

Especialización en Back End II

Ejercicio en vivo

- Ejercitación grupal
- Nivel de complejidad: medio 🔥🔥

Objetivo

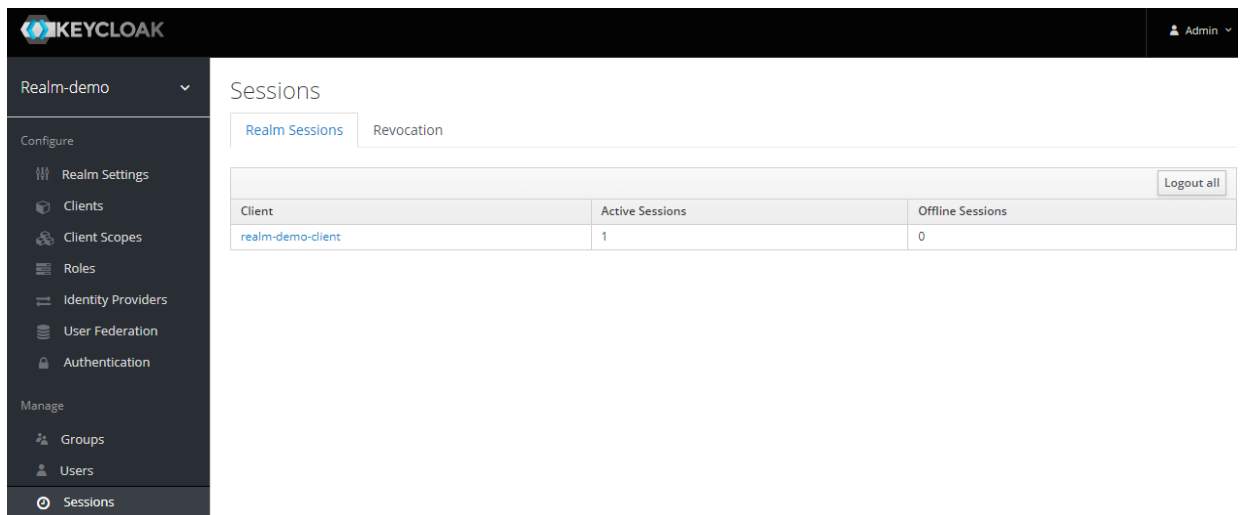
Crear una configuración de un realm, cliente y usuarios en Keycloak para luego loguear un usuario y —como administrador— revocar el acceso, así como gestionar los tokens y sus refresh desde el punto de vista del usuario.

Enunciado

1. Crear un reino indicando en la opción de login que el usuario puede seleccionar la opción de **“Remember me”**.
2. Crear un cliente confidencial del tipo OpenID dentro del reino del punto 1 indicando como URL de redirección **“/*”**.
3. Agregar dentro del cliente un role llamado **“user”**.
4. Agregar 2 usuarios: **user1** y **user2**. Los passwords son el mismo nombre de usuario.
5. Utilizar la URL <http://{host}:9091/realms/{realm-name}/.well-known/openid-configuration> para identificar las URL que disponemos en nuestro nuevo reino que nos permitan autenticarnos.
6. Autenticarnos contra nuestra URL de **authorization_endpoint** del punto anterior:
 - a. Para esto ponemos en nuestro navegador la URL de acuerdo a los pasos 1 y 2:
http://localhost:9091/realms/realm-demo/protocol/openid-connect/auth?response_type=code&client_id=realm-demo-client
 - b. Nos logueamos con **user1**.

7. Como usuarios administradores, vamos a ver los datos de usuarios logueados en nuestro reino desde una nueva pestaña en incógnito (para poder loguearnos como admins):

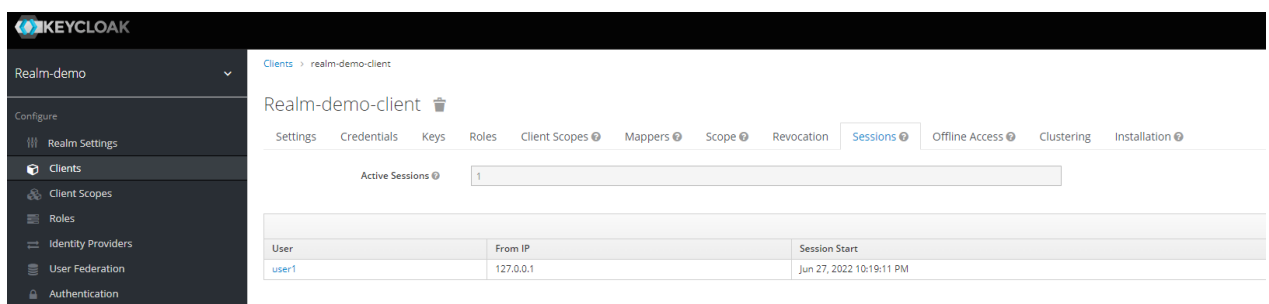
<http://localhost:9091/admin/master/console/#/realms/realm-demo/sessions/realm>



Client	Active Sessions	Offline Sessions
realm-demo-client	1	0

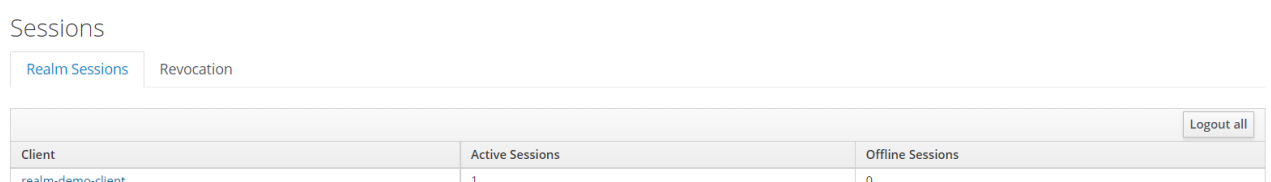
8. Como usuarios administradores, vamos a ver las sesiones del cliente creado en el punto 2.

<http://localhost:9091/admin/master/console/#/realms/realm-demo/clients/4252e6d9-b144-4f6b-b894-e7ceeb4ab561/sessions>



User	From IP	Session Start
user1	127.0.0.1	Jun 27, 2022 10:19:11 PM

9. Desde la vista de sesiones del punto 7, vamos a desloguear a todos los usuarios activos desde el botón “Logout all”.



Client	Active Sessions	Offline Sessions
realm-demo-client	1	0

POST ▼ http://localhost:9091/realms/realm-demo/protocol/openid-connect/token Send ▼

Params Authorization ● Headers (9) ● **Body ●** Pre-request Script Tests Settings

● none ● form-data ● x-www-form-urlencoded ● raw ● binary ● GraphQL

	KEY	VALUE	DESCRIPTION
<input type="checkbox"/>			
<input checked="" type="checkbox"/>	client_id	realm-demo-client	
<input checked="" type="checkbox"/>	client_secret	C0J47cuDphc3MMwWZcr8G7pYgffwAr7X	
<input checked="" type="checkbox"/>	redirect_uri	/* ...	
<input checked="" type="checkbox"/>	grant_type	refresh_token	
<input checked="" type="checkbox"/>	refresh_token	eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6IjxYWfjN...	
<input type="checkbox"/>			
	Key	Value	Description

Este método POST lo podemos invocar tantas veces como queramos de acuerdo a la vida útil de nuestro refresh token.

13. Utilizamos desde Postman la URL

<http://localhost:9091/realms/realm-demo/protocol/openid-connect/userinfo> para solicitar información del token actual. Dicho token se enviará como una cabecera de autorización Bearer.

POST ▼ http://localhost:9091/realms/realm-demo/protocol/openid-connect/userinfo

Params Authorization ● Headers (8) ● **Body ●** Pre-request Script Tests Settings

TYPE

Bearer Token ▼

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Token

eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUIiwia2lkIiA6IjxYWfjN...

Body Cookies Headers (7) Test Results ⚙ Status: 200 OK

Pretty Raw Preview Visualize JSON ⌵ ≡

```

1 {
2   "sub": "ce019c99-1175-4a24-babe-9927b2818149",
3   "email_verified": false,
4   "name": "user user",
5   "preferred_username": "user1",
6   "given_name": "user",
7   "family_name": "user"
8 }
```

14. Como administradores de Keycloak, accedemos a la sección de usuarios, seleccionamos a **user1** y visualizamos las sesiones activas para todas las asociadas.

The screenshot shows the Keycloak user management interface. On the left is a sidebar with a 'Configure' section containing 'Realm Settings', 'Clients', 'Client Scopes', 'Roles', 'Identity Providers', 'User Federation', and 'Authentication'. Below this is a 'Manage' section with 'Groups', 'Users', and 'Sessions'. The 'Users' section is selected, showing 'User1'. The 'Sessions' tab is active, displaying a table of active sessions. The table has columns for IP Address, Started, Last Access, Clients, and Action. There are four sessions listed, all for the 'realm-demo-client'.

IP Address	Started	Last Access	Clients	Action
127.0.0.1	Jun 27, 2022 11:27:17 PM	Jun 27, 2022 11:27:17 PM	realm-demo-client	Logout
127.0.0.1	Jun 27, 2022 11:33:43 PM	Jun 27, 2022 11:33:43 PM	realm-demo-client	Logout
127.0.0.1	Jun 27, 2022 11:34:31 PM	Jun 27, 2022 11:34:31 PM	realm-demo-client	Logout
127.0.0.1	Jun 27, 2022 11:31:25 PM	Jun 27, 2022 11:43:01 PM	realm-demo-client	Logout

15. Si ejecutamos nuevamente el request del punto 12 y 13, vamos a tener un error de token inválido.

The screenshot shows a REST client interface with tabs for 'Body', 'Cookies', 'Headers (8)', and 'Test Results'. The 'Body' tab is selected, showing a JSON response. The response is a 401 Unauthorized error with the message 'Token verification failed'.

```
{
  "error": "invalid_token",
  "error_description": "Token verification failed"
}
```

16. Vamos a configurar nuestro reino para que utilice rotación de tokens. Para esto, dentro de la configuración de este, vamos a activar la opción de **“Revoke Refresh Token”**.

The screenshot shows the Keycloak configuration page for the 'Tokens' tab. The 'Default Signature Algorithm' is set to 'RS256'. The 'Revoke Refresh Token' option is turned 'ON'. Other settings include 'Refresh Token Max Reuse' set to 0, 'SSO Session Idle' set to 30 minutes, 'SSO Session Max' set to 10 hours, 'SSO Session Idle Remember Me' set to 0 minutes, 'SSO Session Max Remember Me' set to 0 minutes, 'Offline Session Idle' set to 30 days, 'Offline Session Max Limited' set to 'OFF', and 'Client Session Idle' set to 0 minutes.

17. Si generamos un token de acceso e intentamos refrescar como en el punto 13, esto será válido solo la primera vez. Si queremos realizarlo más de una vez, debemos tomar el refresh token recibido en el JSON de la respuesta, cada vez que refrescamos el token.