


Open Opened 2 days ago by  **Rubén Montero**

Comenzando con... ¡Inserción de datos!

Resumen

- Añadiremos un componente `<EditText>` a la aplicación
- Haremos `git add`, `git commit` y `git push` para subir los cambios al repositorio

Descripción

Desbloquear la posibilidad de consumir datos de un [API REST](#) y presentarlos al usuario mediante un `RecyclerView` es un logro notable. Pero no es el pastel completo. Abre Twitter, Facebook, Reddit, WhatsApp, Telegram, Instagram, Pinterest, TikTok,... Además de *consumir* datos, permiten *subir* datos.

Nosotros vamos a comenzar implementando una pantalla de *registro*. Así, los usuarios podrán *registrarse* en nuestra aplicación. Para ello, enviaremos una petición `POST` a un *endpoint* del API REST. Los datos *enviados por el cliente* en la petición (nuevo nombre de usuario, nueva contraseña) se guardarán en una base de datos en el *backend*, para su futuro uso.

En esta primera tarea, tan sólo aprenderemos a usar un [EditText](#). Es el componente visual que la plataforma Android nos ofrece para permitir al usuario *introducir* texto.

La tarea

Como primer paso, **abre** un terminal (cmd.exe) en tu ordenador y cambia el directorio activo hasta la ubicación de tu repositorio usando `cd`. Luego, **haz**:

```
git pull
```

...para traer los cambios de remoto a local, que contienen el *esqueleto* del proyecto de este *sprint*.

Abre el proyecto `android-sessions/android-frontend-sessions` desde Android Studio (*no abras `android-sessions` del nivel superior*).

Después, **abre** el fichero de interfaz `activity_register.xml`. Dentro, **elimina** el `"Hello world!"` por defecto:

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.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=".RegisterActivity">
-
-   <TextView
-       android:layout_width="wrap_content"
-       android:layout_height="wrap_content"
-       android:text="Hello World!"
-       app:layout_constraintBottom_toBottomOf="parent"
-       app:layout_constraintEnd_toEndOf="parent"
-       app:layout_constraintStart_toStartOf="parent"
-       app:layout_constraintTop_toTopOf="parent" />
-
</androidx.constraintlayout.widget.ConstraintLayout>
```

Luego, **añade** un `<EditText />` dentro del `<ConstraintLayout>` en `activity_register.xml`. Estará *centrado* vertical y horizontalmente. **Así**:

```
<EditText
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent" />
```

Lanza la aplicación en el emulador. ¿Qué tal se ve?

Seguramente observes que el `<EditText>` aparece *comprimido*. Esto se debe a que `layout_width="wrap_content"` hace que el *ancho* se ajuste a... ¡Nada! No hay texto. Por lo tanto, se ve prácticamente *colapsado*.

Podríamos usar una medida en `dp` estática, como `50dp`, `80dp`, `120dp` ... Pero vamos a aprender [cómo dimensionar un elemento proporcionalmente con respecto a su contenedor](#).

Por ello, **modifica** `layout_width` para tener un tamaño fijo (`0dp`) y **añade** el atributo `app:layout_constraintWidth_percent` como se indica:

```
<EditText
    android:layout_width="0dp"
-    android:layout_width="wrap_content"
+    android:layout_width="0dp"
+    app:layout_constraintWidth_percent="0.6"
    app:layout_constraintTop_toTopOf="parent"
    app:layout_constraintBottom_toBottomOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintEnd_toEndOf="parent"/>
```

(Así conseguimos que `<EditText>` tenga un ancho equivalente al 60% del padre).

hint

¿Como sabe el usuario *qué* introducir en el `<EditText>` ?

Una poderosa herramienta es el atributo `android:hint`. Esto mostrará un texto que aparecerá dentro del `<EditText>`, en gris, y se *esfumará* en cuanto el usuario empiece a teclear.

Añade el siguiente `hint` a tu código `<TextView>` y verifica cómo se ve.

```
android:hint="charlie.doe"
```

Por último

Desde el terminal de Windows (cmd.exe), nos posicionamos en `android-sessions/` y hacemos `git add *`, `git commit` y `git push`.

No está de más visitar la página de GitLab (<https://raspi>) y verificar que el *commit* se ha subido.



Rubén Montero @ruben.montero changed milestone to [%Sprint 3](#) 2 days ago