COREMEDIA CONTENT CLOUD

Deployment Manual

COREMEDIA

List of Figures

List of Tables

1. Preface

1.1 Audience

This manual is intended for architects and developers who want to learn about the concepts of CoreMedia's flexible deployment. You should be familiar with configuration management and application deployment concepts.

Additionally, you should be familiar with *CoreMedia CMS*, *Spring Framework* and *Apache Maven*.

1.2 Typographic Conventions

CoreMedia uses different fonts and types in order to label different elements. The follow-
ing table lists typographic conventions for this documentation:

Element	Typographic format	Example
Source code Command line entries Parameter and values Class and method names Packages and modules	Courier new	`cm systeminfo start`
Menu names and entries	Bold, linked with |	Open the menu entry **Format|Normal**
Field names CoreMedia Components Applications	Italic	Enter in the field *Heading* The *CoreMedia Component* Use *Chef*
Entries	In quotation marks	Enter "On"
(Simultaneously) pressed keys	Bracketed in "<>", linked with "+"	Press the keys <Ctrl>+<A>
Emphasis	Italic	It is *not* saved
Buttons	Bold, with square brackets	Click on the [OK] button
Code lines in code examples which continue in the next line	\	`cm systeminfo \` `-u user`

Table 1.1. Typographic conventions

In addition, these symbols can mark single paragraphs:

Pictograph	Description
	Tip: This denotes a best practice or a recommendation.
	Warning: Please pay special attention to the text.
	Danger: The violation of these rules causes severe damage.

Table 1.2. Pictographs

1.3 CoreMedia Services

This section describes the CoreMedia services that support you in running a CoreMedia system successfully. You will find all the URLs that guide you to the right places. For most of the services you need a CoreMedia account. See Section 1.3.1, "Registration" [5] for details on how to register.

> **NOTE**
>
> CoreMedia User Orientation for CoreMedia Developers and Partners
>
> Find the latest overview of all CoreMedia services and further references at:
>
> http://documentation.coremedia.com/new-user-orientation

- Section 1.3.1, "Registration" [5] describes how to register for the usage of the services.
- Section 1.3.2, "CoreMedia Releases" [6] describes where to find the download of the software.
- Section 1.3.3, "Documentation" [6] describes the CoreMedia documentation. This includes an overview of the manuals and the URL where to find the documentation.
- Section 1.3.4, "CoreMedia Training" [10] describes CoreMedia training. This includes the training calendar,the curriculum and certification information.
- Section 1.3.5, "CoreMedia Support" [10] describes the CoreMedia support.

1.3.1 Registration

In order to use CoreMedia services you need to register. Please, start your initial registration via the CoreMedia website. Afterwards, contact the CoreMedia Support [see Section 1.3.5, "CoreMedia Support" [10]] by email to request further access depending on your customer, partner or freelancer status so that you can use the CoreMedia services.

1.3.2 CoreMedia Releases

Downloading and Upgrading the Blueprint Workspace

CoreMedia provides its software as a Maven based workspace. You can download the current workspace or older releases via the following URL:

https://releases.coremedia.com/cmcc-10

Refer to our Blueprint Github mirror repository for recommendations to upgrade the workspace either via Git or patch files.

> **NOTE**
> If you encounter a 404 error then you are probably not logged in at GitHub or do not have sufficient permissions yet. See Section 1.3.1, "Registration" [5] for details about the registration process. If the problems persist, try clearing your browser cache and cookies.

Maven artifacts

CoreMedia provides its release artifacts via Maven under the following URL:

https://repository.coremedia.com

You have to add your CoreMedia credentials to your Maven settings file as described in section Blueprint Developer Manual .

License files

You need license files to run the CoreMedia system. Contact the support (see Section 1.3.5, "CoreMedia Support" [10]) to get your licences.

1.3.3 Documentation

CoreMedia provides extensive manuals and Javadoc as PDF files and as online documentation at the following URL:

https://documentation.coremedia.com/cmcc-10

The manuals have the following content and use cases:

Manual	Audience	Content
Adaptive Personalization Manual	Developers, architects, administrators	This manual describes the configuration of and development with *Adaptive Personalization*, the CoreMedia module for personalized websites. You will learn how to configure the GUI used in *CoreMedia Studio*, how to use predefined contexts and how to develop your own extensions.
Analytics Connectors Manual	Developers, architects, administrators	This manual describes how you can connect your CoreMedia website with external analytic services, such as Google Analytics.
Blueprint Developer Manual	Developers, architects, administrators	This manual gives an overview over the structure and features of *CoreMedia Content Cloud*. It describes the content type model, the *Studio* extensions, folder and user rights concept and many more details. It also describes administrative tasks for the features.
		It also describes the concepts and usage of the project workspace in which you develop your CoreMedia extensions. You will find a description of the Maven structure, the virtualization concept, learn how to perform a release and many more.
Connector Manuals	Developers, administrators	This manuals gives an overview over the use cases of the eCommerce integration. It describes the deployment of the Commerce Connector and how to connect it with the CoreMedia and eCommerce system.
Content Application Developer Manual	Developers, architects	This manual describes concepts and development of the *Content Application Engine (CAE)*. You will learn how to write JSP or Freemarker templates that access the other CoreMedia modules and use the sophisticated caching mechanisms of the CAE.
Content Server Manual	Developers, architects, administrators	This manual describes the concepts and administration of the main CoreMedia component, the *Content Server*. You will learn about the content type model which lies at the heart of a CoreMedia system, about user and rights management, database configuration, and more.

Manual	Audience	Content
Deployment Manual	Developers, architects, administrators	This manual describes the concepts and usage of the CoreMedia deployment artifacts. That is the deployment archive and the Docker setup. You will also find an overview of the properties required to configure the deployed system.
Elastic Social Manual	Developers, architects, administrators	This manual describes the concepts and administration of the *Elastic Social* module and how you can integrate it into your websites.
Frontend Developer Manual	Frontend Developers	This manual describes the concepts and usage of the Frontend Workspace. You will learn about the structure of this workspace, the CoreMedia themes and bricks concept, the CoreMedia Freemarker facade API, how to develop your own themes and how to upload your themes to the CoreMedia system.
Headless Server Developer Manual	Frontend Developers, administrators	This manual describes the concepts and usage of the *Headless Server*. You will learn how to deploy the Headless Server and how to use its endpoints for your sites.
Importer Manual	Developers, architects	This manual describes the structure of the internal CoreMedia XML format used for storing data, how you set up an *Importer* application and how you define the transformations that convert your content into CoreMedia content.
Operations Basics Manual	Developers, administrators	This manual describes some overall concepts such as the communication between the components, how to set up secure connections, how to start application or the usage of the watchdog component.
Search Manual	Developers, architects, administrators	This manual describes the configuration and customization of the *CoreMedia Search Engine* and the two feeder applications: the *Content Feeder* and the *CAE Feeder*.
Site Manager Developer Manual	Developers, architects, administrators	This manual describes the configuration and customization of *Site Manager*, the Java based stand-alone application for administrative tasks. You will learn how to configure the *Site Manager* with property files and

Manual	Audience	Content
		XML files and how to develop your own extensions using the *Site Manager API*.
		The Site Manager is deprecated for editorial work.
Studio Developer Manual	Developers, architects	This manual describes the concepts and extension of *CoreMedia Studio*. You will learn about the underlying concepts, how to use the development environment and how to customize *Studio* to your needs.
Studio User Manual	Editors	This manual describes the usage of *CoreMedia Studio* for editorial and administrative work. It also describes the usage of the *Adaptive Personalization* and *Elastic Social* GUI that are integrated into *Studio*.
Studio Benutzerhandbuch	Editors	The Studio User Manual but in German.
Supported Environments	Developers, architects, administrators	This document lists the third-party environments with which you can use the CoreMedia system, Java versions or operation systems for example.
Unified API Developer Manual	Developers, architects	This manual describes the concepts and usage of the *CoreMedia Unified API*, which is the recommended API for most applications. This includes access to the content repository, the workflow repository and the user repository.
Utilized Open Source Software & 3rd Party Licenses	Developers, architects, administrators	This manual lists the third-party software used by CoreMedia and lists, when required, the licence texts.
Workflow Manual	Developers, architects, administrators	This manual describes the *Workflow Server*. This includes the administration of the server, the development of workflows using the XML language and the development of extensions.

Table 1.3. CoreMedia manuals

If you have comments or questions about CoreMedia's manuals, contact the Documentation department:

Email: documentation@coremedia.com

1.3.4 CoreMedia Training

CoreMedia's training department provides you with the training for your CoreMedia projects either in the CoreMedia training center or at your own location.

You will find information about the CoreMedia training program, the training schedule and the CoreMedia certification program at the following URL:

http://www.coremedia.com/training

Contact the Training department at the following email address:

Email: training@coremedia.com

1.3.5 CoreMedia Support

CoreMedia's support is located in Hamburg and accepts your support requests between 9 am and 6 pm MET. If you have subscribed to 24/7 support, you can always reach the support using the phone number provided to you.

To submit a support ticket, track your submitted tickets or receive access to our forums visit the CoreMedia Online Support at:

http://support.coremedia.com/

Do not forget to request further access via email after your initial registration as described in Section 1.3.1, "Registration" [5]. The support email address is:

Email: support@coremedia.com

Create a support request

CoreMedia systems are distributed systems that have a rather complex structure. This includes, for example, databases, hardware, operating systems, drivers, virtual machines, class libraries and customized code in many different combinations. That's why CoreMedia needs detailed information about the environment for a support case. In order to track down your problem, provide the following information:

Support request

- Which CoreMedia component(s) did the problem occur with (include the release number)?
- Which database is in use (version, drivers)?
- Which operating system(s) is/are in use?
- Which Java environment is in use?
- Which customizations have been implemented?

- A full description of the problem (as detailed as possible)
- Can the error be reproduced? If yes, give a description please.
- How are the security settings (firewall)?

In addition, log files are the most valuable source of information.

To put it in a nutshell, CoreMedia needs:

Support checklist

1. a person in charge (ideally, the CoreMedia system administrator)

2. extensive and sufficient system specifications

3. detailed error description

4. log files for the affected component(s)

5. if required, system files

An essential feature for the CoreMedia system administration is the output log of Java processes and CoreMedia components. They're often the only source of information for error tracking and solving. All protocolling services should run at the highest log level that is possible in the system context. For a fast breakdown, you should be logging at debug level. The location where component log output is written is specified in its `logback.xml` file.

Log files

Which Log File?

Mostly at least two CoreMedia components are involved in errors. In most cases, the *Content Server* log files together with the log file from the client. If you are able locate the problem exactly, solving the problem becomes much easier.

Where do I Find the Log Files?

By default, log files can be found in the CoreMedia component's installation directory in `/var/logs` or for web applications in the `logs/` directory of the servlet container `.` See Section 4.7, "Logging" in *Operations Basics* for details.

1.4 Changelog

In this chapter you will find a table with all major changes made in this manual.

Section	Version	Description

Table 1.4. Changes

2. Deployment Archive

There are several complete automation solutions in the market for both infrastructure and applications that take you all the way from development to production.

CoreMedia provides an out-of-the-box deployment for systems running a supported Linux OS (see http://bit.ly/cmcc-10-supported-environments). The deployment is based on Spring-Boot applications and the Chef provisioning system. To achieve a simple but flexible solution, all CoreMedia application artifacts are therefore packaged together with the Chef resources into a deployment-archive, which can then be applied to all machines.

The following chapters will explain how to build, configure and install the deployment-archive. In a closing chapter a brief introduction will be given how to extend the provided setup with additional deployment aspect. For further customizations please refer to the official Chef documentation.

2.1 Quickstart

The deployment archive offers you an installer based experience with the possibility to extend it to your needs. See Section 2.2, "Deployment Archive Structure" [15] to get an overview of the deployment archive. To install a system using the deployment-archive approach, follow the steps below:

1. Build the workspace with Maven:

   ```
   mvn clean install -Pwith-chef
   ```

   See Section 2.3, "Building the Deployment Archive" [16] for details.
2. Upload the deployment archive to your target machine. You will find the archive at `global/deployment/chef/target/deployment-archive.zip`
3. Upload a license file to the target machine and make the folder and license file readable for all users.
4. Extract the archive, and configure all necessary attributes in the node file of your choice below `EXTRACTED_DIR/nodes`. At least configure the license and hostname attributes.
5. Make sure that Chef in the appropriate version is installed and run the `EXTRACTED_DIR/deploy.sh` script and chose the configured node file as the installation configuration source. Make sure that you have the appropriate rights when executing the script. Many of the installation steps in the cookbooks require `root` rights. See Section 2.5, "Running the deployment script" [21] for details.

2.2 Deployment Archive Structure

The deployment-archive is a zip based archive that packages all CoreMedia Spring-Boot based applications, all necessary Chef cookbooks and roles together with a simple utility script. The archive though does not package the Chef installation tool and it is required to install Chef 14 in advance.

```
deployment-archive.zip
    |- maven-repo
    |    `com/coremedia/blueprint/boot
    |         |- cae-live-app/<VERSION>/cae-live-app-<VERSION>.jar
    |         `- ...
    |- chef-repo
    |    |- cookbooks
    |    |    |- blueprint-cmcc
    |    |    |- blueprint-dev-tooling
    |    |    |- blueprint
    |    |    |- blueprint-base
    |    |    |- blueprint-yum
    |    |    |- blueprint-mysql
    |    |    |- blueprint-postgresql
    |    |    |- blueprint-mongodb
    |    |    |- blueprint-spring-boot
    |    |    |- blueprint-solr
    |    |    |- blueprint-tools
    |    |    `- blueprint-proxy
    |    |- coremedia-cookbooks
    |    |    `- coremedia_maven
    |    |- thirdparty-cookbooks
    |    |    |- apache2
    |    |    |- ..
    |    |    `- yum
    |    |- environments
    |    |- roles
    |    |- nodes
    |    `- .chef
    |        `- solo.rb
    |- deploy.sh
     `-- chef
```

As you can see from the listing above, the CoreMedia application artifacts are packaged in a Maven repository layout using groupId, artifactId and version folders whereas the Chef resources are packaged using a Chef repository layout with cookbooks, roles, environments folders.

2.3 Building the Deployment Archive

> **Prerequisite**
>
> You have to build the CoreMedia workspace first, otherwise the build will fail due to missing dependencies.

To build the deployment archive, you simply need to build the Maven module below `global/deployment/chef` resulting in a `deployment-archive.zip` file below the `target` folder of that module.

2.4 Configuring the Deployment Archive

To configure the deployment-archive you have to understand a bit about the anatomy of a Chef run. The deployment-archive uses `chef-solo` which itself allows to define all attribute configurations within a JSON file. In the deployment archive you will find these files below the `nodes` folder, which corresponds to the `global/deploy ment/chef/nodes` folder in the workspace.

Before you take a look at some example files, you should understand the basic structure of a node JSON file and the actual structure defined by the CoreMedia Chef cookbooks.

Basic node structure

The basic structure of each node file consists of Chef attributes in form of an attribute hash and a `run_list` array, that can contain either roles or recipes.

CoreMedia attributes structure

The CoreMedia attributes structure is defined by the CoreMedia cookbooks below `global/deployment/chef/cookbooks`. It is recommended to review the `README.md` Markdown files of each cookbook to see what attributes can be set. Because the set of application properties CoreMedia applications can be configured with is large and changing, you won't find application property attributes defined by the cookbooks but instead a generic way of defining those properties using a hash map associated with each application. A detailed documentation how that is done can be found in the `global/deployment/chef/cookbooks/blueprint-spring-boot/README.adoc` file.

> **NOTE** ⓘ
>
> An attribute declaration in Ruby style corresponds to the same nested structure in JSON.
>
> ```
> node['blueprint']['studio']['foo'] = 'bar'
> ```
>
> maps to
>
> ```
> {
>  "blueprint": {
>    "studio": {
>      "foo": "bar"
>    }
>  }
> }
> ```

In the following figure you see the sketched attribute tree. There are 5 different categories of attributes:

Global attributes	These attributes are mostly defined in the `blueprint-base` cookbook, they define common locations, users etc.
Application attributes	These attributes are defined in the `blueprint-base` cookbook and in recipes of the `blueprint-spring-boot` cookbook.
Tool attributes	These attributes are defined in the `blueprint-tools` cookbook.
Spring Boot attributes	These attributes are defined in the `blueprint-spring-boot` cookbook.
Proxy attributes	These attributes are defined in the `blueprint-proxy` cookbook.

```
"blueprint": {
  "user": "coremedia",
  "group": "coremedia",
  "base_dir": "/opt/coremedia",
  "log_dir": "/var/log/coremedia",
  "cache_dir": "/var/cache/coremedia",
  "maven_repository_url": "file:///maven-repo",
  "hostname": "192.168.252.100.xip.io",
  "default_version": "1-SNAPSHOT",              Global Attributes

  "apps": {
    "studio-server": {
      "application.properties": {
        "solr.url": "http://solr:40080/solr"
      }
    },
  },                                        Application Attributes

  "tools": {
    "content-management-server": {},
  },                                           Tool Attributes

  "proxy": {
    "ssl_proxy_verify": false,
    "virtual_host": {
      "studio": {},
      "preview": {}
    }
  },                                           Proxy Attributes

  "spring-boot": {
    "java_home": "/usr/lib/jvm/java",
    "java_opts": {
      "oom_handling": "-XX:+ExitOnOutOfMemoryError"
    },
    "studio-server": {
      "heap": "2048m"
    }
  }                                        Spring Boot Attributes
}
```

Figure 2.1. Attribute Structure

The run_list

The `run_list` array within the node file can contain roles and recipes. In case of a role, all recipes defined in that role will be populated during the chef run. You should always include the `role[base]` at the start of your run list and you are not bound to the roles included in the workspace. You are free to modify, replace or delete them to match your deployment needs. The roles included in the workspace are only a pro-

position for a very simple setup. In most cases it will be more transparent to use the recipes directly.

The node JSON file

If you take a look at an example node file shown in the listing below, you see a configuration how to install a content-management-server configured using a PostgreSQL database.

```
{
  "blueprint": {
    "maven_repository_url": "file://localhost/tmp/maven-repo",
    "apps":{
      "content-management-server": {
        "application.properties": {
          "cap.server.license": "properties/corem/license.zip",
          "sql.store.driver": "org.postgresql.Driver",
          "sql.store.url": "jdbc:postgresql://postgresql:5432/coremedia",
          "sql.store.dbProperties": "corem/postgresql",
          "sql.store.user": "cm_management",
          "sql.store.password": "cm_management"
        }
      }
    }
  },
  "run_list": [
    "role[base]",
    "recipe[blueprint-spring-boot::content-management-server]"
  ]
}
```

If you now want to deploy an application that requires a connection to the content-management-server you can simply create another node json file and configure the connection properties accordingly. Let's install the content-feeder application on a different machine, then you could create a node file like the one below.

```
{
  "blueprint": {
    "maven_repository_url": "file://localhost/tmp/maven-repo",
    "apps":{
      "content-feeder": {
        "application.properties": {
          "repository.url": "http://<CMS HOST>:40180/coremedia/ior",
          "solr.url": "http://<SOLR HOST>:8983/solr"
        }
      }
    }
  },
  "run_list": [
    "role[base]",
    "recipe[blueprint-spring-boot::content-feeder]"
  ]
}
```

2.5 Running the deployment script

After extracting the `deployment-archive.zip` on the target machine, you can run the `deploy.sh`, it will guide you to some very basic questions for the installation. In fact it queries, the environment the node is placed in and the node file that should be used for installation. Let's examine, what choices you have.

Environment Option

The deploy.sh script will prompt you with all available environments and this will be determined by the environment JSON files found below `environments` in the extracted directory.

The Chef environment concept is a topology based construct to group servers into environments. The most common environment set is based on the three environments `development`, `staging` and `production`. In the deployment-archive these environments are represented by the environment cookbook `blueprint`.

An environment cookbook encapsulates attributes per environment in environment specific recipes. By doing so, these attributes are packaged as a cookbook and can be versioned and deployed safely. There can only be one environment cookbook in a *Chef* setup.

Setting the environment to `development` will result in running the `recipe[blueprint::_development]` recipe at the start of the chef run, when the `role[base]` is included.

In order to keep your node files clean and environment independent as possible, you should configure as many aspects as possible in the corresponding environment recipes.

Node Config Option

The deploy.sh script will look for all node files in the `nodes` folder in the extracted directory.

2.6 Validating the Installation

After the chef installation run succeeded, you can validate the correct installation by reviewing the installed application and check the applications log files.

```
/opt/coremedia/<name>
     |- application.properties
     |- <application>.jar
     |- log
     |- post-start-check.sh
     |- jmx-remote.access
     `- jmx-remote.password

/etc/systemd/system/<name>.conf
/var/log/coremedia/<name> -> <path>/log/<name>.log
```

2.7 Customize the Deployment Archive

There are several ways to customize or extend the deployment archive. In this section you will find some quick examples on how to proceed.

Change installation destination

By default, all applications will be installed below `/opt/coremedia`. If you want to change this you simply need to use a different value for the `node['blueprint']['base_dir']` attribute. There are some loosely related attributes you may also change like the Solr home directory `node['blueprint']['solr']['solr_home']`.

Change log file destination

By default, the log files will be written to a directory below the applications directory, for example, `/opt/coremedia/studio-server/log`. For convenience a symbolic link will be created for the to access all logs below a common logging directory configurable by the attribute `node['blueprint']['log_dir']`.

Customize the Roles

The roles provided below the `roles` directory are not mandatory and just represent a minimal example. Please feel free to adapt the setup to your needs. You can remove them completely and directly use the recipes in your run list or define new ones to match your desired topology. Using only chef-solo without a Chef server, roles only have little use.

Adding additional Chef Cookbooks

You can add cookbooks to install any kind of software, for example, by downloading them from the Chef Supermarket and adding them to the `thirdparty-cookbooks` directory, but you have to add all transitive cookbook dependencies as well. Beside that, make sure that the actual Chef version is supported by the cookbook and that there are no dependency resolution errors.

Adding more Spring Boot applications

It is very simple to add more applications to the deployment archive but you have to adapt the right places in the workspace:

- At first, you need to make sure the artifact is packaged into the archive. To achieve this, you need to add a Maven dependency to the application artifact in the `deployment/chef/pom.xml`, that is:

```
<dependency>
  <groupId>my.org</groupId>
  <artifactId>my-application</artifactId>
  <version>1.0</version>
</dependency>
```

If the application artifact is part of your blueprint workspace, you should use `${project.groupId}` and `${project.version}` to follow Maven best practice.

You should verify, that after rebuilding the deployment archive, that your application artifact is packaged within the `maven-repo` folder of the archive.

- Now that the application artifact is available, you can add a new recipe in the `blueprint-spring-boot` cookbook by copying one of the simpler recipes, for example, `headless-server-live.rb`. Now you only need to replace the `service_name` at the top and replace all basic `application.properties` attribute setters with the ones you require. Make sure to you use `node.default_unless` to set them, otherwise you won't be able to overwrite them in the node JSON file.
- The last step is to add the newly created recipe to the runlist. This can be either done in one of the roles, a new role or simply in the node JSON file.

2.8 Deployment Archive Reference

In this chapter, you will find different overviews of the default settings of the deployment archive.

2.8.1 Service and Logfile Names

The service names are defined in the recipe of each service. Although, the name is being set in a variable at the top of the recipe, changing the name will affect notifications and subscription dependencies to or from other resources. By default, there are no such dependencies but renaming should be done with care.

Service names

The services log into files named after the service name. That is, for example, the *Content Management Server* logs into the `content-management-server.log` file.

Logfile names

> **NOTE**
> The CAE Live service has the option to be installed multiple times on the same node, for each instance an incremental number is suffixed to the name. If only one instance is installed, the service name is therefore `cae-live-1`

Component Name	Service Name
Solr Search Engine	solr
Content Management Server	content-management-server
Master Live Server	master-live Server
Workflow Server	workflow-server
Content Feeder	content-feeder
User Changes	user-changes
Elastic Worker	elastic-worker
CAE Feeder Preview	cae-feeder-preview

Component Name	Service Name
CAE Feeder Live	cae-feeder-live
CAE Preview	cae-preview
Studio Rest Server	studio-server
Headless Server Preview	headless-server-preview
Headless Server Live	headless-server-live
Replication Live Server	replication-live-server
CAE Live	cae-live-1
Commerce Adapter IBM WCS	commerce-adapter-wcs
Commerce Adapter SAP Hybris	commerce-adapter-hybris
Commerce Adapter Salesforce Commerce Cloud	commerce-adapter-sfcc

Table 2.1. Service Names

2.8.2 Installation Paths

In the default configuration all Blueprint applications are installed below `/opt/coremedia`.

Component Name	Installation Path
Solr Search Engine	/opt/coremedia/solr
Content Management Server	/opt/coremedia/content-management-server
Master Live Server	/opt/coremedia/master-live-erver
Workflow Server	/opt/coremedia/workflow-server
Content Feeder	/opt/coremedia/content-feeder

Component Name	Installation Path
User Changes	/opt/coremedia/user-changes
Elastic Worker	/opt/coremedia/elastic-worker
CAE Feeder Preview	/opt/coremedia/caef-eeder-preview
CAE Feeder Live	/opt/coremedia/cae-feeder-live
CAE Preview	/opt/coremedia/cae-preview
Studio	/opt/coremedia/studio
Headless Server Preview	/opt/coremedia/headless-server-preview
Headless Server Live	/opt/coremedia/headless-server-live
Replication Live Server	/opt/coremedia/replication-live-server
CAE Live	/opt/coremedia/cae-live-1
Commerce Adapter IBM WCS	/opt/coremedia/commerce-adapter-wcs
Commerce Adapter SAP Hybris	/opt/coremedia/commerce-adapter-hybris
Commerce Adapter Salesforce Commerce Cloud	/opt/coremedia/commerce-adapter-sfcc

Table 2.2. Service Installation Paths

Tool Name	Installation Path
Content Management Server Tools	/opt/coremedia/content-management-server-tools
Master Live Server Tools	/opt/coremedia/master-live-server-tools
Workflow Server	/opt/coremedia/workflow-server-tools
CAE Feeder Preview Tools	/opt/coremedia/caefeeder-preview-tools

Tool Name	Installation Path
Theme Importer	/opt/coremedia/theme-importer-tools

Table 2.3. Tools Installation Paths

2.8.3 Port Conventions

The Deployment Archive allows a simple setup where all services are installed on the same single server instance. To prevent port conflicts each CoreMedia Spring Boot application is therefore configured with a unique set of ports. To keep the port range small, the complete port is built upon a three digit prefix representing the service and a two digit suffix representing the interface. To keep it simple the port prefixes are ascending according to the correct starting order.

> **NOTE**
> CoreMedia recommends to stay with our default configuration based on our recommended port conventions.

Component Name	Port Prefix
Solr Search Engine	400
Content Management Server	401
Master Live Server	402
Workflow Server	403
Content Feeder	404
User Changes	405
Elastic Worker	406
CAE Feeder Preview	407
CAE Feeder Live	408

Component Name	Port Prefix
CAE Preview	409
Studio REST Server	410
Headless Server Preview	411
Headless Server Live	412
Replication Live Server	420
CAE Live	421
Commerce Adapter Salesforce Commerce Cloud	441
Commerce Adapter SAP Hybris	442
Commerce Adapter IBM WCS	443

Table 2.4. Component Port Prefix

> **NOTE**
>
> Remember, that if you install more than one live CAE on a node using the Chef `in stances` attribute, the last digit of the port prefix will be the instance number.

Protocol / Service	Port Suffix
CORBA	83
HTTP	80
HTTP Management	81
gRPC	65
Tomcat JMX Registry Port (RMI)	99

Protocol / Service	Port Suffix
Tomcat JMX Server Port (RMI)	98

Table 2.5. Protocol / Service Port Suffix
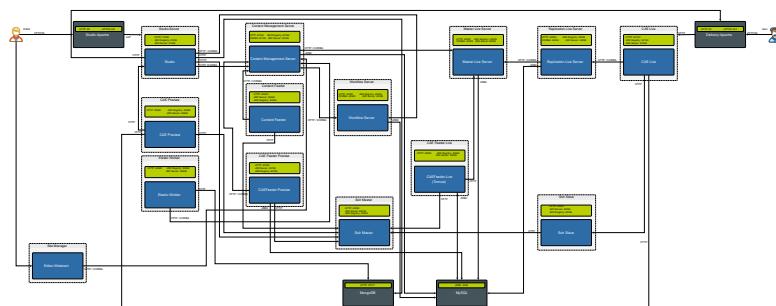
Figure 2.2. CMS Deployment Ports

Service	Port
MySQL	3306
Mongo DB	27017

Table 2.6. Third-Party Services

> **NOTE**
>
> Your ports should not overlap with the default ephemeral port range on your operation systems. You can decrease the ephemeral port range with the cost of not optimizing your TCP/IP stack. This port schema does not overlap with the ephemeral port range so there is no need to adjust the range. However, if you need to decrease the ephemeral port range. In the `blueprint-base::default` recipe, the Chef inbuild `sysctl` resource is being used to decrease the ephemeral port range to start at `45000`.

2.9 Deployment Archive Migration Relevant Changes

Major Changes

- Spring-Boot application `JAR` files instead of webapp `WAR` files as main artifacts.
- `blueprint-spring-boot` cookbook instead of `blueprint-tomcat` cookbook.
- Tomcat version is now defined at build time of the artifacts instead of deployment time. The Tomcat is now embedded in the Spring-Boot application jar.
- SystemD init system instead of SystemV
- Studio has been split up into a `studio-server` app and a `studio-client` part consisting only of static resources served by an Apache HTTPd web server. As a result, all application properties previously defined below `node['blueprint']['webapps']['studio']['application.properties']` must now be defined below `node['blueprint']['apps']['studio-server']['application.properties']`.
- To deploy the Studio Client part, two recipes are required in your run list.
  - `blueprint-spring-boot::studio-client` to download and extract the static resources.
  - `blueprint-proxy::studio` to configure the Apache configuration.

Minor Changes

- The `blueprint-spring-boot` cookbook does not provide a Chef definition (macro) for CoreMedia services as did the `blueprint-tomcat` cookbook. The `blueprint-spring-boot` cookbook instead provides a `spring_boot_application` resource, that abstracts from the process of downloading the Maven artifact, rendering the configuration properties file and the installation of the SystemD service. To prevent unnecessary restarts at first installation, all notifications are directed to a `ruby_block` resource, which aggregates all triggered actions into either a `start` or `restart` action at the corresponding `service` resource.

```
template 'my_service_custom_conf' do
  # renders additional application config. If changed trigger restart check

  notifies :create, 'ruby_block[restart_my_service]', :immediately
end

spring_boot_application 'my_application' do
  # install and configure application. If changed trigger restart check
  notifies :create, 'ruby_block[restart_my_service]', :immediately
end

service 'my_service' do
  # manage service state
  action [:enable, :start]
end

ruby_block 'restart_my_service' do
  # restart check
  block do
    # some code removing unnecessary restart action if start already present

  end
  action :nothing
end
```

Configuration Changes

With the old deployment archive each applications configuration was split into three parts:

- The application configuration:
  - Maven coordinates of the artifact
  - Application properties
- The global installation configuration:
  - JVM options
  - JMX options
  - logging configuration
- The service specific installation configuration which allows overriding the global configuration but also adds specific configuration options like:
  - heap settings
  - enabling / disabling the service for autostart

With the new deployment archive this logic is being kept, except for the hashes being used:

- Old deployment archive
  - application configuration below node['blueprint']['webapps']

- global installation configuration below `node['blueprint']['tomcat']`
- service specific installation configuration below `node['blueprint']['tomcat']`
- New deployment archive
  - application configuration below `node['blueprint']['apps']['<SERVICE KEY>']`
  - global installation configuration below `node['blueprint']['spring-boot']`
  - service specific installation configuration below `node['blueprint']['spring-boot']['<SERVICE KEY>']`

For the application configuration below `node['blueprint']['webapps']['<SERVICE_KEY'>']['application.properties']` an automated migration is being done in the recipes by copying all previous attributes to the new attribute path.

Some of the old installation configurations have been removed or have a new equivalent. This includes:

- `node['blueprint']['tomcat']['shutdown_force']` - not supported anymore
- `node['blueprint']['tomcat']['shutdown_wait']` - not supported anymore
- `node['blueprint']['tomcat']['clean_log_dir_on_start']` - not supported anymore
- `node['blueprint']['tomcat']['keep_old_instances']` - not supported anymore
- `node['blueprint']['tomcat']['context_config']` - not supported anymore, this should be done programatically using Spring Boot
- `node['blueprint']['tomcat']['logback_config']` - not supported anymore, logback configuration can be passed in using `application.properties` or `boot_opts`
- `node['blueprint']['tomcat']['<SERVICE_KEY>']['perm']` - Since Java 8 there is no perm space configurable anymore
- `node['blueprint']['tomcat']['common_libs']` - not supported anymore, package extra libs within your Spring Boot app.
- `node['blueprint']['tomcat']['catalina_opts']` - use one of
  - `node['blueprint']['spring-boot']['java_opts']`
  - `node['blueprint']['spring-boot']['<SERVICE KEY>']['java_opts']`

- `node['blueprint']['tomcat']['<SERVICE_KEY>']['port_prefix']` - not supported anymore. Port prefixes are no longer configurable in the deployment. Ports are being specified using the Spring Boot properties within the application artifacts. However, you can set boot opts or application properties to change the ports:
    - `node['blueprint']['spring-boot']['studio-server']['boot_opts']['server.port']`
    - `node['blueprint']['apps']['studio-server']['application.properties']['server.port']`

    Keep in mind that if you do so, you need manually find references to the old ports and change them, this may include hard coded references in recipes as well as properties of services using an endpoint of the modified service.

3. Docker Setup

CoreMedia 10 introduces an application modularity that supports individual application lifecycles, including a modularized development, release and deployment process. Using Spring-Boot standards and container technologies at its core, our goal is to enable our applications to become more cloud native and easier to integrate with available cloud technologies.

The docker setup provided with CoreMedia 10, consists of:

- a Maven build process to create the images
- a Docker-Compose setup to start a development environment

To bring CoreMedia 10 to production using the Docker images, you have one of the following choices:

- use the Docker images to start containers and benefit from the process abstraction container provide.
- implement a Docker Swarm setup to benefit from the simplicity of a Swarm orchestration layer.
- implement a Kubernetes Deployment to benefit from the flexibility and power, Kubernetes provides to create a cloud native cluster deployment.

Obviously Kubernetes is the most promising technology, but the initial complexity and the steep learning curve in addition with the requirement to build and maintain the knowledge of this technology may not fit each ones requirements.

To help you with this choice, this chapter will provide you with an overview of the build process for our docker images, the conventions and configuration options the docker images provide and how to use them for a production deployment using the containers as a simple process abstraction.

3.1 Docker Images Build Process

To build the Docker images, dockerfile-maven-plugin is used. A plugin that only integrates Maven with the Docker Daemon API to build, tag or push images. It only requires a `Dockerfile` to run and does not provide any kind of abstraction to the directives within a `Dockerfile`.

The benefit the plugin provides for the build process, is that you are able to retrieve Maven artifacts from the Maven reactor or a Maven repository to be consumed by the Docker image build process. It also allows us to pass in Docker build arguments from Maven properties and therefore integrates the Docker image build process into any kind of continuous integration system with the ability to pass in context specific parameters.

3.1.1 The Maven Structure

Each Spring-Boot applications has the following Maven module structure involved in building Docker images:

Filesystem Layout:

```
apps
 `- <application name>
    |- docker
    |  |- <application name>
    |  |  |- Dockerfile
    |  |  |- src
    |  |  |  `- docker
    |  |  `- pom.xml
    |  `- pom.xml
    `- pom.xml
```

- The `Dockerfile` defines the Docker build process. In it, all available Dockerfile directives are available. Please visit Dockerfile Reference for more details. Be aware, that only files beside and beneath the folder where the `Dockerfile` resides, can be copied or added to the Docker image.

- The contents of the `src/docker` folder will be copied as-is into the `/coremedia` within the image. This copying is done by the Dockerfile directive:

```
COPY --chown=coremedia:coremedia src/docker /coremedia
```

- `apps/<application name>/docker/<application naem>/pom.xml`

  This is the Maven module that build the image. It should define the plugin executions to copy artifact to the Maven build dir to be processed from there by the `Dockerfile`. For the cae-live application this looks like the following snippets:

Maven `pom.xml` - dependencies:

```
<dependencies>
    <dependency>
        <groupId>${blueprint.boot.groupId}</groupId>
        <artifactId>cae-live-app</artifactId>
        <version>${blueprint.boot.version}</version>
        <scope>runtime</scope>
    </dependency>
</dependencies>
```

Maven `pom.xml` - build:

```
<plugin>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-dependency-plugin</artifactId>
    <executions>
      <execution>
        <id>copy-boot-jar</id>
        <goals>
          <goal>copy-dependencies</goal>
        </goals>
        <phase>generate-resources</phase>
        <configuration>
          <includeArtifactIds>cae-live-app</includeArtifactIds>
          <stripVersion>true</stripVersion>
          <outputDirectory>target</outputDirectory>
          <excludeTransitive>true</excludeTransitive>
        </configuration>
      </execution>
    </executions>
</plugin>
```

Docker `Dockerfile`:

```
COPY --chown=coremedia:coremedia target/cae-live-app.jar
/coremedia/application.jar
```

- `apps/<application name>/docker/pom.xml`

  This Maven module encapsulates the build process of the `dockerfile-maven-plugin`. The following list of Maven properties can be configured:

  - `docker.repository.prefix`

    This property defines the docker registry part of the Docker image name. By default, it is set to `coremedia` and results in an image name of `coremedia/${project.build.finalName}` for example, for the cae-live `coremedia/cae-live`.

  - `docker.image.tag`

    This property defines the tag of the image. By default, `latest` is used. In a CI environment, you should set this property with a reasonable value. Good candidates are Git hashes or the incremental build number of the build job. Do not use running tags like `latest` or `stable` in production.

  - `docker.java-application-base-image.repo`

This property defines the image repository for the `java-application-base` image, which is the base image for all Spring-Boot application images for Core-Media.

- `docker.java-application-base-image.tag`

The `java-application-base` properties are passed as build args to the Docker build process. This is defined in the `dockerfile-maven-plugin` definition:

Maven `pom.xml` - dockerfile-maven-plugin - here with shortened property names:

```
<plugin>
  <groupId>com.spotify</groupId>
  <artifactId>dockerfile-maven-plugin</artifactId>
  <configuration>
    <buildArgs combine.children="append">
      <BASE_IMAGE_REPO>${base-image.repo}</BASE_IMAGE_REPO>
      <BASE_IMAGE_TAG>${base-image.tag}</BASE_IMAGE_TAG>
    </buildArgs>
  </configuration>
</plugin>
```

And consumed in the `Dockerfile` of the application:

Docker `Dockerfile` - from directive:

```
ARG BASE_IMAGE_REPO
ARG BASE_IMAGE_TAG
FROM ${BASE_IMAGE_REPO}:${BASE_IMAGE_TAG}
```

3.1.2 The java-application-base Image

One of the major inventions Docker brought to software development is the ability to package applications as layers of filesystems, that when stacked upon each other result in the final application image. This technique allows docker to reuse and share common parts between multiple images at build and startup time and reduces the disk and network footprint drastically.

In the CoreMedia Docker setup, this technique is used by providing a common base image on the public DockerHub repository. The `java-application-base` image is the common base image for all our Spring-Boot application based images. It encapsulates our best practice to configure, start and manage CoreMedia applications in most secure way. It is provided in different flavors for different JDK base images on DockerHub and it is possible to fork the source repository on GitHub, to build a customized image.

To summarize the capabilities of the image, it provides:

- a JVM to run our applications

- a startup script chain to integrate arbitrary initialization steps before starting the Spring-Boot application
- a set of convenient script chain steps to let applications delay their startup until service dependencies are available
- a monitoring integration for Prometheus

For a detailed description of the base image, please visit the java-application-base DockerHub page.

3.2 Docker Container Startup

3.2.1 Startup Entrypoint and Command chain

When the Docker container is started, the command of the process is created by inspecting the `Entrypoint` and the `Cmd` metadata fields of the image. If you build the images will all defaults and you run

```
docker image inspect --format='{{.Config.Entrypoint}} - {{.Config.Cmd}}'
coremedia/cae-live
```

You will get a response like this:

```
[./entrypoint.sh] - [application]
```

This is the entrypoint script chain that will be processed at startup.

3.2.2 Health Check

Docker has a built-in health check mechanism that, if defined for an image, probes a defined command regularly to determine the health status of the container. Because containerization is all about automation of processes, health checks are not only important for monitoring purposes but even more important for triggering events in an automation chain in your cluster. Many tools depend on the status information the container daemon provides. The Traefik reverse proxy uses in our Docker development setup for example automatically removes a routing to a backend, when the backends health status is not `healthy`.

If you for example run

```
docker ps --format '{{.Names}} - {{.Status}}'
```

you will get a status report for all running containers like this:

```
elastic-worker - Up 15 hours (healthy)
user-changes - Up 15 hours (healthy)
cae-preview - Up 15 hours (healthy)
cae-live - Up 15 hours (healthy)
site-manager - Up 15 hours
studio-server - Up 15 hours (healthy)
overview - Up 15 hours
content-feeder - Up 15 hours (healthy)
cae-feeder-live - Up 15 hours (healthy)
```

```
commerce-adapter-wcs - Up 15 hours (healthy)
headless-server-preview - Up 15 hours (healthy)
master-live-server - Up 15 hours (healthy)
cae-feeder-preview - Up 15 hours (healthy)
headless-server-live - Up 15 hours (healthy)
studio-client - Up 15 hours (healthy)
workflow-server - Up 15 hours (healthy)
content-management-server - Up 15 hours (healthy)
```

The health check command is defined in the `Dockerfile` of the application, that is

```
HEALTHCHECK --start-period=30s --interval=30s --timeout=3s \
 CMD curl -Lf http://localhost:8081/actuator/health || exit 1
```

3.3 Container Configuration

There are three ways to configure an application:

- build time configuration
- runtime configuration
- start time configuration

Please review the official Spring-Boot documentation for further details how to extern-alize configuration.

3.3.1 Build time configuration

Build time configuration can be achieved by simply predefining properties in one of the configuration files in the workspace. For the Docker setup the locations are:

- `application.properties` in the `app/<app name>/spring-boot/<app name>-app/src/main/resources` directory
- `application.properties` in the `app/<app mame/docker/<app name>/src/docker/comfig` directory

3.3.2 Start time configuration

Start time configuration can be achieved using

- environment variables
- mounted configuration files

Configuring Spring-Boot applications using environment variables is probably the most common case and is the default choice in a container based environment. With Docker you simply add each environment variable on the command-line or pass in a path to a file with the environment variables. With Spring environment variables are defined in upper case and transformed by Spring to the java properties syntax.

- `REPOSITORY_URL` is transformed to `repository.url`
- `LINK_URLPREFIXTYPE` is transformed to `link.urlPrefixType`

To pass in environment variables use the `-e` or the `--env-file` options, that is

passing environment variables on the command-line:

```
docker run \
 -e REPOSITORY_URL=http://my-host:40180/ior coremedia/cae-preview
```

Configuring the application by mounting a configuration file should only be used if passing variables in does not work or if replacing an existing configuration file is desired. To mount a file into the container, the path to the file or directory on the host and in the container must be defined. I.e.

mounting configuration files:

```
docker run \
 -v $PWD/override.properties:/coremedia/config/application.properties \
 coremedia/cae-live
```

3.3.3 Runtime configuration

To support runtime (re)configuration, the applications configuration had to be backed by a k/v store and the application would have to support it. This kind of configuration is not supported by CoreMedia applications except for some special properties that can be configured using struct documents in the content repository.

3.4 Containerized Tools

With the Docker setup all CoreMedia tools are packaged within the `coremedia/management-tools` image. You will find the Maven module to build the image at:

```
global/management-tools/docker/management-tools
```

The image contains the following tools:

- content-management-server tools
- master-live-server tools
- replication-live-server tools
- workflow-server tools
- cae-feeder tools
- theme-importer

3.4.1 Running the Tools

To run the tools, you can either:

- Start the `management-tools` container in interactive mode and run the tools *from within*. This approach has the advantage, that it provides a familiar user experience as if the tool is installed locally.
- Start a tool from the `management-tools` container directly, using the tool as the command argument and remove the container when the tool finishes. This approach has the advantage, that it can be used in the context of a script as it requires no interaction.

Regardless of the approach, you need to keep in mind the following common issues with containers:

- In case you need to read from or write to files on your host, you need to mount volumes or create a bind mount.
- In case the containers are placed within a user-defined network, you need to add the container to that network to access the servers you want to connect. This network configuration depends on the network plugins being used. For the localhost `docker-compose` development setup, this network is named `backend` and prefixed with either the value of the `COMPOSE_PROJECT` environment variable or the name of the directory, where the compose files lies. In the development setup this will be `compose` but you can find out what networks are available, by running:

```
docker network list
NETWORK ID     NAME              DRIVER  SCOPE
181f1af50e6e   bridge            bridge  local
2788cdf4bd7a   compose_backend   bridge  local ❶
3227b513840e   compose_web       bridge  local ❷
494f6bdaafa8   host              host    local
3d8b08335372   none              null    local
```

❶   The network, where all services are put in
❷   The network, where all services with Traefik ingress are put in

Mounting a file system path: You can mount a file system path to a docker container by using the `--volume <local-path>:<container-path>` command-line option.

Adding networks: You can attach the docker container to a network using the `--network <network name>` command-line option.

Start an interactive shell in the container

To start the `management-tools` container with an interactive shell, you need to run:

```
docker run -it --rm coremedia/management-tools
```

Start a tool directly with a custom entrypoint

To start a tool, for example the `version` tool directly, you need to run:

```
docker run --rm \
  coremedia/management-tools \
  tools/bin/cm version
```

Here the tool and its arguments are passed to the entrypoint. Because the default working directory is set to `/coremedia` by the image, you only need to pass the relative path to the `cm` executable. All arguments that follow will be handled as arguments to the `cm` executable. The results of the tools are written to stdout and can easily be processed using standard shell means.

Parse an IOR URL:

```
docker run --rm \
  coremedia/management-tools \
  tools/bin/cm ior <ior string>
```

3.4.2 Configuring the Tools

Many tools have special configuration files and their values cannot be passed into the tool using environment variables. In order to configure the tools, you have two options:

- create configuration files on your host and mount them into the container when executing the tool
- use them embedded `confd` entrypoint chain script together with confd template configurations and render the configuration at startup using one of the many configuration backends, confd provides.

Mount configuration files

1. Create the config file

```
cat << EOF > capclient.properties
cap.client.server.ior.url=http://<host>:<port>/ior
EOF
```

2. Execute the tool with mounted configuration file

```
docker run --rm \
  --entrypoint /coremedia/tools/bin/cm \
  --volume \
    $PWD/capclient.properties:\
    /coremedia/tools/properties/corem/capclient.properties
  coremedia/management-tools \
  dump -u <user> -p <password> <id>
```

With mounted configuration files, you can create a separate configuration folder for each content server and for each environment you want to connect to and mount the whole folder to switch a tool from one environment to another.

Use confd to render configuration files at startup

Since `coremedia/java-application-base:2.1.6` confd is included in the base image. Confd is a tool to render configuration files using GO templates and a special set of template functions from confd.

Confd expects its configuration located at `/etc/confd/conf.d` defined using toml syntax and its templates `/etc/confd/templates`. Each template to render requires a separate configuration file.

The `management-tools` container is by default outfitted with templates for: * UAPI connections using `capclient.properties` * resetcaefeeder using `resetcaefeeder.properties` * sql tools using `sql.properties` * workflowconverter using `workflowserver.properties`

The templates have been designed, so that the environment keys would be the same as if the tool would accept the Spring Boot environment variable to Spring property transformation. For the UAPI connection, this means, that defining `CAP_CLIENT_SERVER_IOR_URL` would result in rendering `cap.client.server.ior.url`.

The advantage of using confd over mounting configuration files is that confd allows to set a key path prefix on the command-line which allows us to define multiple environments within the same configuration source.

For convenience purposes in the development setup, three prefixes are preconfigured within the management-tools image:

- `dev/management`
- `dev/master`
- `dev/replication`

You can easily switch between them by using the convenience function `tools-context` included in the bashrc profile:

```
$ tools-context dev/master
[DOCKER ENTRYPOINT] - entering /coremedia/confd
[DOCKER ENTRYPOINT] - running confd with backend "env" and prefix "dev/master"
[DOCKER ENTRYPOINT] - entrypoint chain finished
```

Using confd to configure tools within a container can be especially useful when developing custom UAPI clients, without Spring configuration techniques, that should run as daemonized services within a container.

3.4.3 Examples with confd rendered configuration

Create config source:

```
cat << EOF > toolconfig.env
UAT_MANAGEMENT_CAP_CLIENT_SERVER_IOR_URL=http://uat-content-management-server:8080/ior
UAT_MASTER_CAP_CLIENT_SERVER_IOR_URL=http://uat-master-live-server:8080/ior
UAT_REPLICATION_1_CAP_CLIENT_SERVER_IOR_URL=http://uat-master-live-server:8080/ior
```

```
UAT_REPLICATION_2_CAP_CLIENT_SERVER_IOR_URL=http://uat-master-live-server:8080/ior
# PROD
PROD_MANAGEMENT_CAP_CLIENT_SERVER_IOR_URL=http://prod-content-management-server:8080/ior
PROD_MASTER_CAP_CLIENT_SERVER_IOR_URL=http://prod-master-live-server:8080/ior
PROD_REPLICATION_1_CAP_CLIENT_SERVER_IOR_URL=http://prod-master-live-server:8080/ior
PROD_REPLICATION_2_CAP_CLIENT_SERVER_IOR_URL=http://prod-master-live-server:8080/ior
EOF
```

Export content to your host from UAT content-management-server:

```
docker run --rm \
  --env-file=toolconfig.env \
  -e CONFD_PREFIX=uat/management \
  --volume $PWD/export:/export \
  coremedia/management-tools confd \
  tools/bincm serverexport -u admin -p admin \
  -r --basedir /export /
```

In the example above we pass the `toolconfig.env` to docker and set the environ-
ment variable `CONFD_PREFIX` to `uat/management`, which translates to the
environment variable prefix `UAT_MANAGEMENT_`. In our env file we defined the
connection to the content-management-server using the key `UAT_MANAGE
MENT_CAP_CLIENT_SERVER_IOR_URL`. If we had set the prefix to
`uat/master` we would have used the contentrepository of the master for the export.

3.4.4 Examples with mounted configuration

Create config:

```
cat << EOF > capclient.properties
cap.client.server.ior.url=http://<host>:<port>/ior
EOF

cat <<EOF > resetcaefeeder.properties
jdbc.driver=com.mysql.cj.jdbc.Driver
jdbc.url=jdbc:mysql://<host>:3306/cm_mcaefeeder
jdbc.user=cm_mcaefeeder
jdbc.password=cm_mcaefeeder
EOF

cat <<EOF > sql.properties
sql.store.driver=com.mysql.cj.jdbc.Driver
sql.store.url=jdbc:mysql://localhost:3306/cm_management
sql.store.user=cm_management
sql.store.password=cm_management
EOF
```

Export content to your host:

```
docker run --rm \
  --entrypoint /coremedia/tools/bin/cm \
  --volume \
    $PWD/capclient.properties:\
    /coremedia/tools/properties/corem/capclient.properties \
  --volume \
    $PWD/export:/export \
  coremedia/management-tools \
```

```
serverexport -u admin -p admin \
-r --basedir /export /
```

Import content to your host:

```
docker run --rm \
  --entrypoint /coremedia/tools/bin/cm \
  --volume \
    $PWD/capclient.properties:\
    /coremedia/tools/properties/corem/capclient.properties \
  --volume \
    $PWD/import:/import \
  coremedia/management-tools \
  serverimport -u admin -p admin \
  -r /import
```

Reset the cae-feeder-preview:

```
docker run --rm \
  --entrypoint /coremedia/tools/bin/cm \
  --volume \
    $PWD/resetcaefeeder.properties:\
    /coremedia/tools/properties/corem/resetcaefeeder.properties \
  coremedia/management-tools \
  resetcaefeeder reset
```

Unlock the content-management-server:

```
docker run --rm \
  --entrypoint /coremedia/tools/bin/cm \
  --volume \
    $PWD/sql.properties:\
    /coremedia/tools/properties/corem/sql.properties \
  coremedia/management-tools \
  unlockcontentserver
```

4. CoreMedia Properties Overview

Properties can be configured via Spring Boot. You can use application.properties, system properties, environment variables in uppercase and many more. See https://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-external-config.html for details.

Spring Boot configuration

Spring's relaxed binding also allows for different notations of property names like snake or camel case, but the default is 'kebap case' (separating words with dashes). Generally a dot in a property name reflects some kind of logical hierarchy. List-valued properties are zero-based and use bracket notation (x.y.1.* -> x.y[0].*)

Spring relaxed binding

4.1 Content Application Engine Properties

4.1.1 General CAE Properties

cae.cookie.force-http-only	
Type	java.lang.Boolean
Default	false
Description	Whether or not to force the 'HttpOnly' attribute on all cookies.

cae.cookie.force-secure	
Type	java.lang.Boolean
Default	false
Description	Whether or not to force the 'Secure' attribute on all cookies.

cae.cookie.same-site	
Type	java.lang.String
Default	
Description	The value of the cookie's 'SameSite' attribute. Valid values are the ones as defined by the spec. In addition, the value 'Unset' can be used to indicate that the attribute should not be set.

cae.cookie.use-rfc6265	
Type	java.lang.Boolean
Default	false

Description	Whether or not to use the RFC6265 cookie processor.

cae.csrf.ignore-paths	
Type	java.util.List<java.lang.String>
Default	
Description	Ant Paths to ignore for CSRF prevention.

cae.http-firewall.allow-semicolon	
Type	java.lang.Boolean
Default	false
Description	Determines if semicolon is allowed in the URL (i.e. matrix variables).

cae.http-firewall.allow-url-encoded-double-slash	
Type	java.lang.Boolean
Default	false
Description	Determines if a double slash (//) that is URL encoded (%2F%2F) should be allowed in the path or not.

cae.http-firewall.allow-url-encoded-percent	
Type	java.lang.Boolean
Default	false
Description	Determines if a percent (%) that is URL encoded (%25) should be allowed in the path or not.

cae.http-firewall.allow-url-encoded-period	
Type	java.lang.Boolean
Default	false

Description	Determines if a period (.) that is URL encoded (%2E) should be allowed in the path or not.

`cae.http-firewall.allow-url-encoded-slash`

Type	java.lang.Boolean
Default	false
Description	Determines if a slash (/) that is URL encoded (%2F) should be allowed in the path or not.

`cae.hashing.secret`

Type	java.lang.String
Default	
Description	A Secret which is used for url parameter hashing. It is recommended to use a secret at least 32 characters long. If not configured a secret will be generated and exposed via warn log on application startup. If multiple CAEs are used, ensure to set the secret instead of trusting a generated one.

`cae.link-transformer.include-params-appender.uri-paths`

Type	java.util.List<java.lang.String>
Default	/servlet/dynamic/
Description	uriPaths the IncludeParamsAppendingLinkTransformer should be applied to.

`cae.link-transformer.serializer-classes`

Type	java.util.List<java.lang.Class<?>>
Default	
Description	A list of fully qualified class names for which an com.fasterxml.jackson.databind.JsonSerializer should be registered for view parameter conversion. Every Class which is configured here, should have a proper com.coremedia.id.IdScheme implementation which known to the com.coremedia.id.IdProvider.

`cae.preview.crossdomain-whitelist`

Type	java.util.List<java.lang.String>
Default	
Description	Sets the list of same origins which are listed in the ACCESS-CONTROL-ALLOW-ORIGIN HTTP header.

`cae.preview.metadata-enabled`

Type	java.lang.Boolean
Default	false
Description	Whether to disable metadata rendering. Disabled by default.

`cae.preview.pbe.include-jquery`

Type	java.lang.Boolean
Default	false
Description	Configures if jquery should be included when rendering the preview related scripts.

`cae.preview.pbe.studio-url-whitelist`

Type	java.util.List<java.lang.String>
Default	
Description	Configures a list of valid Studio URLs. The Studio Preview integration does only work for listed Studio instances. If left blank, any Studio instance is considered valid.

`cae.set-unknown-mime-type`

Type	java.lang.Boolean
Default	false
Description	This property controls if an instance of com.coremedia.blueprint.cae.filter.UnknownMimetypeCharacterEncodingFilter is registered to fix unknown encoding errors in Webshere versions up to and including 8.5.5010.20160721_0036. The UnknownMimetypeCharacter-

Encoding filter will be used when cae.set-unknown-mime-type is set to true. The default is suitable when using Tomcat or Websphere starting from 8.5.5011.20161206_1434.

cae.single-node	
Type	java.lang.Boolean
Default	false
Description	This property is used in com.coremedia.blueprint.cae.handlers.HandlerBase#doCreate-ModelWithView to control if a possibly outdated resource is served or if a redirect is sent. The redirect is only a valid response when cae.single-node is set to true.

cae.view.cycle-check	
Type	java.lang.Boolean
Default	true
Description	Check for cyclic inclusions. You should not disable the check, unless for some good reason, e.g.: - You use a custom ViewDispatcher, whose getView method is not determined only by its arguments. - Your View#render implementation invokes ViewUtils#render with the same bean + viewName, but with an other (delegate) View.

cae.view.debug-enabled	
Type	java.lang.Boolean
Default	false
Description	If set to true, html comments will be written to the rendered pages around included fragments. This is a development feature. With these comments you can easily see which JSP, bean and view was used to render a fragment.

cae.view.errorhandler.enabled	
Type	java.lang.Boolean
Default	true

Description	Enables/disables the view exception handler.

cae.view.errorhandler.output	
Type	java.lang.Boolean
Default	false
Description	If handler is enabled and set to true, exceptions will be displayed in the current page.

cae.view.filter-lookup-by-predicate	
Type	java.lang.Boolean
Default	false
Description	By convention, templates are written for bean interfaces, but views may be named after any type. If set to true, viewlookup will only be done for views named after interfaces, not classes, with configurable excludes and includes. This option is turned off by default.

cae.view.max-depth	
Type	java.lang.Integer
Default	200
Description	Maximum depth of inclusions.

cae.viewdispatcher.cache.enabled	
Type	java.lang.Boolean
Default	true
Description	Defines if the caching of view lookups is enabled. Disabling might be useful when developing templates. Shouldn't be disabled when in production mode!

cae.viewdispatcher.expose-spring-macro-helpers	
Type	java.lang.Boolean
Default	true

Description	Set whether to expose a RequestContext for use by Spring's macro library, under the name "springMacroRequestContext". Default is **"true"**.
	Currently needed for Spring's Velocity and FreeMarker default macros. Note that this is *not* required for templates that use HTML forms *unless* you wish to take advantage of the Spring helper macros.

`cae.viewdispatcher.fallback-to-default-view`

Type	java.lang.Boolean
Default	true
Description	Fallback to default view if requested view name raises view exception.

Table 4.1. Configuration Properties with Prefix cae

4.1.2 Delivery CAE Properties

`delivery.developer-mode`

Type	java.lang.Boolean
Default	false
Description	Setting to enable merging and minification of resources such as JavaScript and CSS. If set to true, these resources will be merged and minfied during delivery and only a single link will be rendered to each type of resource. otherwise, each corresponding script and stylesheet will be rendered as a separate link.

`delivery.local-resources`

Type	java.lang.Boolean
Default	false
Description	true if links to resources such as JavaScript and CSS should be generated to point to files in the application container (e.g. in modules under META-INF/resources/themes/) instead of the content repository.

`delivery.preview-mode`

Type	java.lang.Boolean
Default	false
Description	Indicates whether this is a preview cae.

delivery.standalone

Type	java.lang.Boolean
Default	true
Description	True when this is a "standalone" CAE in a servlet container serving requests directly or false when behind some URL rewriting reverse proxy (e.g. Apache).

Table 4.2. Delivery Properties

4.1.3 Http Cache Control Properties

cae.cache-control.cache-size

Type	java.lang.Integer
Default	10000
Description	Maximum count of cache configuration entries. Cache cleans up automatically considering LRU strategy. Default is 10000 entries.

cae.cache-control.for-type

Type	java.util.Map<java.lang.String,org.springframework.boot.autoconfigure.web.ResourceProperties$Cache$Cachecontrol>
Default	
Description	Map of initial cache control configuration objects for Http cache control Header. See {@link ResourceProperties.Cache.Cachecontrol} for further details. The configuration is type specific, but also can benefit from abstraction along the class hierarchy. That means that you may simply configure a common value for a super class instead of taking care about all the child classes. The type information part of the property name

should be the simple name of the class (no package information) and in lowercase. Configuration of classes with the same name in different packages is not supported.

Example Configuration:

```
  cae.cache-control.for-type.cmlinkable.max-age=1m
cae.cache-control.for-type.blob.max-age=180d
```

Default/Fallback:

```
cae.cache-control.for-type.object.max-age=1m
```

cae.cache-control.for-url-pattern

Type	java.util.Map<java.lang.String,org.springframework.boot.autoconfigure.web.Resource-Properties$Cache$Cachecontrol>
Default	
Description	Map of initial cache control configuration objects for Http cache control Header. See {@link ResourceProperties.Cache.Cachecontrol} for further details. The configuration is URL pattern specific.

cae.cache-control.initial-max-age-for-type

Type	java.util.Map<java.lang.String,java.time.Duration>
Default	
Description	Initial max-age value for Cache Control Header. Does not have to be adjusted necessarily. The initial max-age value should be small, since the effective cache duration is provided with the subsequent request. The configuration is type specific, but also can benefit from abstraction along the class hierarchy. That means that you may simply configure a common value for a super class instead of taking care about all the child classes. The type information part of the property name should be the simple name of the class (no package information) and in lowercase. Configuration of classes with the same name in different packages is not supported.

Example Configuration:

```
    cae.cache-control.initial-max-age-for-type.blob=180d
```

Default/Fallback:

```
cae.cache-control.initial-max-age-for-type.object=1m
```

Table 4.3. Configuration Properties with Prefix cache.control

4.1.4 Blob Transformation properties

com.coremedia.transform.blobCache.size	
Value	Number
Default	1000000000
Description	The maximum allowed size that the *transformed image blob cache* can occupy on the disk. This is a separate cache where results of blob transformations are stored persistently and can survive *CAE* restarts. Note that the file system overhead for storing the files does not count towards this value. So the physical space that has to be reserved on the disk for the cache has to be slightly higher than value of this configuration property.
	If several concurrent threads write large blobs at the same time, the deletion of the folder with the old unused files can be postponed for later, thus this is the second reason why the maximum allowed cache size can grow slightly higher than this configuration property. The size of such deviation depend on the blobs size as well as the amount of parallel threads.

cache.capacities.com.coremedia.transform.image.java2d.LoadedImageCacheKey	
Value	Number
Default	100 MB
Description	The size of the loaded image cache, in bytes.

com.coremedia.transform.blobCache.basePath	
Value	String
Default	not set => cache deactivated

Description	The path to the *transformed image blobs cache* . If not set, than this cache is deactivated and the results of image transformations are stored using *UAPI cache*. It is recommended to set this property to a folder that is not cleared during *CAE* restart, e.g. `/var/cache/coremedia/persistent-transformed-blobcache`.

`com.coremedia.transform.memoryGuard.safetyFactor`

Value	Number
Default	1.2
Description	A safety factor of the memory guard configured for the imageTransformer. Setting this to zero disables memory checks.

`com.coremedia.transform.throttle.permits`

Value	Number
Default	0, falling back to 1/2 total JVM heap space
Description	Capacity of the throttlingBlobTransformer, in megabytes.

`com.coremedia.transform.throttle.hardLimit`

Value	Boolean
Default	true
Description	Specifies whether the configured permits should be considered a hard limit for transformations. Setting this to false allows transformations needing more permits (serialized, one at a time). This would allow larger images to be transformed, but would make OOM exceptions during transformations more likely.

`com.coremedia.transform.throttle.safetyFactor`

Value	Number
Default	3.0
Description	A safety factor for the ImageSizePermitStrategy to multiply the memory size of an image with. Default is 3. A number of 3 takes into account that some image operations need to make a copy of the image, plus one copy for the in-memory cache for loaded images.

com.coremedia.transform.throttle.fallbackFactor	
Value	Number
Default	1.0
Description	A factor to multiply the blob size with if the size of the in-memory representation needed for transformation cannot be determined. A warning is logged if this happens.

Table 4.4. Further Configuration Properties

4.1.5 Renamed CAE Properties

The following list contains renamed *CAE* configuration properties. Use the new names, the old names are deprecated.

Deprecated Name	New Name
cae.developer.mode	delivery.developer-mode
cae.is.preview	delivery.preview-mode
cae.is.standalone	delivery.standalone
cae.use.local.resources	delivery.local-resources
cae.coderesources-max-age	cae.cache-control.*
cae.coderesources.maxAge	cae.coderesources-max-age
cae.crossdomain.whitelist	cae.preview.crossdomain-whitelist
cae.viewdispatcher.cache.size	cache.capacities.com.coremedia.objectserver.view.ViewLookup
filter.viewlookup.by.predicate	cae.view.filter-lookup-by-predicate
metadata.enabled	cae.preview.metadata-enabled

Deprecated Name	New Name
`pbe.studioUrlWhitelist`	`cae.preview.pbe.studio-url-whitelist`
`view.debug.enabled`	`cae.view.debug-enabled`
`view.errorhandler.enabled`	`cae.view.errorhandler.enabled`
`view.errorhandler.output`	`cae.view.errorhandler.output`
`view.maxdepth`	`cae.view.max-depth`
`viewdispatcher.cache.enabled`	`cae.viewdispatcher.cache.enabled`
`viewdispatcher.cache.size`	`cae.viewdispatcher.cache.size`
`viewdispatcher.expose-spring-macro-helpers`	`cae.viewdispatcher.expose-spring-macro-helpers`
`viewdispatcher.fallback-to-default-view`	`cae.viewdispatcher.fallback-to-default-view`
`cae.viewdispatcher.cache.size`	`cache.capacities.com.coremedia.objectserver.view.ViewLookup`

Table 4.5. Renamed CAE Configuration Properties

4.2 Content Server Properties

4.2.1 General Content Server Properties

cap.server.allow-synthetic-replay	
Type	java.lang.Boolean
Default	true
Description	Whether it is allowed for clients to request a synthetic replay of the content repository, for example using the constant Timestamp.SYNTHETIC_REPLAY Unified API. This is a very expensive operation that is rarely used except when setting up a Replication Live Server from scratch.

cap.server.blob-channel-timeout	
Type	java.lang.Integer
Default	60
Description	This property sets the timeout for streaming blobs to and from the database in seconds. In general, you don't have to change the default value. It is provided for exceptional cases, when the connection to the database is unreliable.

cap.server.cache.group-cache-size	
Type	java.lang.Integer
Default	500
Description	This property defines the size of the group cache. It limits the maximum number of groups which can be found in one search for groups in the user window of the Site Manager. Set the property so that all groups connected to rights can be cached in memory.

cap.server.cache.member-folder-rights-cache-size	

Type	java.lang.Integer
Default	1000
Description	This property defines the size of the folder-specific rights cache. This cache stores the results of right calculations per folder and member, aggregating the results for all content types. This cache might help custom code using APIs other than the Unified API, but mainly it affects the performance of the Site Manager in rare cases. Change this setting only if you observe the method getRights(MemberKey) in thread dumps of a slow Content Server.

cap.server.cache.resource-cache-size	
Type	java.lang.Integer
Default	60000
Description	The capacity of the resource cache of the Content Server; the maximum value is 1000000, the minimum value is 100. This property defines the resource cache size, that is, the number of resources the server holds in memory. This value should sometimes be adapted to the increasing number of resources in the actual working set. If the value is too small, the server does not perform well. One resource needs about 2kB of heap space.

cap.server.cache.resource-cache-status-interval	
Type	java.lang.Integer
Default	300
Description	The time between two log messages reporting the current state of the resource cache, in seconds; the maximum value is 3600, the minimum value is 10.

cap.server.cache.rights-cache-size	
Type	java.lang.Integer
Default	3000
Description	This property defines the size of the rights cache. This cache stores the results of right calculations per resource, content type and member. If you have lots of different resources, content types and users you might need to adapt the value of the property. Check the proper size of the cache by examining the cache misses and faults in the

log. To activate the log output of the rights cache set the cap.server.cache.rights-cache-status-interval property to a value larger than zero.

cap.server.cache.rights-cache-status-interval	
Type	java.lang.Integer
Default	0
Description	This property defines the interval (in seconds) at which log output of the rights cache is written. "0" means, that no log output is written.

cap.server.cache.user-cache-size	
Type	java.lang.Integer
Default	500
Description	This property defines the size of the user cache. It limits the maximum number of users which can be found in one search for users in the user window of the Site Manager. Set the property to the size of the largest user search you want to perform, or the number of concurrently working users, whichever is greater.

cap.server.check-unique-db-access	
Type	java.lang.Boolean
Default	true
Description	This property determines whether to check for another server that is running concurrently on the same database on server startup.

cap.server.document-types	
Type	java.lang.String
Default	classpath*:/framework/doctypes/**/*.xml,config/contentserver/doctypes/**/*.xml
Description	This property defines where the server finds the XML file(s) containing the content type definitions. You can specify multiple files as a comma separated list or use Ant-style patterns with wildcards like '*', '?' and '**'.

Example: config/contentserver/doctypes/**/*.xml matches all XML files below the config/contentserver/doctypes directory.

`cap.server.encrypt-passwords-key-file`

Type	java.lang.String
Default	
Description	The location of the key generated by cm encryptpasswords. If empty, defaults to etc/keys/DATABASE_NAME.DATABASE_USER.rijndael

`cap.server.http-port`

Type	java.lang.Integer
Default	0
Description	Defines the HTTP(S) port of the application container containing the Content Server. The entry /Server/Service/Connector@port in server.xml has to have the same value as this property.

`cap.server.init-runlevel`

Type	java.lang.String
Default	online
Description	The initial runlevel that the server will try to reach on startup. Possible runlevels are: online, administration, maintenance. This property does not override the default behavior of the Replication Live Server for the initial replication.

`cap.server.initial-password`

Type	java.util.Map<java.lang.String,java.lang.String>
Default	
Description	The initial password to set for the default user with the indicated name. This password is set when the server is started for the first time. You can change the passwords later on at any time.

cap.server.license	
Type	java.lang.String
Default	properties/corem/license.zip
Description	Defines where the server finds the license file.

cap.server.login-service-webserver-privileged	
Type	java.lang.Boolean
Default	false
Description	This property specifies whether client connections of the login service 'webserver' are privileged and may log in as different users without further authentication. The default is false. It may be set to true to make the 'webserver' login service a privileged service as it was the case in releases before 1907.

cap.server.login.authentication	
Type	java.lang.String
Default	properties/corem/jaas.conf
Description	Arguments for jaas login authentication, will be set on the java.security.auth.login.config System property.

cap.server.login.bouncers	
Type	java.lang.String
Default	
Description	This property points to the optional login bouncer configuration. A login bouncer can grant or deny access to the Content Server based on the characteristics of the user and the set of currently logged in users.

cap.server.login.password-hash-algorithm	
Type	java.lang.String

Default	bcrypt:10
Description	A specification of the hash algorithm used for storing passwords.
	Allowed values are md5 for MD5-based password hashing and bcrypt:N with N being an integer between 4 and 31 (inclusive) for bcrypt-based password hashing. In the latter case N denotes the work factor which should adapted to the available CPU resources. This parameter applies to passwords of users defined in the built-in user repository of the Content Server, only.
	The value md5 is discouraged, because it makes brute-force attacks on passwords of low and medium strength possible. It should only be used if passwords need to be changed by clients (Studio, Site Manager, cm changepassword) that have not been updated to a CMS release that supports configurable password hashing.
	Old clients can login even after a password change without any restrictions. After changing this property, it is recommended to update the passwords of all users to ensure that all hashes have been computed according to the desired algorithm.

`cap.server.maximum-startup-delay`

Type	java.lang.Integer
Default	60
Description	Configured maximum time for Content Server startup in seconds. This is the maximum time after which the Content Server is treated as initialized if it was started as Windows Service or as web application. Dependent Windows Services / other web applications will be started when the Content Server has reached its initial runlevel or after this time.

`cap.server.multiple-live-servers`

Type	java.lang.Boolean
Default	false
Description	This property defines whether the server publishes to multiple live servers. Note that this flag cannot be easily changed after the first start of the Content Management Server.

`cap.server.naming-policy-allow-at`

Type	java.lang.Boolean

Default	false
Description	Allow '@' in member names (may cause confusion with user domains)

cap.server.persistent-property-writers

Type	java.util.List<java.lang.String>
Default	*
Description	A list of names of groups that may write or delete persistent properties. The magic group "*" grants rights to all users. This is the default for compatibility reasons. Connections using the publisher and replicator login service are always allowed to write persistent properties.

cap.server.repository-home

Type	java.lang.String
Default	/Home
Description	Defines the folder which will be used to store the home folders of the users. The whole folder hierarchy of the home folders is only visible to the administrator. Other user will only see one home folder with the path defined in cap.server.repository-home containing his personal files, such as the preferences. The default folder is /Home which will be automatically created by the system. If you define another folder, you need to create this folder by your own.

cap.server.repository-system

Type	java.lang.String
Default	/System
Description	Defines the system folder. It contains for example the public dictionary of the spell checker. The default folder is /System which will be automatically created by the system. If you define another folder, you need to create this folder by your own.

cap.server.search.enable

Type	java.lang.Boolean

Default	false
Description	If true full text search is enabled.

`cap.server.session-ping-interval`

Type	java.lang.Integer
Default	60
Description	The maximum number of seconds that a ping is delayed when there are no available events

`cap.server.termination-timeout-seconds`

Type	java.lang.Integer
Default	30
Description	Timeout for waiting on running threads on shutdown

`cap.server.unique-db-access-write-interval`

Type	java.lang.Integer
Default	0
Description	If unique DB access is checked and if this property is positive, this property determines the number of seconds between two writes of the current timestamp to the database to indicate the liveliness of the server; if 0 or negative the server neither writes a timestamp regularly nor expects a timestamp to be written;

`cap.server.use-strict-workflow`

Type	java.lang.Boolean
Default	false
Description	This property enforces the strict workflow mode. That is, the approver of a resource must be different from the editor. This is checked independently of the workflow engine, and should only be used in cases where a custom workflow definition is not an option.

cap.server.userproviders	
Type	java.util.List<hox.corem.server.ServerConfigurationProperties$Userproviders>
Default	
Description	Configurations for UserProviders For details see {@link Userproviders}

cap.server.blobstore.s3.bucketname	
Type	java.lang.String
Default	
Description	The name of the S3 bucket that is used to store blobs. If this property is not specified, the S3 media store is not enabled.

cap.server.blobstore.s3.rootdir	
Type	java.lang.String
Default	
Description	The name of an S3 path prefix, used to disambiguate multiple media stores in one S3 bucket. If possible, it is recommended to use separate buckets instead, improving performance by avoiding constant path prefixes.

Table 4.6. Content Server Properties

4.2.2 CORBA Properties

com.coremedia.corba.client.local-socket	
Type	java.lang.Boolean
Default	false
Description	Redirect all outgoing connections to localhost.

This is useful for running UAPI clients locally, while forwarding HTTP and CORBA connections to the servers through an SSH tunnel. A typical SSH invocation that allows the use of this socket factory would look like this, forwarding all ports of the *Content Management Server* and the *Workflow Server*:

ssh -L 40180:SERVER:40180 -L 40183:SERVER:40183 -L 40380:SERVER:40380 -L 40383:SERVER:40383 HOST

When using this option, you should also activate com.coremedia.corba.server.setNoSocket(boolean) in order to reduce the attack surface of the JVM.

com.coremedia.corba.client.redirect.original-host	
Type	java.lang.String
Default	
Description	Redirect IIOP invocations to other addresses.
	Useful in tunnelling scenarios: imagine a server X on port P running in a network you cannot reach; but you are able to create a tunnel that forwards your address Y:Q to X:P, e.g. you do user@Y % ssh -g -LQ:X:P
	Then you can configure your clients by the com.coremedia.corba.client.redirect property group:
	- originalHost: The original server's name (X). If set, only IORs pointing to that server will be redirected. - redirectHost: The new server name (Y). If set, this name will be placed in the new target address. - originalPort: The original server's port (P). If set, only IORs pointing to that port will be redirected. - redirectPort: The new server port (Q). If set, this port will be placed in the new target address.

com.coremedia.corba.client.redirect.original-port	
Type	java.lang.Integer
Default	-1
Description	Redirect IIOP invocations to other addresses.
	For details see com.coremedia.corba.client.redirect.originalHost.

com.coremedia.corba.client.redirect.redirect-host

Type	java.lang.String
Default	
Description	Redirect IIOP invocations to other addresses.
	For details see com.coremedia.corba.client.redirect.originalHost.

com.coremedia.corba.client.redirect.redirect-port

Type	java.lang.Integer
Default	-1
Description	Redirect IIOP invocations to other addresses.
	For details see com.coremedia.corba.client.redirect.originalHost.

com.coremedia.corba.client.ssl.clear-text-ports

Type	java.lang.String
Default	
Description	The Corba clear text ports
	Usually one, maybe two (comma separated) values for *Content Server* and *Workflow Server*. In case of multiple values, the order must correspond to the com.core-media.corba.client.ssl.sslPorts.

com.coremedia.corba.client.ssl.keystore

Type	java.lang.String
Default	
Description	The keystore for SSL encrypted communication

com.coremedia.corba.client.ssl.passphrase

Type	java.lang.String

Default	

Description	The passphrase for SSL encrypted communication

`com.coremedia.corba.client.ssl.ssl-ports`

Type	java.lang.String

Default	

Description	The Corba SSL text ports
	Usually one, maybe two (comma separated) values for *Content Server* and *Workflow Server*. In case of multiple values, the order must correspond to the com.core-media.corba.client.ssl.clearTextPorts.

`com.coremedia.corba.server.host`

Type	java.lang.String

Default	

Description	The host of the ORB.
	This hostname is exposed to the client via the IOR. Normally, you do not need to care about this, but in runtime environments with special DNS configurations it may be necessary to set a certain name that the client is able to resolve.

`com.coremedia.corba.server.no-socket`

Type	java.lang.Boolean

Default	true

Description	Disable ORB server sockets.
	This is useful when no incoming requests are expected, so that the server socket would only pose a security risk.
	The default is true, i.e. server sockets are disabled. This is appropriate for all client applications and must be set to false only for *Content Server* (CMS, MLS and RLS) and *Workflow Servery* applications.

`com.coremedia.corba.server.port`

Type	java.lang.Integer
Default	-1
Description	The clear text port of the ORB.

`com.coremedia.corba.server.single-ip`

Type	java.lang.String
Default	
Description	Bind ORB sockets to this IP address.
	Useful to keep control on multi-homed hosts.

`com.coremedia.corba.server.ssl.enforce`

Type	java.lang.Boolean
Default	false
Description	Enforce SSL communication
	By default, the ORB opens an SSL port additionally to the clear text port. With this flag you can suppress the clear text port and thus ensure that clients use SSL connections.

`com.coremedia.corba.server.ssl.keystore`

Type	java.lang.String
Default	
Description	The keystore for SSL encrypted communication

`com.coremedia.corba.server.ssl.passphrase`

Type	java.lang.String
Default	
Description	The passphrase for SSL encrypted communication

com.coremedia.corba.server.ssl.ssl-port	
Type	java.lang.Integer
Default	-1
Description	The port of the ORB for SSL encrypted communication.
	The orb will open the SSL socket *in addition* to the clear text port. There is no way to disable this.

Table 4.7. CORBA Properties

4.2.3 Properties for the Publisher

In the properties given below, some properties take a different form when using *CoreMedia Multi-Master Management*. In that case, an index number is used to indicate the publication target that is configured using this property value. The index numbers are consecutive integers starting with 1 and running up to the number of publication targets.

publisher.autoextend.latest-approved-version	
Type	java.lang.Boolean
Default	false
Description	This property affects the automatic extension of publication sets. The default behavior ["false"] is as follows: If the referenced content item is not published yet, publish its earliest approved version. If the referenced content item is already published, do nothing If set to "true", not the earliest but the latest (= newest) approved version is published.

publisher.autoextend.max-failures	
Type	java.lang.Integer
Default	0
Description	The maximum number of implicitly added content items and folders whose publication may fail before no further content items and folders may be added implicitly. If set to 0 [the default], the publisher never stops adding content items and folders.

publisher.autoextend.update-linked-documents

Type	java.lang.Boolean
Default	false
Description	This property affects the automatic extension of publication sets. By default, ["false"] only direct linked and approved content items are added to the publication set. If set to "true", all linked content items are recursively added to the publication set. The recursion stops when a version is already published and may lead to surprisingly large publication sets. Setting updateLinkedDocuments implicitly also sets latestApprovedVersion.

publisher.connection-timeout-seconds

Type	java.lang.Integer
Default	300
Description	The timeout in seconds for the connection to the Master Live Server ior.

publisher.destroy-intermediate-versions

Type	java.lang.String
Default	strict
Description	Whether intermediate content item versions between two publications will be destroyed or not on the Content Management Server.

Example for dumb mode: Version 1 of content item A was published. In the meantime, the versions 2, 3, 4 and 5 have been created. When you now publish version 5, the versions 2, 3, 4 are destroyed and only version 1 and 5 remain on the Content Server. Old flags true and false are supported, too. false maps to off while true maps to strict.

Available Modes

- off

  Turns destruction of intermediate versions off.

- dumb

  Simply destroys all versions between the currently published one and the previously published one; if you run into performance issues on publication or you do not use a multi-site set up this should be chosen. Mind that for multi-site set up this setting might break translation states.

- strict

  Destroys all versions between the currently published one and the previously published one, but only if the versions are not referenced by master/masterVersion properties. This ensures that for a multi-site set up your translation state will be valid. This setting is recommended for multi-site set ups.

publisher.destroy-older-versions-on-live-servers	
Type	java.lang.Boolean
Default	true
Description	Whether older published versions should be destroyed ("true") on the Master Live Server. That is, only two published versions (the current and the one before) of each content item remain on the Master Live Server. Only change if you have a valid reason.

publisher.enable-bypass-previews	
Type	java.lang.Boolean
Default	true
Description	Whether publication previews bypass ("true") or not ("false") the publication queue for faster response times.

publisher.local.domain	
Type	java.lang.String
Default	
Description	The domain for logging in locally to the Content Management Server.

publisher.local.password	
Type	java.lang.String
Default	publisher
Description	The password for logging in locally to the Content Management Server.

publisher.local.user	
Type	java.lang.String
Default	publisher
Description	The user name for logging in locally to the Content Management Server.

publisher.priority	
Type	java.util.Map<java.lang.String,java.lang.Integer>
Default	
Description	Priorities of publications requested through:

- guiEditor: 60
- uapi: 40
- generator: 40
- jpython: 20
- importer: 20
- utility: 20
- unknown: 20

publisher.target	
Type	java.util.List<hox.corem.server.publish.PublisherConfigurationProperties$Target>
Default	
Description	target is a list-valued property. You configure it with publisher.target[0].name, for example. Values other than "0" are only required for the multi-master feature, that is publication to different Master Live Servers. Target has the following properties:

name

- Value: java.lang.String
- Default:
- Description: The permanent and unique name of the publication target. Once set, it should never be changed, as this name is used for target identification in the APIs and in JMX.

displayName

- **Value:** java.lang.String
- **Default:**
- **Description:** The display name is shown to users when no localized information about a publication target is available; display names, too, should be unique, but they may well change to better illustrate the current uses of a publication target.

user

- **Value:** java.lang.String
- **Default:** publisher
- **Description:** The URL where the publisher can obtain the IOR of the Master Live Server.

password

- **Value:** java.lang.String
- **Default:** publisher
- **Description:** The password for logging in to the Master Live Server.

domain

- **Value:** java.lang.String
- **Default:**
- **Description:** The domain for logging in to the Master Live Server.

iorUrl

- **Value:** java.lang.String
- **Default:**
- **Description:** The URL where the publisher can obtain the IOR of the Master Live Server.

folders

- **Value:** java.lang.String
- **Default:**
- **Description:** The base folders that are assigned to a publication target. This property typically references exactly one top-level folder, either by name or by its numerical ID. If more than one site is generated from a single Live Server, multiple top-level folders may be given, separated by commas. When indicating a folder by name, that name is blocked for rename operations on the top-level folder. Once you have assigned a folder to a publication target, it must not be reassigned to another target.

Doing so would result in inconsistencies between Content Management Server and Master Live Server.

Table 4.8. Publisher Properties

4.2.4 Properties for the Connection to the Database

> **NOTE**
>
> sql.properties for Command Line Tools
>
> Be aware that the command line tools that access the database directly (like `cm unlockcontentserver` or `cm schemaaccess`) still need the `sql.properties` file in their `properties/corem` directory, as the tools are not (yet) configurable via Spring.

`sql.pool.check-timeout`

Type	java.lang.Integer
Default	5
Description	Maximum time in seconds the check is allowed to take.

`sql.pool.log-queries`

Type	java.lang.Boolean
Default	false
Description	If the property is "true", messages concerning queries (search of content item versions) will be generated.

`sql.pool.log-query-statements`

Type	java.lang.Boolean
Default	false

Description	If the property is "true", SQL statements concerning queries will be written to the log.

sql.pool.log-schedule-messages	
Type	java.lang.Boolean
Default	false
Description	If the property is "true", write messages of the connection pool to the log file.

sql.pool.log-verbose	
Type	java.lang.Boolean
Default	false
Description	If the property is "true", more debug messages will be generated.

sql.pool.max-connections	
Type	java.lang.Integer
Default	4
Description	Maximum number of connections to the database.

sql.pool.max-queries	
Type	java.lang.Integer
Default	4
Description	Maximum number of connections used for queries, that is, the maximum number of parallel queries.

sql.pool.min-connections	
Type	java.lang.Integer
Default	2

Description	Minimum number of connections to the database.

`sql.pool.open-timeout`	
Type	java.lang.Integer
Default	30
Description	Maximum time in seconds the opening of connections is allowed to take.

`sql.pool.reaper-interval`	
Type	java.lang.Long
Default	120
Description	The interval in seconds in which it is checked if connections can be closed.

`sql.pool.reaper-timeout`	
Type	java.lang.Long
Default	180
Description	The time in seconds a connection must be idle before it will be closed.

`sql.pool.validator-interval`	
Type	java.lang.Long
Default	300
Description	The interval in seconds in which existing connections will be checked for function.

`sql.pool.validator-timeout`	
Type	java.lang.Long
Default	120
Description	The time in seconds a connection must be idle before it will be checked for function.

sql.schema.alter-table

Type	java.lang.Boolean
Default	false
Description	Setting the property to "true", causes the Content Server to automatically add database columns for newly defined content type properties and to fix wrong widths of String properties. This will only work if you set sql.schema.checkColumns to "true".

sql.schema.check-columns

Type	java.lang.Boolean
Default	true
Description	Setting the property sql.schema.checkColumns to "true", causes the Content Server to compare the content type definition with the existing database schema for missing columns and matching widths of String properties. If there are differences, it depends on the setting of sql.schema.alterTables if the Content Server refuses to start ["false"] or if it adds and changes the columns automatically ["true"]. Checking the tables consumes a considerable amount of time, so that the server starts up more slowly. If sql.schema.checkColumns is set to "false", the Content Server will not check the columns. If there are differences, you will run into Content Server exceptions later on.

sql.schema.create-drop-indexes

Type	java.lang.Boolean
Default	false
Description	Setting the property to "true", causes the Content Server to automatically create and drop indexes on content type properties, according to the Index attribute in the doc-types.xml. This flag only affects existing columns. For newly added columns and tables, an index is always created if the Index attribute is set.

sql.schema.create-table

Type	java.lang.Boolean
Default	true

Description	The Content Server always checks if tables for the content types are missing. Setting the property to "true", causes the Content Server to create missing tables for new content types. If the setting is "false" and there are missing tables the Content Server refuses to start

`sql.store.collector.blob-preservation-period`

Type	java.lang.Integer
Default	86400
Description	The time in seconds, a blob, that has no reference to a resource, will be preserved.

`sql.store.collector.delay`

Type	java.lang.Integer
Default	86400
Description	The delay between two collections in seconds.

`sql.store.collector.initial-delay`

Type	java.lang.Integer
Default	600
Description	The delay in seconds before starting the first collection.

`sql.store.collector.start-time`

Type	java.lang.Integer
Default	-1
Description	The time of the day, in seconds from 0:00h in the default time zone, when the collector should run every day. If set to -1, it does not run at a fixed time. The initial delay is always enforced as a minimal delay after server startup.

`sql.store.collector.suspend`

Type	java.lang.Boolean

Default	false
Description	If set to true, unused blobs will not be deleted in the blob store. This should be used during backup if you have a non-transactional blob store.

`sql.store.convert`	
Type	java.util.Map<java.lang.String,java.lang.String>
Default	
Description	Converters which are used to convert custom XML formats which have been defined for the obsolete coremedia-sgmltext.dtd. **Example**: sql.store.convert.DocumentType.PropertyType=com.customer.XMLConverter

`sql.store.convert-correct-rich-text`	
Type	java.lang.Boolean
Default	true
Description	The editor in versions before SCI 4.1.38 created XML text, which was not valid according to the coremedia-richtext-1.0.dtd. If you have created content with versions before SCI 4.1.38, set the property to "true" to correct these errors (CoreMedia recommends to always use "true").

`sql.store.driver`	
Type	java.lang.String
Default	
Description	The JDBC driver used to connect to the database. **Example:** oracle.jdbc.driver.OracleDriver

`sql.store.isolation`	
Type	java.lang.String
Default	

Description	Define the transaction isolation level. An empty value uses the driver default setting.

sql.store.log-driver-messages

Type	java.lang.Boolean
Default	false
Description	Write messages of the JDBC driver to the logs

- hox.corem.server.sql.SQLStore and
- com.coremedia.cotopaxi.server.DatabaseProperties.

sql.store.login-user-name

Type	java.lang.String
Default	
Description	The user name for a database login. If not set, the value of "sql.store.user" will be used to log in to the database. In some cases the login username differs from the actual user, e.g. with PostgreSQL on Azure a postfix on the user name is necessary to log in.

Example:

- sql.store.login-user-name=username@domain
- sql.store.user=username

sql.store.password

Type	java.lang.String
Default	
Description	The password of the user at the database.

sql.store.prepared-statement-cache-size

Type	java.lang.Integer
Default	-1

Description	If set, denotes the maximum number of prepared statements that is kept open per database connection.

`sql.store.replace-substitute`

Type	java.lang.Boolean
Default	false
Description	There are problems with zOS DB2 databases to store characters that are not in the databases character set. Such characters are converted to the character with the code "0x001A" upon read. If the property is set to "true", this character will be replaced with the character defined in sql.store.substituteCharacter, thereby avoiding the replacement character that is illegal in XML texts.

`sql.store.sgml-cache-interval-seconds`

Type	java.lang.Integer
Default	0
Description	Seconds between cache statistics log entries to facility cap.server.store.sgmlcache

`sql.store.sgml-cache-size-bytes`

Type	java.lang.Long
Default	10000000
Description	Total size of XML objects cached in memory in bytes.

`sql.store.substitute-character`

Type	java.lang.String
Default	?
Description	The character, with which "0x001A" should be replaced.

`sql.store.url`

Type	java.lang.String

Default	
Description	The URL of the database to connect to.
	Example: jdbc:oracle:thin@HostName:Port:CM Replace HostName and Port with the appropriate values of the database host. Don't replace HostName with "localhost", this may cause problems with some JDBC drivers.

`sql.store.user`	
Type	java.lang.String
Default	
Description	The user name at the database, which must match the schema.

Table 4.9. SQL Properties

4.2.5 Properties for Replicator Configuration

`replicator.auto-restart`	
Type	java.lang.Boolean
Default	true
Description	Restarts the Replicator sessions if lost (true).

`replicator.check-blob-hashes`	
Type	java.lang.Boolean
Default	true
Description	Whether the replicator checks that the hashes of uploaded blobs are the same as the hashes stored on the Master Live Server. A failed check indicates that the blob store configuration differs between the Master Live Server and the Replication Live Server. Defaults to true. Only set to false to force replication to resume in case of an emergency. The resulting inconsistent blobs ids may affect blob retrieval and the affected Replication Live Server should be recreated afterwards.

replicator.check-stream

Type	java.lang.Boolean
Default	true
Description	Defines if the Replication Live Server checks the event queue for connection (true). The event stream is not checked during the initial replication.

replicator.check-timeout

Type	java.lang.Long
Default	300
Description	The time in seconds after which the Replication Live Server checks the event queue for connection.

replicator.chunking-threshold

Type	java.lang.Integer
Default	10000
Description	Sets the maximum number of events that is fetched from the Master Live Server in one chunk during startup; 0 indicates no limit. Lowering the threshold will reduce main memory consumption at the cost of startup times.

replicator.enable

Type	java.lang.Boolean
Default	true
Description	Defines if the Replicator should be started on start of the Replication Live Server (true).

replicator.force-online-switch

Type	java.lang.Boolean
Default	false

Description	Switch to runlevel online even if the repository contents have not yet been replicated from the Master Live Server. Ignored if not cap.server.init-runlevel=online.

replicator.log-events	
Type	java.lang.Boolean
Default	false
Description	All repository events will be logged (true).

replicator.max-accepted-lag	
Type	java.lang.Long
Default	100
Description	If the Replication Live Server is offline after a consistent replication but it is more than the given number of events behind the current timestamp, then the Replication Live Server will not go online until it catches up.

replicator.packager-flush-size	
Type	java.lang.Integer
Default	500
Description	Sets the maximum number of events that the packager holds in main memory before flushing to disk. Lowering the flush size will reduce main memory consumption at the cost of write performance.

replicator.password	
Type	java.lang.String
Default	replicator
Description	Configures the password to authenticate against the Master Live Server.

replicator.publication-ior-url	
Type	java.lang.String

Default	
Description	Sets the location where the Replicator can find the IOR of the Master Live Server.

`replicator.restart-replicator-on-error`

Type	java.lang.Boolean
Default	true
Description	Restart the Replicator if an error occurs on Replication Live Server side (true). Otherwise, the Replicator will be stopped.

`replicator.shutdown-server-on-error`

Type	java.lang.Boolean
Default	false
Description	Stop the Replication Live Server if an error occurs on Replication Live Server side.

`replicator.startup-timeout`

Type	java.lang.Boolean
Default	false
Description	Defines if the Replication Live Server waits on start for the Replicator to connect to the Master Live Server.

`replicator.tmp-dir`

Type	java.lang.String
Default	
Description	The path to the folder where the Replication Live Server should write temporary data during replication.
	The path is relative to the installation directory of the replicator application, so you will probably use an absolute path. During initial replication of a complete repository, a huge amount of data will be written to this directory, in the order of the repository size. So,

this directory should be located on a suitable file system with enough space left. During normal operation, only newly published blob content will be buffered on local disk

replicator.user	
Type	java.lang.String
Default	replicator
Description	Configures the user name to authenticate against the Master Live Server.

Table 4.10. Replicator Properties

4.2.6 Properties for Timezone and IOR

cap.client.server.ior.url	
Value	URL format `http://<server>:<port> /ior`
Default	
Description	This property determines where to get the IOR of the *contentserver*. `<server>` must be the name of the *Content Server* host. For <port> you have to set the server's web server HTTP port.

cap.client.timezone.default	
Value	TimeZone
Default	Europe/Berlin
Description	This parameter determines the used timezone. The standard value is *Europe/Berlin*. More time zones are documented in the class `java.util.TimeZone`.

Table 4.11. capclient.properties

4.2.7 Renamed Properties

Deprecated Name	New Name
`cap.server.ORBServerHost`	`com.coremedia.corba.server.host`
`cap.server.ORBServerPort`	`com.coremedia.corba.server.port`
`cap.server.ORBServerSSLPort`	`com.coremedia.corba.server.ssl.ssl-port`
`cap.server.blob.channel.timeout`	`cap.server.blob-channel-timeout`
`cap.server.encryptpasswords.key file`	`cap.server.encrypt-passwords-key-file`
`cap.server.groupcache.size`	`cap.server.cache.group-cache-size`
`cap.server.groupcache.status.interval`	`cap.server.cache.group-cache-status-interval`
`cap.server.http.port`	`cap.server.http-port`
`cap.server.init.runlevel`	`cap.server.init-runlevel`
`cap.server.memberfolderrights cache.size`	`cap.server.cache.member-folder-rights-cache-size`
`cap.server.namingpolicy.allowat`	`cap.server.naming-policy-allow-at`
`cap.server.repository.home`	`cap.server.repository-home`
`cap.server.repository.system`	`cap.server.repository-system`
`cap.server.resourcecache.size`	`cap.server.cache.resource-cache-size`
`cap.server.resourcecache.status.interval`	`cap.server.cache.resource-cache-status-interval`

Deprecated Name	New Name
`cap.server.rightscache.size`	`cap.server.cache.rights-cache-size`
`cap.server.rightscache.status.in terval`	`cap.server.cache.rights-cache-status-interval`
`cap.server.session.ping.interval`	`cap.server.session-ping-interval`
`cap.server.session.transi ent.threshold`	`cap.server.session-transient-threshold`
`cap.server.termina tion.timeout.seconds`	`cap.server.termination-timeout-seconds`
`cap.server.usercache.size`	`cap.server.cache.user-cache-size`
`cap.server.usercache.status.inter val`	`cap.server.cache.user-cache-status-interval`
`com.coremedia.corba.ORBRedirect or.original.host`	`com.coremedia.corba.client.redir ect.original-host`
`com.coremedia.corba.ORBRedirect or.original.port`	`com.coremedia.corba.client.redir ect.original-port`
`com.coremedia.corba.ORBRedirect or.redirect.host`	`com.coremedia.corba.client.redir ect.redirect-host`
`com.coremedia.corba.ORBRedirect or.redirect.port`	`com.coremedia.corba.client.redir ect.redirect-port`
`com.coremedia.corba.SSLClientSock etFactory.clearTextPort`	`com.coremedia.corba.cli ent.ssl.clear-text-ports`
`com.coremedia.corba.SSLClientSock etFactory.keystore`	`com.coremedia.corba.client.ssl.key store`
`com.coremedia.corba.SSLClientSock etFactory.passphrase`	`com.coremedia.corba.cli ent.ssl.passphrase`

Deprecated Name	New Name
`com.coremedia.corba.SSLClientSock etFactory.sslPort`	`com.coremedia.corba.cli ent.ssl.ssl-ports`
`com.coremedia.corba.SSLServerSock etFactory.keystore`	`com.coremedia.corba.server.ssl.key store`
`com.coremedia.corba.SSLServerSock etFactory.passphrase`	`com.coremedia.corba.serv er.ssl.passphrase`
`com.coremedia.corba.SingleIpSock etFactory.ip`	`com.coremedia.corba.server.single- ip`
`publisher.connection.timeout Seconds`	`publisher.connection-timeout- seconds`
`publisher.linkValidator.class`	`publisher.link-validator-class`
`publisher.status.interval`	`publisher.status-interval`
`replicator.Packager.flushSize`	`replicator.packager-flush-size`
`sql.modules.textEngine`	`sql.modules-text-engine`
`sql.store.collector.blob.preserva tionperiod`	`sql.store.collector.blob-preserva tion-period`
`sql.store.convert.correct-prefer ences`	`sql.store.convert-correct-prefer ences`
`sql.store.convert.correctRichText`	`sql.store.convert-correct-rich- text`
`sql.store.convert.defaultHoxImage Property`	`sql.store.convert-default-hox-im age-property`
`sql.store.convert.filterXmlChars`	`sql.store.convert-filter-xml-chars`
`sql.store.sgmlcache.interval`	`sql.store.sgml-cache-interval- seconds`

Deprecated Name	New Name
`sql.store.sgmlcache.size`	`sql.store.sgml-cache-size-bytes`
`sql.store.sgmlcache.sizeBytes`	`sql.store.sgml-cache-size-bytes`

Table 4.12. Renamed Content Server Properties

4.3 Headless Server Properties

4.3.1 Headless Server Spring Boot Properties

caas.graphql.max-execution-timeout	
Type	java.lang.Long
Default	
Description	Limits the allowed execution time for a query, set in milliseconds. 0 = no timeout.

caas.graphql.max-query-complexity	
Type	java.lang.Integer
Default	0
Description	Limits the complexity of a graphql query if set to a value greater than 0. 0 = off.

caas.graphql.max-query-depth	
Type	java.lang.Integer
Default	30
Description	Limits the depth of a graphql query if set to a value greater than 0. 0 = off.

caas.graphql.max-search-limit	
Type	java.lang.Integer
Default	200
Description	Limit the count of search results hits. Defaults to 200.

caas.search.cache.seconds	

Type	java.lang.Integer
Default	-1
Description	Time to cache search query results in seconds. Set it to "-1" to deactivate the search-query cache.

caas.stax.context-trace-enabled	
Type	java.lang.Boolean
Default	false
Description	When 'true', wraps ContextHandlers and OutputHandlers while parsing in order to tracelog all parsing events. This is for debugging purposes only. Defaults to false. Additionally this requires the loglevel for 'com.coremedia.caas.richtext.stax.handler.output' to be set to trace level: logging.level.com.coremedia.caas.richtext.stax.handler.output = trace

caas-rest.jslt-enabled	
Type	java.lang.Boolean
Default	true
Description	En/disables the JSLT transformation processor on REST endpoints

caas.cache-specs	
Type	java.util.Map<java.lang.String,java.lang.String>
Default	
Description	[Caffeine Cache] configuration: Cache specs by cache name.

Size based eviction:

- maximumSize=[long] - eviction occurs when the configured size limit of the cache is exceeded
- maximumWeight=[long] - The values are removed from the cache when the weight is exceeded

Time based eviction:

- expireAfterWrite = entry is expired after period is passed since the last write occurs
- expireAfterAccess = entry is expired after period is passed since the last read or write occurs
- refreshAfterWrite = refresh entries after a defined period automatically

Durations are represented by an integer, followed by one of "d", "h", "m", or "s", representing days, hours, minutes, or seconds respectively.

caas.forward-cookies

Type	java.lang.Boolean
Default	false
Description	Enables http cookie forwarding of incoming requests on endpoint 'graphql' and for persisted queries on '/caas/v1/*' towards outgoing, secondary requests to 'graphql' (triggered by '/preview') and cae remote link resolving.

caas.forward-header-names

Type	java.util.List<java.lang.String>
Default	
Description	Comaseparated enumeration of http headernames to be forwarded for incoming requests on endpoint 'graphql' and for persisted queries on '/caas/v1/*' towards outgoing, secondary requests to 'graphql' (triggered by /preview) and cae remote link resolving.

caas.graphql-restmapping-controller.enabled

Type	java.lang.Boolean
Default	true
Description	En-/disable REST mapping controller. Enabled by default (true).

caas.log-requests

Type	java.lang.Boolean
Default	false
Description	Enables logging for HTTP requests (excluding OPTIONS)

caas.media-cache-time

Type	java.time.Duration
Default	7d
Description	Maximum cache time to set in the com.coremedia.caas.web.controller.MediaController responses via the Cache-Control max-age header.

caas.metadata.property-mapping-location

Type	java.lang.String
Default	classpath*:graphql/metadata/propertyMapping.json
Description	Location of the property mapping file

caas.preview

Type	java.lang.Boolean
Default	false
Description	Enables preview mode

caas.querylist-search-cache-for-seconds

Type	java.lang.Integer
Default	0
Description	Time to cache query list search results in seconds. Set it to "-1" to deactivate the query list cache.

caas.site-filter-config-properties.excluded-field-names

Type	java.util.Collection<java.lang.String>
Default	[localizedVariant, localizedVariants, derivedSites]
Description	Fields to be excluded from site restrictions check.

To disable all fields (thus prohibiting all objects not belonging to the filtered site), add the config property without any field names (=empty list).

Adding additional fields requires to add the defaults as mentioned above to the config property! Defaults to "localizedVariant", "localizedVariants" and "derivedSites".

caas.solr.collection	
Type	java.lang.String
Default	preview
Description	Solr collection to use for CaaS Search

caas.swagger.enabled	
Type	java.lang.Boolean
Default	false
Description	Enables Swagger.

Table 4.13. Headless Server Properties

4.3.2 Persisted Query Properties

caas.persisted-queries.apollo-query-map-resources-pattern	
Type	java.lang.String
Default	classpath*:graphql/queries/apollo*.json
Description	Resource pattern for persisted queries in Apollo Persisted Queries format

caas.persisted-queries.automatic	
Type	java.lang.Boolean
Default	true

Description	Enable Apollo Automatic Persisted Queries

`caas.persisted-queries.exclude-file-name-pattern`	
Type	java.lang.String
Default	.*Fragment(s)?.graphql
Description	Resource pattern for persisted queries in Relay Compiler format

`caas.persisted-queries.query-resources-pattern`	
Type	java.lang.String
Default	classpath*:graphql/queries/*.graphql
Description	Resource pattern for persisted queries, one query per resource file. The filename w/o extension serves as the query id. See also: Spring PathMatchingResourcePatternResolver class.

`caas.persisted-queries.relay-query-map-resources-pattern`	
Type	java.lang.String
Default	graphql/queries/relay*.json
Description	Resource pattern for persisted queries in Relay Compiler format

`caas.persisted-queries.whitelist`	
Type	java.lang.Boolean
Default	false
Description	Query whitelisting. Set this to true to disallow any queries not loaded by some of the persisted queries pattern above.

Table 4.14. Persisted Query Properties

4.3.3 Remote Service Adapter Properties

caas.remote.base-url	
Type	java.lang.String
Default	http://localhost:8080/webapp/servlet/service-endpoint
Description	Base URL to the remote service handler

caas.remote.http-client-config.connection-pool-size	
Type	java.lang.Integer
Default	200
Description	The maximum total number of connections

caas.remote.http-client-config.connection-request-timeout	
Type	java.lang.Integer
Default	-1
Description	The timeout in milliseconds used when requesting a connection from the connection manager.

caas.remote.http-client-config.connection-time-to-live-ms	
Type	java.lang.Integer
Default	-1
Description	The maximum time to live of a connection.

caas.remote.http-client-config.connection-timeout	
Type	java.lang.Integer
Default	-1

Description	The timeout in milliseconds until a connection is established.

`caas.remote.http-client-config.socket-timeout`	
Type	java.lang.Integer
Default	-1
Description	The socket timeout in milliseconds.

`caas.remote.http-client-config.trust-all-ssl-certificates`	
Type	java.lang.Boolean
Default	false
Description	Enables a trust all manager for SSL connections. For development use only.

`caas.remote.http-client-config.validate-after-inactivity`	
Type	java.lang.Integer
Default	-1
Description	The period of inactivity in milliseconds after which connections must be re-validated.

Table 4.15. Headless Server Remote Properties

4.3.4 Properties of External Frameworks

`graphiql.enabled`	
Type	java.lang.Boolean
Default	false
Description	Enables GraphiQL.

`graphql.url`	

Type	null
Default	graphql
Description	GraphQL URL path.

Table 4.16. Headless Server External Framework Properties

4.3.5 Renamed Properties

Deprecated Name	New Name
`caas.context.trace.enabled`	`caas.stax.context-trace-enabled`
`caas.cache-capacities`	`cache.capacities.com.core` `media.caas.search.solr.SolrQueryCacheKey`
`caas.queryl` `ist.search.cache.seconds`	`caas.querylist-search-cache-for-` `seconds`

Table 4.17. Renamed Headless Server Properties

4.4 Workflow Server Properties

workflow.blob-cache-size	
Type	java.lang.Long
Default	32000000
Description	The size of the main disk cache of the UAPI

workflow.blob-streaming-size-threshold	
Type	java.lang.Integer
Default	131072
Description	The minimum size of streamed blobs

workflow.blob-streaming-threads	
Type	java.lang.Integer
Default	2
Description	The number of streaming threads

workflow.domain	
Type	java.lang.String
Default	
Description	The server's domain name

workflow.heap-cache-size	
Type	java.lang.Integer
Default	20000000

Description	The size of the main memory cache of the UAPI

`workflow.ior-url`

Type	java.lang.String
Default	
Description	The *Content Server*'s IOR URL

`workflow.map-role`

Type	java.util.Map<java.lang.String,java.lang.String>
Default	
Description	Role mappings are used to replace group names in a process definition file by other group names without having to actually modify the textual definition

`workflow.max-cached-blob-size`

Type	java.lang.Integer
Default	
Description	The maximum size of cached blobs

`workflow.navigatethrough`

Type	java.lang.Boolean
Default	true
Description	Whether to enable navigate through cache of the UAPI. This cache is used for computing read rights on folders in case no explicit read rights are defined and read rights on subfolders force read rights on folders on the path to these folders. Because these computations are expensive and sometimes unnecessary, this flag allows you to ignore navigate-through rights in the Workflow Server.

`workflow.password`

Type	java.lang.String

Default	workflow
Description	Define a password for the connection with the *Content Management Server*.

`workflow.pool.queue-size`

Type	java.lang.Integer
Default	0
Description	Pool queue limit.

`workflow.pool.threads-max`

Type	java.lang.Integer
Default	
Description	Pool thread max limit.

`workflow.pool.threads-min`

Type	java.lang.Integer
Default	0
Description	Pool thread min limit.

`workflow.server.adapters.directory`

Type	java.lang.String
Default	com.coremedia.workflow.impl.server.adapters.directory.UnifiedUserManagerAdapter
Description	The directory service adapter.

`workflow.server.adapters.persistence`

Type	java.lang.String
Default	com.coremedia.workflow.impl.server.adapters.persistence.GenericRDBMSAdapter

Description	The persistence adapter.

workflow.server.allow-live	
Type	java.lang.Boolean
Default	false
Description	Whether to allow connecting to a live *Content Server*

workflow.server.remote-action-handler	
Type	java.lang.String
Default	com.coremedia.cotopaxi.workflow.BuiltInRemoteActionHandler
Description	The remote action handler.

workflow.server.session.reaper-timeout	
Type	java.lang.Integer
Default	60
Description	The interval of the session reaper checks.

workflow.server.session.timeout	
Type	java.lang.Integer
Default	300
Description	The timeout until an inactive session is closed.

workflow.server.standalone	
Type	java.lang.String
Default	
Description	Possible values: 'true'|'false'|'embedded'

workflow.server.sweeper.concurrent	
Type	java.lang.Boolean
Default	false
Description	The concurrent sweep mode.

workflow.server.sweeper.delay	
Type	java.lang.Long
Default	0
Description	The sweep delay.

workflow.server.termination-timeout-seconds	
Type	java.lang.Integer
Default	30
Description	Timeout for waiting on running threads on shutdown

workflow.server.tx.idletimeout	
Type	java.lang.Integer
Default	30
Description	Transaction handler idle timeout.

workflow.server.tx.max	
Type	java.lang.Integer
Default	20
Description	Limits the maximum number of database connections.

workflow.server.uapi.connect-retry-delay-seconds	

Type	java.lang.Long
Default	10
Description	The delay in seconds the *Workflow Server* waits between retries to connect to the *Content Server* when starting.

`workflow.server.uapi.session-pool.max-idle-count`

Type	java.lang.Integer
Default	1000
Description	The maximum number of idle sessions to keep in the *Content Server* session pool.

`workflow.server.uapi.session-pool.max-idle-time-m-s`

Type	java.lang.Long
Default	60000
Description	The time in milliseconds after which an idle session of the *Content Server* session pool should be closed.

`workflow.server.uapi.session-pool.max-open-count`

Type	java.lang.Integer
Default	1000
Description	The maximum number of sessions to hold at any time (busy + idle) in the *Content Server* session pool.

`workflow.serviceindicator`

Type	java.lang.String
Default	
Description	The server's service indicator

`workflow.user`

Type	java.lang.String
Default	workflow
Description	Define a user for the connection with the *Content Management Server*.

Table 4.18. Workflow Server Properties

4.5 Studio Properties

4.5.1 Studio Configuration

The following list contains configuration properties for the *Studio*.

`studio.auto-logout.delay`

Type	java.lang.Integer
Default	1800
Description	The delay in seconds before a Studio session is automatically disconnected after inactivity. Inactivity is checked on the client. Set to 0 to disable auto logout (not recommended). For clients that cannot reach the Studio web application, the timeout of the server session determines the timeout of the Studio session. Defaults to 1800.

`studio.available-locales-content-path`

Type	java.lang.String
Default	/Settings/Options/Settings/LocaleSettings
Description	Path of the content that defines the available locales. Defaults to '/Settings/Options/Settings/LocaleSettings'.

`studio.available-locales-property-path`

Type	java.lang.String
Default	settings.availableLocales
Description	Property path to the Struct-StringListProperty containing the locales. Defaults to 'settings.availableLocales'.

`studio.dashboard.refresh.interval`

Type	java.lang.Integer

Default	30000
Description	The interval at which the dashboard is refreshed when visible in milliseconds (0 to disable automatic refresh) Defaults to 30.000 milliseconds.

`studio.default-time-zone`

Type	java.lang.String
Default	Europe/Berlin
Description	Default time zone in CoreMedia Studio. Make sure that the default time zone is included in the studio.timeZones list. Defaults to 'Europe/Berlin'.

`studio.differencing.max-markup-size`

Type	java.lang.Integer
Default	300000
Description	The maximum size of a markup object in bytes for which differences with other markup can be computed. By default, this value is set to 300000, which amounts to approximately 15000 words.

`studio.global-configuration-path`

Type	java.lang.String
Default	/Settings
Description	A global folder where additional settings are read from. Several Studio plugins lookup settings content from here. Defaults to '/Settings.

`studio.locales`

Type	java.util.List<java.lang.String>
Default	en,de
Description	A comma-separated list of locales from which the user can choose the locale in which the Studio is displayed. If unset, the list locales defaults to English or German.

```
studio.login-url
```

Type	java.lang.String
Default	
Description	If non-empty, a URL to which the Studio client should redirect a user who is not yet logged in. This supports single-sign-on solutions using an external login page. If empty, Studio shows a login form itself.

```
studio.preview-controller-pattern
```

Type	java.lang.String
Default	preview?id={0}
Description	Get the preview controller pattern. If it is empty or not defined, then use the default preview controller pattern.

```
studio.preview-url-prefix
```

Type	java.lang.String
Default	
Description	The URI prefix of the CAE preview web app. This prefix is used to compose preview URIs for contents, assuming that a standard CAE preview controller is mapped at the path 'preview'. Since studio and cae are usually deployed as independent spring boot application or as a containerized application (docker/kubernetes), it is neccesary to provide an absolute URL to load the preview, like 'https://your-host-name.com:40980/blue-print/servlet'

```
studio.preview-url-whitelist
```

Type	java.util.List<java.lang.String>
Default	
Description	Configures a list of valid preview origins. The preview integration does only work for contents from listed origins. Wildcards (*) may be used for valid origin entries. If left blank (or if the property is not listed at all), the origin of studio.previewUrlPrefix is the only accepted origin. NOTE: Once a whitelist is configured, the Studio has no chance to

set a target origin in outgoing messages anymore. Be aware that this is a minor security drawback. EXAMPLE: studio.previewUrlWhitelist=https://host1:port1,\ https://host2:port2,\ http://localhost.*,\ *.company.com Enabling Elastic Social tenants in the embedded preview requires including a placeholder CoreMedia Studio then replaces the token with the current tenant. In a &blueprintname; related project, this could be: studio.previewUrlPrefix=http://{0}.localhost:40081/blueprint/servlet

studio.site-configuration-path

Type	java.lang.String
Default	Options/Settings
Description	A local folder where additional settings are read from. Several Studio plugins lookup settings content from here. Defaults to 'Options/Settings.

studio.time-zones

Type	java.util.List<java.lang.String>
Default	Europe/Berlin
Description	Supported time zones in CoreMedia Studio. Provide a comma separated list of supported Java time zone IDs. Defaults to 'Europe/Berlin'.

studio.translation.show-pull-translation-start-window

Type	java.lang.Boolean
Default	true
Description	Configures Pull Translation (translation into preferred site) behavior: When set to true the workflow dialog is displayed, otherwise the workflow is started automatically. Defaults to true.

studio.upload.default-blob-property

Type	java.lang.String
Default	data

Description	The default blob property name of the default content to be used for bulk uploads, used if no mapping is found in the upload settings defined in content. Defaults to 'data'.

`studio.upload.default-content-type`	
Type	java.lang.String
Default	CMDownload
Description	The default content type to be used for bulk uploads, used if no mapping is found in the upload settings defined in content. Defaults to 'CMDownload'.

`studio.upload.upload-settings-struct-property`	
Type	java.lang.String
Default	settings
Description	The name of the struct property to be used when the UploadSettings document is read. Defaults to 'settings'.

`studio.validate-before`	
Type	com.coremedia.rest.cap.config.ValidatedActionLevel
Default	
Description	Defines which content actions are not allowed if the content has error issues. As some actions are in a dependency relationship (e.g. APPROVE depends on CHECKIN), only the least inclusive actions need to be declared (e.g. CHECKIN entails the check of CHECKIN and APPROVE actions). Currently, the only supported OPTIONS are CHECKIN, APPROVE or nothing for no restriction. Defaults to APPROVE, i.e. content approval and publication is not allowed in case of error issues.

`studio.validators.navigation.ignore-path`	
Type	java.util.List<java.lang.String>
Default	/Settings,/Home,/System

Description	This list contains comma-separated string values. If a content path matches one of these values or is a subfolder of it, the corresponding validator won't continue it's validation.
	Wildcard annotations are allowed here, e.g. '/Home/*' matches '/Home/Adam' or '/Sites/*/*/Navigation/*' matches '/Sites/Chef Corp./English/Navigation/Content ABC'.

`studio.workflow.translation.extended-workflow`

Type	java.lang.Integer
Default	100
Description	The value that defines how many dependent contents will be returned in "Translation Workflow Start window" when starting a translation workflow. Defaults to 100.

Table 4.19. Studio Properties

4.5.2 Preview URL Service Properties

The following list contains configuration properties related to the Multi Preview Menu in *Studio*.

`preview.urlservice.content-type`

Type	java.lang.String
Default	CMSettings
Description	Defines the content type that is used to configure previews.

`preview.urlservice.global-path`

Type	java.lang.String
Default	/Settings/Options/Settings/Multi Preview
Description	Defines the repository folder path with contents that configure previews for all sites and contents without a site. Subfolders are ignored.

`preview.urlservice.headless-preview-host`

Type	java.lang.String
Default	
Description	Defined the host of the headless preview server.

`preview.urlservice.preview-url-allow-list`

Type	java.util.List<java.lang.String>
Default	
Description	Defines the commercePreviewUrlAllowList. In order to limit the urls in the preview frame and prevent CSRF.

`preview.urlservice.property`

Type	java.lang.String
Default	settings
Description	Defines the name of the struct property in the configured content type.

`preview.urlservice.site-path`

Type	java.lang.String
Default	Options/Settings/Multi Preview
Description	Defines the repository folder path below a site root folder with contents that configure previews for one site. Subfolders are ignored.

Table 4.20. Preview URL Service Properties

4.5.3 Content Security Policy Configuration

The following list contains configuration properties related to Content Security Policy (CSP) in the *Studio*.

```
studio.security.csp.connect-src
```

Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'connect-src' policy directive. Defaults to 'self'.

studio.security.csp.csp-mode

Type	com.coremedia.rest.security.util.CSPMode
Default	
Description	Level of Content Security Policy protection (CSP). For further details about CSP and the default policy settings please refer to the Studio Developer Manual. Allowed values are:

- ENFORCE - Enable CSP protection. This is the default.
- ENFORCE_ALLOW_DISABLE - Enable CSP protection unless the 'disableCsp' query parameter is 'true'.
- REPORT - Enable CSP report only mode without enforcing CSP protection.
- DISABLE - Disable CSP protection and reporting.

studio.security.csp.font-src

Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'font-src' policy directive. Defaults to 'self'.

studio.security.csp.frame-ancestors

Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'frame-ancestors' policy directive. Defaults to 'self'. @deprecated Configuring this setting does not have an effect anymore. Please configure this directive in deployment.

studio.security.csp.frame-src

Type	java.util.List<java.lang.String>

Default	
Description	List of values for the 'frame-src' policy directive. The hierarchy of default values for this directive is as follows

1. studio.previewUrlWhitelist values if specified
2. schema and authority of studio.previewUrlPrefix if specified
3. 'self'

To allow YouTube videos inside the external preview, add the Youtube URL: studio.security.csp.frameSrc=http://localhost:40980,*.coremedia.vm:40980, *.coremedia.vm,*.coremedia.com,*.coremedia.com:8000,*.coremedia.vm:8000, 'self',www.youtube.com

studio.security.csp.img-src	
Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'img-src' policy directive. Defaults to 'self'.

studio.security.csp.manifest-src	
Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'manifest-src' policy directive. Defaults to 'self'.

studio.security.csp.media-src	
Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'media-src' policy directive. Defaults to 'self'.

studio.security.csp.object-src	
Type	java.util.List<java.lang.String>
Default	

Description	List of values for the 'object-src' policy directive. Defaults to 'self'.

studio.security.csp.report-uri	
Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'report-uri' policy directive. If no custom list is provided the directive is not included.

studio.security.csp.script-src	
Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'script-src' policy directive. Defaults to 'self','unsafe-eval'.

studio.security.csp.style-src	
Type	java.util.List<java.lang.String>
Default	
Description	List of values for the 'style-src' policy directive. Defaults to 'self','unsafe-inline'.

Table 4.21. Content Security Policy Related Studio Properties

4.5.4 Content Hub Configuration

The following table lists the configuration properties for the *CoreMedia Content Hub*.

contenthub.studio.global-configuration-path	
Type	java.lang.String
Default	/Settings/Options/Settings/Content Hub

Description	Defines the global configuration path. The Content Hub will look up 'Connection' settings document in this folders. The default value is /Settings/Options/Settings/Content Hub.

`contenthub.studio.site-configuration-path`	
Type	java.lang.String
Default	/Options/Settings/Content Hub
Description	Defines the site specific configuration path. If a Content Hub connection should only be available sites, the 'Connection' settings document can be but in this subfolder of a site. The default value is /Options/Settings.

Table 4.22. Content Hub Properties

4.5.5 Feedback Hub Configuration

The following table lists the configuration properties for the *CoreMedia Feedback Hub*.

`feedbackhub.bindings.content-type`	
Type	java.lang.String
Default	CMSettings
Description	Defines the content type that is used to configure Feedback Hub bindings.

`feedbackhub.bindings.global-path`	
Type	java.lang.String
Default	/Settings/Options/Settings/Feedback Hub
Description	Defines the repository folder path with contents that configure Feedback Hub bindings for all sites and contents without a site. Subfolders are ignored.

`feedbackhub.bindings.property`	
Type	java.lang.String

Default	settings
Description	Defines the name of the struct property in the configured content type with the Feedback Hub binding configuration.

`feedbackhub.bindings.site-path`

Type	java.lang.String
Default	Options/Settings/Feedback Hub
Description	Defines the repository folder path below a site root folder with contents that configure Feedback Hub bindings for one site. Subfolders are ignored.

Table 4.23. Feedback Hub Properties

4.5.6 Editorial Comments Configuration

The following table lists the configuration properties for the *CoreMedia Editorial Comments* feature, which establishes a connection to the relational database.

The most important property is `editorial.comments.datasource.url` which sets the URL to connect to.

In case another *schema/username/password* has to be configured use the properties `editorial.comments.db.schema/editorial.comments.db.user name/editorial.comments.db.password`.

`editorial.comments.datasource.driver-class-name`

Type	java.lang.String
Default	
Description	Required to be set according to your database. See official spring documentation 'spring.datasource.driver-class-name' for detailed information

`editorial.comments.datasource.hikari.connection-timeout`

Type	java.lang.Integer
Default	20000ms

Description	Value must not be greater then the studio request timeout. See official spring document-ation 'spring.datasource.hikari.connection-timeout' for more information

editorial.comments.datasource.password

Type	java.lang.String
Default	cm_editorial_comments
Description	See official spring documentation 'spring.datasource.password' for more information

editorial.comments.datasource.url

Type	java.lang.String
Default	
Description	Required to be set according to your database. See official spring documentation 'spring.datasource.url' for detailed information

editorial.comments.datasource.username

Type	java.lang.String
Default	cm_editorial_comments
Description	See official spring documentation 'spring.datasource.username' for more information

editorial.comments.db.password

Type	java.lang.String
Default	cm_editorial_comments
Description	Use to set the password for hibernate and liquibase.

editorial.comments.db.schema

Type	java.lang.String
Default	cm_editorial_comments

Description	Use to set the schema for hibernate and liquibase.

`editorial.comments.db.username`

Type	java.lang.String
Default	cm_editorial_comments
Description	Use to set the username for hibernate and liquibase.

`editorial.comments.jpa.database-platform`

Type	java.lang.String
Default	
Description	See official spring documentation 'spring.jpa.properties.hibernate.database-platform' for detailed information

`editorial.comments.jpa.properties.hibernate.default_schema`

Type	java.lang.String
Default	cm_editorial_comments
Description	See official spring documentation 'spring.jpa.properties.hibernate.default_schema' for more information

`editorial.comments.liquibase.change-log`

Type	java.lang.String
Default	classpath:db/changelog/db.changelog-editorial-comments.xml
Description	See official liquibase documentation 'liquibase.change-log' for more information

`editorial.comments.liquibase.default-schema`

Type	java.lang.String
Default	cm_editorial_comments

| Description | See official liquibase documentation 'liquibase.default-schema' for more information |

`editorial.comments.liquibase.enabled`

Type	java.lang.Boolean
Default	true
Description	Use this property to disable liquibase, however this means that you need to apply the changesets to the database manually. You can either do that by activating liquibase for at least one startup of a Studio-Server, or run liquibase manually (https://docs.li-quibase.com/tools-integrations/cli/home.html) after an upgrade.

`editorial.comments.liquibase.password`

Type	java.lang.String
Default	cm_editorial_comments
Description	See official liquibase documentation 'liquibase.password' for more information

`editorial.comments.liquibase.user`

Type	java.lang.String
Default	cm_editorial_comments
Description	See official liquibase documentation 'liquibase.user' for more information

`editorial.comments.notification-strategies.created-comment-on-content`

Type	java.lang.Boolean
Default	true
Description	This property enables the CreatedCommentOnContentCollectUsersToNotifyStrategy which collects all users who wrote a comment at this content in the last 30 days. True if the strategy is enabled, false if the strategy is disabled.

`editorial.comments.notification-strategies.edited-content-in-last-thirty-days`

Type	java.lang.Boolean
Default	true
Description	This property enables the EditedContentInLastThirtyDaysCollectUsersToNotifyStrategy which collects all users who edited the given content in last 30 days. True if the strategy is enabled, false if the strategy is disabled.

`editorial.comments.notification-strategies.my-edited-contents`	
Type	java.lang.Boolean
Default	true
Description	This property enables the MyEditedContentsCollectUsersToNotifyStrategy, which notifies every user that has the Content where an EditorialComment was created for in his "myEditedContent" List. true if the strategy is enabled, false if the strategy is disabled.

Table 4.24. Editorial Comments Properties

> **NOTE**
>
> If the startup of a Studio-Server instance has been interrupted, it is possible that a lock is left by Liquibase on the database schema cm_editorial_comments. In this case it is necessary to remove the lock manually, as described at https://docs.liquibase.com/con-cepts/basic/databasechangeloglock-table.html. It is also possible to disable Liquibase with the configuration `editorial.comments.liquibase.enabled` (for further details, refer to Section 3.5, "Editorial Comments Database Configuration" in *Studio Developer Manual*).

4.6 Commerce Hub Properties

commerce.hub.cache.capacities	
Type	java.util.Map<java.lang.String,java.lang.Long>
Default	
Description	The capacity configuration of the caching for the *Commerce Hub*.
	Please refer to the java documentation of the class
	• com.coremedia.blueprint.base.livecontext.ecommerce.common.AbstractCommerceCacheKey

commerce.hub.cache.enabled	
Type	java.lang.Boolean
Default	true
Description	Activates/deactivates caching for the *Commerce Hub*.

commerce.hub.cache.timeout-seconds	
Type	java.util.Map<java.lang.String,java.lang.Long>
Default	
Description	The timeout configuration of the caching for the *Commerce Hub*.

commerce.hub.data.custom-entity-params	
Type	java.util.Map<java.lang.String,java.lang.String>
Default	
Description	The custom entity parameters.

commerce.hub.data.deadline	

Type	java.time.Duration
Default	20s
Description	Deadline value to be used when calling a grpc service method. After this time, the client no longer expects an answer and cancels the call. Default is 20 seconds. If no unit is given the value is taken as seconds.

`commerce.hub.data.endpoint-shutdown-timeout-millis`	
Type	java.lang.Integer
Default	1000
Description	Timeout millis to be used when shutting down the data endpoints.

`commerce.hub.data.endpoints`	
Type	java.util.Map<java.lang.String,java.lang.String>
Default	
Description	The endpoints of the commerce adapter services to connect to. Each key-value pair represents one commerce adapter service to be connected. The keys can freely be defined, but they should easily identify the connected commerce system. The keys are also used as health-indicator keys. Please refer to the java documentation of the method • com.coremedia.blueprint.base.livecontext.client.settings.CommerceSettings#getEndpoint() The value consists of the host name and the port of the connected commerce adapter service.

Table 4.25. Commerce Hub Properties

4.7 Elastic Social Properties

4.7.1 General Elastic Social Properties

elastic.core.event-collection-ttl-sec	
Type	java.lang.Integer
Default	300
Description	Time-To-Live index (TTL) for the collection event in seconds. Set it to 0 to disable automatic deletion.

elastic.social.comments.auto-reject-interval-ms	
Type	java.time.Duration
Default	
Description	Interval in milliseconds in which comments will be automatically rejected when the number of complaints reaches the limit {@code elastic.social.comments.auto-reject-limit}.

elastic.social.comments.auto-reject-limit	
Type	java.lang.Integer
Default	0
Description	Number of complaints after which a comment will automatically be rejected. Set to 0 to disable automatic rejection.

elastic.social.mails.on-profile-changes	
Type	java.lang.Boolean
Default	true

Description	Flag which states if a mail should be send on profile changes

`elastic.social.mails.on-registration`

Type	java.lang.Boolean
Default	true
Description	Flag which states if a mail should be send on registration

`elastic.social.password-hash-algorithm`

Type	java.lang.String
Default	
Description	The password hashing algorithm for elastic social user passwords. Allowed values are: {@code sha1} and {@code bcrypt:N}. The algorithm {@code sha1} is not recommended as it is vulnerable to brute-force attacks. The work factor N for bcrypt salt generation can be changed from the default value to an integer value between 4 and 31 (inclusive). This should be adapted to the available CPU resources.

`elastic.social.users.auto-block-interval-ms`

Type	java.time.Duration
Default	
Description	Interval in milliseconds in which users will be automatically blocked when the number of complaints reaches the limit {@code elastic.social.users.auto-block-limit}

`elastic.social.users.auto-block-limit`

Type	java.lang.Integer
Default	0
Description	A number of complaints after which a user will automatically be blocked. Set to 0 to disable automatic rejection.

`elastic.social.users.pre-moderation-properties`

Type	java.util.List<java.lang.String>
Default	
Description	Properties for the pre-moderation of users.

elastic.social.users.token-expiration-time-ms	
Type	java.time.Duration
Default	
Description	Token expiration time in milliseconds, defines the lifetime of the user token used for user activation and password reset. After expiration, the token becomes invalid

Table 4.26. Elastic Social Properties

4.7.2 MongoDB Properties

mongodb.client-uri	
Type	java.lang.String
Default	mongodb://localhost:27017
Description	The standard MongoDB connection string URI is used to configure your MongoDB connection, for example, it allows you to configure read preferences and write concerns. The format of a client URI is documented under the following link: http://docs.mongodb.org/manual/reference/connection-string/.

mongodb.host	
Type	java.lang.String
Default	localhost
Description	Set the hostname of the mongoDb server. Only relevant for development purposes.

mongodb.models.create-indexes	

Type	java.lang.Boolean
Default	true
Description	Set to false to skip index creation on startup. This can be useful to speed up initial data import. Make sure to set this true before production use.

`mongodb.prefix`

Type	java.lang.String
Default	elastic
Description	Prefix for MongoDB database names. Change this when sharing a MongoDB installation with other Elastic Core applications.

Table 4.27. MongoDb Properties

4.7.3 Counter Properties

`counters.aggregation-interval-milliseconds.daily`

Type	java.time.Duration
Default	
Description	Interval in milliseconds in which aggregated values for counter events of the last 24 hours will be updated. Defaults to 300000 (5 min).

`counters.aggregation-interval-milliseconds.for-all`

Type	java.time.Duration
Default	
Description	Interval in milliseconds in which aggregated values for all historic counter events will be updated. Defaults to 86400000 (24 h).

`counters.aggregation-interval-milliseconds.monthly`

Type	java.time.Duration
Default	
Description	Interval in milliseconds in which aggregated values for counter events of the last 30 days will be updated. Defaults to 86400000 (24 hours).

`counters.aggregation-interval-milliseconds.weekly`

Type	java.time.Duration
Default	
Description	Interval in milliseconds in which aggregated values for counter events of the last 7 days will be updated. Defaults to 10800000 (3 hours).

`counters.aggregation-interval-milliseconds.yearly`

Type	java.time.Duration
Default	
Description	Interval in milliseconds in which aggregated values for counter events of the last 365 days will be updated. Defaults to 86400000 (24 hours).

Table 4.28. Counters Properties

4.7.4 Task Queue Properties

`taskqueues.execution-timeout-milli-seconds`

Type	java.time.Duration
Default	
Description	Specifies the task queue execution timeout in milliseconds. Defaults to 600000 (10 m).

`taskqueues.execution-warning-timeout-milli-seconds`

Type	java.time.Duration

Default	

Description	Specifies duration in milliseconds after log warning for all long running tasks should be shown. Defaults to 300000 (5 m).

`taskqueues.number-of-retries`

Type	java.lang.Integer

Default	10

Description	Specifies the number of retries for a MongoDB task.

`taskqueues.polling-interval-milli-seconds`

Type	java.time.Duration

Default	

Description	Specifies the duration of the task queue polling interval in milliseconds. Used by the scheduler in the MongoDbTaskQueueService. Defaults to 500 (0.5 s).

`taskqueues.recovery-interval-milli-seconds`

Type	java.time.Duration

Default	

Description	Specifies the duration of the task queue recovery interval in milliseconds. Used by the scheduler in the MongoDbTaskQueueService. Defaults to 60000 (1 m).

`taskqueues.retry-interval-milli-seconds`

Type	java.time.Duration

Default	

Description	Specifies task queue retry interval in milliseconds. Defaults to 300000 (5 m).

`taskqueues.worker-node`

Type	java.lang.Boolean

Default	false
Description	Set to false to disable execution of background tasks. This setting can be used to differ-entiate applications nodes into worker nodes which only execute background tasks and rendering nodes which serve requests.

Table 4.29. Task-Queues Properties

4.7.5 Elastic Social Solr Properties

elastic.solr.cloud	
Type	java.lang.Boolean
Default	
Description	Whether to connect to SolrCloud. Do not set this property to let Elastic Core use the standard CoreMedia setting. If true, connect to a SolrCloud cluster. SolrCloud connection details must be set either as ZooKeeper addresses (elastic.solr.zookeeper.addresses) or, if the former is unset or empty as HTTP URLs (elastic.solr.url). If false, connect to stand-alone Solr nodes via HTTP URLs (elastic.solr.url).

elastic.solr.connection-timeout	
Type	java.lang.Integer
Default	
Description	Connection timeout in milliseconds, or 0 for no timeout, or a negative value to use SolrClient default. Do not set this property to let Elastic Core use the standard CoreMedia setting.

elastic.solr.index-config	
Type	java.lang.String
Default	elastic
Description	Name of the Apache Solr config set for Elastic Core applications. This config set must exist in the Solr server, typically as subdirectory of "$SOLR_HOME/configsets".

```
elastic.solr.index-data-directory
```

Type	java.lang.String
Default	data
Description	Value for the "dataDir" parameter of the Solr CoreAdmin API / Collection API request to create a Solr index.

```
elastic.solr.index-prefix
```

Type	java.lang.String
Default	elastic
Description	Prefix for Apache Solr index names. Change this when sharing an Apache Solr installation with other Elastic Core applications.

```
elastic.solr.lazy-index-creation
```

Type	java.lang.Boolean
Default	false
Description	Whether all indices are created lazily. The default is false, and all indices are created eagerly except indices for tenants configured with elastic.solr.tenants-with-lazy-index-creation.

```
elastic.solr.password
```

Type	java.lang.String
Default	
Description	Password for HTTP basic authentication, used if a non-empty solr.username has been specified. The value may have been encrypted with the tool "cm encryptpasswordproperty". Do not set this property to let Elastic Core use the standard CoreMedia setting.

```
elastic.solr.socket-timeout
```

Type	java.lang.Integer

Default	
Description	Socket timeout in milliseconds, or 0 for no timeout, or a negative value to use SolrClient default. Do not set this property to let Elastic Core use the standard CoreMedia setting.

`elastic.solr.tenants-with-lazy-index-creation`

Type	java.util.List<java.lang.String>

Default	
Description	List of tenants for which indices are not eagerly created when the application starts, but only upon first access.

`elastic.solr.url`

Type	java.lang.String

Default	
Description	The Solr URL to connect to, or null to let Elastic Core use the standard CoreMedia setting. In a Solr master/slave setup, this must be the URL of the Solr master. For SolrCloud, it's recommended to set elastic.solr.zookeeper.addresses instead.

`elastic.solr.username`

Type	java.lang.String

Default	
Description	Username for HTTP basic authentication, or empty string for no authentication. Do not set this property to let Elastic Core use the standard CoreMedia setting.

`elastic.solr.zookeeper.addresses`

Type	java.util.List<java.lang.String>

Default	
Description	ZooKeeper addresses for connecting to SolrCloud. Only used if elastic.solr.cloud=true. Do not set this property to let Elastic Core use the standard CoreMedia setting.

elastic.solr.zookeeper.chroot	
Type	java.lang.String
Default	
Description	Optional ZooKeeper chroot path for Solr. ZooKeeper chroot support makes it possible to isolate the SolrCloud tree in a ZooKeeper instance that is Only used if elastic.solr.cloud=true and elastic.solr.zookeeper.addresses is set to non-empty value. Do not set this property to let Elastic Core use the standard CoreMedia setting.

elastic.solr.zookeeper.client-timeout	
Type	java.lang.Integer
Default	
Description	Client-timeout for ZooKeeper in milliseconds, or a negative value to use SolrClient default. Only used if elastic.solr.cloud=true and elastic.solr.zookeeper.addresses is set to non-empty value. Do not set this property to let Elastic Core use the standard CoreMedia setting.

elastic.solr.zookeeper.connect-timeout	
Type	java.lang.Integer
Default	
Description	Connect-timeout for ZooKeeper in milliseconds, or a negative value to use SolrClient default. Only used if elastic.solr.cloud=true and elastic.solr.zookeeper.addresses is set to non-empty value. Do not set this property to let Elastic Core use the standard CoreMedia setting.

Table 4.30. Elastic Solr Properties

4.7.6 Renamed Properties

Deprecated Name	New Name
`counters.aggregationInterval`	`counters.aggregation-interval-milliseconds.for-all`
`counters.aggregationInterval.daily`	`counters.aggregation-interval-milliseconds.daily`
`counters.aggregationInterval.monthly`	`counters.aggregation-interval-milliseconds.monthly`
`counters.aggregationInterval.weekly`	`counters.aggregation-interval-milliseconds.weekly`
`counters.aggregationInterval.yearly`	`counters.aggregation-interval-milliseconds.yearly`

Table 4.31. Renamed Elastic Social Properties

4.8 Importer Properties

import.user	
Value	String
Default	importer
Description	The name of the CoreMedia user with which the importer logs on. Make sure that the user has the rights required to carry out operations triggered by the import process, for example, creating a new document, editing, approving, publishing. For this purpose, the standard CoreMedia installation offers a predefined user called importer (password also importer).

import.password	
Value	String
Default	importer
Description	The password of the user to log in with.

import.autoLogoutSeconds	
Value	int
Default	-1
Description	This property defines the time of inactivity in seconds after which the importer should log out. When the importer is active again, it will log in at the server automatically. A value of "-1" means that the importer will not log out.

import.multiResultGeneratorFactory.property.sleepingSeconds	
Value	int
Default	-1
Description	An importer remains logged in per default, whether data are imported or not. When configuring SubDirGenerators, the property defines the number of seconds for

the importer to be inactive after the completion of the import. If the number of seconds is very large, it is reasonable to log out the importer automatically. In this case, the released importer license can be used by another importer. Note that the special value "-1" will cause the importer to terminate after importing the contents of the inbox directories.

Table 4.32. Properties of the cm-xmlimport.properties file

import.loginTimeoutSeconds	
Value	long
Default	-1
Description	This property defines the timeout for login attempts after which the importer aborts. If `import.loginTimeoutSeconds=-1`, the importer tries to login forever without abortion.

import.enforceCompleteVersion	
Value	Boolean
Default	true
Description	This property handles the processing of XML importer files. See Section 4.1, "The CoreMedia XML Format" in *Importer Manual* for details on the CoreMedia XML format. • `import.enforceCompleteVersion=true` For each <version> element in the importer file a new version will be created in the CoreMedia repository. For all properties of a version the values must be given. It is not allowed to omit a property. • `import.enforceCompleteVersion=false` Now it is possible to omit even all property elements of a version. If there are only action elements and the document already exists on the server, then no new version is created and the corresponding actions are applied to the document (delete) or to the latest document version on the server (approve, delete). If there is at least one <property> element in the <version> element then for every property that is specified in the document type but missing in the XML importer file, the property value of the predecessor document version is taken. If there is no predecessor version, then a default value is inserted, that depends on the property type.

import.validate-textproperty	

Value	Boolean
Default	false
Description	If "true" the importer validates all XML text properties against the associated DTD. If a validation fails, no document is created on the server. For big XML properties the validation may take some time.

import.removeBrokenLinks	
Value	Boolean
Default	false
Description	If "true" the importer removes broken content links in link list and markup properties. In markup properties only the link tag (a or img) is removed, not the containing link text. Be careful when enabling this option, as it may lead to invalid XML in markup properties.

import.entityResolverClass	
Value	class name
Default	see description
Description	Configures the name of a class of type org.xml.sax.EntityResolver used to resolve entities in markup properties during XML validation. The default value is com.coremedia.xml.ClasspathURLEntityResolver.

Table 4.33. Properties of the cm-xmlimport.properties file

4.9 Search Related Properties

4.9.1 Content Feeder Properties

Solr specific properties for Content Feeder

feeder.solr.partial-updates.enabled	
Type	java.lang.Boolean
Default	true
Description	Whether partial updates are supported for updating content metadata in Solr. This requires that all fields in the Solr index are configured as stored="true" or docValues="true" except fields that are copyField destinations, which must be configured as stored="false". This is because partial updates are applied to the index document reconstructed from the existing stored field values.

feeder.solr.partial-updates.skip-index-check	
Type	java.lang.Boolean
Default	false
Description	If feeder.solr.partial-updates.enabled is true, the Solr index schema is analyzed whether fields are stored as required for partial updates. The Feeder will log a warning and not use partial update functionality if the index seems to not support it. You can set this property to true to skip the check.

feeder.solr.send-retry-delay	
Type	java.time.Duration
Default	30s
Description	The delay to wait before the Feeder retries to send data after failures from Solr.

solr.cloud	

Type	java.lang.Boolean
Default	false
Description	Whether to connect to SolrCloud. If true, connect to a SolrCloud cluster. SolrCloud connection details must be set either as ZooKeeper addresses (solr.zookeeper.addresses) or, if the former is unset or empty as HTTP URLs (solr.url). If false, connect to stand-alone Solr nodes via HTTP URLs (solr.url).

`solr.connection-timeout`

Type	java.lang.Integer
Default	0
Description	Connection timeout in milliseconds, or 0 for no timeout, or a negative value to use SolrClient default.

`solr.content.collection`

Type	java.lang.String
Default	studio
Description	The name of the Solr collection for editorial search.

`solr.content.config-set`

Type	java.lang.String
Default	content
Description	The name of the Solr config set to use when creating the collection for editorial search. This property is used by the Content Feeder.

`solr.index-data-directory`

Type	java.lang.String
Default	data

Description	Value for the "dataDir" parameter of the Solr CoreAdmin API / Collection API request to create a Solr index.

solr.password

Type	java.lang.String
Default	
Description	Password for HTTP basic authentication, used if a non-empty solr.username has been specified. The value may have been encrypted with the tool "cm encryptpasswordproperty".

solr.socket-timeout

Type	java.lang.Integer
Default	600000
Description	Socket timeout in milliseconds, or 0 for no timeout, or a negative value to use SolrClient default.

solr.url

Type	java.util.List<java.lang.String>
Default	http://localhost:40080/solr
Description	The list of Solr URLs to connect to. These URLs are ignored if connecting to SolrCloud (solr.cloud=true) and non-empty ZooKeeper addresses (solr.zookeeper.addresses) have been set. For a Feeder application that is not connected to a SolrCloud cluster, a single URL to the Solr master must be configured.

solr.use-xml-response-writer

Type	java.lang.Boolean
Default	false
Description	Whether SolrJ should use XML response format instead of Javabin format.

solr.username

Type	java.lang.String
Default	
Description	Username for HTTP basic authentication, or empty string for no authentication.

solr.zookeeper.addresses	
Type	java.util.List<java.lang.String>
Default	
Description	ZooKeeper addresses for connecting to SolrCloud. Only used if solr.cloud=true.

solr.zookeeper.chroot	
Type	java.lang.String
Default	
Description	Optional ZooKeeper chroot path for Solr. ZooKeeper chroot support makes it possible to isolate the SolrCloud tree in a ZooKeeper instance that is Only used if solr.cloud=true and solr.zookeeper.addresses is set to non-empty value.

solr.zookeeper.client-timeout	
Type	java.lang.Integer
Default	10000
Description	Client-timeout for ZooKeeper in milliseconds, or a negative value to use SolrClient default. Only used if solr.cloud=true and solr.zookeeper.addresses is set to non-empty value.

solr.zookeeper.connect-timeout	
Type	java.lang.Integer
Default	10000

Description	Connect-timeout for ZooKeeper in milliseconds, or a negative value to use SolrClient default. Only used if solr.cloud=true and solr.zookeeper.addresses is set to non-empty value.

Table 4.34. Content Feeder Solr Configuration Properties

Login properties for Content Feeder

The following properties are used to define the login data for the *Content Server* and the administration page of the *Search Engine*.

`feeder.management.user`	
Value	user name
Default	feeder
Description	The user name to be used in the HTTP authentication of the administration page of the *Content Feeder*. This is not an account from the user management of the *Content Server*.

`feeder.management.password`	
Value	password
Default	feeder
Description	The password to be used in the HTTP authentication of the administration page of the *Content Feeder*.

`repository.user`	
Value	user name
Default	feeder
Description	The user account the *Content Feeder* uses to read content.

`repository.password`	
Value	password

Default	feeder
Description	The password for the user account of the *Content Feeder*.

Table 4.35. Properties for login

Partial Update specific Properties for Content Feeder

With this property you can configure the usage of partial updates, if supported by the connected Indexer - for example for Solr as configured with property `feeder.solr.partial-updates.enabled`.

`feeder.partialUpdate.aspects`	
Value	comma-separated list of index document aspects or *
Default	*
Description	The aspects of index documents that can be updated with a partial update, provided that the connected Indexer supports partial updates [for example, `feeder.solr.partial-updates.enabled=true` for Solr]. Multiple values are separated by comma. Use the special value * to use partial updates for all aspects, if possible. An empty value means that partial updates are not used. See the API documentation of `Feedable.isPartialUpdate`, `FeedableAspect` and `ContentFeedableAspect` in package `com.coremedia.cap.feeder` for more details.

Table 4.36. Partial update configuration

Batch configuration properties for Content Feeder

With these properties you can configure the processing of batches.

`feeder.batch.max-bytes`	
Type	org.springframework.util.unit.DataSize
Default	5MB
Description	The maximum batch size in bytes. The Feeder sends a batch to the search engine if its maximum size would be exceeded when adding more entries. Note, that byte computa-

tion is a rough estimate only. A smaller batch may be sent if the maximum number of index documents is reached before, or if configured delays are reached.

feeder.batch.max-open	
Type	java.lang.Integer
Default	5
Description	The maximum number of batches indexed in parallel. This setting is not used with the default integration of Apache Solr but only with custom implementations of the com.coremedia.cap.feeder.index.async.AsyncIndexer interface. The Feeder does not call the index method of the AsyncIndexer interface to index another batch if the maximum number of parallel batches has been reached. The method will not be called until a callback about the persistence of one of these batches has been received.

feeder.batch.max-processed	
Type	java.lang.Integer
Default	1
Description	The maximum number of batches processed by the Indexer in parallel. This setting is not used with the default integration of Apache Solr but only with custom implementations of the com.coremedia.cap.feeder.index.async.AsyncIndexer interface. The Feeder does not call the index method of the AsyncIndexer interface to index another batch if the configured number of currently processed batches has been reached. The method will not be called until a callback about completed processing or persistence of one of these batches has been received.

feeder.batch.max-size	
Type	java.lang.Integer
Default	500
Description	The maximum number of index documents in a batch. If the maximum number is reached, the Feeder sends the batch to the search engine. A smaller batch may be sent if the maximum byte size is reached before, or if configured delays are reached.

feeder.batch.retry-send-idle-delay	
Type	java.time.Duration

Default	1m
Description	The time to wait before retrying to send index documents to the search engine after failures. This delay is used if the feeder is idle.

`feeder.batch.retry-send-max-delay`

Type	java.time.Duration
Default	10m
Description	The maximum time to wait before retrying to send index documents to the search engine after failures. This delay is used if the feeder is not idle. The setting is typically larger than retry-send-idle-delay.

`feeder.batch.send-idle-delay`

Type	java.time.Duration
Default	3s
Description	The time between adding an index document to a batch and sending that batch to the search engine, if the batch is not yet full according to the max-size and max-bytes configuration properties, and if the feeder is idle. If a change needs to be sent to the search engine, and no further changes were made within the specified time, then an index document for the change will be sent after that time to the search engine. A small delay ensures low latency for changes to become visible in the search engine, as long as the system is not too busy.

`feeder.batch.send-max-delay`

Type	java.time.Duration
Default	20s
Description	The maximum time between adding an index document to a batch and sending that batch to the search engine. This setting is typically larger than send-idle-delay to allow batches to grow and increase throughput, for example when large amounts of content are created by an import process. The configured value may still be exceeded under high load, or if there are problems connecting to the search engine.

Table 4.37. Feeder Batch Configuration Properties

Solr specific properties to define Feedable for Content Feeder

You can use the following properties to define which elements the *Content Feeder* should feed to the *Search Engine*.

feeder.indexDeleted	
Value	`true` or `false`
Default	`true`
Description	`true` if contents in the trash should be indexed. If you do not need to find contents in the trash and want to keep your index smaller, you can change this to `false`.

feeder.indexReferrers	
Value	`true` or `false`
Default	`false`
Description	`true` to reindex a content after its referrers have changed.

feeder.indexNameInTextBody	
Value	`true` or `false`
Default	`true`
Description	Configures whether the content name should be indexed in index field textbody. It can make sense to disable this if lots of content names contain unique identifiers (from third-party systems, for example) to avoid problems with too many unique terms in field textbody.

feeder.indexGroups	
Value	`true` or `false`
Default	`true`
Description	`true` to index the groups with potential read rights for the content in the index field `groups`. This set of groups is then used to narrow a user's search to the contents where he might have read rights to. This is an optimization to get smaller search results

for some queries and content structures and to get more accurate search suggestion counts. The client has to check for read rights anyway.

If set to `false`, then you must also configure *Studio* and *Content Server* to not add a query condition for the indexed groups. To this end, set the *Studio* property `stu dio.rest.searchService.useGroupsFilterQuery` and the *Content Server* property `solr.useGroupsFilterQuery` to `false`.

feeder.updateGroups.immediately	
Value	`true` or `false`
Default	`false`
Description	If `feeder.indexGroups` is `true`, configures whether the field `groups` is updated immediately after a change of a folder's right rule. It is recommended to keep this set to `false` and let the *Content Feeder* update the index field `groups` in the background with lower priority than updates for editorial changes. It is quite expensive to set this to `true` because all contents below the folder will be reindexed.

Table 4.38. Properties to feed additional items

Properties to define content types for feeding

You can restrict the indexed contents by their type using the `includes` and `ex cludes` properties.

feeder.content.type.includes	
Value	content type name
Default	Content_
Description	The name of the abstract or concrete content type whose contents should be indexed. Regular expressions are not allowed.

feeder.content.type.excludes	
Value	content type name
Default	Preferences, EditorPreferences, Dictionary, Query

Description	The name of the abstract or concrete content type whose contents should **not** be indexed. Regular expressions are not allowed.

Table 4.39. Properties to specify content types.

Properties to define property types for feeding

The default configuration feeds all properties for all specified content types. For configuration of indexed properties by their name, see the section for XML configuration below.

Property types to feed

You can only select a content property from a content type if its property type is specified with the following rules.

`feeder.content.propertyType.string`

Value	`true` or `false`
Default	`true`
Description	Set this property to `false` in order to exclude `String` properties from indexing.

`feeder.content.propertyType.integer`

Value	`true` or `false`
Default	`false`
Description	Set this property to `true` in order to include `Integer` properties when indexing.

`feeder.content.propertyType.date`

Value	`true` or `false`
Default	`false`
Description	Set this property to `true` in order to include `Date` properties when indexing.

`feeder.content.propertyType.linkList`

Value	`true` or `false`

Default	`false`
Description	Set this property to `true` in order to include `LinkList` properties when indexing.

`feeder.content.propertyType.struct`	
Value	`true` or `false`
Default	`false`
Description	Set this property to `true` in order to include `Struct` properties when indexing.

`feeder.content.propertyType.xmlGrammars`	
Value	List of included grammar names separated by comma
Default	`coremedia-richtext-1.0`
Description	You can define which XML properties should be indexed by specifying their grammar.

Example

```
feeder.content.propertyType.xmlGrammars=coremedia-
richtext-1.0
```

`feeder.content.propertyType.blobMimeType.includes`	
Value	List of included MIME types separated by comma
Default	`text/*,application/pdf,application/msword,application/vnd.openxmlformats-officedocument.wordprocessingml.document`
Description	You can define which blob properties are indexed, depending on the MIME type.

Example

```
feeder.content.propertyType.blobMimeType.includes=text/*
```

All blobs of MIME type `text/*` are indexed.

`feeder.content.propertyType.blobMimeType.excludes`	
Value	List of excluded MIME types separated by comma

Default	(empty)

Description	Exclude some blobs from indexing depending on the MIME type. If you've included a primary MIME type such as `text/*` or even the catch all type `*/*`, you can exclude some concrete types with this property.
	
	Example
	
	`feeder.content.propertyType.blobMimeType.ex`
	`cludes=text/plain`
	
	Blobs of MIME type `text/plain` will not be indexed.

`feeder.content.propertyType.blobMaxSize`

Value	size in bytes
Default	5242880 (5 MB)
Description	Configure the maximum size of indexed blob properties. Larger values will be skipped. This configuration can be overridden in a Spring XML configuration file where you can configure the maximum size per MIME type by customizing the bean `feederCon` `tentBlobMaxSizePerMimeType`. See XML configuration for an example.

Table 4.40. Include property types

Properties to configure Apache Tika

You can customize text extraction with Apache Tika using the following properties:

`feeder.tika.append-metadata`

Type	java.lang.String
Default	
Description	Comma-separated list of metadata identifiers returned by Apache Tika to append to the extracted body text.

`feeder.tika.config`

Type	org.springframework.core.io.Resource

Default	

Description	The location of a custom Tika Config XML, for example to customize the default Tika parsers. See Apache Tika documentation for details on configuring Tika. The value of this property must be a Spring Resource location (e.g. file:/path/to/local/file) or empty for defaults.

`feeder.tika.copy-metadata`

Type	java.lang.String

Default	

Description	Comma-separated list of metadata identifiers returned by Apache Tika and names of Feedable elements to copy the metadata to. Entries in the comma separated list have the following format: "metadata identifier"="element name". With Apache Solr, target index fields must be defined as multiValued="true" to avoid indexing errors if there are multiple metadata values with the same identifier.

`feeder.tika.timeout`

Type	java.time.Duration

Default	2m

Description	The maximum time after which text extraction from binary data with Apache Tika fails. If extraction fails, the binary data will be skipped for the index document. Lower values will avoid that the Feeder is blocked for a long time in text extraction.

`feeder.tika.warn-time-threshold`

Type	java.time.Duration

Default	15s

Description	The time after which a warning is logged when text extraction from binary data with Apache Tika takes some time.

`feeder.tika.zip-bomb-prevention.enabled`

Type	java.lang.Boolean

Default	true
Description	Sets whether Apache Tika's "Zip bomb" prevention is enabled. When a "Zip bomb" is detected, no text will be extracted from the Blob, but a warning will be logged. Note that "Zip bombs" are not restricted to ZIP files but also apply to PDFs or other formats. Disabled "Zip bomb" prevention bears the risk of OutOfMemoryError-s. Note that false positives are possible.

`feeder.tika.zip-bomb-prevention.maximum-compression-ratio`

Type	java.lang.Long
Default	-1
Description	Sets the ratio between output characters and input bytes for the Apache Tika "Zip bomb" prevention. If this ratio is exceeded (after the output threshold has been reached) then no text will be extracted and a warning will be logged. Set to -1 to use the default of Apache Tika.

`feeder.tika.zip-bomb-prevention.maximum-depth`

Type	java.lang.Integer
Default	-1
Description	Sets the maximum XML element nesting level for the Apache Tika "Zip bomb" prevention. If this depth level is exceeded then no text will be extracted, and a warning will be logged. Set to -1 to use the default of Apache Tika.

`feeder.tika.zip-bomb-prevention.maximum-package-entry-depth`

Type	java.lang.Integer
Default	-1
Description	Sets the maximum package entry nesting level for the Apache Tika "Zip bomb" prevention. If this depth level is exceeded then no text will be extracted, and a warning will be logged. Set to -1 to use the default of Apache Tika.

Table 4.41. Feeder Tika Configuration Properties

Feeder Core Properties

You can use the following properties to customize some internal settings of the *Content Feeder*.

feeder.core.executor-queue-capacity	
Type	java.lang.Integer
Default	100
Description	Maximum capacity of the Feeder's executor queue, which is internally used to transfer evaluated values.

feeder.core.executor-retry-delay	
Type	java.time.Duration
Default	1m
Description	The delay to wait before the Feeder retries to access the source data after failures.

Table 4.42. Feeder Core Configuration Properties

Error behavior specific Properties for Content Feeder

You can use the following properties to customize the *Content Feeder* behavior in case of errors.

feeder.retryConnectToIndexDelay.seconds	
Value	time in seconds
Default	10
Description	The time to wait between retries to connect to the search engine on startup.

Table 4.43. Error Handling Configuration Properties

Renamed Content Feeder Properties

Deprecated Name	New Name
`feeder.executorQueueCapacity`	`feeder.core.executor-queue-capacity`
`feeder.executorRetryDelay`	`feeder.core.executor-retry-delay`
`feeder.maxBatchByteSize`	`feeder.batch.max-bytes`
`feeder.maxBatchBytes`	`feeder.batch.max-bytes`
`feeder.maxBatchSize`	`feeder.batch.max-size`
`feeder.maxOpenBatches`	`feeder.batch.max-open`
`feeder.maxProcessedBatches`	`feeder.batch.max-processed`
`feeder.retrySendIdleDelay`	`feeder.batch.retry-send-idle-delay`
`feeder.retrySendMaxDelay`	`feeder.batch.retry-send-max-delay`
`feeder.sendIdleDelay`	`feeder.batch.send-idle-delay`
`feeder.sendMaxDelay`	`feeder.batch.send-max-delay`
`solr.partialUpdates`	`feeder.solr.partial-updates.enabled`
`solr.partialUpdatesSkipIndexCheck`	`feeder.solr.partial-updates.skip-index-check`
`feeder.tika.timeout.milliseconds`	`feeder.tika.timeout`
`feeder.tika.warn.milliseconds`	`feeder.tika.warn-time-threshold`
`solr.collection.content`	`solr.content.collection`

Deprecated Name	New Name

`solr.configSet`	`solr.cae.config-set (CAE Feeder),`
	`solr.content.config-set (Content`
	`Feeder)`

Table 4.44. Renamed Content Feeder Configuration Properties

4.9.2 CAE Feeder Properties

Properties for general configuration

`repository.user`	
Value	user name
Default	none
Description	The name of the user to connect to the *CoreMedia Content Server*.

`repository.password`	
Value	password
Default	none
Description	The password of the user to connect to the *CoreMedia Content Server*.

`repository.domain`	
Value	domain
Default	none
Description	The domain of the user to connect to the *CoreMedia Content Server*. Empty String for a built-in user.

`repository.url`	

Value	URL
Default	none
Description	The URL to the IOR of the *CoreMedia Content Server*.

`jdbc.driver`	
Value	driver class
Default	none
Description	The class of the database driver. For example: `oracle.jdbc.driver.Oracle Driver`

`jdbc.url`	
Value	URL
Default	none
Description	The URL to connect to the database.

`jdbc.user`	
Value	user name
Default	none
Description	The name of the user to connect to the database.

`jdbc.login-user-name`	
Value	the user name for the database login
Default	value of jdbc.user
Description	The user name for a database login. If not set, the value of "jdbc.user" will be used to log in to the database. In some cases the login username differs from the actual user, e.g. with PostgreSQL on Azure a postfix on the user name is necessary to log in. Set this property additionally to jdbc.user. (e.g. jdbc.login-user-name=username@domain jdbc.user=username).

jdbc.password	
Value	password
Default	none
Description	The password of the user to connect to the database.

feeder.contentSelector.basePath	
Value	String
Default	/Sites
Description	A comma-separated list of base folders for which content beans are indexed.

feeder.contentSelector.contentTypes	
Value	String
Default	Document_
Description	A comma-separated list of content types for which content beans are indexed.

feeder.contentSelector.includeSubTypes	
Value	Boolean
Default	true
Description	Specifies whether the sub types of the content types configured with property `feeder.contentSelector.contentTypes` are selected as well.

feeder.core.executor-queue-capacity	
Value	int
Default	2000
Description	Capacity of the *CAE Feeder*'s executor queue, which is internally used to transfer evaluated values

feeder.core.executor-retry-delay

Value	milliseconds
Default	60000
Description	The delay in milliseconds to wait before the *CAE Feeder* retries to access the source data after failures to do so.

feeder.batch.max-bytes

Value	bytes
Default	20971520 [20 MB]
Description	The maximum size of a batch in bytes. The *CAE Feeder* sends a batch to the *Search Engine* if its maximum size would be exceeded when adding more entries. Note, that byte computation is a rough estimate only.

feeder.batch.max-size

Value	int
Default	500
Description	The maximum number of entries in a batch. If the maximum number is reached, the *CAE Feeder* sends the batch to the *Search Engine*.

feeder.batch.max-open

Value	int
Default	5
Description	The maximum number of batches indexed in parallel. This setting is not used with the default integration of Apache Solr but only with custom implementations of the `com.coremedia.cap.feeder.index.async.AsyncIndexer` interface. The *CAE Feeder* does not call the index method of the AsyncIndexer interface to index another batch if the maximum number of parallel batches has been reached. The method will not be called until a callback about the persistence of one of these batches has been received.

feeder.batch.max-processed	
Value	int
Default	1
Description	The maximum number of batches processed by the Indexer in parallel. This setting is not used with the default integration of Apache Solr but only with custom implementations of the `com.coremedia.cap.feeder.index.async.AsyncIndexer` interface. The *CAE Feeder* does not call the index method of the AsyncIndexer interface to index another batch if the configured number of currently processed batches has been reached. The method will not be called until a callback about completed processing or persistence of one of these batches has been received.

feeder.batch.retry-send-idle-delay	
Value	milliseconds
Default	60000
Description	The *CAE Feeder* sends a batch which only contains retried entries and is not full with regard to the `feeder.batch.max-size` and `feeder.batch.max-bytes` properties after the *CAE Feeder* was idle for the time configured in this property. A retried entry is an entry which was sent to the *Search Engine* before but could not be indexed successfully. If the batch contains entries which are not retried, the value of property `feeder.batch.send-idle-delay` is used instead.

feeder.batch.retry-send-max-delay	
Value	milliseconds
Default	600000
Description	The maximum time in milliseconds between the time the *CAE Feeder* received an error from the *Search Engine* and the time, the *CAE Feeder* tries to send the failed entry as part of a batch to the *Search Engine* again. The time is exceeded if an error occurs while contacting the *Search Engine*. If the batch contains entries which are not retried, the value of property `feeder.batch.send-max-delay` is used instead.

feeder.beanPropertyMaxBytes	
Value	number of bytes

Default	5242880 (5 MB)
Description	The maximum size in bytes for the value of a bean property or -1 for no limitation. Larger values are ignored and will not be sent to the Search Engine.

`feeder.beanMapping.mimeType.includes`	
Value	comma-separated list of included MIME types
Default	*/*
Description	List of included MIME types for blob properties configured for indexing at the BeanMappingFeedablePopulator. For details, see the API documentation of method `setMimeTypeIncludes` of `com.coremedia.cap.feeder.bean.BeanMappingFeedablePopulator`

Example

`feeder.beanMapping.mimeType.includes=text/*`

Only indexes blobs of MIME type `text/*`. |

`feeder.beanMapping.mimeType.excludes`	
Value	comma-separated list of excluded MIME types
Default	
Description	List of excluded MIME types for blob properties configured for indexing at the BeanMappingFeedablePopulator. For details, see the API documentation of method `setMimeTypeExcludes` of `com.coremedia.cap.feeder.bean.BeanMappingFeedablePopulator`

Example

`feeder.beanMapping.mimeType.excludes=text/xml`

Indexes all blobs except blobs of MIME type `text/xml`. |

`feeder.batch.send-idle-delay`	
Value	milliseconds
Default	10000

Description	The *CAE Feeder* sends a batch which is not full with regard to the `feed er.batch.max-size` and `feeder.batch.max-bytes` properties after the *CAE Feeder* was idle for the configured time in milliseconds.

`feeder.batch.send-max-delay`	
Value	milliseconds
Default	120000
Description	The maximum time in milliseconds after which the *CAE Feeder* sends a batch which is not full with regard to the `feeder.batch.max-size` and `feed er.batch.max-bytes` properties. The time may be exceeded if an error occurs while contacting the *Search Engine* or if the *CAE Feeder* is under high load.

`proactiveengine.log.progress.interval.seconds`	
Value	seconds
Default	600
Description	Set the time interval to log some statistics about the progress, including the number of keys that are currently invalid and still need to be computed.

`proactiveengine.senders.evaluators`	
Value	number of threads
Default	50
Description	Number of evaluator threads in the *CAE Feeder*. The number of threads influences performance not only because evaluations can execute concurrently but also because higher values increase the probability that the *CAE Feeder* writes the state of multiple evaluations to the database in one database transaction.

`proactiveengine.senders.delay`	
Value	milliseconds
Default	0

Description	Minimum delay in milliseconds between notifications of the Feeder by the internal *Proactive Engine* sub component. Higher values lead to reduced throughput.

`proactiveengine.senders.idledelay`	
Value	milliseconds
Default	10000
Description	Delay in milliseconds between notifications of the Feeder by the internal *Proactive Engine* sub component if the application is idle. Smaller values can be configured to reduce the latency of the *CAE Feeder* but may lead to increased load on the database.

`dependencyStore.maxTransactionWeight`	
Value	maximum number of changed keys per database transaction
Default	2500
Description	The maximum weight of a database transaction to change stored dependencies. The weight is interpreted as the number of changed keys, that is, a transaction with one deleted key has weight 1. Multiple transactions will be used to process an event that causes the invalidation of more keys.

Table 4.45. Configuration of general properties independent from the type of the search engine

Properties to configure Apache Tika

You can customize text extraction with Apache Tika using the following properties:

`feeder.tika.append-metadata`	
Type	java.lang.String
Default	
Description	Comma-separated list of metadata identifiers returned by Apache Tika to append to the extracted body text.

`feeder.tika.config`	

Type	org.springframework.core.io.Resource
Default	
Description	The location of a custom Tika Config XML, for example to customize the default Tika parsers. See Apache Tika documentation for details on configuring Tika. The value of this property must be a Spring Resource location (e.g. file:/path/to/local/file) or empty for defaults.

`feeder.tika.copy-metadata`

Type	java.lang.String
Default	
Description	Comma-separated list of metadata identifiers returned by Apache Tika and names of Feedable elements to copy the metadata to. Entries in the comma separated list have the following format: "metadata identifier"="element name". With Apache Solr, target index fields must be defined as multiValued="true" to avoid indexing errors if there are multiple metadata values with the same identifier.

`feeder.tika.timeout`

Type	java.time.Duration
Default	2m
Description	The maximum time after which text extraction from binary data with Apache Tika fails. If extraction fails, the binary data will be skipped for the index document. Lower values will avoid that the Feeder is blocked for a long time in text extraction.

`feeder.tika.warn-time-threshold`

Type	java.time.Duration
Default	15s
Description	The time after which a warning is logged when text extraction from binary data with Apache Tika takes some time.

`feeder.tika.zip-bomb-prevention.enabled`

Type	java.lang.Boolean
Default	true
Description	Sets whether Apache Tika's "Zip bomb" prevention is enabled. When a "Zip bomb" is detected, no text will be extracted from the Blob, but a warning will be logged. Note that "Zip bombs" are not restricted to ZIP files but also apply to PDFs or other formats. Disabled "Zip bomb" prevention bears the risk of OutOfMemoryError-s. Note that false positives are possible.

`feeder.tika.zip-bomb-prevention.maximum-compression-ratio`

Type	java.lang.Long
Default	-1
Description	Sets the ratio between output characters and input bytes for the Apache Tika "Zip bomb" prevention. If this ratio is exceeded (after the output threshold has been reached) then no text will be extracted and a warning will be logged. Set to -1 to use the default of Apache Tika.

`feeder.tika.zip-bomb-prevention.maximum-depth`

Type	java.lang.Integer
Default	-1
Description	Sets the maximum XML element nesting level for the Apache Tika "Zip bomb" prevention. If this depth level is exceeded then no text will be extracted, and a warning will be logged. Set to -1 to use the default of Apache Tika.

`feeder.tika.zip-bomb-prevention.maximum-package-entry-depth`

Type	java.lang.Integer
Default	-1
Description	Sets the maximum package entry nesting level for the Apache Tika "Zip bomb" prevention. If this depth level is exceeded then no text will be extracted, and a warning will be logged. Set to -1 to use the default of Apache Tika.

Table 4.46. Feeder Tika Configuration Properties

Properties for Solr configuration

The following properties are only used for a *CoreMedia Search Engine* based on Apache Solr:

feeder.solr.send-retry-delay	
Type	java.time.Duration
Default	30s
Description	The delay to wait before the Feeder retries to send data after failures from Solr.

solr.cae.collection	
Type	java.lang.String
Default	
Description	The name of the Solr collection for web site search. This property does not have a default. It's typically set to 'preview' or 'live'.

solr.cae.config-set	
Type	java.lang.String
Default	cae
Description	The name of the Solr config set to use when creating the CAE collection. This property is used by the CAE Feeder.

solr.cloud	
Type	java.lang.Boolean
Default	false
Description	Whether to connect to SolrCloud. If true, connect to a SolrCloud cluster. SolrCloud connection details must be set either as ZooKeeper addresses (solr.zookeeper.addresses) or, if the former is unset or empty as HTTP URLs (solr.url). If false, connect to stand-alone Solr nodes via HTTP URLs (solr.url).

solr.connection-timeout	
Type	java.lang.Integer
Default	0
Description	Connection timeout in milliseconds, or 0 for no timeout, or a negative value to use SolrClient default.

solr.index-data-directory	
Type	java.lang.String
Default	data
Description	Value for the "dataDir" parameter of the Solr CoreAdmin API / Collection API request to create a Solr index.

solr.password	
Type	java.lang.String
Default	
Description	Password for HTTP basic authentication, used if a non-empty solr.username has been specified. The value may have been encrypted with the tool "cm encryptpasswordproperty".

solr.socket-timeout	
Type	java.lang.Integer
Default	600000
Description	Socket timeout in milliseconds, or 0 for no timeout, or a negative value to use SolrClient default.

solr.url	
Type	java.util.List<java.lang.String>
Default	http://localhost:40080/solr

Description	The list of Solr URLs to connect to. These URLs are ignored if connecting to SolrCloud (solr.cloud=true) and non-empty ZooKeeper addresses (solr.zookeeper.addresses) have been set. For a Feeder application that is not connected to a SolrCloud cluster, a single URL to the Solr master must be configured.

`solr.use-xml-response-writer`

Type	java.lang.Boolean
Default	false
Description	Whether SolrJ should use XML response format instead of Javabin format.

`solr.username`

Type	java.lang.String
Default	
Description	Username for HTTP basic authentication, or empty string for no authentication.

`solr.zookeeper.addresses`

Type	java.util.List<java.lang.String>
Default	
Description	ZooKeeper addresses for connecting to SolrCloud. Only used if solr.cloud=true.

`solr.zookeeper.chroot`

Type	java.lang.String
Default	
Description	Optional ZooKeeper chroot path for Solr. ZooKeeper chroot support makes it possible to isolate the SolrCloud tree in a ZooKeeper instance that is Only used if solr.cloud=true and solr.zookeeper.addresses is set to non-empty value.

`solr.zookeeper.client-timeout`

Type	java.lang.Integer

Default	10000
Description	Client-timeout for ZooKeeper in milliseconds, or a negative value to use SolrClient default. Only used if solr.cloud=true and solr.zookeeper.addresses is set to non-empty value.

solr.zookeeper.connect-timeout	
Type	java.lang.Integer
Default	10000
Description	Connect-timeout for ZooKeeper in milliseconds, or a negative value to use SolrClient default. Only used if solr.cloud=true and solr.zookeeper.addresses is set to non-empty value.

Table 4.47. CAE Feeder Solr Configuration Properties

Renamed CAE Feeder Properties

Deprecated Name	New Name
feeder.executorQueueCapacity	feeder.core.executor-queue-capacity
feeder.executorRetryDelay	feeder.core.executor-retry-delay
feeder.maxBatchByteSize	feeder.batch.max-bytes
feeder.maxBatchBytes	feeder.batch.max-bytes
feeder.maxBatchSize	feeder.batch.max-size
feeder.maxOpenBatches	feeder.batch.max-open
feeder.maxProcessedBatches	feeder.batch.max-processed
feeder.retrySendIdleDelay	feeder.batch.retry-send-idle-delay
feeder.retrySendMaxDelay	feeder.batch.retry-send-max-delay
feeder.sendIdleDelay	feeder.batch.send-idle-delay

Deprecated Name	New Name
`feeder.sendMaxDelay`	`feeder.batch.send-max-delay`
`feeder.tika.timeout.milliseconds`	`feeder.tika.timeout`
`feeder.tika.warn.milliseconds`	`feeder.tika.warn-time-threshold`
`solr.collection.cae`	`solr.cae.collection`

`solr.configSet`	`solr.cae.config-set (CAE Feeder),`
	`solr.content.config-set (Content`
	`Feeder)`

Table 4.48. Renamed Content Feeder Configuration Properties

4.10 UAPI Client Properties

4.10.1 Unified API Spring Boot Client Properties

repository.blob-cache-path	
Type	java.lang.String
Default	
Description	The directory in which cached blobs are stored. Make sure that the file system for this directory is large enough. Note that forced shutdowns of component's web application may result in leftover files in this directory, which should be cleared while the components are down. The configured directory is shared among components, because the actual cache content is placed in dynamically allocated subdirectories.

repository.blob-cache-size	
Type	java.lang.Long
Default	-1
Description	The maximum allowed size that the *transformed image blob cache* can occupy on the disk. This is a separate cache where results of blob transformations are stored persistently. Note that the file system overhead for storing the files does not count towards this value. So the physical space that has to be reserved on the disk for the cache has to be slightly higher than value of this configuration property.
	If several concurrent threads write large blobs at the same time, the deletion of the folder with the old unused files can be postponed for later, thus this is the second reason why the maximum allowed cache size can grow slightly higher than this configuration property. The size of such deviation depend on the blobs size as well as the amount of parallel threads.

repository.blob-streaming-size-threshold	
Type	java.lang.Integer

Default	-1
Description	The minimum size of streamed blobs in bytes. blobs less than or equal to this size will be downloaded completely to disk before the first byte can be read. Larger blobs will be downloaded in the background.

repository.blob-streaming-threads

Type	java.lang.Integer
Default	-1
Description	The number of threads reserved for streaming blob.

repository.caplist.mongo-db-client-uri

Type	java.lang.String
Default	mongodb://localhost:27017
Description	The Cap List MongoDB connection string URI is used to configure your MongoDB connection. Property 'readpreference' must be 'primary'. See also mongoDb documentation.

repository.caplist.mongo-db-prefix

Type	java.lang.String
Default	elastic
Description	Prefix for Cap List MongoDB database names.

repository.connect-retry-delay-seconds

Type	java.lang.Long
Default	10
Description	The delay between UAPI connect retries.

repository.domain

Type	java.lang.String

Default	
Description	The domain of the *Content Server*

repository.force-immediate-login

Type	java.lang.Boolean
Default	false
Description	Configuration option that forces the UAPI connection to connect immediately instead of waiting for the *Content Server* to come up. This is useful for command line tools like the workflow converter that should provide quick feedback.

repository.heap-cache-size

Type	java.lang.Long
Default	-1
Description	The total number of bytes used by the main memory cache. For 32 bit JVMs this value is exact, for 64 bit JVMs, the actual memory consumption may be up to 2 times the configured value.

repository.max-cached-blob-size

Type	java.lang.Integer
Default	-1
Description	The maximum size of blobs that are cached on the local disk. Larger blobs are downloaded from the Content Server on every request.

repository.password

Type	java.lang.String
Default	
Description	The password of the user.

repository.url

Type	java.lang.String
Default	
Description	The URL of the *Content Server*.
	This property determines where to get the IOR of the *Content Server* (format: http://<server>:<port>/ior).
	• <server> must be the name of the *Content Server* host.
	• <port> must be the server's web server HTTP port.

repository.user	
Type	java.lang.String
Default	
Description	Define the user which connects to the *Content Server*.

Table 4.49. UAPI Spring Boot Client Properties

4.10.2 Renamed Properties

Deprecated Name	New Name
repository.con nect.retry.delay.seconds	repository.connect-retry-delay-seconds

Table 4.50. Renamed UAPI Spring Boot Client Properties

4.11 Cache Properties

cache.capacities	
Type	java.util.Map<java.lang.String,java.lang.Long>
Default	
Description	Number of cache entries per cache class until cache eviction takes place. The keys must match the cache classes as defined by the cache keys. Please refer to javadoc of com.coremedia.cache.CacheKey.
cache.timeout-seconds	
Type	java.util.Map<java.lang.String,java.lang.Long>
Default	
Description	TTL in seconds until certain cache entries are invalidated.

Table 4.51. Cache Properties

4.12 Image Transformation Properties

imagetransformation.default-jpeg-quality	
Type	java.lang.Float
Default	0.8
Description	The default jpeg quality used for cropping.

imagetransformation.dynamic-variants	
Type	java.lang.Boolean
Default	true
Description	If true, resolve transformations from content, otherwise only programmatically configured transformations are used.

imagetransformation.remove-metadata	
Type	java.lang.Boolean
Default	true
Description	The fallback value for removeMetadata. Will be effective for transformations where neither the given Breakpoint nor the given Transformation have a removeMetadata value. If true, the "rm" image operation will be applied to remove the image metadata

imagetransformation.sharpen	
Type	java.lang.Boolean
Default	true

| Description | The fallback value for sharpen. Will be effective for transformations where neither the given Breakpoint nor the given Transformation have a sharpen value. If true, the "usm" image operation will be applied and the image will be sharpened. |

Table 4.52. Image Transformation Properties

Index

L
logfile names, 25