


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

# Petición GET y SharedPreferences

## Resumen

- Lanzaremos una petición GET en `StatusActivity.java` para obtener el 'Estado' del usuario
- Usaremos las `SharedPreferences` para *persistir* el nombre del usuario que inició sesión y usarlo en la petición
- Añadiremos una `LauncherActivity` para que se inicie `LoginActivity` ó `StatusActivity`, dependiendo de si el usuario ya se ha *logueado* con anterioridad
- No nos desanimaremos a pesar de que, al final de la tarea, todavía no funcionará nuestra petición de 'Estado'

## Descripción

Hemos mencionado que el propósito de esta aplicación es que el usuario *almacene* un 'Estado'.

- Sólo podrá tener *un* 'Estado'.
- Podrá grabar *un nuevo* 'Estado', pero al hacerlo, reemplazará el anterior.

Para ello, se lanzan peticiones HTTP a estos dos *endpoints*:

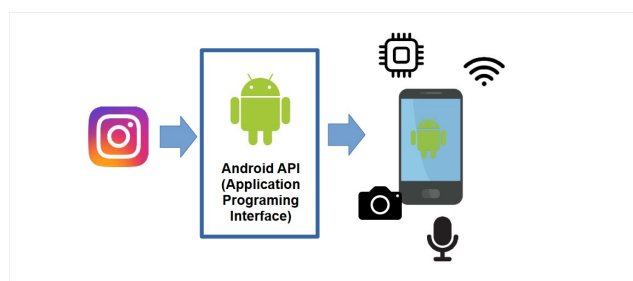
- GET `http://raspi:8000/users/<nombreUsuario>/status` (recuperar 'Estado' actual)
- PUT `http://raspi:8000/users/<nombreUsuario>/status` (registrar nuevo 'Estado')

Por ejemplo, si queremos *recuperar* el estado de `john.doe`, mandaremos un GET a `/users/john.doe/status`

En esta tarea trabajaremos en ello, y de paso nos familiarizaremos con un nuevo recurso del API de Android: Las [SharedPreferences](#).

## API de Android y SharedPreferences

¡El API de Android lo hemos estado usando todo el rato! Se trata del conjunto de clases y métodos que están a nuestra disposición y sirven de intermediarios entre nuestra *app* y los recursos del teléfono, como *cámara, micrófono, Wi-Fi*...



Con [SharedPreferences](#) puedes acceder al *almacenamiento secundario*. Es decir, la tarjeta SD ó memoria interna del teléfono donde los datos *no* se eliminan de manera *volátil*.

¡Vamos a trabajar en ello!

## La tarea

En `StatusActivity.java`, **crea** un atributo `RequestQueue` e **inicialízalo**. También, **añade** un método `retrieveStatus` que será invocado al final de `onCreate`. Así:

```
public class StatusActivity extends AppCompatActivity {
    private Context context = this;
    private RequestQueue queue;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_status);
        // Inicializamos la cola de peticiones y preparamos la petición inicial
        queue = Volley.newRequestQueue(this);
        retrieveStatus();
    }
}
```

```
private void retrieveStatus() {
    // Mandaremos la petición GET

}
}
```

Ahora, podríamos ponernos a trabajar en `retrieveStatus`, donde enviamos la petición `GET` mencionada arriba:

```
private void retrieveStatus() {
    // Mandaremos la petición GET
    JsonObjectRequest request = new JsonObjectRequest(
        Request.Method.GET,
        Server.name + "/users/" + "???" + "/status" // ?????? qué pongo aquí (?)

    )
}
```

¡Ups! Parece que hay un problema.

## La URL debe contener el *username*

Sí, así es.

## Hay que sacarlo de algún sitio

Exacto.

¡El usuario introdujo su *username* en `LoginActivity.java` !

¡Correcto!

## Guardando el *username* tras hacer *Login* en `LoginActivity.java`

Vamos a usar las [SharedPreferences](#) para persistir el *username*.

Primero, en `LoginActivity.java`, después de un *Login* exitoso ( `onResponse` ), **almacena** el nombre de usuario que se *logueó*. **Empieza** *instanciando* unas `SharedPreferences` así:

```
new Response.Listener<JSONObject>() {
    @Override
    public void onResponse(JSONObject response) {
        String receivedToken;
        // Aquí tendremos el código asociado a:
        // (1) Mostrar un Toast con el Token de sesión
        // (2) Iniciar la StatusActivity
        /* ... */

        // Instanciamos un objeto de tipo SharedPreferences
        // En el constructor pasamos un String. SIEMPRE será el mismo para nuestra aplicación.
        SharedPreferences preferences = context.getSharedPreferences("SESSIONS_APP_PREFS", MODE_PRIVATE);
    }
}
```

El objeto `SharedPreferences` es suficiente para *recuperar* valores, pero para *almacenarlos* necesitamos usar una clase distinta. Lleva el sufijo `.Editor`.

**Añade** la siguiente línea después de la indicada arriba:

```
SharedPreferences.Editor editor = preferences.edit();
```

Después de esto, basta con invocar `.putXXX` para guardar un dato concreto.

**Invoca** `putString` y **usa** la clave `"VALID_USERNAME"`. Como valor, **guarda** el contenido del `EditText` donde el usuario tecleó su *username*:

```
editor.putString("VALID_USERNAME", /* .getText().toString() sobre tu instancia de editText para el nombre de usu
```

Y finalmente, **invoca** `.commit()` (e **invoca** `finish()`, como se indica):

```
// .apply() es preferible y cuando se usa, el sistema almacena
// los datos en segundo plano de forma más lenta
```

```
// .commit() es bloqueante y lo almacena inmediatamente. Debe
// usarse sólo si es crítico que los datos se guarden de forma
// instantánea. ¡En nuestro caso es así!
editor.commit();

// Un detalle extra:
// Añadamos 'finish()' para que, tras un Login exitoso, la
// LoginActivity se termine. Así evitamos que al pulsar Atrás
// el usuario regrese a dicha LoginActivity.
// ¡Sería confuso!
finish();
```

## Volviendo a `StatusActivity.java`

¡Ahora estamos listos para mandar la petición a la URL correcta!

En `StatusActivity.java`, instancia un `JsonObjectRequest` con los parámetros correctos en `retrieveStatus` y añádelo a la cola de peticiones.

Muestra un `Toast` de éxito y uno de error como se expone a continuación:

```
private void retrieveStatus() {
    // Recuperamos el nombre de usuario de las preferencias
    SharedPreferences preferences = getSharedPreferences("SESSIONS_APP_PREFS", MODE_PRIVATE);
    String username = preferences.getString("VALID_USERNAME", null); // null será el valor por defecto
    // Mandaremos la petición GET
    JsonObjectRequest request = new JsonObjectRequest(
        Request.Method.GET,
        Server.name + "/users/" + username + "/status",
        null,
        new Response.Listener<JsonObject>() {
            @Override
            public void onResponse(JsonObject response) {
                Toast.makeText(context, "Estado obtenido", Toast.LENGTH_LONG).show();
            }
        },
        new Response.ErrorListener() {
            @Override
            public void onErrorResponse(VolleyError error) {
                Toast.makeText(context, "Problema recibiendo el estado", Toast.LENGTH_LONG).show();
            }
        }
    );
    queue.add(request);
}
```

Antes de verificar que funciona, dejémonos llevar por la emoción y trabajemos en una `LauncherActivity`.

## LauncherActivity

Declara en `AndroidManifest.xml` una `LauncherActivity`, e indica que es la nueva `LAUNCHER` y `MAIN`:

```
+ <activity android:name=".LauncherActivity"
+     android:exported="true"
+     android:noHistory="true"> <!-- con noHistory evitamos que, al pulsar Atrás, el usuario regrese a esta actividad -->
+     <intent-filter>
+         <action android:name="android.intent.action.MAIN" />
+
+         <category android:name="android.intent.category.LAUNCHER" />
+     </intent-filter>
+ </activity>
+ <activity
+     android:name=".LoginActivity"
+     android:exported="true"> <!-- exported debe indicarse como "true" para actividades LAUNCHER desde Android 12 -->
-     <intent-filter>
-         <action android:name="android.intent.action.MAIN" />
-
-         <category android:name="android.intent.category.LAUNCHER" />
-     </intent-filter>
- </activity>
```

Luego, **crea** la clase `LauncherActivity.java` y **provéela** del siguiente código:

```
public class LauncherActivity extends AppCompatActivity {

    @Override
    protected void onCreate(@Nullable Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        // Recuperamos el nombre de usuario de las preferencias
        SharedPreferences preferences = getSharedPreferences("SESSIONS_APP_PREFS", MODE_PRIVATE);
        String username = preferences.getString("VALID_USERNAME", null); // null será el valor por defecto

        // Si el usuario NO se ha logueado, el valor es 'null' por defecto
        // ¡Vamos a iniciar la pantalla de Login!
        if (username == null) {
            Intent loginActivity = new Intent(this, LoginActivity.class);
            startActivity(loginActivity);

            // Si el usuario SÍ se ha logueado, ya disponemos de su nombre de usuario
            // ¡Vamos a iniciar la pantalla principal!
        } else {
            Intent statusActivity = new Intent(this, StatusActivity.class);
            startActivity(statusActivity);
        }
    }
}
```

Presta atención a este código y compréndelo. ¡Sirve para que la *app* vaya a la pantalla correcta tras iniciarse!

Si lanzas tu *app*, verás que aparece la `StatusActivity` en caso de que ya te hayas *logueado*.

Puedes *borrar* las `SharedPreferences` yendo, en el dispositivo, a *Settings > Apps & Notifications > Sessions > Storage > **Clear Storage***. ¡Es como cerrar la sesión!

Tras esto, puedes *abrir* de nuevo la aplicación y verás la `LoginActivity`.

## ¡Probémoslo!

Prueba a lanzar la *app* y hacer *Login*. Verás que aparece el Toast `"Problema recibiendo el estado"`

Parece que la petición `GET` para obtener el 'Estado', ¡falla!

Lo arreglaremos en la siguiente tarea.

## Por último

Sube tus cambios al repositorio en un nuevo *commit*.



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