

## Especialización en Back End II

# Ejercicio en vivo

- Ejercitación grupal
- Nivel de complejidad: medio 🔥🔥

## Objetivo

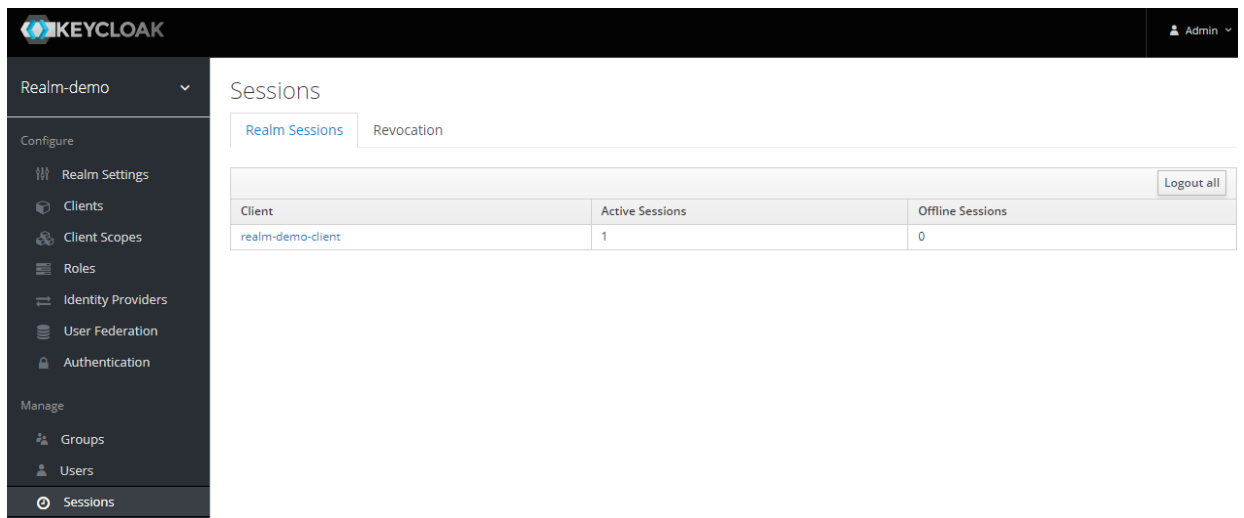
Crear una configuración de un realm, cliente y usuarios en Keycloak para luego loguear un usuario y —como administrador— revocar el acceso, así como gestionar los tokens y sus refresh desde el punto de vista del usuario.

## 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](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>



Keycloak Admin Console - Sessions

Realm: realm-demo

Configure: Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, Authentication

Manage: Groups, Users, Sessions

Sessions

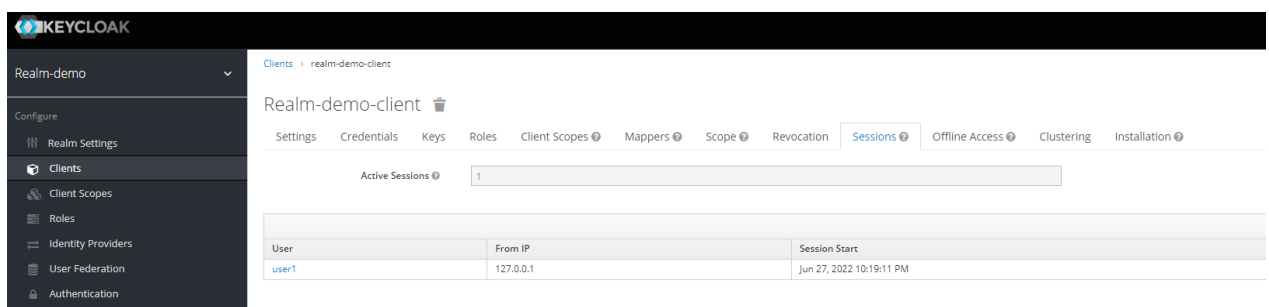
Realm Sessions | Revocation

Logout all

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>



Keycloak Admin Console - Sessions

Realm: realm-demo

Configure: Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, Authentication

Manage: Groups, Users, Sessions

Clients > realm-demo-client

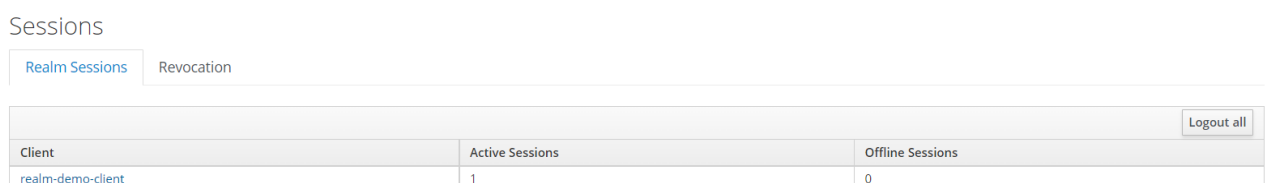
Realm-demo-client

Settings | Credentials | Keys | Roles | Client Scopes | Mappers | Scope | Revocation | Sessions | Offline Access | Clustering | Installation

Active Sessions: 1

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”.



Sessions

Realm Sessions | Revocation

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	eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUiIiwia2lkIiA6IjxYWFlN...	
<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**

eyJhbGciOiJIUzI1NiIsInR5cCIgOiAiSldUiIiwia2lkIiA6IjxYWFlN...

**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 navigation options: Configure (Realm Settings, Clients, Client Scopes, Roles, Identity Providers, User Federation, Authentication), Manage (Groups, Users, Sessions), and a top bar with 'Realm-demo' and 'Users > user1'. The main content area is titled 'User1' and has tabs for Details, Attributes, Credentials, Role Mappings, Groups, Consents, and Sessions. The 'Sessions' tab is active, displaying a table of active sessions. A 'Log out all sessions' button is in the top right of the table.

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 active, showing a JSON response in 'Pretty' format. The response indicates an 'invalid\_token' error.

```
1 {
2   "error": "invalid_token",
3   "error_description": "Token verification failed"
4 }
```

At the top right, the status is '401 Unauthorized', time is '13 ms', size is '458 B', and there is a 'Save Response' button.

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 'Tokens' configuration page in Keycloak. The left sidebar is the same as in the previous screenshot. The main content area has tabs for General, Login, Keys, Email, Themes, Localization, Cache, Tokens, Client Registration, Client Policies, and Security Defenses. The 'Tokens' tab is active, showing various token-related settings. The 'Revoke Refresh Token' toggle is turned 'ON'.

Setting	Value
Default Signature Algorithm	RS256
Revoke Refresh Token	ON
Refresh Token Max Reuse	0
SSO Session Idle	30 Minutes
SSO Session Max	10 Hours
SSO Session Idle Remember Me	0 Minutes
SSO Session Max Remember Me	0 Minutes
Offline Session Idle	30 Days
Offline Session Max Limited	OFF
Client Session Idle	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.