffset, int gravity_yOffset) - Toast toast - Toast.makeText (context, message, duration); toast.setGravity (gravity, gravity_xOffset, gravity_yOffset); toast.show(); 'toastNavigation'(this, My message, Toast.LENGTH_SHORT, Gravity.NO_GRAVITY, 0, 0); Why would my toast be centered even if I go into the constant that says ... no seriousness has been fixed. »? Is there another constant that I should spend to clear the gravity constants inherited from the context? Context?