11. Pesos

El atributo `android:layout_weight` reparte el espacio sobrante entre las vistas de un contenedor. Utilizado adecuadamente, constituye una forma sencilla y elegante de adaptar la interfaz a distintos tamaños de pantalla. El valor por omisión de `android:layout_weight` es 0, lo cual indica que el espacio sobrante no se debe utilizar. Para que entiendas su significado vamos a crear un proyecto sencillo, `ccc11`, con cuatro interfaces de usuario diferentes (`activity_main_i.xml`, donde i va de 1 a 4).

Empecemos con una interfaz que muestra tres botones dispuestos horizontalmente dentro de un contenedor `LinearLayout` sin repartir el espacio sobrante, marcado en la figura con una línea horizontal:

`/res/layout/activity_main_2.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="0"
        android:text="w=0" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="0"
        android:text="w=0" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="0"
        android:text="w=0" />
</LinearLayout>
```

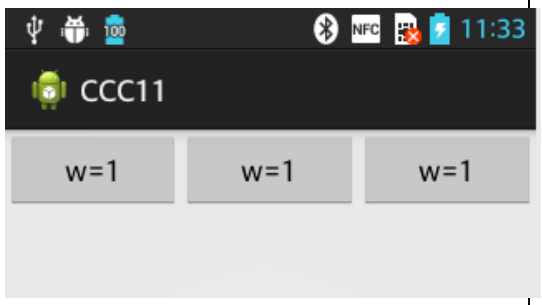
El contenedor `LinearLayout` hace dos pasadas antes de asignar la anchura de sus vistas en pantalla. Primero utiliza el atributo `layout_width`. En este caso, las vistas reciben espacio suficiente para su contenido, pues el valor asignado a este atributo es `wrap_content`. En el segundo paso, el contenedor utiliza la información del atributo `layout_weight` para repartir el espacio sobrante. Como en este caso todos los pesos valen 0, este espacio sobrante no se utiliza.

Si lo que se quiere es que el contenedor reparta el espacio de una sola vez, basándose exclusivamente en los pesos, basta con igualar la anchura de las vistas a 0 (`android:layout_width="0dp"`).

En el siguiente ejemplo, la asignación `android:layout_weight="1"` en todos los botones garantiza que el espacio sobrante se repartirá uniformemente, es decir, cada botón recibirá 1/3 del espacio sobrante. Cualquier otro valor entero o real surtirá el mismo efecto (1.0, 8, …), con tal de ser el mismo para las tres vistas. La interfaz de usuario es la siguiente:

/res/layout/activity_main_2.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
</LinearLayout>
```



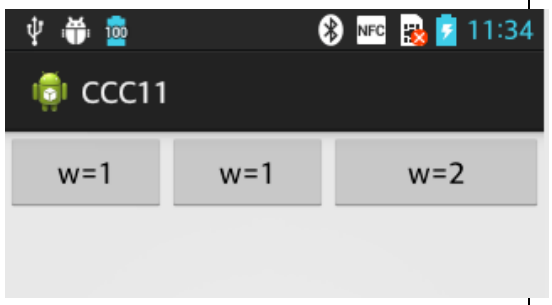
Si los dos primeros botones tienen un peso igual a 1 y el tercero igual a 2, el espacio sobrante se repartirá como ¼, ¼ y ½, respectivamente. Lo mismo se consigue asignando los pesos de la siguiente manera: (25, 25, 50). En general, ajustando el valor de los pesos, w_i, conseguimos que los elementos del contenedor rellenen el espacio libre en su interior de acuerdo con la siguiente formula:

$$\text{espacio sobrante}_i = \frac{w_i}{\sum_j w_j}$$

donde w_i es el peso del hijo número i, y el denominador es la suma de los pesos de cada hijo del contenedor. El resultado de esta ecuación es el espacio sobrante que se concede al hijo número i. De acuerdo con esta fórmula, si los pesos valen 0, no se reparte el espacio sobrante. Si todos los pesos se igualan a 1, el espacio sobrante se reparte uniformemente, 1/3 para cada botón. Los siguientes ejemplos corresponden a las distribuciones (¼, ¼, ½) y (0, ½, ½), respectivamente:

/res/layout/activity_main_3.xml

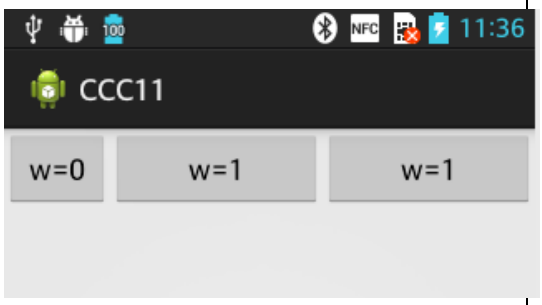
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="2"
        android:text="w=2" />
```



```
</LinearLayout>
```

/res/layout/activity_main_4.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="0"
        android:text="w=0" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:text="w=1" />
</LinearLayout>
```



El fichero Java para el proyecto de esta unidad es sumamente sencillo, reduciéndose a inflar la interfaz correspondiente a cada uno de los ficheros de diseño:

/src/MainActivity.java

```
package es.uam.eps.dadm.ccc11;

import android.app.Activity;
import android.os.Bundle;

public class MainActivity extends Activity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_1);
    }
}
```