


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

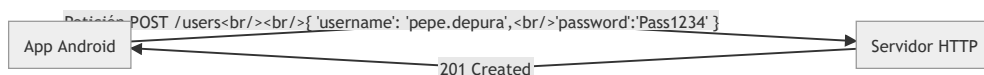
Petición POST de inicio de sesión

Resumen

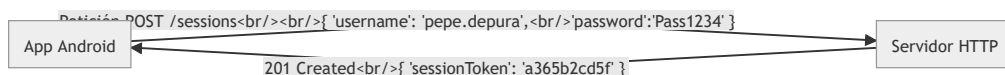
- Cuando el usuario pulse *Loguearse* enviaremos una petición HTTP al API REST para que genere una nueva sesión
- La petición contendrá un cuerpo JSON
- Si la respuesta es exitosa, mostraremos un **Toast** con un dato de la respuesta llamado *Token de sesión*

Descripción

Anteriormente hemos creado una petición **POST** para *registrar un nuevo usuario*:



Ahora, implementaremos una petición **POST** que, empleando las credenciales de la misma forma, *genera* una nueva sesión en el servidor y tras ello *recibe* un **Token de sesión**:



Profundizaremos más adelante en *cómo se usa* dicho *Token*. De momento, únicamente lo mostraremos en un **Toast**.

La tarea

En `LoginActivity.java`, **añade** un atributo tipo `RequestQueue`. **Inicialízalo** en `onCreate`:

```

public class LoginActivity extends AppCompatActivity {
    /* ... */
    private RequestQueue requestQueue;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        /* ... */

        requestQueue = Volley.newRequestQueue(this);
    }
}

```

Luego, **crea** un nuevo método privado `sendPostRequest`. De momento estará vacío, pero servirá para *encapsular* la lógica relacionada a enviar el **POST** de *login*:

```

public class LoginActivity extends AppCompatActivity {
    /* ... */

    private void sendPostRequest() {

    }
}

```

A continuación, **añade** un atributo tipo `Button` e **inicialízalo** conectándolo al botón de **"Loguearse"**. Algo parecido a lo que hiciste en la tarea anterior con el otro botón.

Después, para este botón, **establece** un `OnClickListener`. Dentro de `onClick`, **invoca** el método `sendPostRequest` recién creado.

Así, cuando el usuario pulse **"Loguearse"**, ¡se ejecutará el método `sendPostRequest`!

Trabajemos ahora en `sendPostRequest`.

Instancia un objeto `JSONObject` y **almacena** el *nombre de usuario* y *contraseña* que el usuario ha introducido en los campos de texto. Lo haremos bajo las claves `"username"` y `"password"`, **igual** que en la pantalla de registro:

```
private void sendPostRequest() {
    JSONObject requestBody = new JSONObject();
    try {
        requestBody.put("username", /* .getText().toString() sobre tu instancia de editText para el nombre de usuario */
        requestBody.put("password", /* .getText().toString() sobre tu instancia de editText para la contraseña */ );
    } catch (JSONException e) {
        throw new RuntimeException(e);
    }
}
}
```

¡Envíemos la petición HTTP!

A continuación del código anterior, **instancia** un `JsonObjectRequest`. Los parámetros del constructor serán:

- (1) `Request.Method.POST`.
- (2) `Server.name + "/sessions"`.
- (3) `requestBody`, creado en las líneas anteriores.
- (4) Un `Response.Listener` *anónimo*.
- (5) Un `Response.ErrorListener` *anónimo*.

Quedará así:

```
JsonObjectRequest request = new JsonObjectRequest(
    Request.Method.POST,
    Server.name + "/sessions",
    requestBody,
    new Response.Listener<JSONObject>() {
        @Override
        public void onResponse(JSONObject response) {

        }
    },
    new Response.ErrorListener() {
        @Override
        public void onErrorResponse(VolleyError error) {

        }
    }
);
```

¡Ya tenemos prácticamente diseñada nuestra petición HTTP! Ahora, para completar la tarea:

1. **Añade** la petición a `this.requestQueue` para que la petición se envíe
2. En `onResponse`, **muestra** un `Toast` que indique `"Token: XXX"` donde `XXX` es el *Token de sesión* recibido en la respuesta. **Así:**

```
new Response.Listener<JSONObject>() {
    @Override
    public void onResponse(JSONObject response) {
        String receivedToken;
        try {
            receivedToken = response.getString("sessionToken");
        } catch (JSONException e) {
            // Si el JSON de la respuesta NO contiene "sessionToken", vamos a lanzar
            // una RuntimeException para que la aplicación rompa.
            // En preferible que sea NOTORIO el problema del servidor, pues desde
            // la aplicación no podemos hacer nada. Estamos 'vendidos'.
            throw new RuntimeException(e);
        }
        // Si la respuesta está OK, mostramos un Toast
        // Esta línea asume que private Context context = this; está definido
        Toast.makeText(context, "Token: " + receivedToken, Toast.LENGTH_SHORT).show();
    }
},
```

Cuando termines estos pasos, puedes lanzar la *app* y verificar que el `Toast` aparece tras hacer *Login*.

Por último

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



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