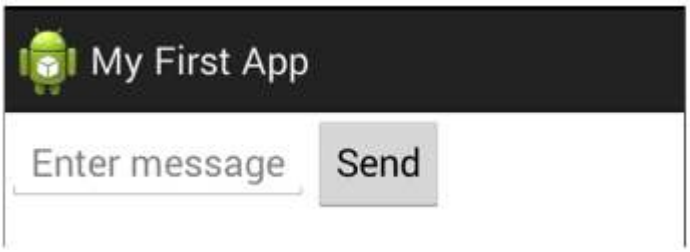
Android View'leri Yatay Olarak Hizalamak

View'leri yatay olarak hizalamak için **bazı attribute(özellik)'lerini** kullanmamız gerekir. Bu özellikleri açıklamadan önce, **LinearLayout** kullanmak **zorunda** olduğumuzu ifade edelim.

LinearLayout child view'lerin genişliklerini ayarlarken, **iki özelliğe bakar:** `android:layout_width ve android:layout_weight` . İlk aşamada, **LinearLayout** `android:layout_width` değerine bakar. Eğer view'lerin `android:layout_width` değerleri **wrap_content** olursa, her biri sadece yeterli miktarda alanı kaplar. Bu ise aşağıda görüldüğü gibi **boşluk oluşmasına** neden olur:



İkinci aşamada **LinearLayout** `android_layout_weight` değerlerine bakar. Eğer **EditText** view'in `android_layout_weight` değeri **"1"** olursa ve **Button**'da `android_layout_weight` özelliğini kullanmazsak tüm boşluğu **EditText'in** kaplaması sağlanmış olur. Çünkü `android_layout_weight` özelliği, bir view'in oluşan boşluğu kaplaması için **önemlilik** değerini temsil eder. Örneğin, bu özellik değeri **"1"** değerine eşit olduğu zaman ilgili view, boşluğu doldurmada öncelik kazanmış olur. View'lerin `android_layout_weight` özelliği **default** olarak **"0"** değerine eşittir.

Eğer birden fazla view'in `android_layout_weight` özellik değeri **"1"** olursa, bu view'ler eşit olarak oluşan boşluğu paylaşırlar. Bu view'lerden birinin değeri **"2"** olursa ve diğer view'in değeri **"1"** olarak ayarlanırsa, o zaman boşluğu, değeri **"2"** olan view daha çok işgal eder. Bu konu ile ilgili ayrıntılı bilgi için **tıklayınız** (/en/tutorial/Android-How-Much-Of-The-Extra-Space-In-The-Layout-To-Be-Allocated-To-The-View)

Sonuç olarak layout'un son hali aşağıdaki gibidir:

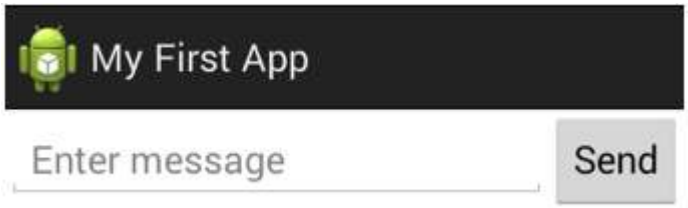
```
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:orientation="horizontal">

  <EditText
      android:id="@+id/edit_message"
      android:layout_weight="1"
      android:layout_width="0dp"
      android:layout_height="wrap_content"
      android:hint="@string/edittext_hint" />

  <Button
      android:layout_width="wrap_content"
      android:layout_height="wrap_content"
      android:text="@string/button_send" />

</LinearLayout>
```

Ekran çıktısı



EditText için, **LinearLayout** gereksiz yere genişlik hesaplaması yapar. Bunu önlemek için, yukarıdaki layout'ta görüldüğü **EditText'in** `android:layout_width` özelliği **"0dp"** yapılır. Çünkü `android:layout_width` değeri **"0dp"** olduğu zaman, genişlik hesaplaması yapılmaz. Sonuç olarak view'lerin gösterilmesi hızlanmış olur.

Not: LinearLayout'un `android:orientation` özelliği **horizontal** olduğu zaman `android:layout_width` ignore edilir. Bu özellik **vertical** olduğu zaman `android:layout_height` ignore edilir. Bunun gerçekte anlamı şudur: `android:layout_width` veya `android:layout_height` özellikleri hangi değeri alırsa alsın, **"0dp"**, "fill_parent","wrap_content", önemli değildir. Fakat tavsiye edilen değer yukarda belirtildiği gibi **"0dp"** dir.

Not: Eğer view'leri **eşit genişlikte** ayarlamak istiyorsak, `android_layout_weight` özelliğine **yüzdelik değerler** verebiliriz. Örneğin **EditText** ve **Button** için `android_layout_weight` özelliklerine **".50"** demiş olsaydık, eşit şekilde satırı paylaşırlardı. Eğer **EditText'in** kapladığı genişliğin, **Button**'un kapladığı genişliğinin **3** katı olmasını isteseydik **EditText'in** `android_layout_weight` özelliğine **"3"** değeri verip, **Button**'un `android_layout_weight` özelliğine **"1"** değerini vermemiz gerekecekti.