tts performance suite

Creator 3.39.4 Installation Manual - tts performance suite 2026

Contents

1 General Information

1.1 Introduction

1.1.1 Objective

This document describes the requirements of the tts performance suite component named **Creator**, and its installation process.

The goal of this document is to provide an understanding of the server configuration and to guarantee its successful installation. For installation instructions of other tts performance suite components, Curator and WebAccess, refer to the corresponding manuals.

1.1.2 Target audience

System administrators, developers and all interested parties.

1.1.3 Prerequisites

It is expected that system administrators, developers and who else may be interested have good knowledge of

- Windows and/or Unix-based operating systems
- Administration and handling of application server Apache Tomcat
- Deployment of web applications (WAR, external web application)

⚠ **Please read this installation guide completely and carefully!**

1.1.4 Contact

- tts Support
  Phone: +49 (0) 2 21 / 17 09 30 - 110
  Fax: +49 (0) 2 21 / 17 09 30 – 170
  `support@tt-s.com`
- Application consultant
  Application consultants are very experienced with the tts performance suite and can guide you through the installation process, providing professional solutions to match your requirements. If no application consultant has yet been assigned, please contact your key account manager.

1.1.5 Structure

The first chapter sheds light on the functionalities and technologies used in the Creator and finished by presenting the System overview and deployments scenarios.

The next section points to the System requirements concerning hardware and software. They should be checked thoroughly and carefully.

The Installation chapter guides you step-by-step through the deployment of the Creator, including defining server configuration. Post installation steps deals with initial administration tasks.

Due to the fact that the Creator is a complex application, many configuration options are provided. A variety of settings are explained in detail in the Appendix. Each property or parameter of the application's services is listed with its name, a description and the possible values.

1.1.6 Conventions

1.1.6.1 Symbols

To highlight important information on the one hand, and "nice to know" details on the other, the following icons are used:

Attention

Hint or note

Tip

1.1.6.2 Variables

Variables are marked with a leading $

Variable	Description
$TTPS_HOME	installation directory of tts performance suite
$TOMCAT_HOME	installation directory of Apache Tomcat

1.1.7 Your feedback is highly welcome

tts welcomes your feedback concerning the quality and usefulness of this manual. Your comments and suggestions will be considered as valuable input for future revisions of this manual.

- Found an error? Please let us know where.
- A topic is not described clearly enough? Please let us know which one.
- Need more information? On which topic?
- Something doesn't work for you? Please let us know, so we can provide additional examples.

Please feel free to send us your feedback: `support@tt-s.com`. We appreciate your help!

1.2 Application description

The Creator consists of a backend component, written in Java (Spring Boot) and a frontend created with Typescript. The backend is responsible for the connection to the curator and the repository (S3/MinIO), the frontend presents the Creator editor to the user.

1.3 System overview and deployment scenarios

The Creator is a standalone application, which is deployed as a separate war file to an application server. Nevertheless, you'll need the tts performance suite installed and accessible to use it.

There are two different deployment scenarios.

The first scenario, which is the minimum way in practice, involves one application server hosting the tts performance suite (Curator and WebAccess) and also the Creator.

The second, recommended, scenario would be to deploy the Creator to its own application server. It differs from the first scenario by being more scalable and dynamic, thanks to using separate application servers plus a load balancer that distributes all requests among the reverse proxies.

Both scenarios, minimum and recommended, are field-tested concepts. The usage depends on the customer's requirements. Nevertheless, we recommend the deployment including a reverse proxy in any case. Moreover, if there is a huge number of end users expected, an installation in a clustered environment is advised, since this scenario offers more flexibility, better scalability, and higher performance.

2 System requirements

For the current hardware and software requirements of tts server, please refer to the system requirements.

3 Installation step-by-step

3.1 Checklist

Before you start deploying the Creator, please ensure that the following requirements are met:

- The installation files are available
- The application server is installed properly
- The MinIO Server is installed and configured
- The tts performance suite is installed

3.2 Installation environment

3.2.1 MinIO installation

For full MinIO installation instructions, refer to the tts performance suite installation manual. It is required that the Creator use the same MinIO instance as the Curator.

Create a bucket named "bg-creator", which is the default name. To use another bucket name use the corresponding value within the creator.application.properties.

The properties are explained in detail in the appendix.

The Creator uses Deno to run JavaScript on the backend. If you want to encrypt the traffic between MinIO and Creator using TLS with a self-signed root certificate, you need to tell Deno about it. This can be done in two different ways:

- Add your root certificate to your system's trusted root certificates, and set the environment variable `DENO_TLS_CA_STORE` to `system` on your system.
- Set the path to your root certificate in the `creator.deno.run-options.cert` application property:

```
creator.deno.run-options.cert=file:/absolute/path/to/your/root.crt
```

3.2.2 Connecting the tts performance suite and the Creator

These parameters need to be added to the application-config.properties of the tts performance suite to connect to the Creator:

3.2.2.1 2022

```
ttkf.server.businessguidancecreator.endpoint=https://<creator-domain>:<port>/creator/v1/session/open
```

3.2.2.2 2022r2

```
ttkf.server.businessguidancecreator.endpoint=https://<creator-domain>:<port>/creator
```

3.2.2.3 2023 and later

```
ttkf.server.businessguidancecreator.endpoint=https://<creator-domain>:<port>/creator
ttkf.server.businessguidancecreator.presignedurl.publickey.endpoint=https://<creator-domain>:<port>/creator/v1/presigned-url/certificate
```

3.2.3 Configuring the Creator

The actual configuration is carried out via a file named creator.*application.properties*. Place it directly in $TTPS_HOME.

The following example shows a snippet of a creator.*application.properties* file for the minimum deployment scenario with the creator placed on the same application server as the tts performance suite.

For all available parameters check the creator.application.properties reference in the appendix.

3.2.3.1 2025r2 and later

Additional parameters are required.

```
# url equals jwt token iss field value
spring.security.oauth2.resourceserver.jwt.issuer-uri=<jwt issuer uri>
```

It should be equal to the authserver `spring.security.oauth2.authorizationserver.issuer` property.

3.2.3.2 2023r2 and later

```
# optional, add customizing specific folder.
# The customizing folder has the following structure:
#  customizing-root-dir
#      └── template-name
#          └── configuration-id
#
#  e.g:
#  customizing-root-dir
#      └── steplist
```

```
#          └── eeefffaaa
# The configuration id is equals the client configuration id from the producer/curator.
creator.templates.customizing.location=file:/<path-to-templates-customizing-directory>
```

3.2.3.3 2023 and later

```
# Internal URL of the Curator
creator.curator.internal-address=https://<curator-domain>:<port>/curator
# External URL of the Curator
creator.curator.external-address=https://<curator-domain>:<port>/curator
# Version of the Curator where creator.curator.internal-address points to
creator.curator.version=<curator-version>
# Internal address of minio server.
# Must be exactly the same value as the
#   ttkf.server.repository.endpoint.internal
# parameter in the Curator configuration.
# E.g.
# if
#   ttkf.server.repository.endpoint.internal=http://127.0.0.1:9000
# is used in the Curator, use
#   creator.s3.minio.internal-address= http://127.0.0.1:9000
# if
#   ttkf.server.repository.endpoint.internal=http://localhost:9000
# is used in the Curator, use
#   creator.s3.minio.internal-address= http://localhost:9000
creator.s3.minio.internal-address=https://<minio-domain>:9000
# External address of minio server
creator.s3.minio.external-address=https://<minio-domain>:9000
# MionIO authentication
creator.s3.access-key={MinIO credentials}
creator.s3.secret-key={MinIO credentials}
# Path to the templates directory
creator.templates.location=file:/<path-to-templates-directory>
```

3.2.3.4 2022r2 and before

Contains old mandatory parameter `creator.jwt-max-age`, `creator.keep-alive-interval` and `creator.keep-alive-retry-delay`

```
# Internal URL of the Curator
creator.curator.internal-address=https://<curator-domain>:<port>/curator
# External URL of the Curator
creator.curator.external-address=https://<curator-domain>:<port>/curator
# Version of the Curator where creator.curator.internal-address points to
creator.curator.version=<curator-version>
# Seconds until the jwt expires. Must be 0 < jwt-max-age <= Curator JSESSIONID Max-Age.
creator.jwt-max-age=300
# Send a keep-alive request every keep-alive-interval seconds.
creator.keep-alive-interval=150
# Time in seconds to wait before retrying a failed keep-alive
```

```
creator.keep-alive-retry-delay=10
# Internal address of minio server.
# Must be exactly the same value as the
#   ttkf.server.repository.endpoint.internal
# parameter in the Curator configuration.
# E.g.
# if
#   ttkf.server.repository.endpoint.internal=http://127.0.0.1:9000
# is used in the Curator, use
#   creator.s3.minio.internal-address= http://127.0.0.1:9000
# if
#   ttkf.server.repository.endpoint.internal=http://localhost:9000
# is used in the Curator, use
#   creator.s3.minio.internal-address= http://localhost:9000
creator.s3.minio.internal-address=https://<minio-domain>:9000
# External address of minio server
creator.s3.minio.external-address=https://<minio-domain>:9000
# MionIO authentication
creator.s3.access-key={MinIO credentials}
creator.s3.secret-key={MinIO credentials}
# Path to the templates directory
creator.templates.location=file:///<path-to-templates-directory>
```

3.2.4 Creator permission to create a S3 lifecycle rule

The creator has a preview for documents. It stores temporary files on S3 in `s3://<defined_bucket_in_application_config.properties>/<defined_prefix_in_application_config.properties>/previews/<preview_files>`. These files are no longer needed after a user is done with the preview.

Therefore, a lifecycle rule is created that says that after one day, everything with the key prefix `<defined_prefix_in_application_config.properties>/previews/*` will be deleted.

If the creator does not have permissions or something else goes wrong when setting the lifecycle rule is set, it will appear as an info in the log and the creator will continue as normal. So it is optional to give the creator the permission. If he does not have it, some else is responsible to clean up the files himself.

3.2.4.1 Disable s3 lifecycle rule

If you want to disable the complete s3 lifecycle mechanism use following `application-config.properties`.

```
creator.preview.s3.lifecycle.skip=true
```

3.2.5 Configuring the steplist template

The steplist template is configured using the file referenced by *creator.templates.options* in *creator.application.properties*. The following shows a example configuration.

The settings are described in the *Appendix*.

```
steplist:
```

```
    use-informal-auto-texts: true|false
    recorder:
        port: <recorderPort>
```

3.2.6 How to make creator.application.properties accessible?

There exist two options to provide the application with the configuration file externally. Either you extend the class path so the application will find that properties file; or you directly define its path within the context file as a parameter.

For better maintenance, we suggest to configure the path to the location of creator.application.properties directly in the corresponding Creator context file.

To do so, add the *spring.config.location* context parameter in the context file during deployment (see the deployment chapter):

```
...
<Parameter name="spring.config.location" override="false" value="file:/C:/tts/
creator.application.properties"/>
    ...
```

3.3 Application server settings

3.3.1 JVM settings

The JVM of each server instance must meet at least the following memory settings and might be adapted to the environment's requirements:

- Maximum Java heap size: 1024 MB (-Xmx1024m)
- Initial Java heap size: 256 MB (-Xms256m)
- Maximum thread stack size: (-Xss256k)

Depending on the Java distribution UTF-8 might not be the default encoding. Thus, the following JVM parameter has to be set:

```
-Dfile.encoding=utf-8
```

3.3.2 Deno settings

The template backend scripts will be executed in a javascript runtime called deno. The creator provides a bundled deno version for following platform and arch.

platform	arch
windows	amd64 (x86)
linux	amd64 (x86)
linux	aarch64

If another deno executable is required, it can be set via following application config property: `creator.deno.executable` This could be the case if a creator update is not possible, but the bundled deno version has a security issue.

3.3.2.1 Deno performance settings

If the creator runs on low performance compute unit following application config properties can be adjusted to ensure better operation.

`creator.deno.timeout` Maximum number of seconds a single deno process is allowed to run. Increase this, if backend scripts runs regularly in some timeouts. On low performance machines, the execution could take some more time.

`creator.deno.v8.max-heap-size` Maximum amount of memory (in Mbytes) used by a single deno process. Reduce this, if the server runs in out of memory issues. On low performance machines, the default settings could consume too much memory.

3.4 Deployment

Usually, tts provides pre-configured .WAR files for the server components.

Starting with Creator, release 2022r2, there's a templates.zip which has to be deployed additionally to the server.

1. Stop Tomcat if it is running.
2. Unzip the templates.zip and copy its content to $TOMCAT_HOME/
3. Copy *creator.war* to $TOMCAT_HOME/webapps.
4. Create the *creator.xml* file in the *$TOMCAT_HOME/conf/Catalina/localhost* directory.
5. Configure the location of the creator.*application.properties* within this context file.

   Example:

```
1  <?xml version="1.0" encoding="UTF-8"?>
2
3     <Context reloadable="false">
4
5     <Parameter name="spring.config.location" override="false" value="file:/C:/
↪ tts/creator.application.properties" />
6
7  </Context>
```

6. Restart Tomcat.

3.4.1 Tomcat 10

There is a significant breaking change between Tomcat 9.0.x and Tomcat 10.0.x. The Java package used by the specification APIs has changed from javax... to jakarta....

The Creator supports only the javax API.

Tomcat can convert an existing web application from Java EE 8 to Jakarta EE 9 at deployment time using the Apache Tomcat migration tool for Jakarta EE. To make use of the feature, the web application should be placed in the Host legacyAppBase folder (by default named webapps-javaee) and they will be converted to an equivalent Jakarta EE web application in the Host appBase folder (by default named webapps).

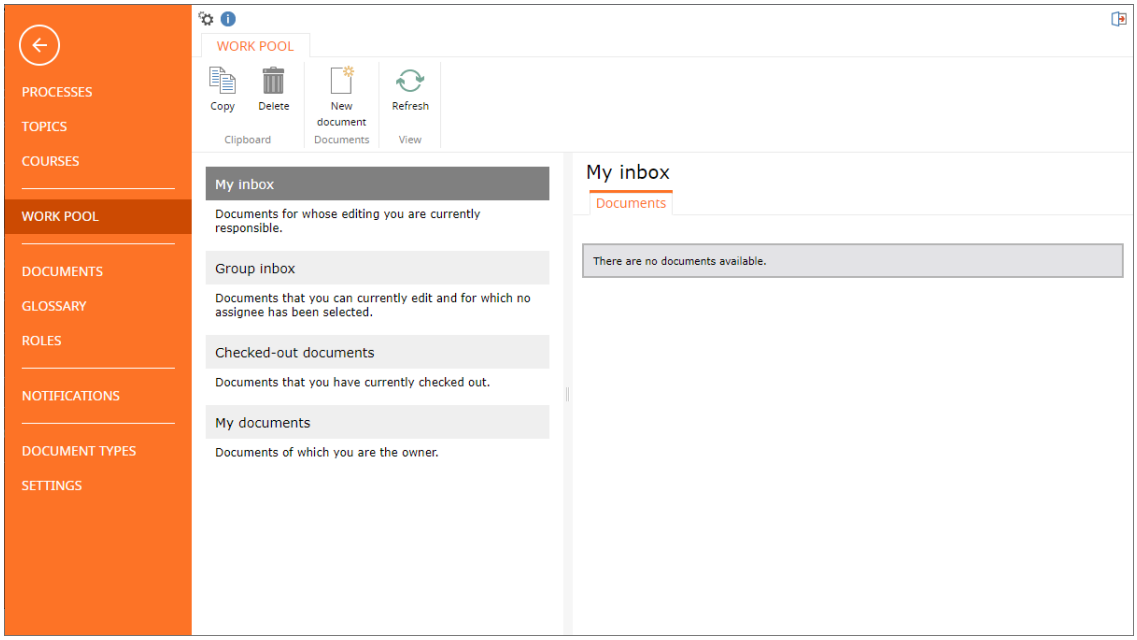
Example:

```
1
2 <Host name="localhost" appBase="webapps-javaee" unpackWARs="true" autoDeploy="true
↪ ">
3     <Valve className="org.apache.tomcat.jakartaee.TomcatMigrationToolValve"
↪ jakartaEeForward="true"/>
4 </Host>
```

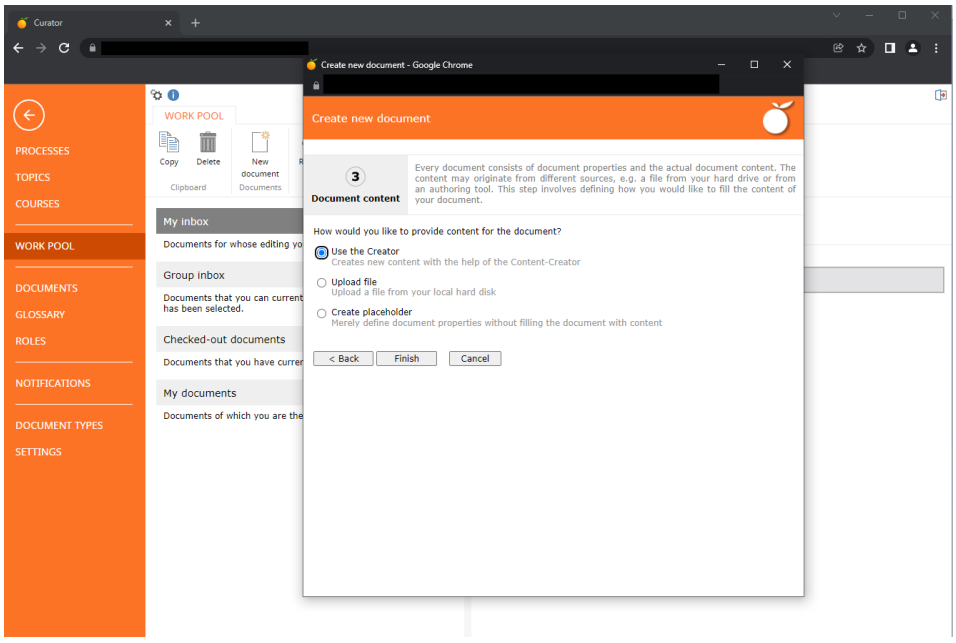
3.5 Post installation steps

3.5.1 Test the creator

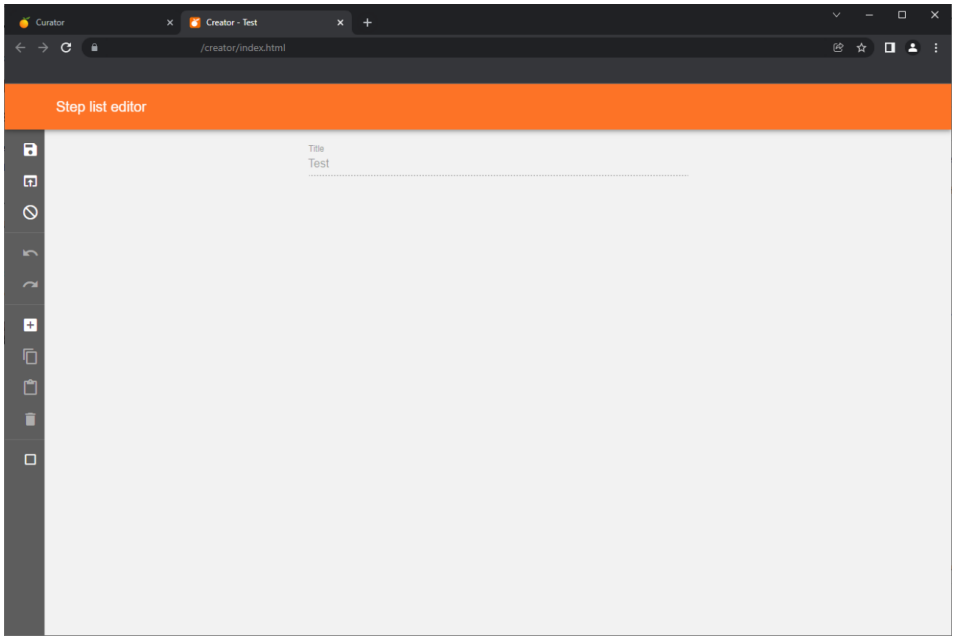
Log in to the Curator



When creating a new document, on step three of the wizard, there should be the option "Use the Creator"



After clicking "Finish" the Creator editor should open with a new document in a separate browser window.



4 Authorization Server (since Version 2025r2)

The Creator is a Spring Resource Server, which can communicate with the Auth Server to validate JWT tokens. For more information about Spring Resource Server and JWT see: https://-docs.spring.io/spring-security/reference/servlet/oauth2/resource-server/jwt.html

5 Templates Customizing

The customizing feature allows the user to customize the templates to their needs. It will be applied when a document is exported. To change the customizing for a document, the user has to open the document in the Creator and export it again.

Add the following configuration parameter to the application server settings:

```
creator.templates.customizing.location=file:/<
↪ absolute_path_to_templates_customizing_directory>
```

The customizing folder has the following structure:

```
customizing-root-dir
    └── template-name
        └── configuration-id
```

The configuration id is equals the client configuration id from the producer/curator. In this folder, customizing files could be deposited.

Which files are applied/supported is template specific. Please look into the template specific manual. As template developer see the “Template Developer Guide”.

5.1 Example

Folder structure

```
customizing-root-dir
    └── steplist
        └── eeefffaaa
```

6 template.yml

Each template contains a `template.yml` file that describes the template.

6.1 required-api-version field

Describes which template api verison is required. If the api version is not satisfied, the server will not start.

6.2 options field

It describes which options are available for a template. These options can be set via the template options file.

6.3 features field

It describes which features are available for a template. These features can be allowed or denied via the templates access file.

6.4 Example

```yaml
template-schema-version: 1

name: some-template
description: ""
author: ""
version: "0.0.1"
commit-sha: "abc"
since: ""
required-api-version: "^1.0.0"

options:
  - name: required-option
    required: true
    server-only: false
    schema:
      type: string

features:
    - some-feature
...
```

7 Templates Options

With the templates options file, one can configure the behavior of the templates. The template must support the specific options. Which options are supported is stored in the `template.yml`

file for the given template.

Add the following configuration parameter to the application server settings:

```
creator.templates.options=file:/path/to/templates-options.yml
```

If the `creator.templates.options` is set, the file must exist.

See application config properties reference which values are supported.

The template options file has the following structure:

```
template-name:
    single-option: "value"
    object-option:
        some-value: "another-value"
```

7.1 Example

```
steplist:
    use-informal-auto-texts: true|false
    recorder:
        port: <recorderPort>
```

8 Templates Access

With the templates access file, one can configure the access to the templates.

The file will be configured with the `creator.templates.access` application config properties.

```
creator.templates.access=file:/path/to/templates-access.yml
```

If the `creator.templates.access` is set, the file must exist.

See application config properties reference which values are supported.

Following permissions exists

8.1 Access

8.1.1 `ImplicitAllow`

Default, if no role matched. An `ImplicitAllow` occurs when there is no applicable `Deny` statement but also no applicable `Allow` statement.

8.1.2 `Deny`

If a role/template is matched, the user is denied access to the template.

8.1.3 `Allow`

If a role/template is matched, the user is allowed access to the template.

8.1.4 Overview

Access	Overrides
`Allow`	`ImplicitAllow` `Deny`
`Deny`	`ImplicitAllow`
`ImplicitAllow`	-

Access Precedence `Allow` < `Deny` < `ImplicitAllow`

8.2 Permissions

Currently, the following Permissions exists

Permission	Description
`create`	Create a template
`edit`	Edit a template
`features`	Template specific Features. Contains subfields. See

8.2.1 Creator Create Permission / Curator replace content

To perform the Curator action `Replace Content` the Creator user needs the **Create permission** in the Creator. From the Creator perspective, a new document is created. The template selection page will appear. From the Curator perspective, the document is edited with replaced content.

8.2.1.1 Explanation

The Creator has three entry modes

Create Used when a Creator document is created from QA. The Curator does not create the document until it is saved by the Creator. Meta information needs to be filled out in the upload dialog.

Edit Used when a document is edited from the Curator. Meta information does not need to be filled out in the upload dialog because it reused the existing meta information.

Replace Used when the Curator creates a new Creator document. Technically, the Curator creates a `Placeholder` before opening the Creator, and this is replaced by the Creator when saving to the Creator. Meta information does not need to be filled out in the upload dialog because it reused the existing meta information from the placeholder.

Also used when the Curator action `Replace Content` will be used on an existing document in the Curator. Meta information does not need to be filled out in the upload dialog because it reused the existing meta information.

8.2.2 Creator Template Features Permission

A template could have specific features that need to be allowed or denied. The features will be listed in the template.yml file under `features`.

8.3 template-access format

A JSON schema can be found in the creator repo (`templates-access-v1-0.yml`)

```
templates-access-schema-version: "1.0"
roles:
    <given_role>:
        templates:
            # short form
            <template-name>: allow | deny
            # long form
            <another-template-name>:
                create: allow | deny
                edit: allow | deny
                features:
                  <feature-name>: allow | deny
           ...
    ...
```

8.3.1 Example

```
templates-access-schema-version: "1.0"
roles:
    some-role:
        templates:
            steplist: deny
            imagemap:
                create: allow
                edit: allow
            foobar:
                create: deny
                edit: allow
                features:
                    some-feature: allow
    another-role:
        templates:
            steplist: deny
            imagemap:
                create: deny
                edit: allow
            foobar:
                create: deny
                edit: allow
```

9 Configuring the First View Note (since Version 2023)

Since Version 2023 a note can be configured to be shown the first time one visits the template selection.

Add the following configuration parameter to the application server settings:

```
creator.first-visit-note.options=file:/path/to/first-visit-note-options.yml
```

If the `creator.first-visit-note.options` is set, the file must exist.

See application config properties reference which values are supported.

⚠ Important: Even if the feature is enabled and configured. If only one template is installed, no template selection appears and therefore the note is **not** displayed. The one template will be opened directly.

9.0.1 Settings

General configuration for the note.

Option	required/optional	Description
fallbackLanguage	required	The message can be displayed in different languages. If a message is not found in a specific language, this language shall be used. References to a message in a specified langauge. Possible values are `en`, `de`, `fr`, `es`.
showWhenOpenedFrom	required	The template selection can be opened both from the curator and from the QA. One can configure if the note will be shown when the template selection was opened from the QA, or the curator.
showAgainInDays	optional	If set this option defines when to show the note again after one has confirmed it. Will only expect positiv integer values such as e.g. `30` for having the note reappear after 30 days.
firstVisitNote	required	Array of elements defining the messages in different languages. See chapter *firstVisitNote* for details.

9.0.1.1 firstVisitNote

Option	required/optional	Description
en	optional	Message defined in English. See chapter *Message*.
de	optional	Message defined in German. See chapter *Message*.
fr	optional	Message defined in French. See chapter *Message*.
es	optional	Message defined in Spanish. See chapter *Message*.

9.0.1.2 Message

Option	required/optional	Description
message	required	Message to be displayed in the given language. One may add HTML elements to the message.
button-Text	required	Text for the confirmation Button in the given language

9.0.1.3 Sample configuration

Message is configured in english and in german. The fallback is english and the message will be repeatedly shown every 60 days. The message will only be shown if the template selection was opened from the QA and not if it was opened from the Curator

```
fallbackLanguage: "en"
showNoteWhenOpenedFrom:
    - qa
showAgainInDays: 60
firstVisitNote:
    en:
        message: |
            Good morning.<br />
            We are <a href="https://tt-s.com" target="_blank">TTS</a>
        buttonText: "Confirm"
    de:
        message: "Guten Morgen.<br>Wir sind <a href='https://tt-s.com' target='
↪ _blank'>TTS</a>"
        buttonText: "Bestätigen"
```

10 Troubleshooting

10.1 Proxies and Cookies

The following cookies must not be set to http only:

```
jwtMaxAge
jwtKeepAliveInterval
jwtKeepAliveRetryDelay
```

10.2 Deno throws `invalid peer certificate: UnknownIssuer` error

If you are using MinIO with TLS and a self-signed certificate, and Deno throws the `invalid peer certificate: UnknownIssuer` error, please make sure you informed the Creator and Deno about your self-signed certificate correctly. Have a look at the MinIO section on how to do this.

If you are still getting this error, please ensure that your MinIO answers with a valid certificate chain. You can verify it using openssl:

```
# Replace tt-s.com:443 with the address and port of your MinIO server.
openssl s_client -showcerts -connect tt-s.com:443
```

If it is valid, the output should end with something like this:

```
Verify return code: 0 (ok)
```

11 Appendix

11.1 creator.application.properties reference

11.1.1 creator.curator.external-address

External URL of the Curator.

11.1.2 creator.curator.internal-address

Internal URL of the Curator.

11.1.3 creator.curator.version

Version of the used Curator. For Example "22.1".

11.1.4 creator.deno.executable

Path to the deno executable. If not configured, the bundled deno executable will be used.

11.1.5 creator.deno.timeout

Maximum number of seconds a single deno process is allowed to run.

Default: `60`

11.1.6 creator.deno.v8.max-heap-size

Maximum amount of memory (in MBytes) used by a single deno process.

Default: `128`

11.1.7 creator.deno.run-options.cert

Load certificate authority from PEM encoded file. See Deno documentation for more details.

Example: `file:/absolute/path/to/given.pem`

11.1.8 creator.first-visit-note.options

Path to the one and only first visit yaml file. e.g. file:/

11.1.9 creator.s3.access-key

Access key ID to access s3/minio bucket. If creator.s3.accessKey or creator.s3.secretKey is omitted DefaultCredentialProvider is used to find credentials.

11.1.10 creator.s3.arn-kms-key

If an encrypted bucket is used this needs to contain the ARN of the KMS key

11.1.11 creator.s3.arn-role

To grant users access to the s3 bucket, temporary credentials are being created using STS AssumeRole. This property must contain the ARN of the parent role that limits the permissions of the temporary credentials. The temporary credentials cannot assume broader permissions than the parent role. The parent role needs to allow s3:ListBucket access on the bucket, and

s3:GetObject and s3:PutObject on the prefixed key. If an encrypted bucket is used, it needs to allow ksm:Encrypt, kms:Decrypt and kms:GenerateDataKey with the KMS key ARN. arn-role can be omitted if minio is used.

11.1.12 creator.s3.bucket-name

Default: `bg-creator`

11.1.13 creator.s3.key-prefix

Path used by the creator inside the bucket. Must be empty or end with a `/`.

11.1.14 creator.s3.minio

Minio options. Must be omitted if AWS S3 buckets are used.

11.1.15 creator.s3.minio.external-address

External address of minio server.

11.1.16 creator.s3.minio.internal-address

Internal address of minio server.

11.1.17 creator.s3.region

Default: `us-east-1`

11.1.18 creator.s3.secret-key

Secret access key to access s3/minio bucket. If creator.s3.accessKey or creator.s3.secretKey is omitted DefaultCredentialProvider is used to find credentials.

11.1.19 creator.preview.s3.lifecycle.skip

Prevent that the creator tries to add a s3 lifecycle rule for the preview path rule.

Default: `true`

11.1.20 creator.templates.access

Path to the one and only templates access yaml file.

Example: `file:/absolute/path/to/templates-access.yml`

11.1.21 creator.templates.external-upload

Enable the external document Upload in the template selection if mode is 'create'. Only 'true' is true.

Default: `true`

11.1.22 creator.templates.location

Path to the given templates.

Example: `file:/absolute/path/to/directory/of/templates`

11.1.23 creator.templates.options

Path to the one and only templates options yaml file.

Example: `file:/absolute/path/tp/templates-options.yml`

11.1.24 creator.templates.customizing.location

Path to the customizing directory.

Example: `file:/absolute/path/to/directory/with/customizing`

11.1.25 creator.web-log-level

Only log entries with a level equal or 'above' will be logged on the server. Must be debug, info, warn or error.

Default: `info`

11.1.26 logging.config

Path to custom log4j2.xml configuration file. If not set a default log4j2.xml is used.

11.1.27 server.servlet.context-path

Default: `/creator`

11.1.28 spring.security.oauth2.resourceserver.jwt.issuer-uri

URL of the JWT issuer. This is used to verify the JWT token.

11.1.29 creator.jwt-max-age

Deprecated since 2023

Seconds until the jwt expires. Must be 0 < jwt-max-age <= Curator JSESSIONID Max-Age.

Default: `0`

11.1.30 creator.keep-alive-interval

Deprecated since 2023

Send a keep-alive request every keep-alive-interval seconds. Must be 0 < keep-alive-interval < Curator JSESSIONID Max-Age.

Default: `0`

11.1.31 creator.keep-alive-retry-delay

Deprecated since 2023

Time in seconds to wait before retrying a failed keep-alive request. Must be 0 < keep-alive-retry-delay < Curator JSESSIONID Max-Age.

Default: `0`

11.2 Cookies

Name	Description	HTTP-ONLY
jwt	Used for authentication	true
jwtMaxAge	Used to retrieve new jwts	false
jwtKeepAliveInterval	Used to retrieve new jwts	false
jwtKeepAliveRetryDelay	Used to retrieve new jwts	false