This document has the purpose to help integrating Frontend-kit (FroK) library in Angular applications.

This document was written to integrate the library FroK 2.1.2.

FroK library website: https://frok.westeurope.cloudapp.azure.com

user: bosch

password: uneo2019

The npm package of the FroK is stored in Bosch Artifactory. To be able to use it, please use the following configuration in your project .npmrc file:

```
registry=https://registry.npmjs.org

@bosch:registry=https://fe-artifactoryha.de.bosch.com/artifactory/api/npm/npm-repo/

strict-ssl = false
```

To install FroK npm package and add its dependency to package.json, in the root folder of your project, run the command:

```
npm install --save @bosch/frontend.kit-npm@2.1.2
```

To integrate it in a JHipster generated web application, you will need to copy FroK library from node_modules/@bosch/frontend.kit-npm/dist to src/main/webapp/content/frok. For automating this copy, in pom.xml add:

```
<build>…<pluginManagement>…<plugins>…
 <plugin>
  <groupId>org.apache.maven.plugins</groupId>
  <artifactId>maven-resources-plugin</artifactId>
  <version>${maven-resources-plugin.version}</version>
  <executions>
  <execution>
   <id>copy-frok-library</id>
   <phase>validate</phase>
   <goals>
    <goal>copy-resources</goal>
   </goals>
   <configuration>
    <outputDirectory>${basedir}/src/main/webapp/content/frok</outputDirectory>
    <resources>
     <resource>
      <directory>${basedir}/node_modules/@bosch/frontend.kit-npm/dist</directory>
      <!--filtering>true</filtering-->
     </resource>
    </resources>
   </configuration>
  </execution>
 </executions>
</plugin></plugins></pluginManagement></build>
```

- src/main/webapp/content/scss/vendor.scss

```
@import '~@bosch/frontend.kit-npm/dist/frontend-kit.css';
```

- src/main/webapp/index.html
  Right before the end body tag (</body>) add the script call.

```
<script id="frok_js" src="content/frok/frontend-kit.js"></script>
```

  Note that: the value of the attribute id in the element is used in the main.component.ts. If you give a different id make sure you adapt it also in the main.component.ts code.

- src/main/webapp/app/layouts/main/main.component.ts
  In the *ngOnInit* method, add the following code

```
ngOnInit(): void {
    . . .
    this.router.events.subscribe(event => {
      if (event instanceof NavigationEnd) {
        . . .
        /* FroK library needs to be loaded after Angular components is loaded.
This is a work around forces FroK library to be loaded on each route change.
*/
        if (document.getElementById("frok_js") != null) {
          document.getElementById("frok_js")?.remove();
        }
        const scriptTag = document.createElement("script");
        scriptTag.src = "content/frok/frontend-kit.js";
        scriptTag.id = "frok_js";
        document.body.appendChild(scriptTag);
        /* ---------- */
      }
      . . .
    });
  . . .
  }
```

To use FROK components that have Static API implemented, in frameworks like Angular, one needs to have interfaces specification for those FROK components (because Angular is “typed typescript”). For example: to have Modal Dialog working (see FROK documentation https://frok.westeurope.cloudapp.azure.com/molecules/dialog/guide), one needs to create a typescript file (for example, content/frok/types.ts) where the needed interfaces are declared (note that the methods of Dialog are the ones specified in the FROK’s Static API

documentation). Then, in the **.component.ts* where you want to use the FROK component, you just need to import the type specifications (e.g., *import { WindowWithFrontendKit } from 'content/frok/types'*) and follows the examples in the FROK documentation.

```
interface Dialog {
  dialogId(id: string): string;
  findDialog(id: string): string;
  showDialog(id: string): void;
  hideDialog(id: string): void;
}

export interface WindowWithFrontendKit extends Window {
  boschFrontendKit: {
    Dialog: Dialog;
  };
}
```