RelativeLayout

RelativeLayout lays out elements based on their relationships with one another, and with the parent container. This is arguably the most complicated layout, and we need several properties to actually get the layout we want.

Relative To Container

These properties will layout elements relative to the parent container.

* android:layout\_alignParentBottom – Places the bottom of the element on the bottom of the container
* android:layout\_alignParentLeft – Places the left of the element on the left side of the container
* android:layout\_alignParentRight – Places the right of the element on the right side of the container
* android:layout\_alignParentTop – Places the element at the top of the container
* android:layout\_centerHorizontal – Centers the element horizontally within its parent container
* android:layout\_centerInParent – Centers the element both horizontally and vertically within its container
* android:layout\_centerVertical – Centers the element vertically within its parent container

Relative To Other Elements

These properties allow you to layout elements relative to other elements on screen. The value for each of these elements is the id of the element you are using to layout the new element. Each element that is used in this way must have an ID defined using android:id=”@+id/XXXXX” where XXXXX is replaced with the desired id. You use “@id/XXXXX” to reference an element by its id. One thing to remember is that referencing an element before it has been declared will produce an error.

* android:layout\_above – Places the element above the specified element
* android:layout\_below – Places the element below the specified element
* android:layout\_toLeftOf – Places the element to the left of the specified element
* android:layout\_toRightOf – Places the element to the right of the specified element

Alignment With Other Elements

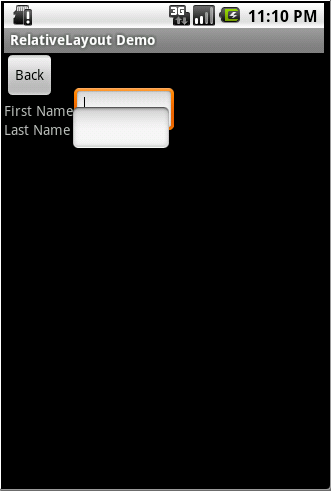
These properties allow you to specify how elements are aligned in relation to other elements.

* android:layout\_alignBaseline – Aligns baseline of the new element with the baseline of the specified element
* android:layout\_alignBottom – Aligns the bottom of new element in with the bottom of the specified element
* android:layout\_alignLeft – Aligns left edge of the new element with the left edge of the specified element
* android:layout\_alignRight – Aligns right edge of the new element with the right edge of the specified element
* android:layout\_alignTop – Places top of the new element in alignment with the top of the specified element

Here is a sample XML Layout

|  |
| --- |
| **<RelativeLayout**  android:layout\_width="fill\_parent"  android:layout\_height="fill\_parent"  xmlns:android="http://schemas.android.com/apk/res/android"**>**  **<Button**  android:id="@+id/backbutton"  android:text="Back"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" **/>**  **<TextView**  android:id="@+id/firstName"  android:text="First Name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@id/backbutton" **/>**  **<EditText**  android:width="100px"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_toRightOf="@id/firstName"  android:layout\_alignBaseline="@id/firstName" **/>**  **<TextView**  android:id="@+id/lastName"  android:text="Last Name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@id/firstName" **/>**  **<EditText**  android:width="100px"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_toRightOf="@id/lastName"  android:layout\_alignBaseline="@id/lastName" **/>**  **</RelativeLayout>** |

Here is the screen produced by that XML.



I wanted to show this to you because the first time I made a RelativeLayout I did exactly this and then looked at the screen and said, “Hang on a minute, that’s not what I wanted!” The problem here is that when Android draws the TextView lastName below the TextView firstName it only sets aside the space it needs for the TextView. Android only reads the Layout XML one time so it doesn’t know that an EditView is the next item and doesn’t plan for it. So when the EditView is drawn to the right of the TextView it only has the height of the TextView to work with so it overlaps the EditView above it. Here is the Layout XML I wrote to create the form the way it should look.

|  |
| --- |
| **<RelativeLayout**  android:layout\_width="fill\_parent"  android:layout\_height="fill\_parent"  xmlns:android="http://schemas.android.com/apk/res/android"**>**  **<Button**  android:id="@+id/backbutton"  android:text="Back"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content" **/>**  **<TextView**  android:id="@+id/firstName"  android:text="First Name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@id/backbutton" **/>**  **<EditText**  android:id="@+id/editFirstName"  android:width="100px"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_toRightOf="@id/firstName"  android:layout\_below="@id/backbutton"**/>**  **<EditText**  android:id="@+id/editLastName"  android:width="100px"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_below="@id/editFirstName"  android:layout\_alignLeft="@id/editFirstName"**/>**  **<TextView**  android:id="@+id/lastName"  android:text="Last Name"  android:layout\_width="wrap\_content"  android:layout\_height="wrap\_content"  android:layout\_toLeftOf="@id/editLastName"  android:layout\_below="@id/editFirstName" **/>**  **</RelativeLayout>** |

You probably noticed that I had to rearrange the elements in the XML since, as I already mentioned, you cannot reference an element that has not already been laid out. Here is what the updated RelativeLayout produces.

