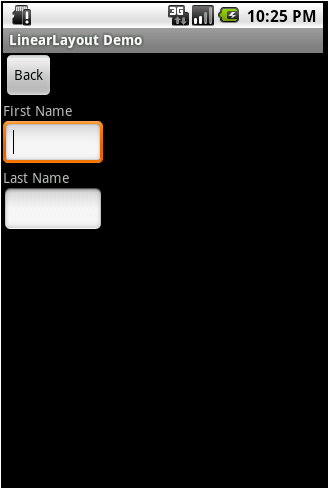
LinearLayout

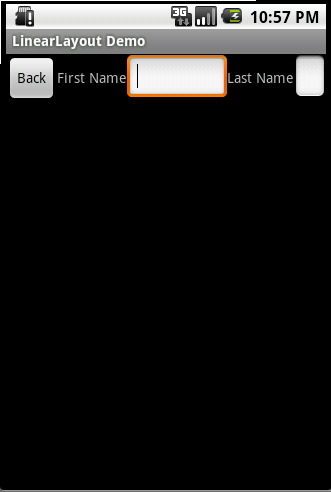
LinearLayout organizes elements along a single line. You specify whether that line is verticle or horizontal using android:orientation. Here is a sample Layout XML using LinearLayout.

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

Here is a screenshot of the result of the above XML.



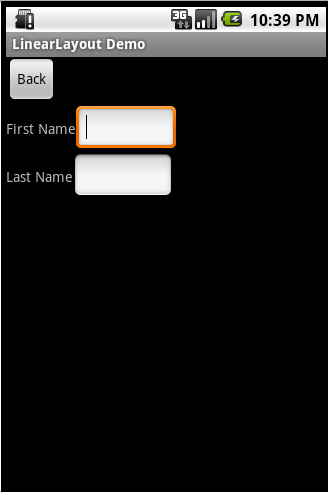
Here is a screenshot of the same XML except that the android:orientation has been changed to horizontal.

  
You might note that the EditText field at the end of the line has had its width reduced in order to fit. Android will try to make adjustments when necessary to fit items on screen. The last page of this tutorial will cover one method to help deal with this.

I mentioned on the first page that Layouts can be nested. LinearLayout is frequently nested, with horizontal and vertical layouts mixed. Here is an example of this.

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

As you can see we have a Vertical LinearLayout whose children are a button, and two horizontal LinearLayouts. Each horizontal LinearLayout has two child controls. You should note that in the child LinearLayout elements I used android:layout\_height=”wrap\_content” instead of fill\_parent. If I had used fill\_parent the first name TextView and EditView would have taken all of the available space on the screen, and the Last Name would not have been displayed. Here is what this XML does display.



Nested Layouts do not have to be of one type. I could, for example, have a LinearLayout as one of the children in a FrameLayout.