

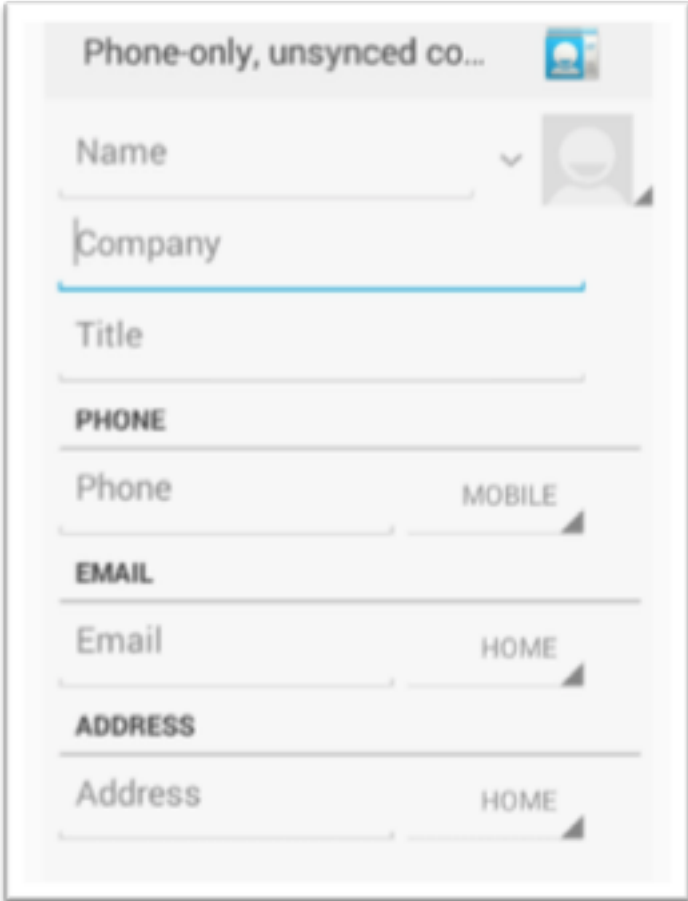


Android Tutorials

By : Piyush Khandelwal

Widgets

In computer programming, a widget (or control) is an element of a graphical user interface (GUI) that displays an information arrangement changeable by the user, such as a window or a text box. The defining characteristic of a widget is to provide a single interaction point for the direct manipulation of a given kind of data. In other words, widgets are basic visual building blocks which, combined in an application, hold all the data processed by the application and the available interactions on this data.



The image shows a contact form with the following fields and sections:

- Header:** "Phone-only, unsynced co..." with a small icon.
- Name:** A text input field with a dropdown arrow and a profile picture icon.
- Company:** A text input field with a blue underline.
- Title:** A text input field.
- PHONE:** A section header.
- Phone:** A text input field with a "MOBILE" label and a dropdown arrow.
- EMAIL:** A section header.
- Email:** A text input field with a "HOME" label and a dropdown arrow.
- ADDRESS:** A section header.
- Address:** A text input field with a "HOME" label and a dropdown arrow.

What Are Containers?

Containers are ways of organizing multiple widgets into some sort of structure.

Widgets do not naturally line themselves up in some specific pattern — we have to define that pattern ourselves.

Common container patterns include:

- put all children in a row, one after the next
- put all children in a column, one below the next
- arrange the children into a table or grid with some number of rows and columns
- anchor the children to the sides of the container, according to requests made by those children
- anchor the children to other children in the container, according to requests made by those children
- stack all children, one on top of the next
- and so on

The Absolute Positioning Anti-Pattern

Example: Windows apps, back in the 1990's

In mobile, particularly with Android, we have a wide range of possible screen resolutions, from QVGA (320x240) to beyond 1080p (1920x1080), and many values in between.

Themes

Web development Cascading Style Sheets (CSS)

Android Styles and themes

Styles are a collection of values for properties (e.g., have a foreground color of red).

These can be applied to specific widgets (e.g., this label should adopt this style),

OR they can be employed by “themes” that provide the default look for all sorts of widgets and other elements of our UI.

Of course, you do not have to declare any theme for your app. Android will give you a default look-and-feel without any specific theme

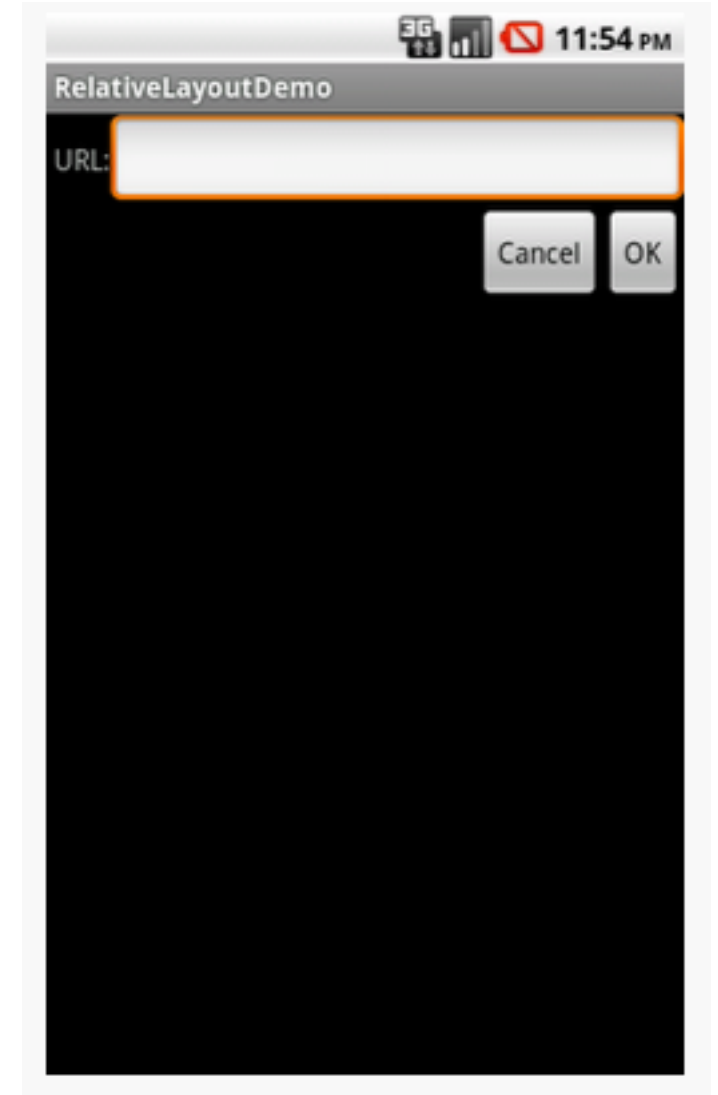
In the Beginning, There Was “Theme”

In Android 1.0, the default theme was known simply as Theme

At the top of the screen, we had a thin gray “title bar” with the name of our app

The focused field (an EditText widget) had a bright orange outline, whereas normally it was a plain white rectangle

The buttons (“OK” and “Cancel”) were... well... buttons



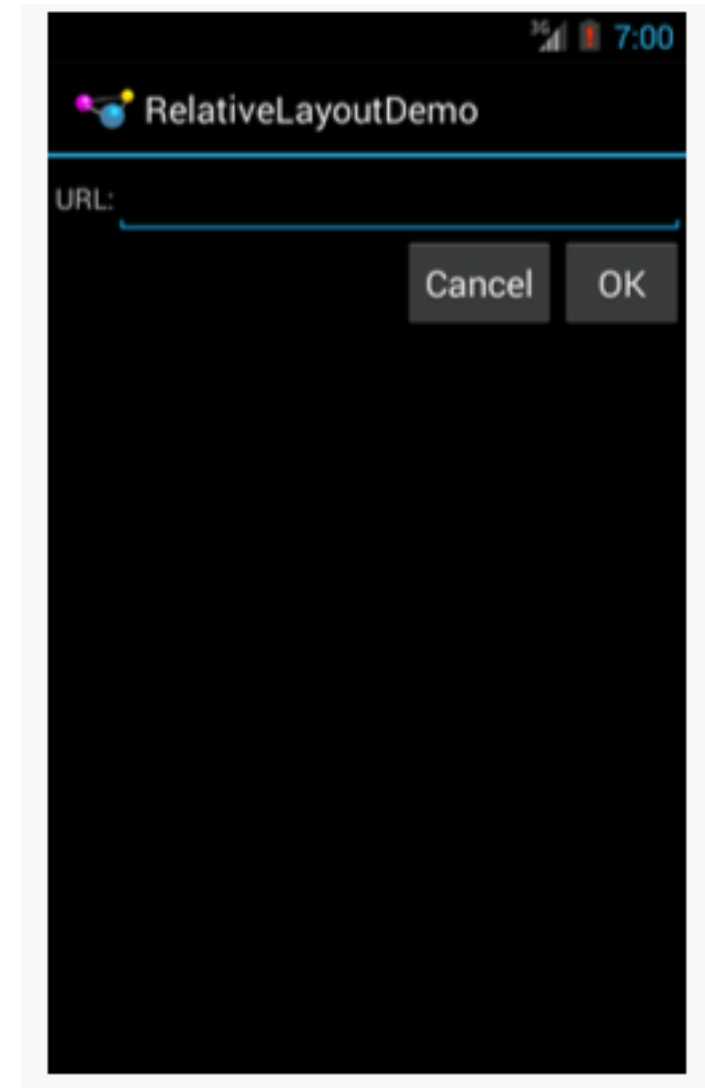
Holo, There!

Android 3.0 (API Level 11) introduced a new default theme, Theme.Holo

At the top of the screen, we have an “action bar”, containing our app’s logo and name

The focused field has a blue “underbracket”, whereas normally it is gray

The buttons are styled slightly differently, with a bigger font, alternative backgrounds, etc.



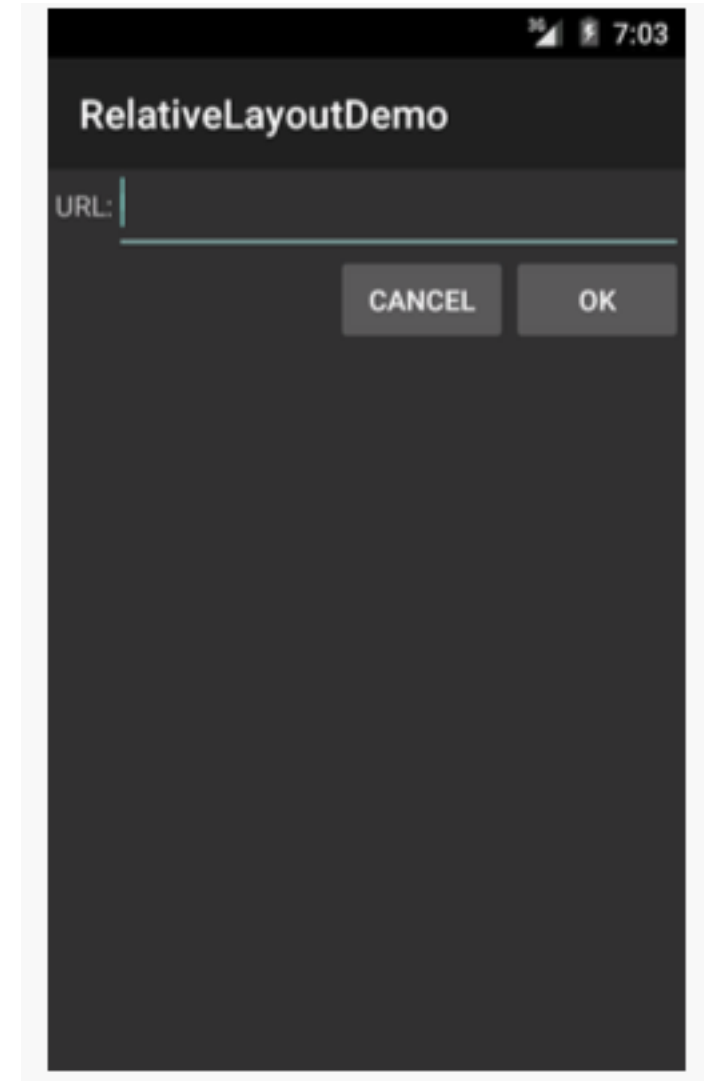
Material Theme

Android 5.0 changed the default theme yet again, to Theme.Material

The action bar at the top of the screen no longer shows the app icon

Our field is indicated by an underline, which is teal when focused or gray when unfocused

The buttons are now forced into all-caps font, with a slightly smaller font size and subtly different background than we had with Theme.Holo



Lets Play with Themes