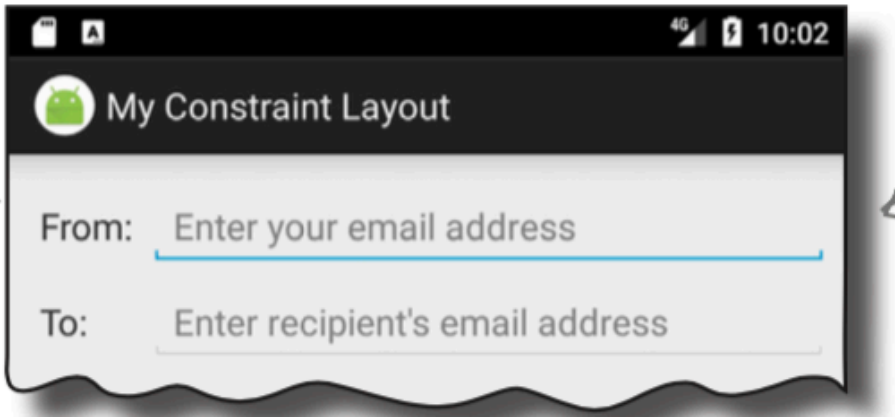
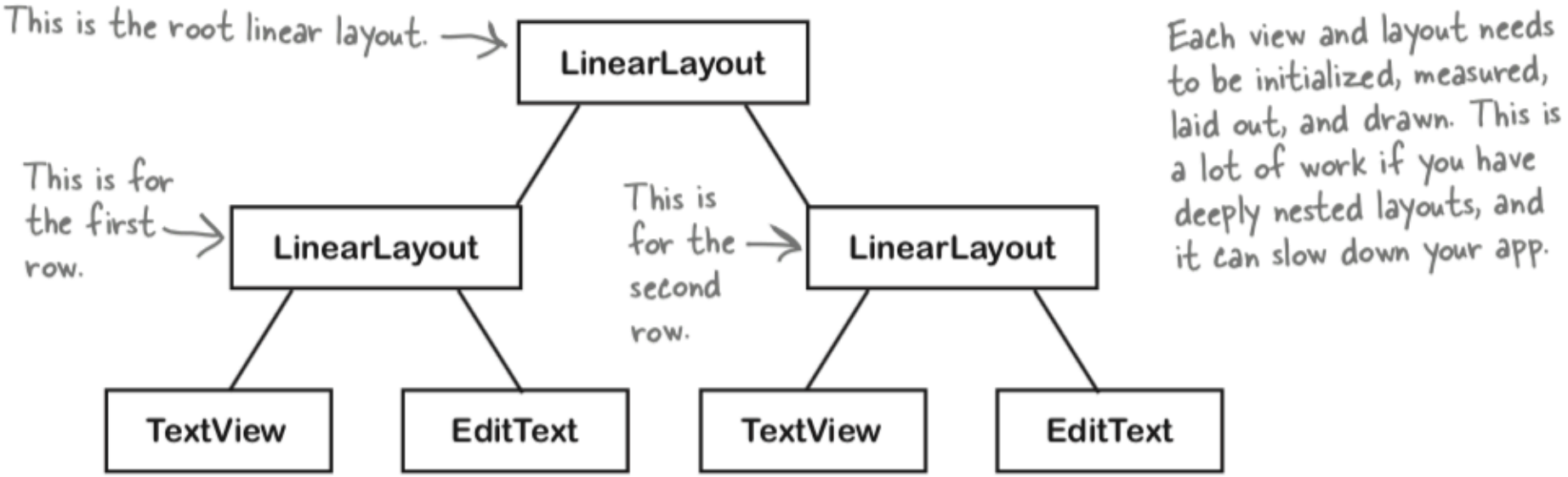
6
Diseño restringido (constraint layout)

Los diseños anidados pueden ser ineficientes

Linear layouts only allow you to display views in a single column or row, so you can't use a single linear layout to construct layouts like this. You can, however, nest linear layouts to produce the results you need.

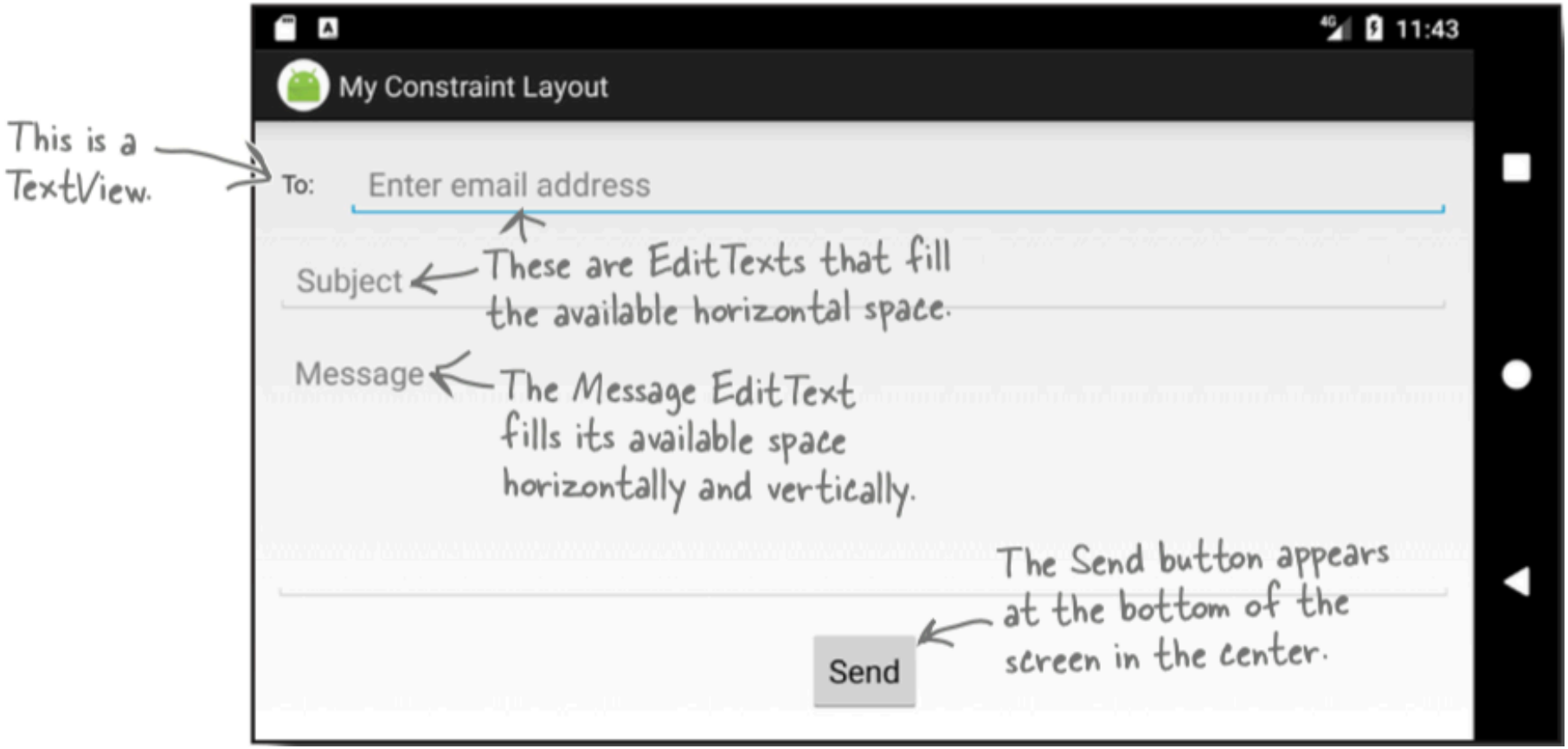


For this layout, you could use one linear layout at the root, and one nested linear layout for each row.



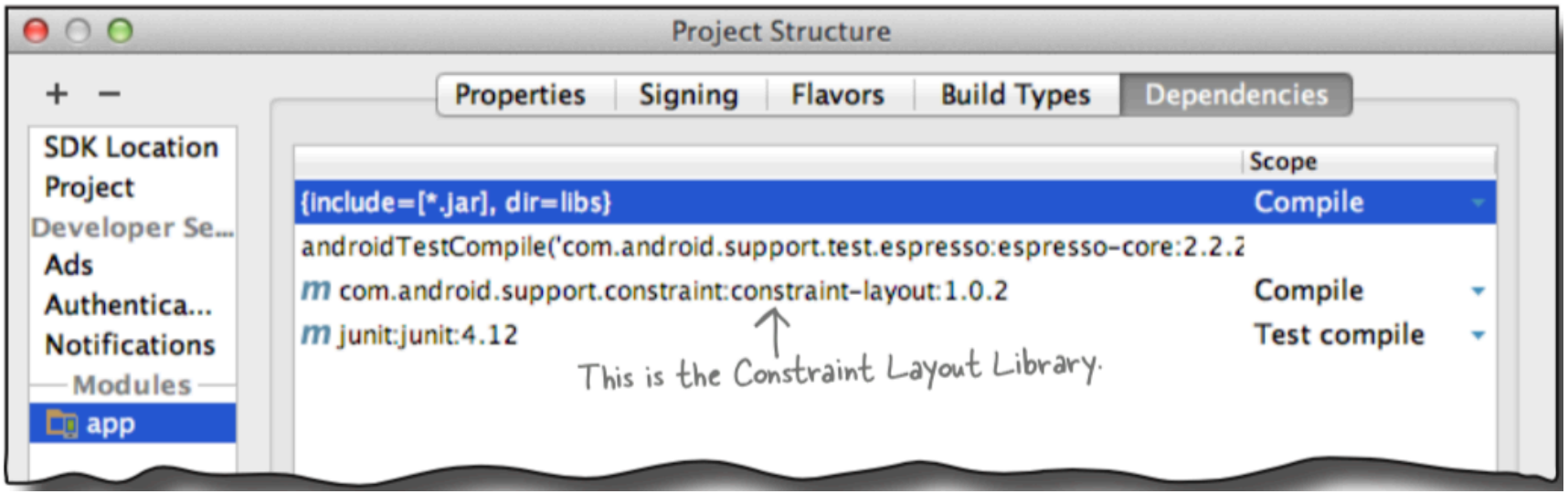
Each view and layout needs to be initialized, measured, laid out, and drawn. This is a lot of work if you have deeply nested layouts, and it can slow down your app.

Presentando el constraint layout



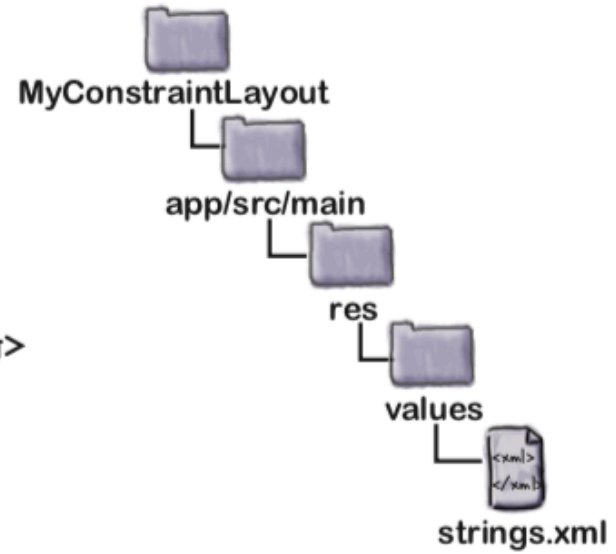
Crear un nuevo proyecto llamado “My Constraint Layout”, con la actividad principal llamada “MainActivity”, el layout llamado “activity_main” y desactivando la opción AppCompat.

Asegurarse que el proyecto incluya la biblioteca constraint layout.

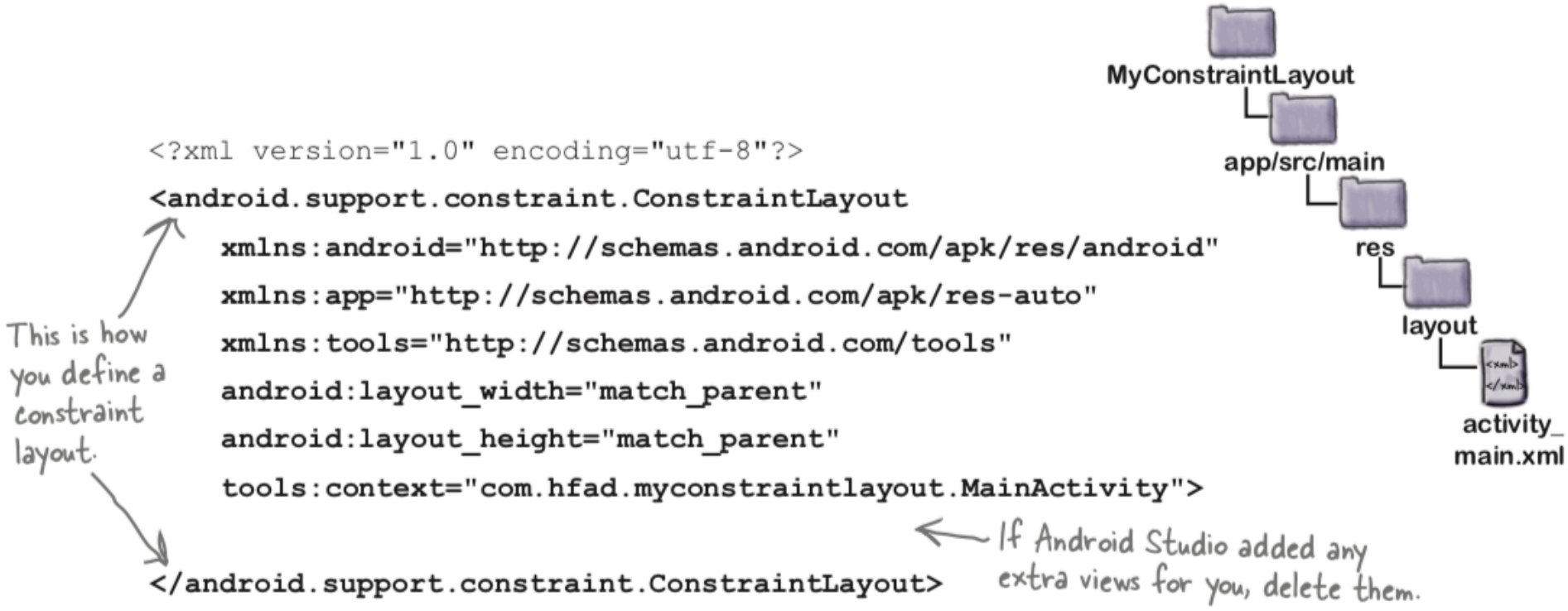


Agregar los recursos a strings.xml

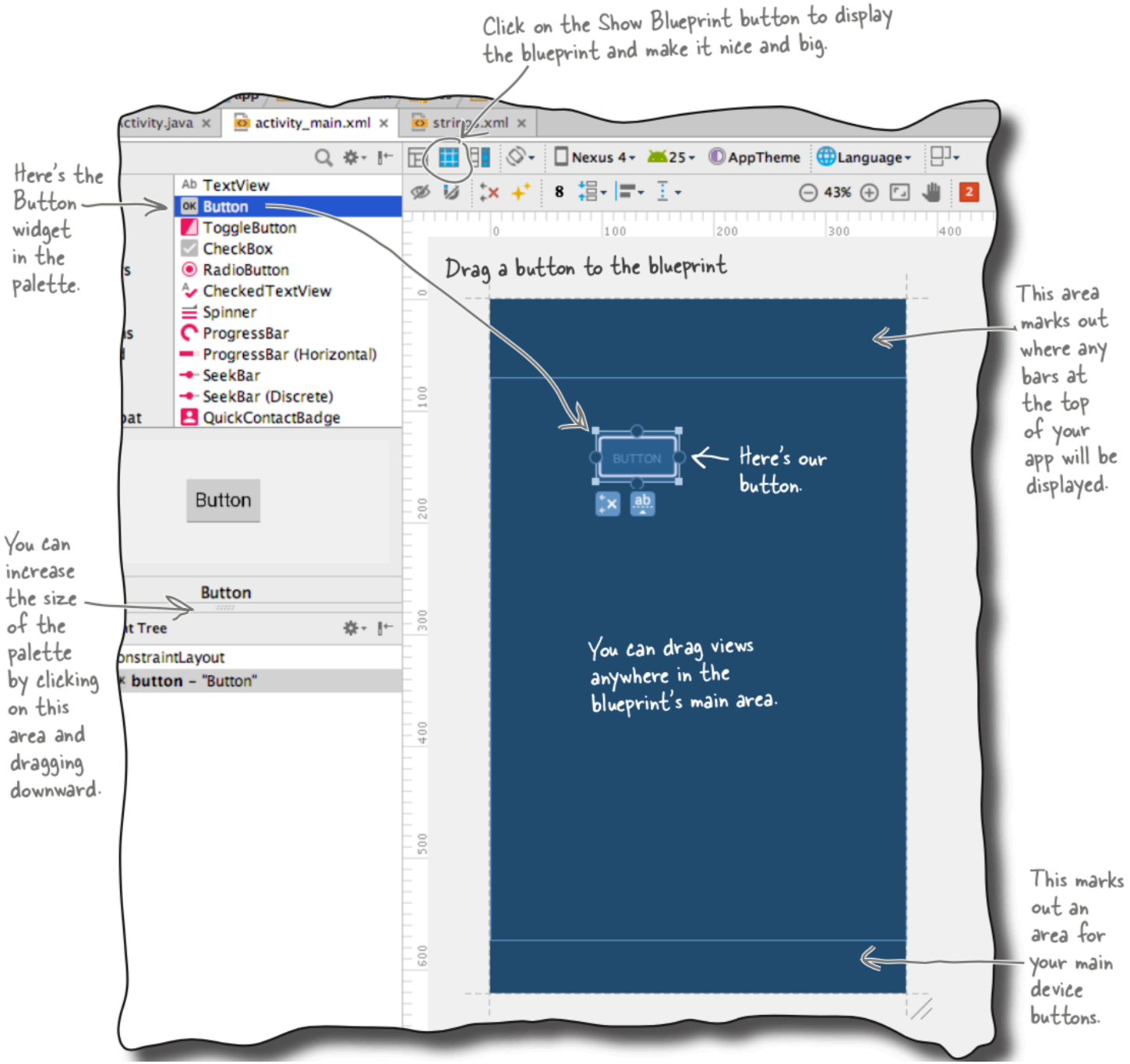
```
...
    <string name="to_label">To:</string>
    <string name="email_hint">Enter email address</string>
    <string name="subject_hint">Subject</string>
    <string name="message_hint">Message</string>
    <string name="send_button">Send</string>
...
```



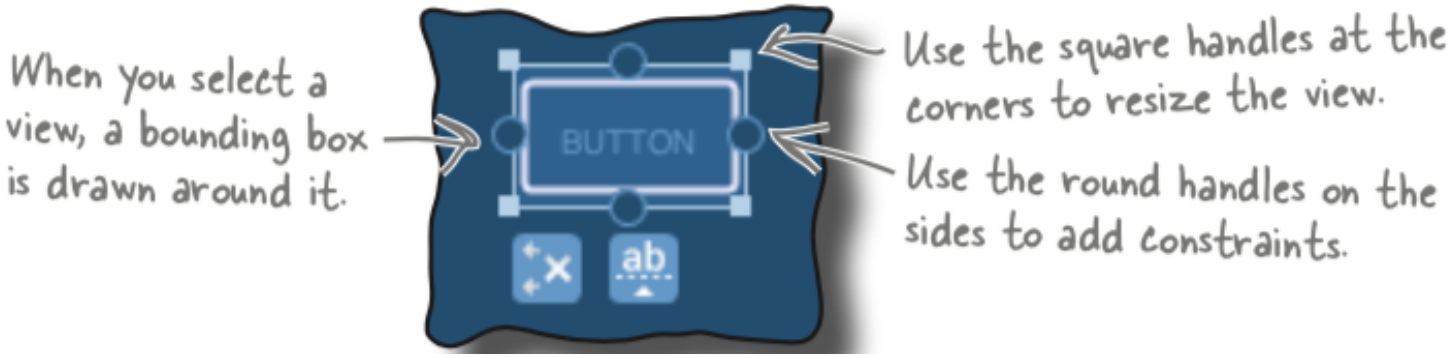
Modificando activity_main.xml para utilizar un constraint layout

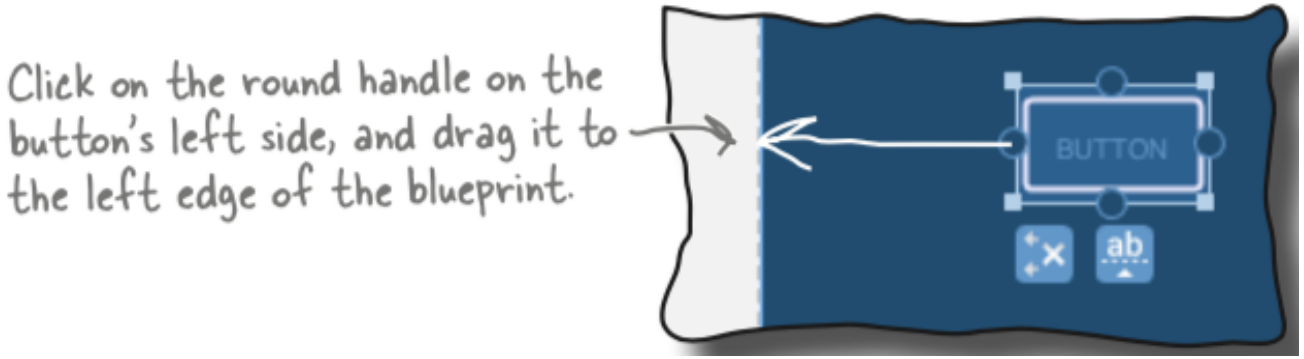


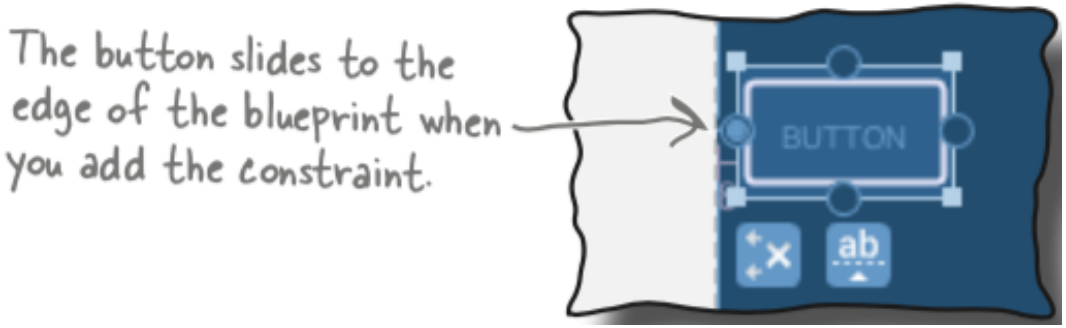
Utilizando la herramienta blueprint



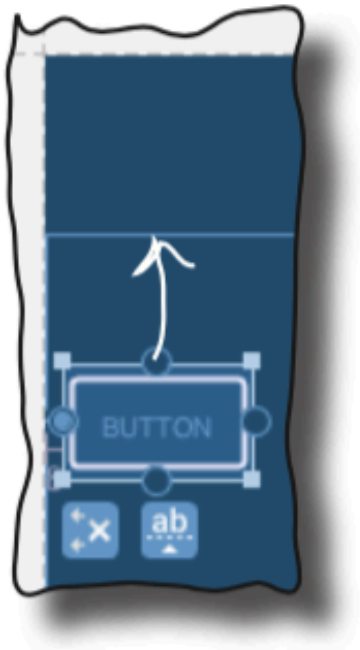
Posicionando vistas a través de restricciones (constraints)
Restricción horizontal



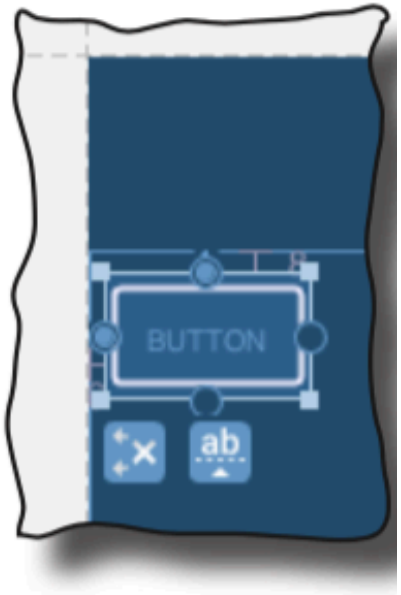




Posicionando vistas a través de restricciones (constraints)
Restricción vertical

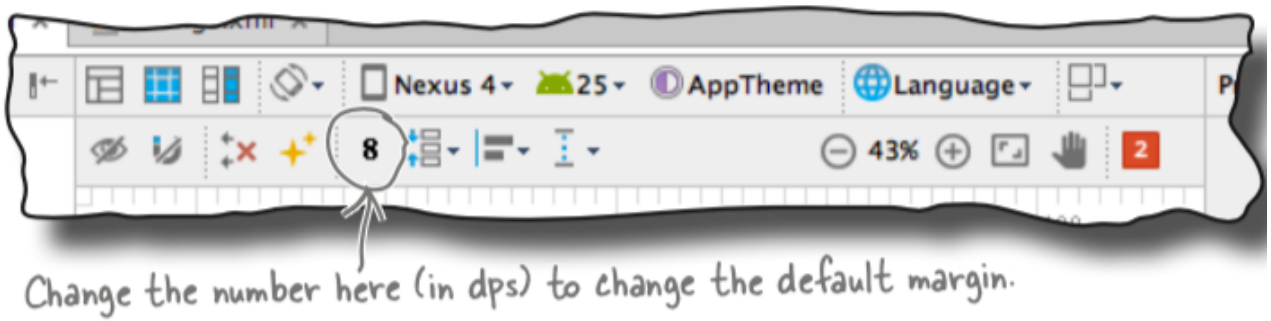


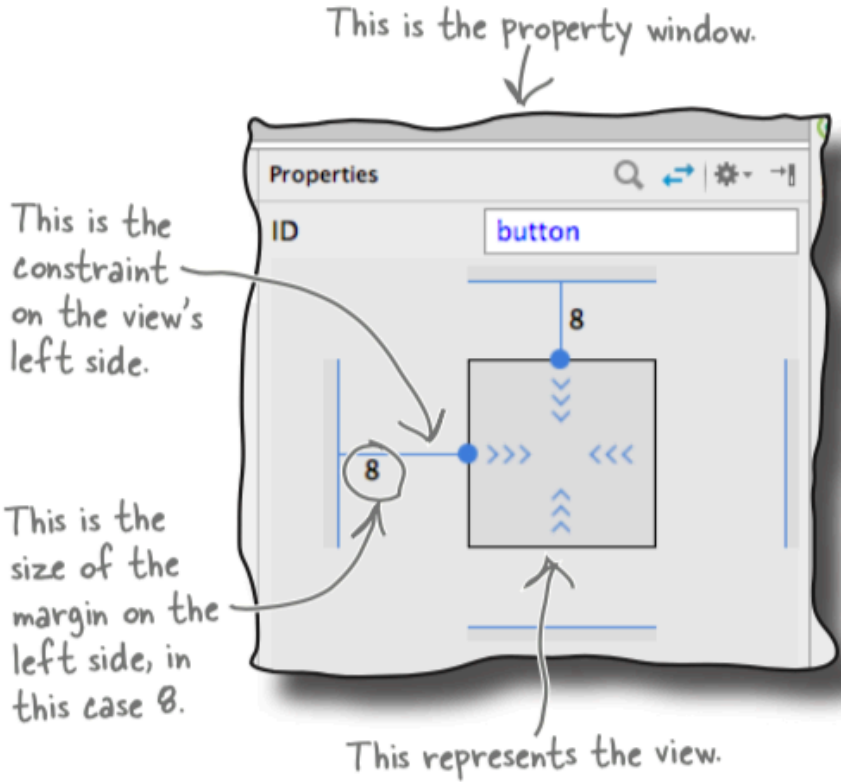
Click on the round handle on the button's top edge, and drag it to the top of the blueprint.



The button slides to the top of the blueprint's main area.

Cambiando los márgenes de las vistas





Los cambios se reflejan en XML

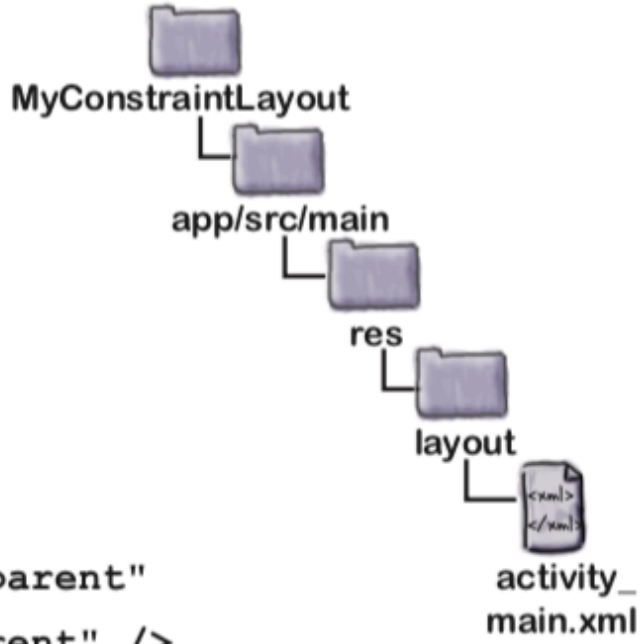
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    ...>
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginLeft="8dp"
        android:layout_marginTop="32dp"
        android:text="Button"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</android.support.constraint.ConstraintLayout>
```

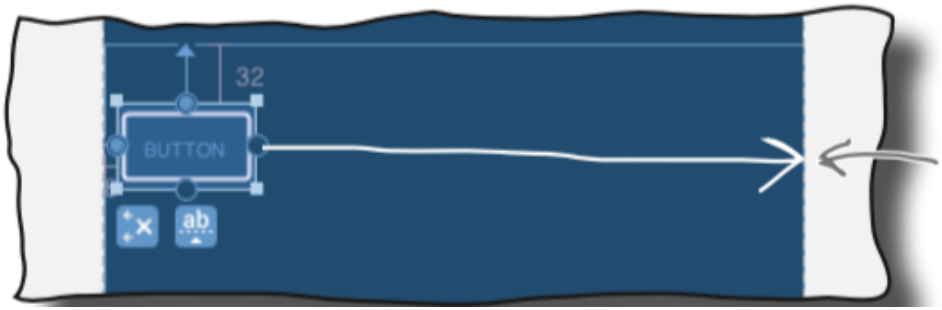
The design editor's added a button.

These are all attributes you've seen before.

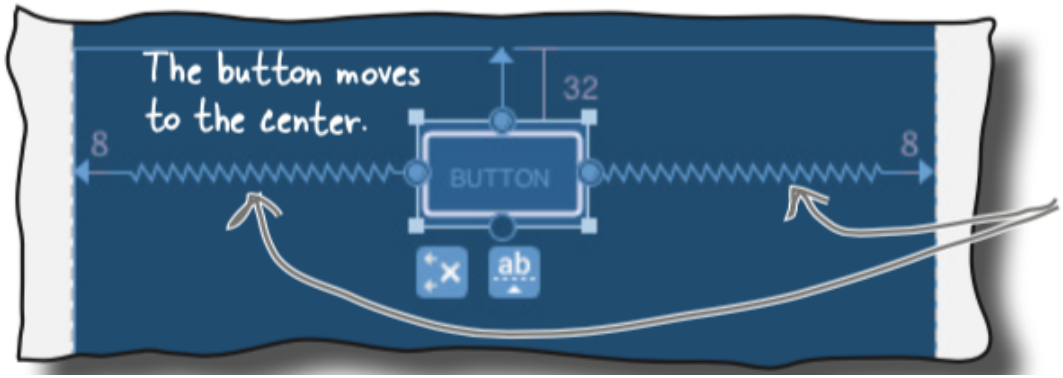
These lines only apply to constraint layouts.



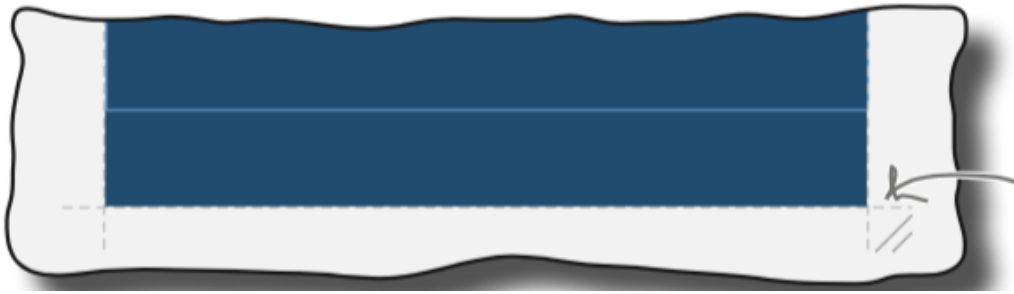
Centrando una vista



Click on the constraint handle on the button's right edge, and drag it to the right edge of the blueprint.

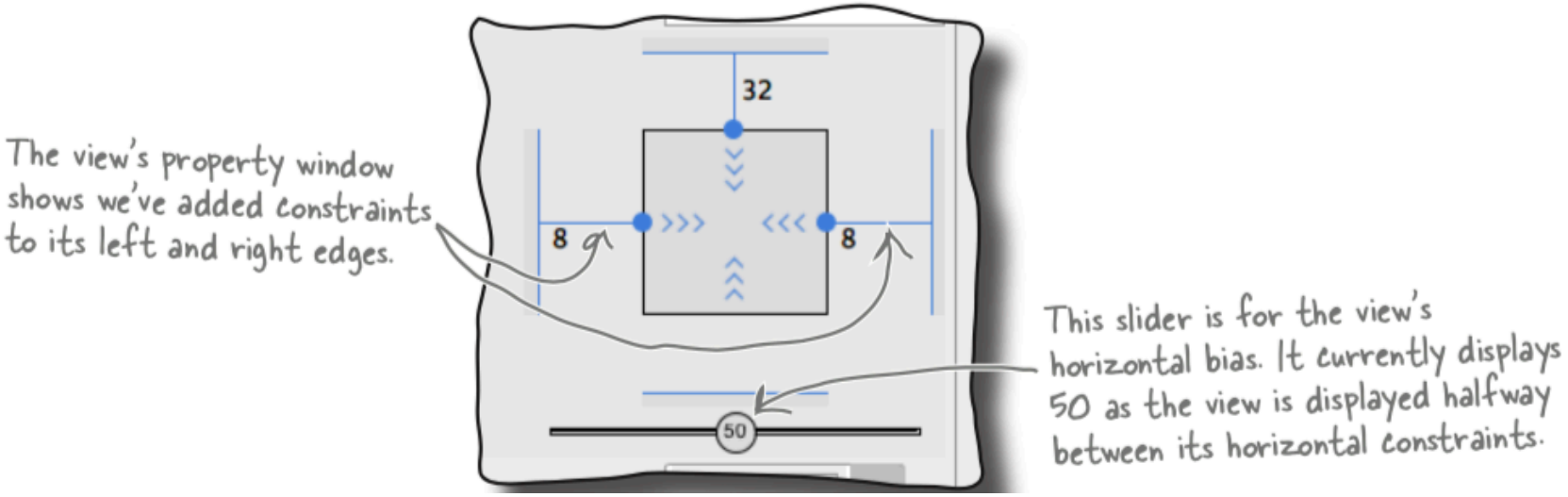


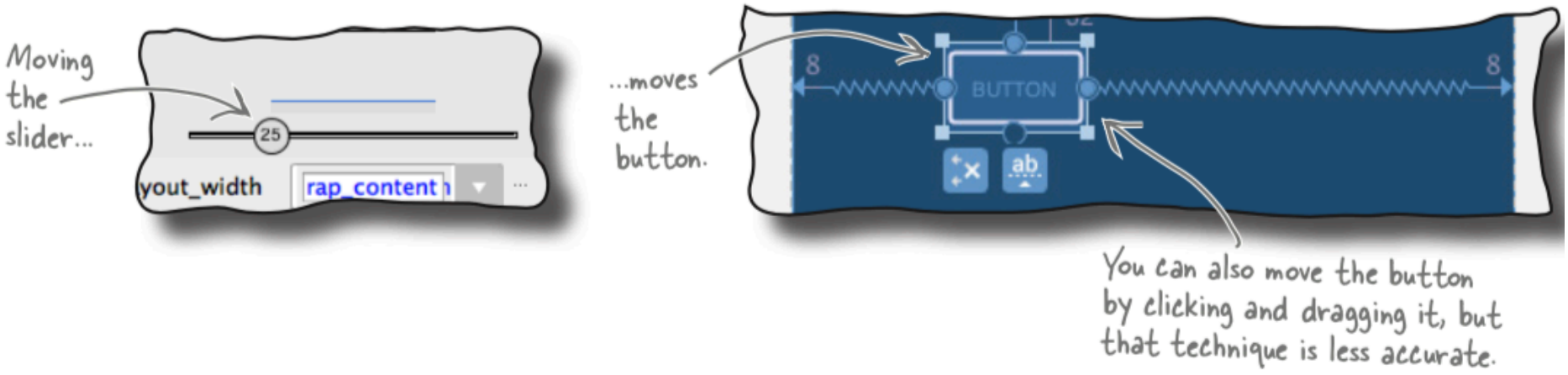
Constraints on opposite sides of a view are displayed as springs.



You can resize the blueprint by clicking and dragging its bottom-right corner.

Ajustando la posición



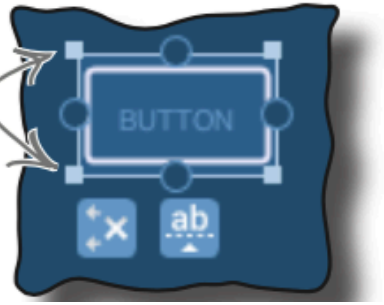


Ajustando el tamaño

1. Make the view a fixed size

There are a couple of ways of using the design editor to make the view a fixed size. One way is to simply resize the view in the blueprint by clicking and dragging the square resizing handles on its corners. The other way is to type values into the `layout_width` and `layout_height` fields in the properties window:

You can resize a view using the square resizing handles on its corners.

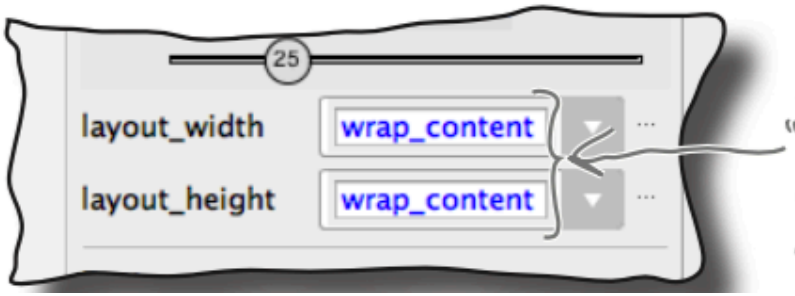


You can also hardcode the width and height in the view's property window.

Ajustando el tamaño

2. Make it just big enough

To make the view just large enough to display its contents, change the view's `layout_width` and `layout_height` properties to `wrap_content`. You do this in the view's property window as shown here:



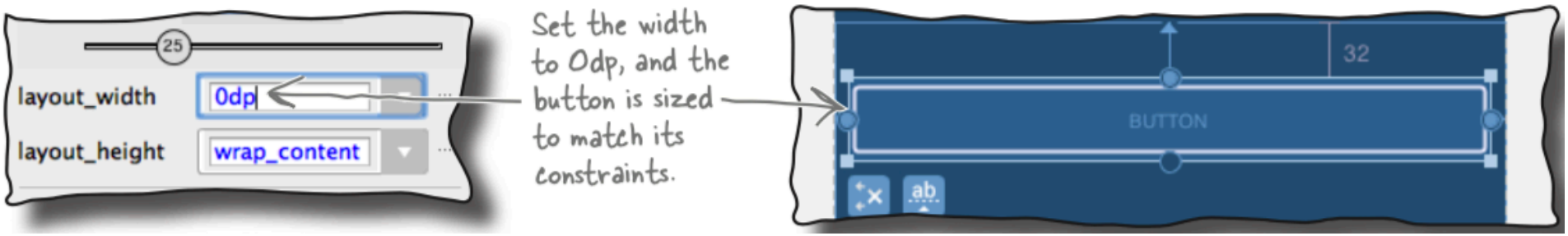
Setting the width and height to "wrap_content" makes it just large enough to display its contents, just as it does in other layouts.

Ajustando el tamaño

3. Match the view's constraints

If you've added constraints to opposite sides of your view, you can make the view as wide as its constraints. You do this by setting its width and/or height to 0dp: set its width to 0dp to get the view to match the size of its horizontal constraints, and set its height to 0dp to get it to match the size of its vertical constraints.

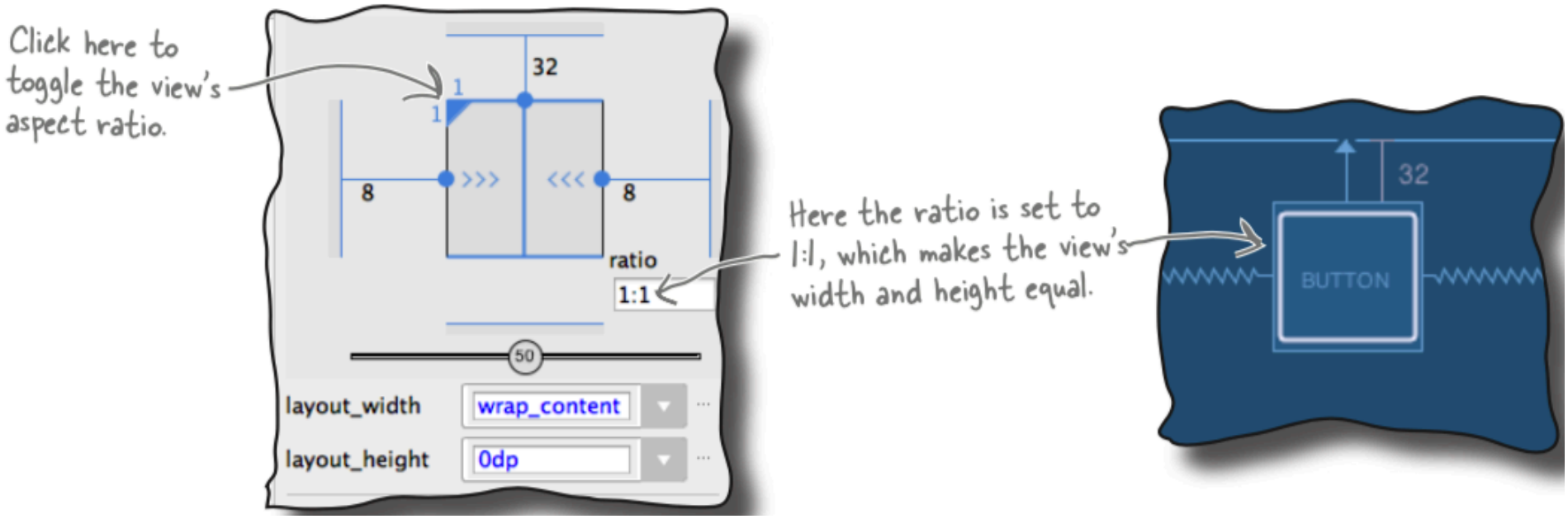
In our case, we've added constraints to the left and right sides of our button, so we can get the button to match the size of these constraints. To do this, go to the view's property window, and change the `layout_width` property to 0dp. In the blueprint, the button should expand to fill the available horizontal space (allowing for any margins):



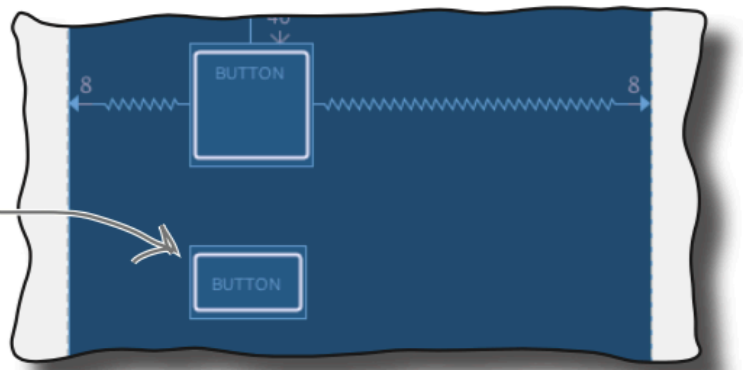
Ajustando el tamaño

4. Specify the width:height ratio

Finally, you can specify an aspect ratio for the view's width and height. To do this, change the view's `layout_width` *or* `layout_height` to 0dp as you did above, then click in the top-left corner of the view diagram that's displayed in the property window. This should display a ratio field, which you can then update:

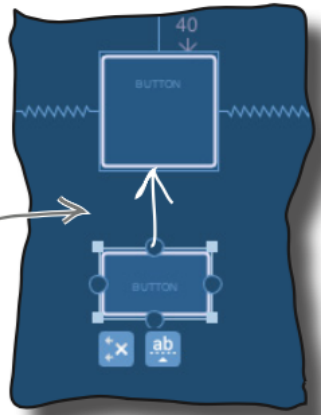


Alineando las vistas



Add a second button to the blueprint, underneath the first.

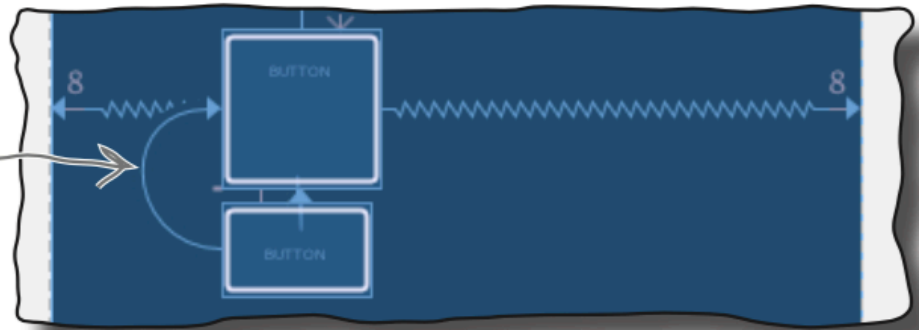
This is the Show Constraints button. Clicking on it shows (or hides) all the constraints in the layout.



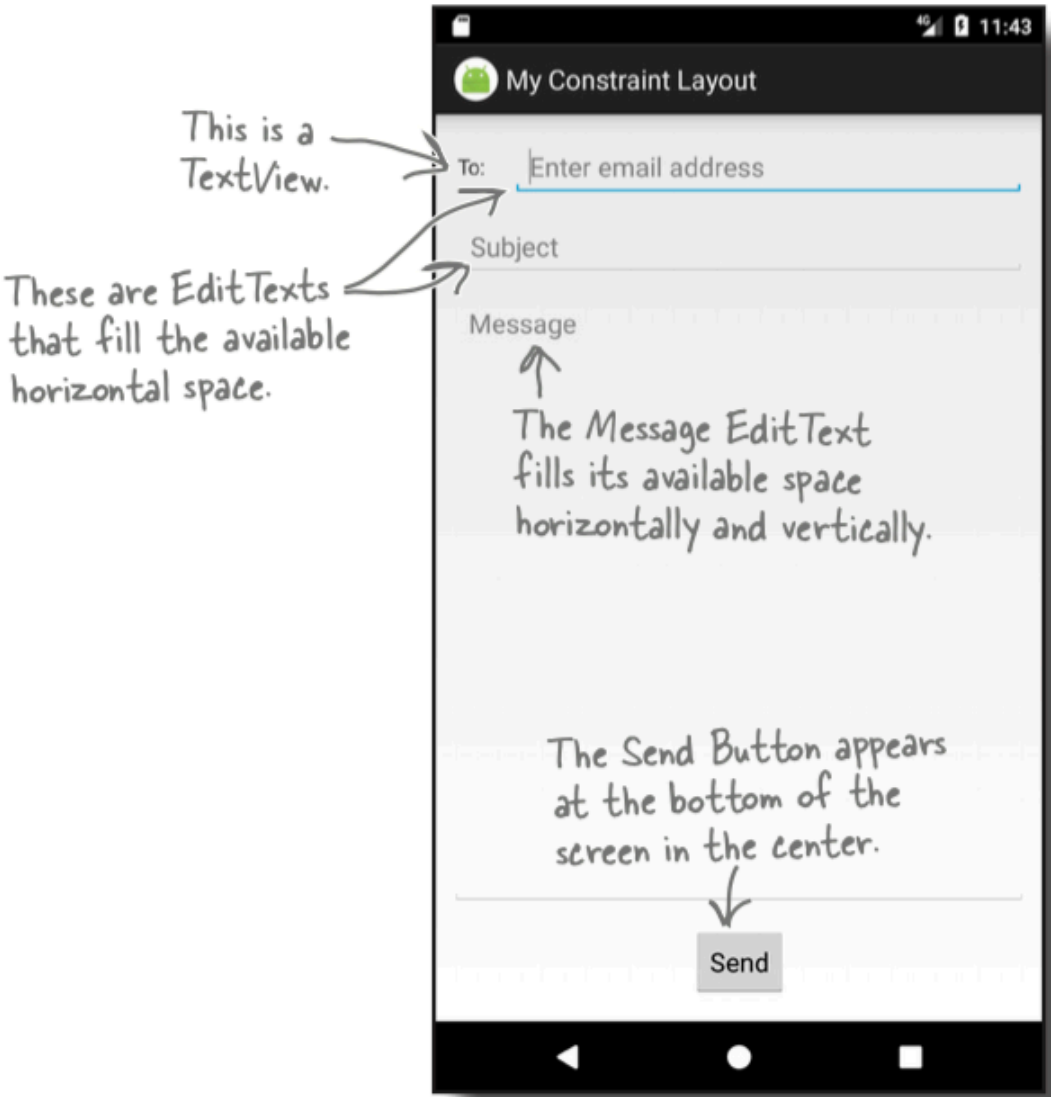
This adds a constraint attaching the top of one button to the bottom edge of the other.

Clicking on this button gives you different options for aligning views.

Aligning the view's left edges adds another constraint.



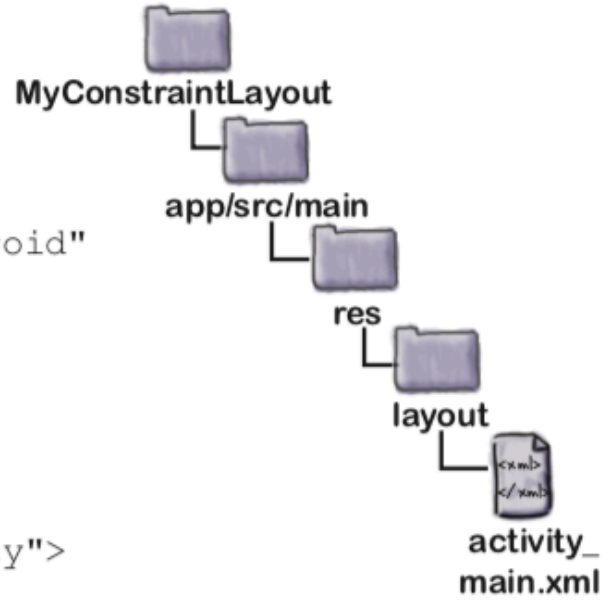
Construyendo un diseño real



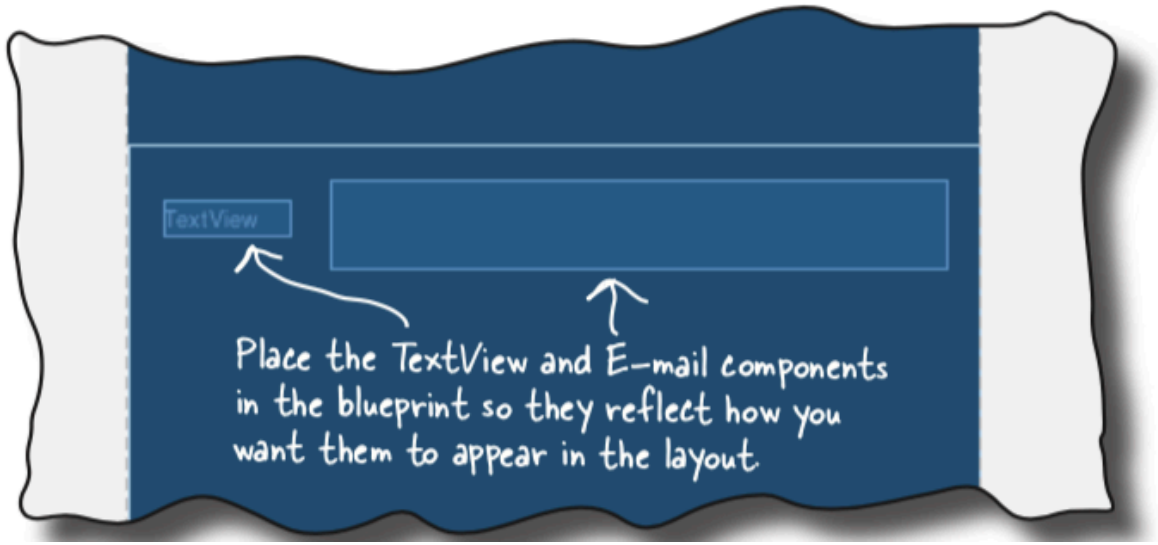
Construyendo un diseño real

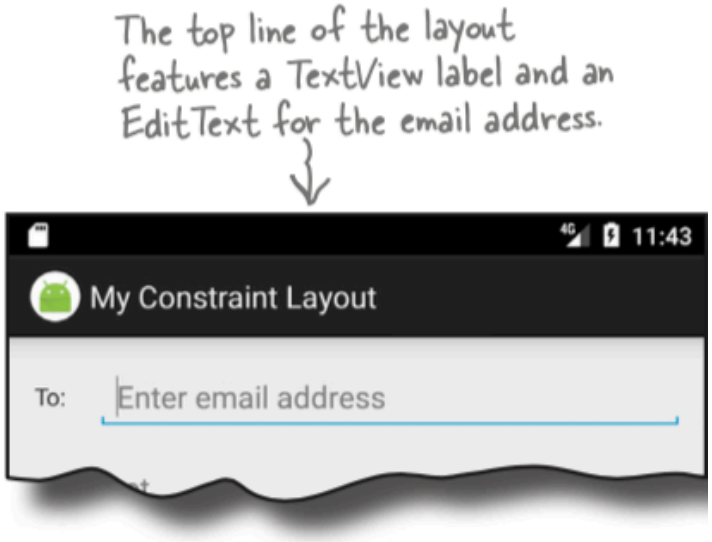
```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context="com.hfad.myconstraintlayout.MainActivity">

</android.support.constraint.ConstraintLayout>
```

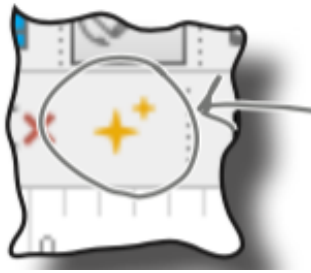


Colocando las vistas superiores



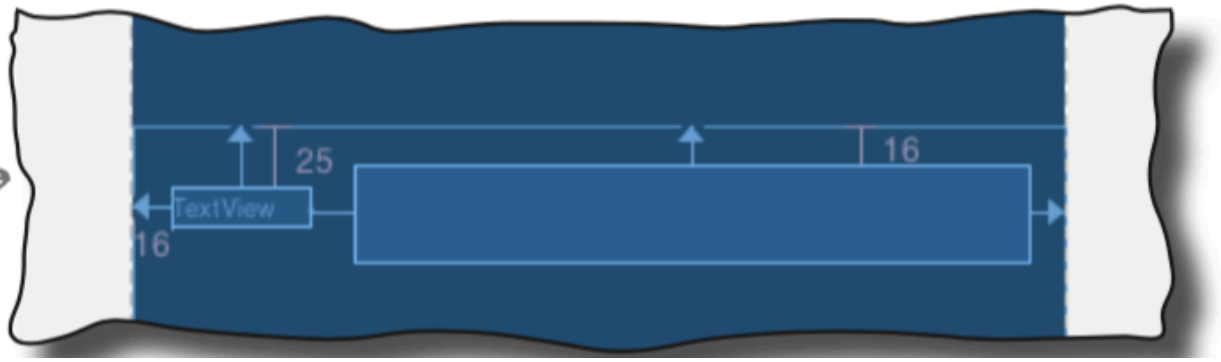


Utilizando la herramienta de inferencia



This is the Infer Constraints button. Click it now.

Clicking on the Infer Constraints button added constraints to both views.



You can check the details of each constraint by selecting each view in turn and looking at its values in the property window.

Modificando las propiedades de una vista

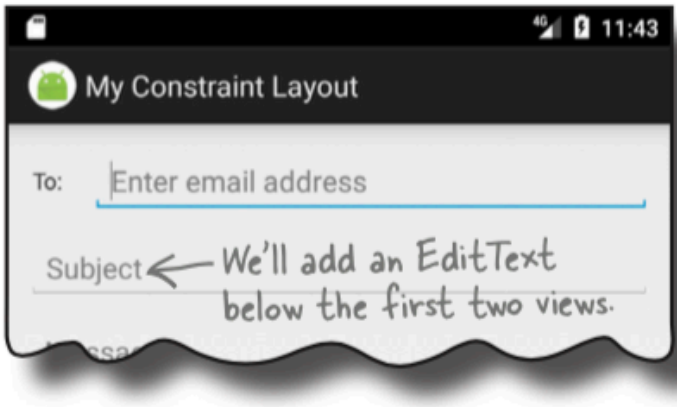
```
android:id="@+id/to_label"
android:text="@string/to_label"
```

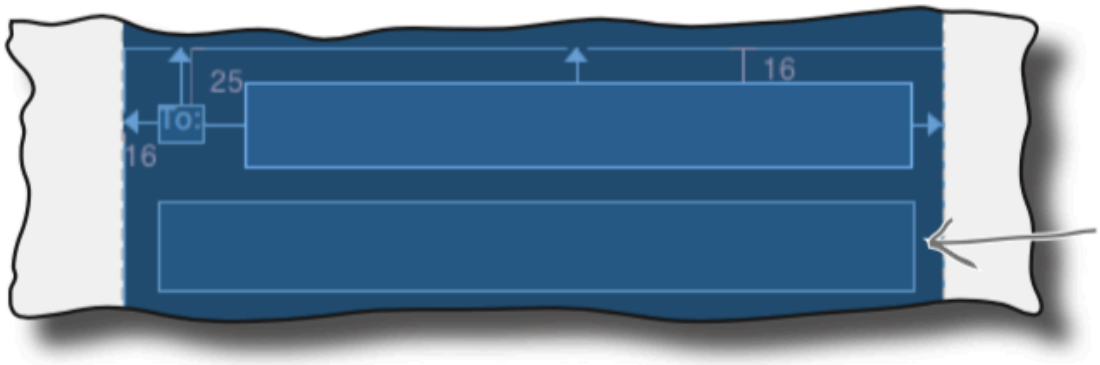
Android Studio adds these lines of code when you change the view's ID and text value.

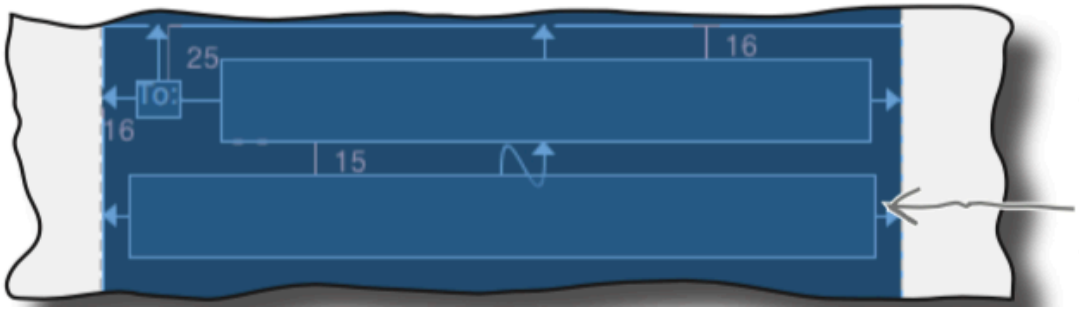

```
android:id="@+id/email_address"
android:layout_height="wrap_content"
android:hint="@string/email_hint"
```

Android Studio adds these lines of code when you change the view's layout_height and hint value.

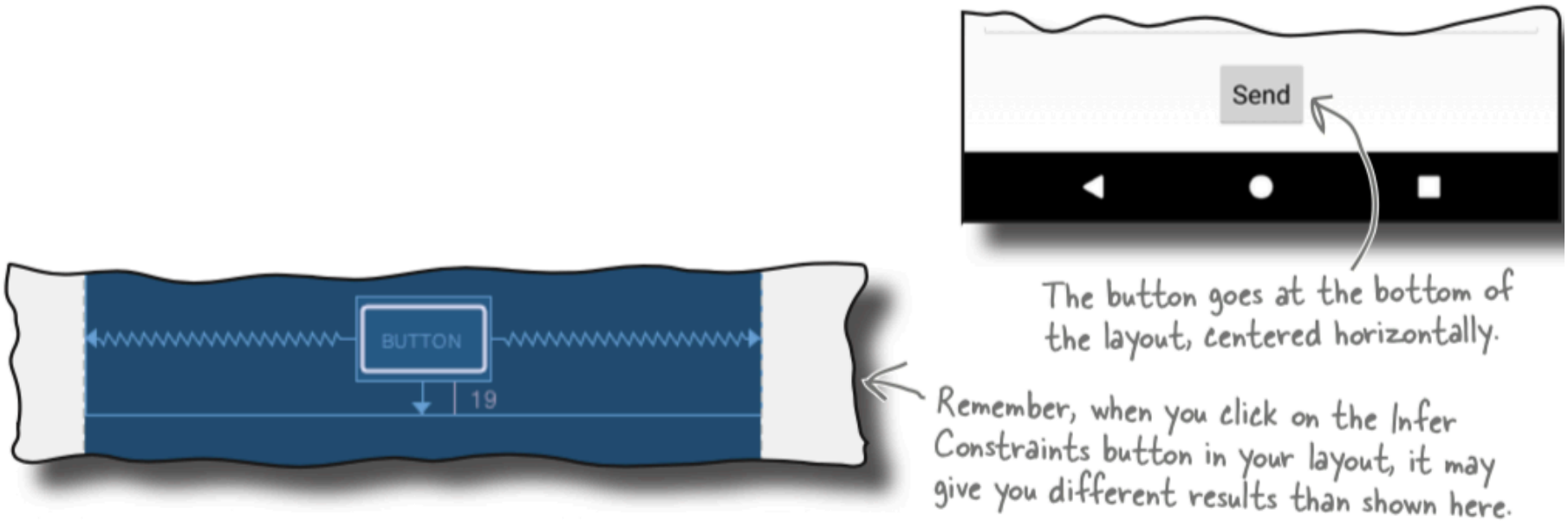
Colocando la segunda vista



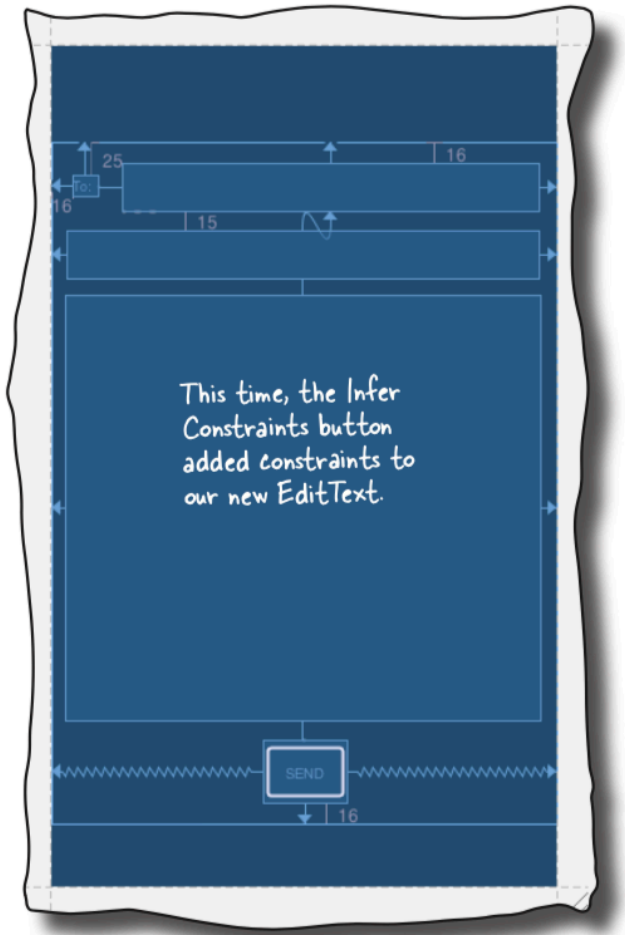




Colocando la última vista

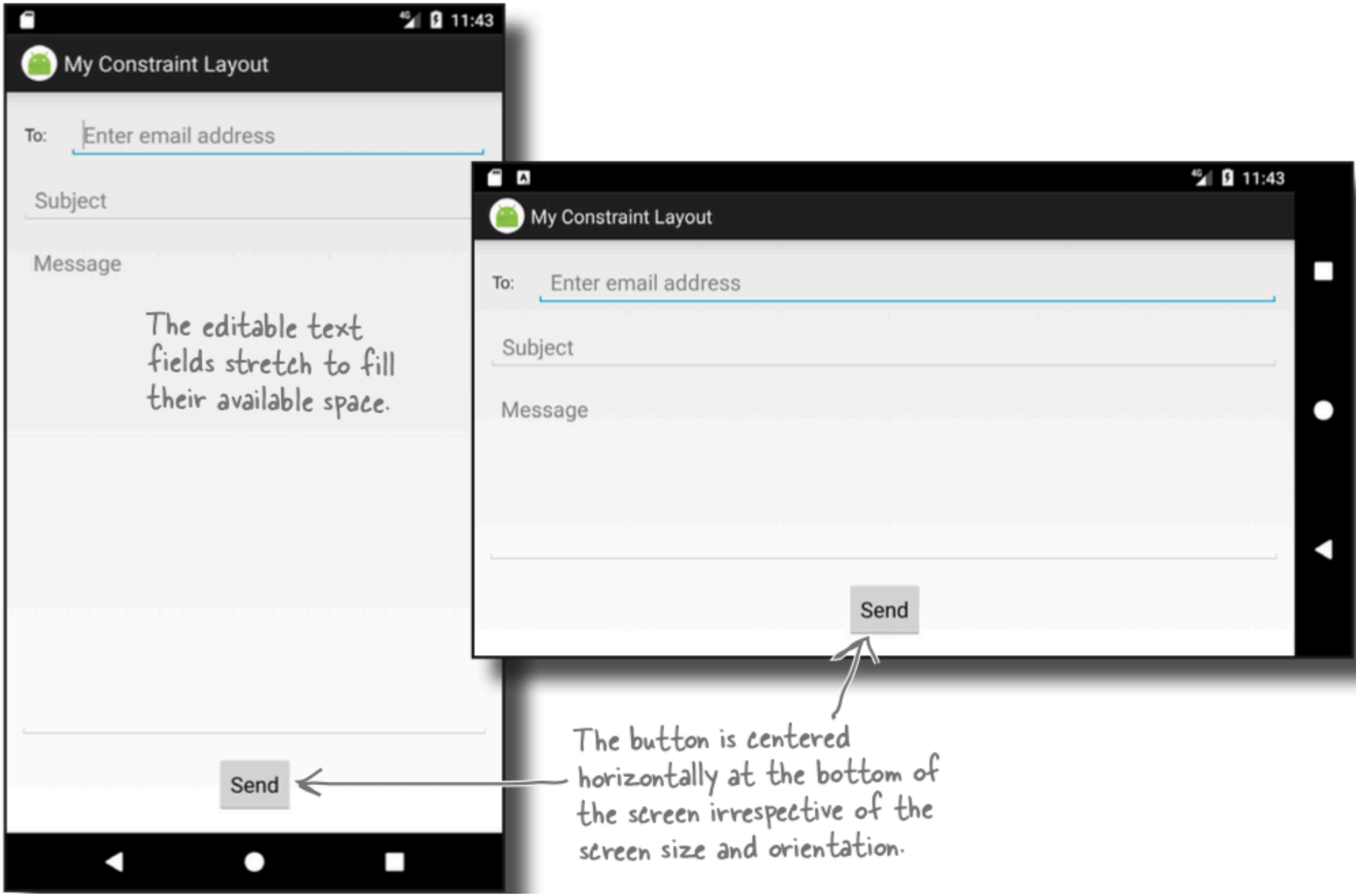


Colocando la vista del mensaje (penúltima)



Note that we could have added all these views in one go, and clicked the Infer Constraints button when we reached the end. We've found, however, that building it up step-by-step gives the best results. Why not experiment and try this out for yourself?

Prueba


11:43
My Constraint Layout
To:
Enter email address
Subject
Message
The editable text fields stretch to fill their available space.
Send
11:43
My Constraint Layout
To:
Enter email address
Subject
Message
Send
The button is centered horizontally at the bottom of the screen irrespective of the screen size and orientation.

BULLET POINTS

- Constraint layouts are designed to work with Android Studio's design editor. They have their own library and can be used in apps where the minimum SDK is API level 9 or above.
- Position views by adding constraints. Each view needs at least one horizontal and one vertical constraint.
- Center views by adding constraints to opposite sides of the view. Change the view's bias to update its position between the constraints.
- You can change a view's size to match its constraints if the view has constraints on opposing sides.
- You can specify a width:height aspect ratio for the view's size.
- Clicking on the Infer Constraints button adds constraints to views based on their position in the blueprint.