



EUROPEAN COMMISSION
DIRECTORATE-GENERAL FOR INFORMATICS

Directorate A – Digital Transformation
DIGIT.A.3 – Solutions for Legislation, Policy & HR



Getting started with LEOS

Technical guide

Contents

Introduction to LEOS.....	3
Install LEOS.....	4
1. Download the source code of both modules in the same folder.	5
2. Select demo installation option.	6
3. OPTION A.....	6
3.1. Run script.	6
3.2. Use in browser.	6
4. OPTION B	6
4.1. Running user database	6
4.2. Running repository.....	7
4.2.1. Use the InMemory repository version included with this LEOS distribution.	7
4.3. Running LEOS.....	7
4.4. RUNNING AKN4EUUTIL.....	8
4.5. VIEW H2 CONSOLE.....	8
5. Configure LEOS	8
5.1. Document templates and element configuration.....	8
5.2. Structure File.....	11
5.2.1. List of elements – TOC items list <tocItems>	12
5.2.2. List of numbering configurations <numberingConfig>.....	13
5.2.3. List of table of content rules – <tocRules>.....	13
5.3. Document repository Data Model.....	14
5.3.1. InMememory data configuration	14

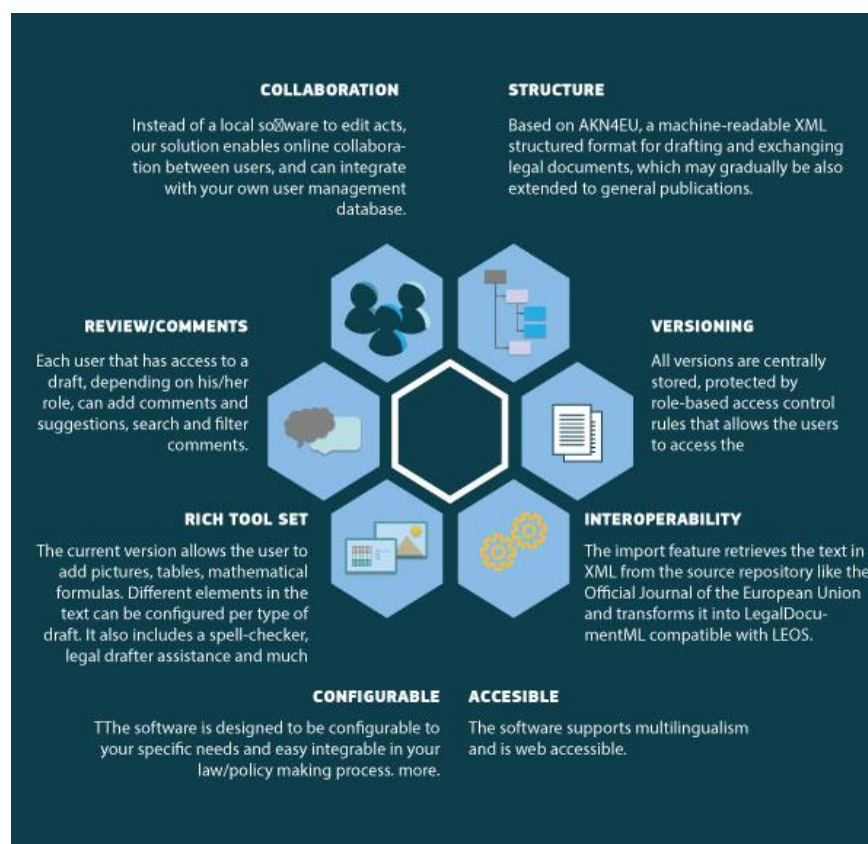
Introduction to LEOS

LEOS (Legislation Editing Open Software) is a project under [Interoperable Europe](#) initiative of the European Commission for a reinforced public sector interoperability policy, funded by the [Digital Europe Programme](#) (DIGITAL) and which was created to address the need of the public administration and European Institutions to generate draft legislation in a legal XML format.

The LEOS project under **Interoperable Europe** focuses on supporting the co-development co-design, and co-deployment of an 'IT eco-system centred on augmented LEOS'.

LEOS has been created to addresses the **modernisation and digital transformation of the drafting and revising of legislation in EU Institutions, EU agencies and bodies and Member States**.

LEOS ensures that the content drafted by the users follows the drafting guidelines by offering features like enforcing predefined document structures, predefined layout , numbering rules. All of this to ensure that the author would first and foremost focus on drafting itself and much less on layout management (or checking). In order to facilitate efficient online collaboration LEOS also contains some others features like comments, suggestions, version control, co-edition, etc.



Content is stored in an XML format, currently in AKN4EU which is based on the Akoma Ntoso OASIS standard namespace: <http://docs.oasis-open.org/legaldocml/ns/akn/3.0>. This facilitates document exchange, element retrieval, etc.

While Akoma Ntoso defines representations for parliamentary, legislative and judiciary documents, AKN4EU is a localisation of Akoma Ntoso and currently focus on the documents exchanged in the scope of the Ordinary Legislative Procedure (OLP).

- Regulation, Directive, Decision,
- Proposal for a Regulation, Proposal for a Directive, Proposal for a Decision.
- Some cases of the EP Texts Adopted.
- Council documents including the Position and the Statements of Council reasons.

(see AKN4EU 4.0 Baseline – General structure of documents, 0 Scope).

Each of the above legal documents have a certain structure, and it is usually composed of a cover page and one or more components. The components have their own structure.

Therefore, the main concepts LEOS is managing are the documents (the legal documents) and their components (or subdocuments).

Install LEOS

The current modules of augmented LEOS are:

- [LEOS core](#) and
- [Annotations utils](#).

IMPORTANT NOTES: *This release is intended to provide an experience with the software and is stripped of several important components to enable ease of use.*

Additionally, this software is adapted to run on a local server for demo purposes and without proper security mechanisms.

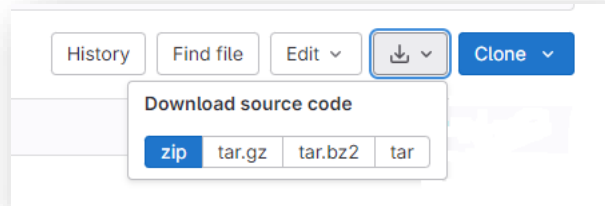
DEPENDENCIES

To compile the supplied source files and run the generated WAR the following software should be configured:

- Java SDK version 8.0.
- Maven version 3.3.9+ (Maven runtime memory might need to be set: MAVEN_OPTS=-Xms256m -Xmx512m) (Maven settings, proxy and mirrors, might need to be adjusted to your environment and internet access requirements).
- Supported browser is Google Chrome version 45+ (Mozilla Firefox ESR version 38.3 and Microsoft Internet Explorer version 11 are known to work with minor issues).

1. Download the source code of both modules in the same folder.

Create a new directory {AUGMENTED_LEOS} in the local file system.

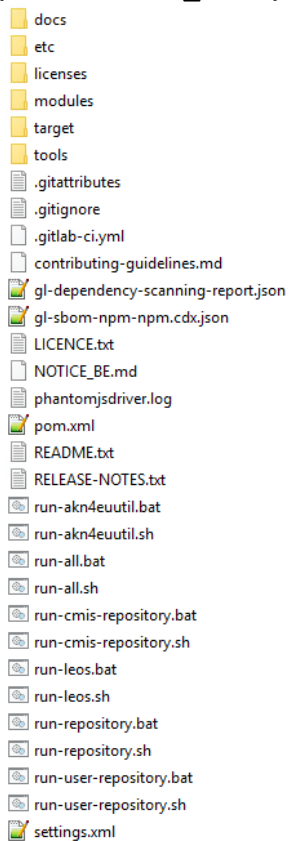


Download and unzip the following modules in the new directory { AUGMENTED_LEOS }:

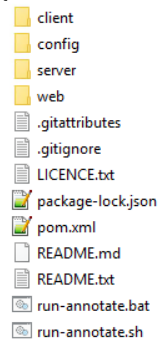
- [LEOS core](#)
- [Annotations utils](#)

Such that you'll end up with the following folder structure:

{ AUGMENTED_LEOS }/leos



{ AUGMENTED_LEOS }/annotate



2. Select demo installation option.

- If your machine's operating system is Microsoft Windows or Linux, you can use [Option A](#).
- If your machine's operating system is not windows or Linux based or you want to run components one by one, you should follow the steps of [Option B](#).

3. OPTION A

3.1. Run script.

If your machine's operating system is Microsoft Windows, you can simply execute the provided script:

```
run-all.bat.
```

If your machine's operating system is Linux based, you can simply execute the provided script:

```
run all.sh.
```

This script will execute individual scripts that will compile AND run each of the required software components.

3.2. Use in browser.

Open the browser and navigate to the LEOS web interface available at the following URL:

<http://localhost:8080/leos-pilot/ui>

LEOS is pre-configured with these demo users:

NAME	LOGIN	PASSWORD	ROLE
Demo User	demo	demo	Normal
John Doe	john	demo	Normal
Jane Doe	jane	demo	Support
S Leo	leos	demo	Support

4. OPTION B

If your machine's operating system is not windows or Linux based or you want to run components one by one, you should follow below steps.

4.1. Running user database

You must compile and run the user database on the command line.

1. Traverse to folder { AUGMENTED_LEOS}/leos/tools/user-repo
2. Execute the following command to compile source code:

```
mvn clean install
```

3. Execute the following command to run it:

```
mvn spring-boot:run -Drun.profiles=h2
```

4.2. Running repository

- Use the InMemory repository version included with this LEOS distribution.

4.2.1. Use the InMemory repository version included with this LEOS distribution.

To run InMemory repository server, you must compile and run the repository on the command line.

- a. Traverse to folder { AUGMENTED_LEOS}/leos/tools/repository
- b. Execute the following command to clean source code.

```
mvn clean install
```

- c. Execute the following command to run it.

```
mvn spring-boot:run -Dspring-boot.run.directories=../config/target/generated-config
```

- d. Normally, Access Inmemory repository console on web browser
<http://localhost:9097/repository/h2-console/>

4.3. Running LEOS

Notes:

- User database and repository must already be running.
- LEOS can use SAML protocol to authenticate with an IDP.
- To use SAML it is needed modify SAML configuration defined in "securityContext.xml" file inside module "security-saml".

You must run LEOS on the command line.

- 1) Traverse to folder { AUGMENTED_LEOS}/leos
- 2) Execute the following command to compile source code.
 - a. For default authentication:

```
mvn clean install
```

- b. For SAML authentication:

```
mvn clean install
```

```
Dsaml
```

- 3) Traverse to folder { AUGMENTED_LEOS}/leos/modules/web
- 4) Execute the following command to run LEOS.
 - a. For default authentication:

```
mvn jetty:run | war
```

- b. For SAML authentication:

```
mvn jetty:run war Dsaml
```

4.4. RUNNING AKN4EUUTIL

You must compile and run akn4euutil on the command line.

- Traverse to folder { AUGMENTED_LEOS}/leos/tools/akn4euutil
- Execute the following command to compile source code.

```
mvn clean install
```

- Execute the following command to run it.

```
mvn spring boot:run
```

Open the browser and navigate to the LEOS web interface available at the following URL:

<http://localhost:8080/leos-pilot/ui>

4.5. VIEW H2 CONSOLE

1. Make sure application.properties has console enable:

```
spring.h2.console.enabled=true
```

2. Enter the below url in browser (don't forget the context of the application)

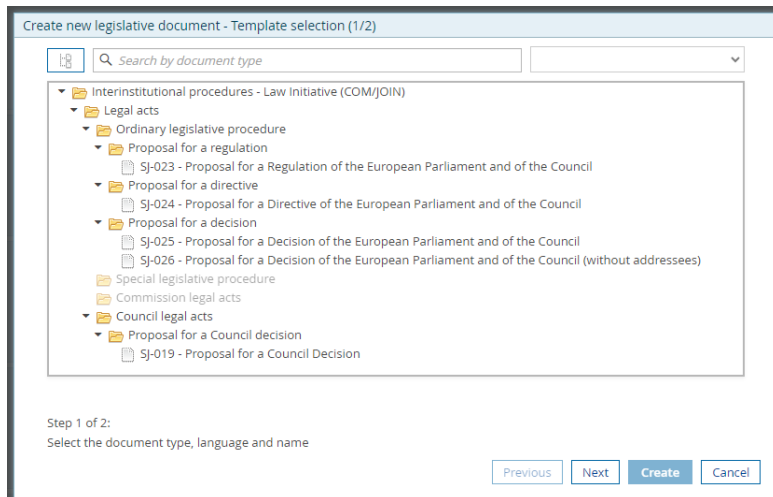
- **USER REPO** : <http://localhost:9095/ud-repo/h2-console>
- **ANNOTATIONS**: <http://localhost:9099/annotate/h2-console>
- **JDBC URL** : jdbc:h2:mem:~/test;Mode=Oracle;DB_CLOSE_ON_EXIT=FALSE

5. Configure LEOS

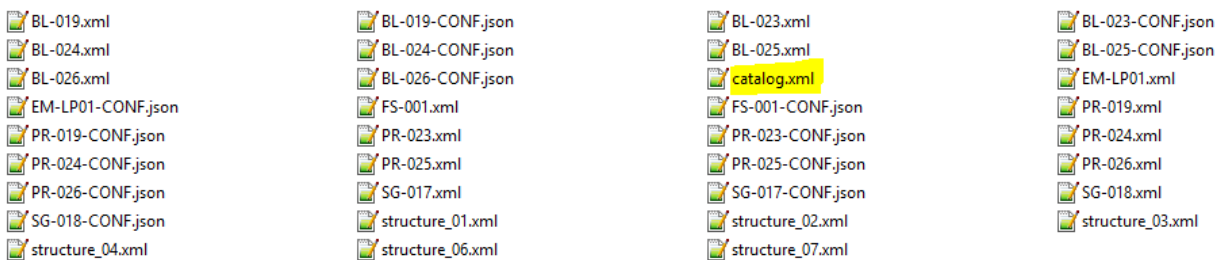
5.1. Document templates and element configuration

Document templates are in format Akoma Ntoso files stored in the document repository and they are used for the creation of documents and all the components/subdocuments that belong to it.

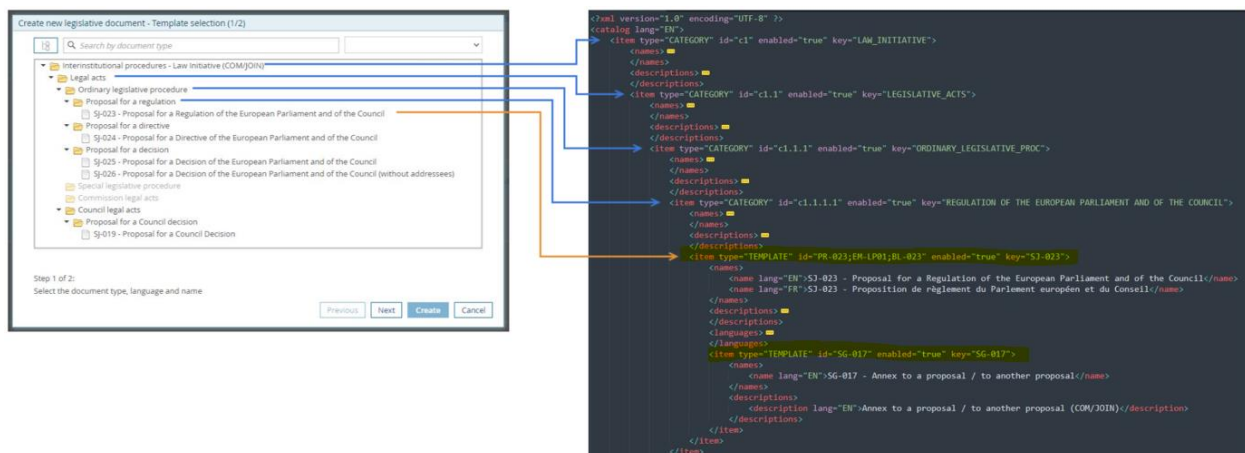
In LEOS application a user can select the type of document to be created by clicking the "**Create proposal**" button. The term "proposal" is the type of document that is currently used by default in LEOS, but that can be changed, depending on the specific needs.



All the information needed to compose the template selection screen and the documents that belongs to each of the proposal are store in the "catalog.xml", located in {AUGMENTED_LEOS}/leos/tools/repository/server/src/main/resources/leos/templates/os

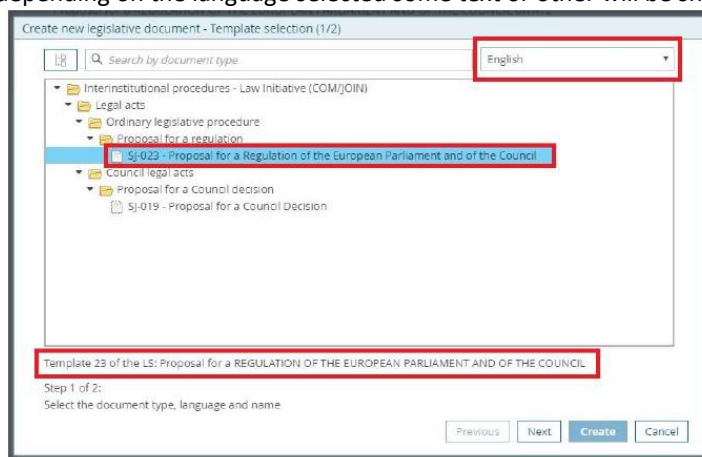


This is the entry point to describe all existing templates, for example proposal "SJ-023 - Proposal for a Regulation of the European Parliament and of the Council" is composed of documents and subdocuments "PR- 23;EM-LP01;BL-023" and can contain annexes "SG-017".

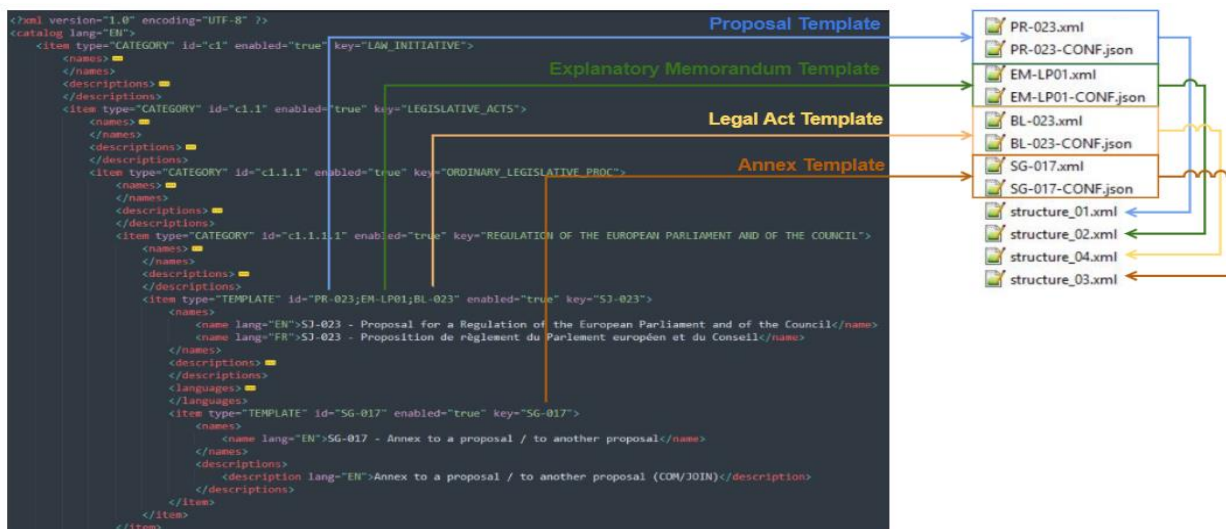


Important to notice that:

- The "id" attribute of the item contains a list of the templates to be used when a document of type proposal is created:
 - Proposal document template PR-023.
 - Memorandum (*sub*)document template EM-LP01. (*The Memorandum document template can be removed and then no memorandum will be created.*)
 - Legal Text (*sub*)document template BL-023.
- There are some tags related with the text that appears in the template selection screen like names, descriptions and depending on the language selected some text or other will be shown.



These files PR-023, EMP-LP01, BL-023 and SG-017 are document and (*sub*)document templates in akn format and each of them can have associated a JSON configuration file. In these JSON files some configuration like guidance or the structure file to be used to define all the elements that the document can contain can be added.

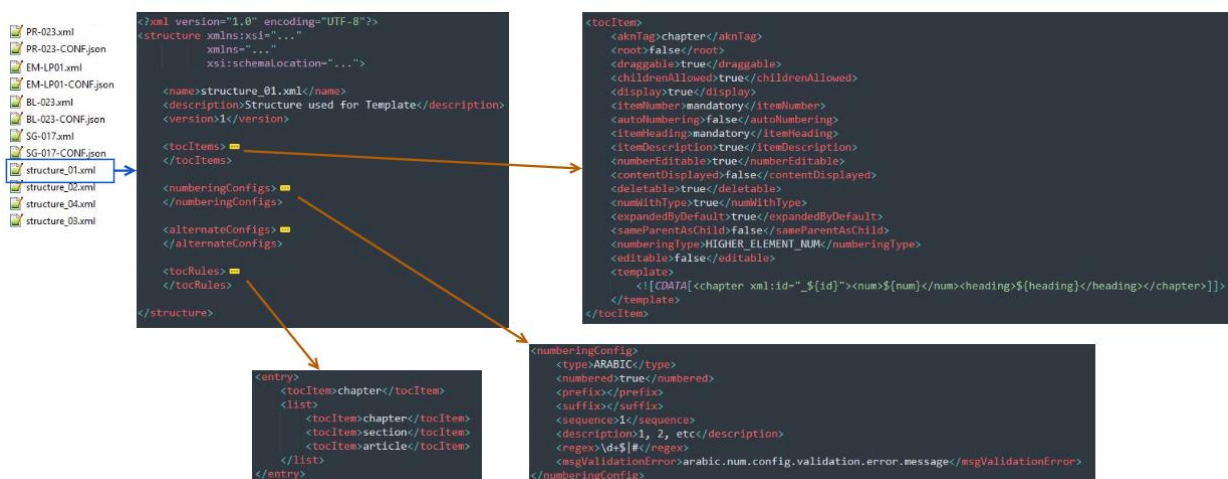


Sample BL-019-CONF.json

<pre> { "guidance": [{ "targets": [...], "content": "<guidance id=\"guidance-cit-3\" class=\"guidance-green-block\">[where necessary]</guidance>" }, { "targets": [...], "content": "<guidance id=\"guidance-cit-4\" class=\"guidance-green-block\">[where necessary]</guidance>" }] //..... }, </pre>	User Guidance
<pre> "alternatives": [{ "name": "listOption1", "list": [{ "index": "1", "content": "This Decision is addressed to the Member States." }, { "index": "2", "content": "This Decision is addressed to the Member States in accordance with the Treaties." }] } //...] </pre>	Alternatives
<pre> "structure": [{ "name": "structure_01" }] </pre>	Structure file

While LEOS comes by default with a set of document templates and configuration elements, based on your requirements, you can create your own document templates and configure according to your needs.

5.2. Structure File



The structure file further defines what elements shall be available in the in the document or subdocument it is being associated with, what structural and numbering rules should each of them obey to.

This schema is used to validate the structure files that configure all elements that can be used inside a document and that it is linked to a document template through template JSON configuration files. Some comments about this schema must be done:

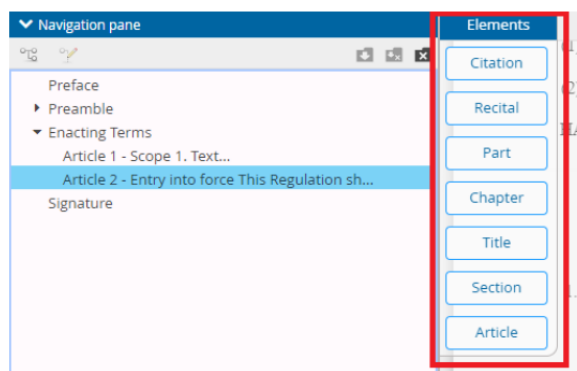
- If a new element must be added to a document, then this new element must be added to `aknTag` simpletype object.
- If a new numbering type must be added, then it must be added as a new `numberingType` object.

5.2.1. List of elements – TOC items list <tocItems>

Allows to define and configure a set of elements that can be used in the document or subdocument. For each element there's a set of properties to define its behaviour.

```
<tocItem>
  <aknTag>preface</aknTag>
  <root>true</root>
  <draggable>false</draggable>
  <childrenAllowed>false</childrenAllowed>
  <display>true</display>
  <itemNumber>none</itemNumber>
  <itemHeading>none</itemHeading>
  <itemDescription>true</itemDescription>
  <numberEditable>false</numberEditable>
  <contentDisplayed>false</contentDisplayed>
  <deletable>false</deletable>
  <numWithType>false</numWithType>
  <expandedByDefault>true</expandedByDefault>
  <sameParentAsChild>false</sameParentAsChild>
  <numberingType>NONE</numberingType>
  <editable>false</editable>
</tocItem>
```

- **aknTag**: Element name. It should be added or should exists in the structure schema too (structure_1.xsd inside domain module).
- **root**: If element it is a root element or not. In LEOS code there's some actions that cannot be done over root elements.
- **draggable**: If element is draggable it will appear in the elements pane. User can drag and drop element in the table of content tree.



- **childrenAllowed**: Element can contain other elements.
- **display**: Element is showed in the table of content tree.
- **itemNumber**: Element contains <num> tag.
- **itemHeading**: Element contains <heading> tag.

- **itemDescription:** Element text (ex. Article) is shown in the table of content tree.
- **numberEditable:** Number is editable by the user.
- **contentDisplayed:** Content inside the element is showed in the TOC.
- **deletable:** Element can be deleted.
- **numWithType:** Inside the num tag the element name (ex. Article) is with the number.
- **expandedByDefault:** Element is showed expanded in the TOC.
- **sameParentAsChild:** Element can contain same elements.
- **numberingType:** Numbering type used for numbering the element.
- **editable:** Element is editable.
- **template:** XML structure used when element is created in the document. By default, current template is used if no template is configured.

5.2.2. List of numbering configurations <numberingConfig>

Through the numberingConfigs list a set of different numbering configuration can be defined. A concrete number configuration is applied to an element using the <numberingType> tag inside the tocItem.

For example, ARABIC numbering type.

```
<numberingConfig>
  <type>ARABIC-PARENTHESES</type>
  <prefix></prefix>
  <suffix></suffix>
  <sequence>1</sequence>
  <description>(1), (2), etc</description>
  <regex>\d+$|<#</regex>
  <msgValidationError>arabic.num.config.validation.error.message</msgValidationError>
</numberingConfig>
```

- **type:** Numbering configuration name. It should be added or should exists in the structure schema too (structure_1.xsd inside domain module).
- **prefix:** If numbering should include some prefix.
- **suffix:** If numbering should include some suffix.
- **sequence:** Numbering sequence (1 by 1 in this example).
- **description:** Just a description.
- **regex:** Regular expression used to validate number introduced by the user.
- **msgValidationError:** Error validation message showed defined inside application message.properties file.

5.2.3. List of table of content rules – <tocRules>

Allows to define which kind of elements can contain a particular element. For example, body element:

```

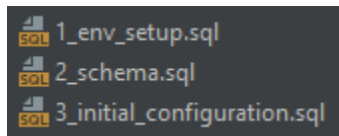
<entry>
  <tocItem>body</tocItem>
  <list>
    <tocItem>part</tocItem>
    <tocItem>title</tocItem>
    <tocItem>chapter</tocItem>
    <tocItem>section</tocItem>
    <tocItem>article</tocItem>
  </list>
</entry>

```

Body element can contain part, title, chapter, section and article elements.

5.3. Document repository Data Model

The Document repository stores all XML documents in a supporting database that needs to be configured and initialized prior of using LEOS. The steps to setup a MySQL database for the purpose can be found in {AUGMENTED_LEOS}/leos/tools/repository/web/src/etc/database/mysql/*



1_env_setup.sql
2_schema.sql
3_initial_configuration.sql

5.3.1. InMemory data configuration

Alternatively, and recommended to be used for testing purposes only, you can use the h2-in memory database to load the templates. To achieve this, besides adding the necessary AKN4EU template files as aforementioned on [Section 5.1](#), a reference to the files should be added in the file {AUGMENTED_LEOS}/leos/tools/repository/server/src/main/resources/ **data-property-h2.sql**. These reference implies an entry into :

- CONFIG table
- CONFIG_VERSION table

