Keycloak SSO + Angular 6 + Spring Boot 2

1. Keycloak

Download and install standalone keycloak server (v.4.0.0 final).

<https://www.keycloak.org/downloads.html>

Run server from command line

../keycloak-4.0.0.Final/bin/standalone.bat

Keycloak UI will be available at the localhost:8080.

At first run create a admin account with login and password.

Enter to the Administration Console.

Add a Realm.  
Input a realm name: KAS-Demo-Realm. In that page, you can also import data from external source to create realm.  
Stay with default settings for demo.

Clients.  
Create first clients. Input Client ID: KAS-Demo-Frontend. Save. Input Valid Redirect URLs: \*. Input Web Origins: \*. Save.  
Stay with Client Protocol: opened-connect.   
Create second client. Input Client ID: KAS-Demo-Backend. Select Access Type: bearer-only. Save.

Roles.  
Add realm Role. Input Role Name: admin. Save.  
Add one more Role. Input Role Name: manager. Save.

Users.  
Add user. Input Username: admin. Save.   
Go to Credentials. Input password: admin. Set temporary: off. Reset Password. Change Password.  
Go to Role Mappings. Select Available Roles: admin. Add selected to Assigned Roles.

Return to Users. You can view all created users when click on button “View all users”.  
Add one more user. Input Username: manager. Save.  
Go to Credentials. Input password: manager. Set temporary: off. Reset password. Change Password.  
Go to Role Mappings. Select Available Roles: manager. Add selected to Assigned Roles.

Return to Users.  
Add one more user. Input Username: user. Save.  
Go to Credentials. Input password: user. Set temporary: off. Reset password. Change Password.  
No custom role will be available for that user.

2. Angular 6 demo project.

Install npm. <https://www.npmjs.com/get-npm>

Install Angular Client. <https://cli.angular.io/>  
# npm install –g @angular/cli

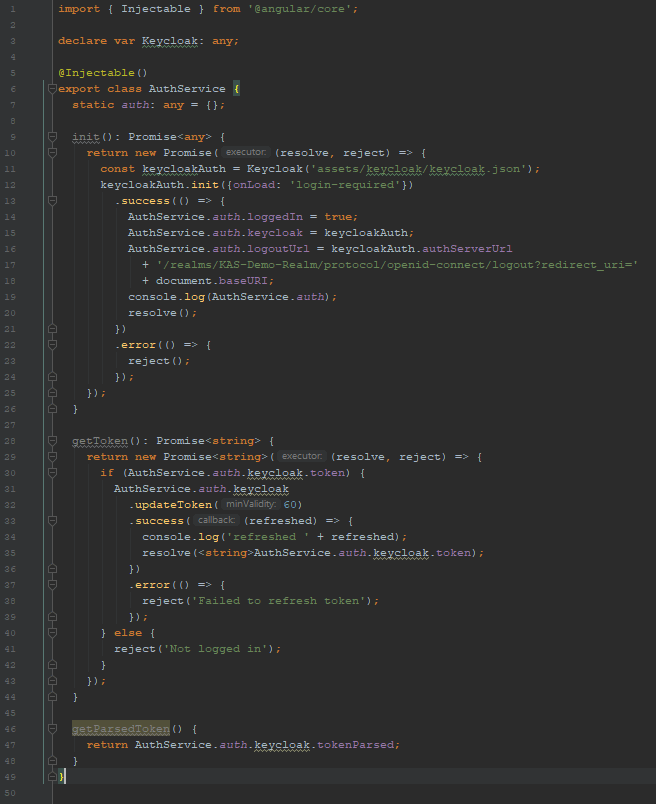
Create project:   
# ng new KAS-Demo-Frontend

Now you can open project with IDE (I use an IntellijIdea).  
Add keycloak-js into dependency.  
# npm install –save [keycloak-js@4.0.0](mailto:keycloak-js@4.0.0)

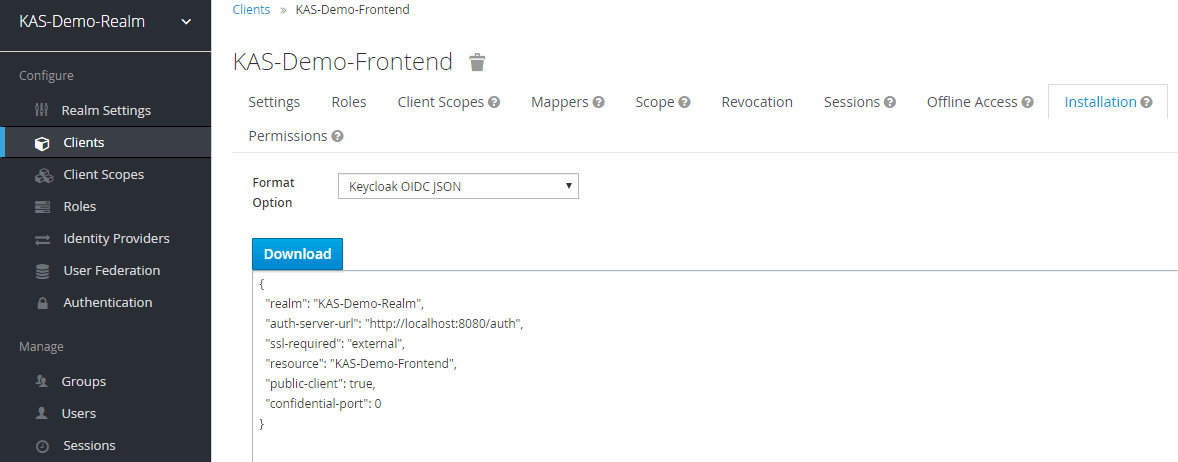
Go to angular.json. Add keycloak scripts into “architect”:”build”:”options”:"scripts": ["node\_modules/keycloak-js/dist/keycloak.min.js"]

Create keycloak authentication service:  
# ng generate service services/auth

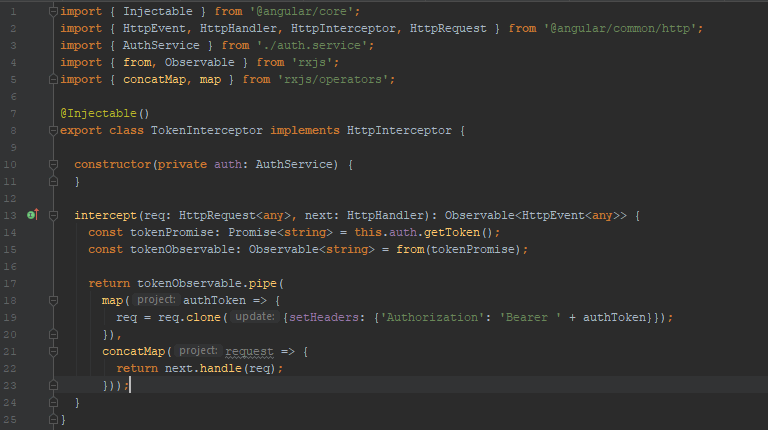
Create AuthService:



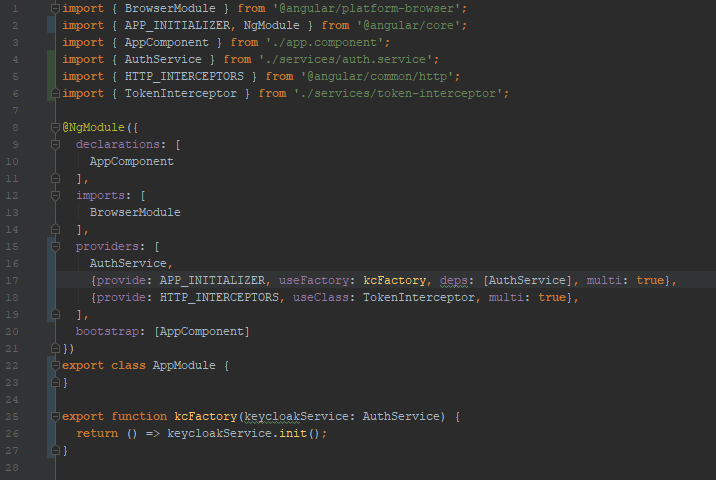
As keycloakAuth constant was declared, keycloak.json file with keycloak realm properties must be into assets/keycloak directory. Keycloak.json may be generated with keycloak server in KAS-Demo-Frontend client in the Installation tab with Keycloak OIDC JSON option.



Create an interceptor that put keycloak token into request header for backend authentication:  
# ng generate class services/TokenInterceptor

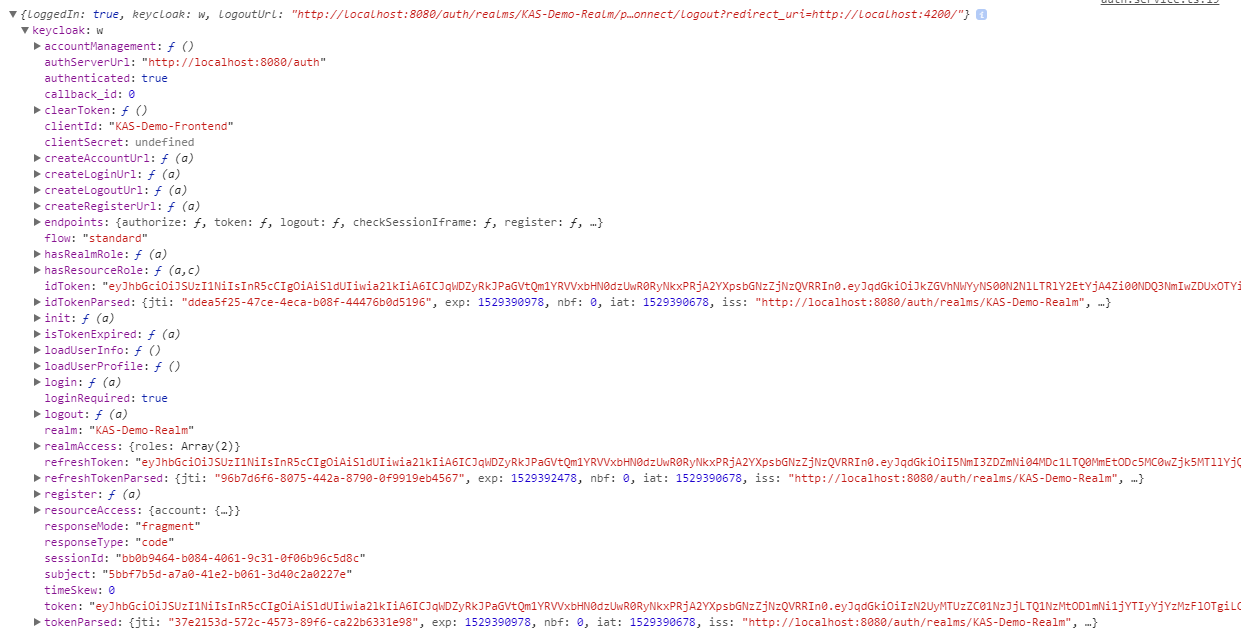


Go to app.module.ts.  
Create keycloak factory. Add providers: AuthService, APP\_INITIALIZER, HTTP\_INTERCEPTORS.



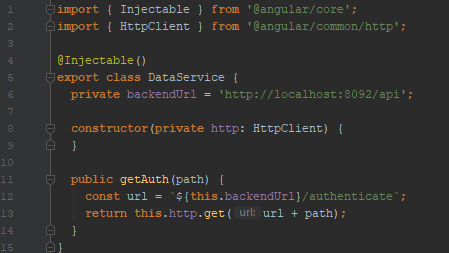
Now you can run app with keycloak authentication.   
# ng serve

In the browser developer tool (console) you can find AuthService.auth object where you can find keycloak token, realm roles of the logged in user and other information.

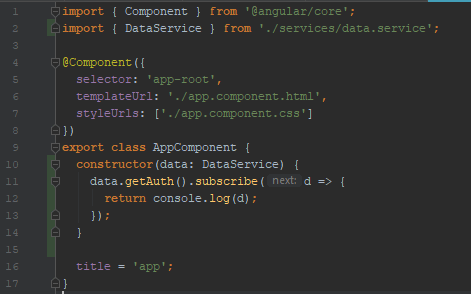


Add data.service for REST request to the backend.

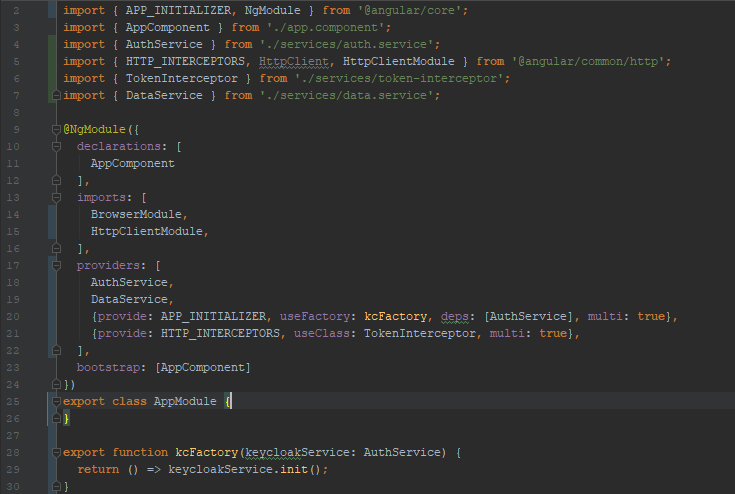
# ng generate service services/data



Get getAuth request when main page was loading. Fix app.component.ts as shown below.

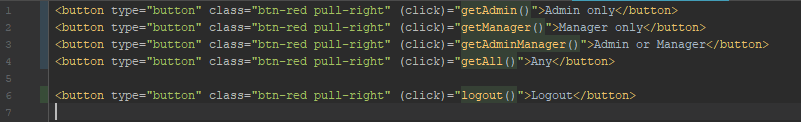


Add HttpClient and DataService to the app.module.ts as “providers”.

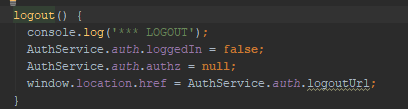


Add logout button and other buttons for backend requests.

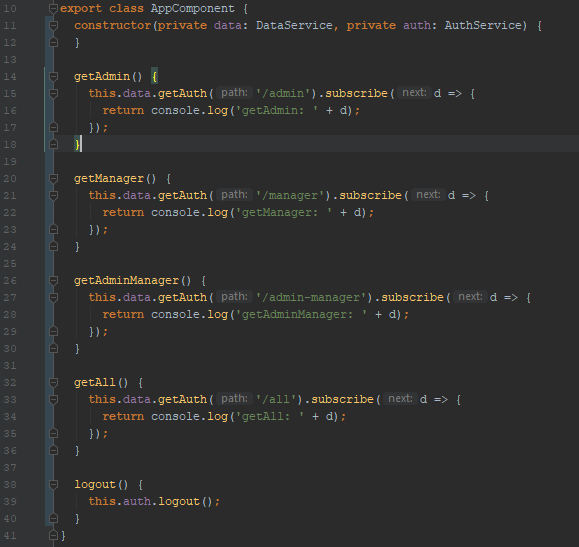
App.component.ts:



Add to auth.service.ts:

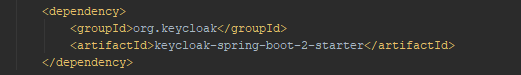


Add to app.compponent.ts:

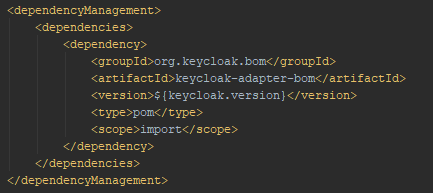


3. Spring Boot 2

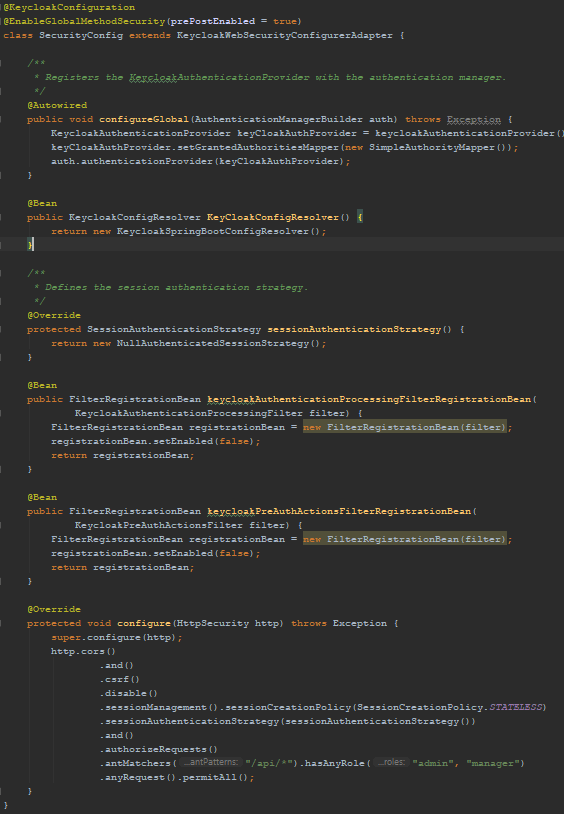
Generate Spring Boot 2 (v.2.0.3) KAS-Demo-Backend maven project. <https://start.spring.io/>  
add keycloak dependency (keycloak version 4.0.0.Final)



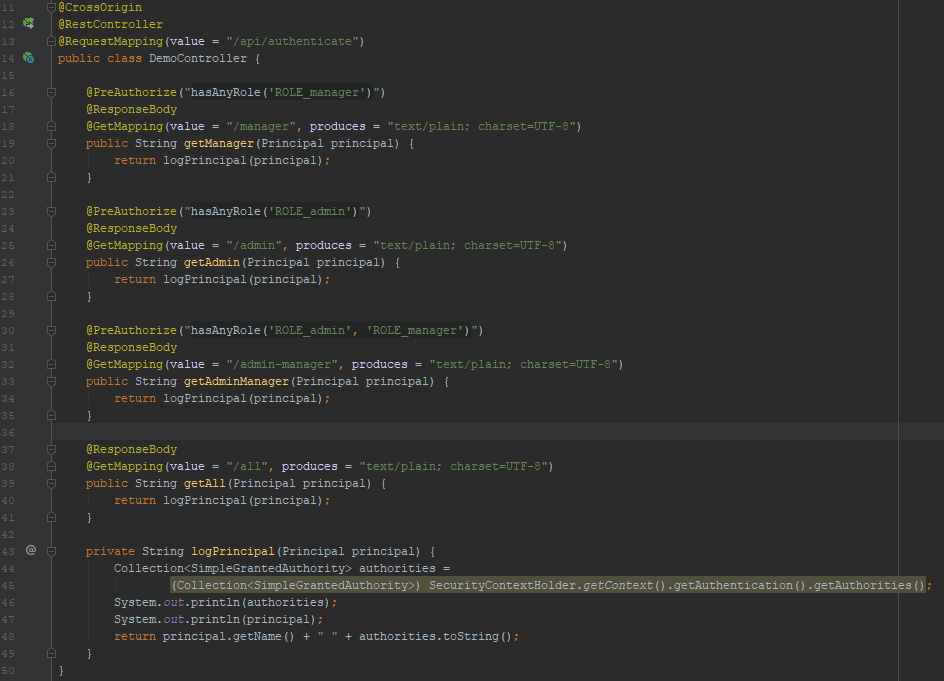
And dependency management



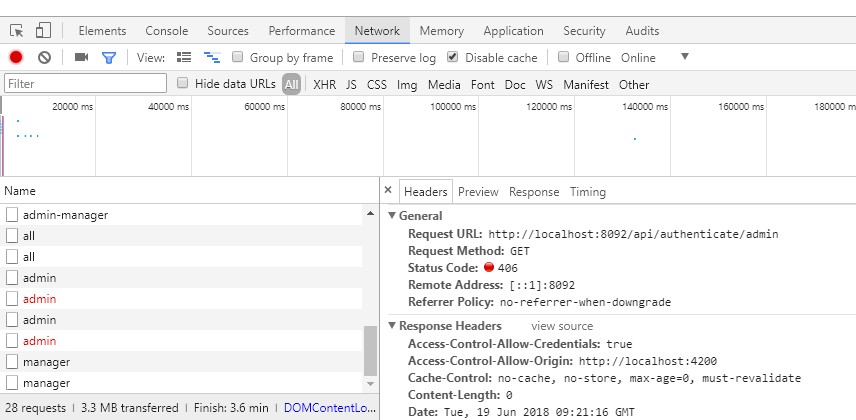
Create SecurityConfig class.



Controller:



Now you can try login with other users and test authorized or forbidden requests to the backend. Click buttons and watch the result.



If you are logged in as “user” all request get **403** status code error because it is forbidden by antMatchers here:



For example, if you are logged in as “manager” you get 200 status code for authorized requests and **406** for forbidden requests because preauthorization of methods is enabled:

