10. APRIL 2024

PRINT MANAGER VER. 4.0

INSTALLATION, CONFIGURATION, AND OPERATION

ANDREAS HARTMANN
HMEDIA
mail@hmedia.de

PREAMBLE

The information in this document and any other document belonging to the PrintManager software is subject to change without notice. It does not contain any liabilities incurred by Hmedia. Hmedia takes no charge for any errors that may appear in these documents. Errors and omissions are expected. Any documentation accompanying the PrintManager software is licensed for internal, non-commercial reference purposes only.

The system described in this documentation is subject to technical developments. There may be differences between the documentation and the actual implementation of the system.

Any part of the documentation delivered along with the PrintManager software or accessible on the Hmedia website may not be modified.

No part of the PrintManager software or this document may be passed on to third parties in any form (print, photocopy, microfilm, or in any other process) without the prior written permission of Hmedia. The user is permitted to reproduce this document for his purposes.

Hmedia reserves all rights not expressly granted in this license.

The software described in this document is subject to the following license:

1. The software is left for free use on the customer's systems.
2. Any modification or extension of the PrintManager software is not permitted.
3. No part of the PrintManager software or any PrintManager software artifacts may be in any form given to third parties.
4. You may not reverse engineer, decompile, or disassemble this software.

This license applies to all updates and supplements that Hmedia will provide.

Hmedia may terminate your license if you fail to comply with the terms and conditions of this license.

In this case, you have to uninstall all your instances of the PrintManager software and destroy all copies of the software and any of its artifacts.

THIS SOFTWARE AND THE ACCOMPANYING DOCUMENTATION ARE PROVIDED "AS IS" AND WITHOUT WARRANTY OF ANY KIND.

FURTHERMORE, HMEDIA DOES NOT WARRANT, GUARANTEE OR ASSUME ANY LIABILITY WITH RESPECT TO THE USE AND RESULTS OF THE USE OF THE SOFTWARE AND DOCUMENTATION IN TERMS OF CORRECTNESS, ACCURACY, RELIABILITY, TIMELINESS OR OTHERWISE. THE ENTIRE RISK AS TO THE RESULTS AND PERFORMANCE OF THE SOFTWARE IS WITH THE USER. IF THE SOFTWARE OR USER DOCUMENTATION IS DEFECTIVE, THE USER ALONE, AND NOT HMEDIA, ASSUMES THE ENTIRE COST OF ALL NECESSARY REPAIRS AND CORRECTIONS.

NEITHER HMEDIA NOR ANY OTHER PARTY INVOLVED IN THE DEVELOPMENT, PRODUCTION, OR DELIVERY OF THIS PRODUCT SHALL BE LIABLE FOR ANY DAMAGES WHATSOEVER, WHETHER DIRECT,

INDIRECT, CONSEQUENTIAL OR INCIDENTAL (INCLUDING DAMAGES FOR LOSS OF PROFITS, BUSINESS INTERRUPTION, LOSS OF DATA, LOSS OF BUSINESS INFORMATION AND OTHER PECUNIARY LOSS), ARISING OUT OF THE USE OF OR INABILITY TO USE THIS PRODUCT, EVEN IF HMEDIA HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

This product includes software developed by The Apache Software Foundation (http://www.apache.org/).

Hmedia
Andreas Hartmann media solutions
Vorwerkstrasse 1
01936 Koenigsbrueck – Germany
mail@hmedia.de

The designations and brand names of the respective companies used in this document are generally subject to trademark, brand, or patent protection.

TABLE OF CONTENTS

PREFACE

GENERAL

The Hmedia PrintManager accomplishes the Avid Newsroom Management System with versatile PDF rendering and printing functions. It offers PDF renders of particular iNEWS queues or selected, individual stories. The created PDF documents can be printed, previewed in Adobe Acrobat, or sent out via email straight from within the PrintClient software.

ABOUT THIS DOCUMENT

The chapters of this document cover following objectives:

- System Overview: Gives a general description of the components of the PrintManager and how these components act together.
- The PrintServer – Native Deployment: Explains the installation and configuration of the PrintServer component straight in a Java Servlet Provider.
- The PrintServer – Containerized Application: Explains the various options to deploy the PrintServer as a containerized application.
- Der PrintClient: Installation, configuration and operation of the PrintClient.
- Printstyles: Explains how Printstyles are used to convert the content of an iNEWS queue into a printable document and how such PrintStyles are structured..

SYMBOLS AND CONVENTIONS

Terms

The Avid iNEWS system is now rebranded to MediaCentral | Newsroom Management. The term iNEWS is still common and get used in many documents. This PrintManager Documentation utilizes both terms **iNEWS** and **Newsroom Management** synonymously.

The data structure in iNEWS has three fundamental elements: Directory – **Queue** – Story. Directories don't matter at all for the PrintManager. A queue is a collection of individual stories and the most common element to print. However, the PrintManager can also handle single or multiple selected stories.

Printstyle is the rule set to convert the source data into a PDF layout. Technically, the printstyles are XSLT stylesheet documents.

Bold

Bold text emphasizes important terms, letters, signs, or numbers.

Tip

INFORMATION FORMATTED LIKE THIS HELP THE READER NOT TO MISS ADDITIONAL FACTS.

Warning

A WARNING LOOKS LIKE THIS HELPS YOU TO AVOID COMMON PROBLEMS AND KNOWN ISSUES.

Links

Hypertext links look like: https://hmedia.de

Console Commands and Console Output

Another font is used to indicate command line phrases and related outputs, like here for the command **`java -version`**:

```
c:\>java -version
```

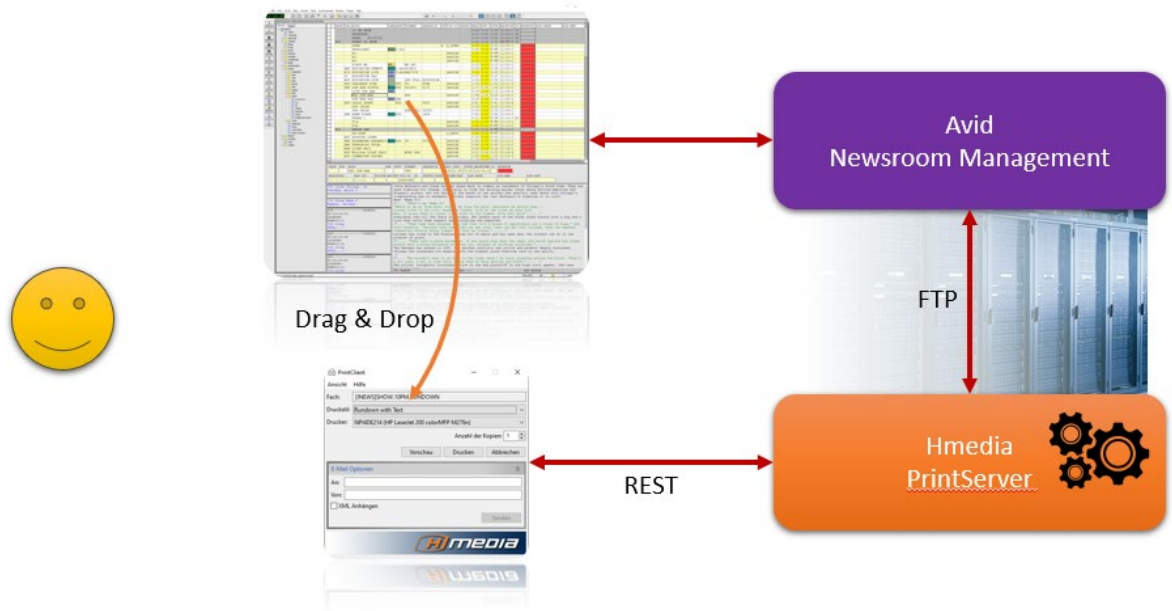
```
openjdk version "16" 2021-03-16
OpenJDK Runtime Environment (build 16+36-2231)
OpenJDK 64-Bit Server VM (build 16+36-2231, mixed mode, sharing)
```

SYSTEM OVERVIEW

FEATURES AND COMPONENTS

Originally built as an iNEWS extension to print any kind of data in any format, one can now view the results in Adobe Acrobat and process it from there, or send it as email straight out of the PrintClient.

The PrintManager consists of two componentes – the PrintServer and the PrintClient. The PrintServer executes on a central host and process all print requests. The PrintClient gets installed together with the Avid iNEWS Workstation software. It maintains the communication with iNEWS, the backend, Adobe Acrobat, and the printing infrastructure.



A typical print process consists of following steps:

1. The user specifies what data from iNEWS to render (an iNEWS Queue or any number of stories) and drag&drop them into the PrintClient. Alternatively, the PrintClient offers command line parameters to hand over the objects to print.
2. In a next step, the user chooses the print style.
3. If anything is complete, the PrintClient sends a request to the PrintServer.
4. The PrintServer reads the requested content from the iNEWS system and converts them, using the print style, into a PDF document. That PDF document is then transferred back to the PrintClient.
5. The PrintClient shows the PDF in Adobe Acrobat, prints it on any connected printer, or sends it out via email.

RESILIENCE

The PrintServer follows a high-available and scalable concept.

PrintServer – iNEWS

The communication with the Newsroom Management system is implemented in a fail-tolerant fashion and supports the typical iNEWS architecture with two mirrored servers. If a server in a dual-iNEWS system crashes or is not responding for any reason, then the PrintServer goes ahead and try to establish a connection to the other iNEWS server.

PrintClient – PrintServer

The PrintClient can talk to multiple PrintServers. If the PrintClient gets a request for a particular iNEWS system, it tries to establish a connection to all configured PrintServers – one after another, until it gets a successful response from one of the servers.

In total, one PrintServer can work for multiple iNEWS systems, and multiple PrintServers can connect into the same iNEWS system. This architecture provides scalability and redundancy.

NETWORK

The PrintClient communicates with the REST API of the PrintServer over HTTP or HTTPS. The default port of the PrintServer is 8080, but could be customized. To achieve an encrypted and secured connection, the PrintServer can be paired with a reverse proxy software like Nginx. Then, the communication between PrintClient and PrintServer is routed through that reverse proxy. This component is also responsible for managing the TLS certificates. In such a scenario, the REST API could be also protected with Http Basic Authentication.

The PrintServer retrieves data from the iNEWS servers on their FTP(S) interface.

THE PRINTSERVER – NATIVE DEPLOYMENT

GENERAL

This section covers the PrintServer when installed directly on a host system inside of a Java servlet container.

Hmedia develops and qualifies the software against Apache Tomcat. While other Java application servers could also host the Hmedia PrintServer, this documentation covers installation and operation in Apache Tomcat only.

The installation and features of Apache Tomcat is not part of this document. The text might contain references to public internet pages for the related topics.

PREREQUISITES

Operating System

Apache Tomcat and Hmedia PrintServer are Java programs. Hence, you are free of choice for the underlying operating system.

At Hmedia, we test the PrintServer 4.0 with Apache Tomcat 9 on **Windows Server 2019** and **Ubuntu 22.04 LTS**.

Java

The PrintServer needs a **Java Runtime Environment (JRE) 11** or higher. The current version (March 2024) is Java 21. The Java installation must be finished before installing the Tomcat.

The owner of Java, Oracle, has changed its license model for the Java Runtime in 2019. Since then, the Oracle JRE is only available with a commercial license.

Hmedia suggests to use the Community-version named OpenJDK. The OpenJDK can be retrieved from https://jdk.java.net. The OpenJDK doesn't come with an install program. You need to follow the installation procedure explained in various internet tutorials, depending on your server operating system.

If you need help installing Java on your PrintServer computer, please contact us at mail@hmedia.de.

ENSURE, THAT THE BIN FOLDER OF JAVA IS REGISTERED IN THE PATH ENVIRONMENT VARIABLE.

The command j**ava** must be executable on the shell. Verify the active version of your Java installation with the command **`java -version`**. The result should look like:

```
openjdk version "16" 2021-03-16
OpenJDK Runtime Environment (build 16+36-2231)
OpenJDK 64-Bit Server VM (build 16+36-2231, mixed mode, sharing)
```

Tomcat

As already mentioned, the PrintServer executes in a Java servlet container. Hmedia develops and tests the software with Apache Tomcat. We utilize the version Tomcat 7 and 9.

TOMCAT 10 CAN'T BE USED! IN THAT VERSION, MANY COMPONENTS HAVE DIFFERENT NAMESPACES (MIGRATED FROM JAVA TO JAKARTA). HENCE IT IS NOT BACKWARD COMPATIBLE.

With Linux, Tomcat comes usually with distribution-specific packages. That makes installation rather simple and follows the common procedures of your Linux package manager (like yum or apt). If you want to install a newer version than the one provided by your distribution, you can install precompiled binaries. You can find them on the download section of the Apache Tomcat website https://tomcat.apache.org. If you need help with such an installation, please contact us at mail@hmedia.de.

For Microsoft Windows, you'll find the binary packages available in the download section of the Apache Tomcat web site https://tomcat.apache.org. If you need help with such an installation, please contact us at mail@hmedia.de.

Log4J Vulnerability

The version 4.0 of the PrintManagers utilizes an updated version of the Log4J framework.

ADDITIONALLY, CHECK YOUR INSTALLATION OF JAVA AND TOMCAT FOR POTENTIALLY INCLUDED LOG4J LIBRARIES AND REPLACE THEM WITH A NEWER VERSION >= 2.17!

INSTALLATION

The following chapter describes the installation procedure of the PrintServer software. Here, we use the Linux path syntax with the delimiter **/**. If you apply the instructions to a Windows system, please adjust the path names accordingly.

Preparation of Tomcat

1. Localize the folder, where Tomcat is installed into. This folder will be referred as **<TOMCAT_HOME>**.
2. Create the folder **<TOMCAT_HOME>/shared/classes**.
3. Edit the file **<TOMCAT_HOME>/conf/catalina.properties** and replace the line
   shared.loader=
   with
   shared.loader=${catalina.base}/shared/classes.

LATER ON, WHEN YOU MODIFY FILES WITHIN THE SHARED FOLDER, YOU NEED TO RESTART TOMCAT TO APPLY THEM PROPERLY!

PrintServer Deployment

To deploy the PrintServer software in Tomcat, execute the following steps:

1. Extract the installation archive **printserver-xyz.zip**. xyz stands for the version number like 4.0-512. The extracted PrintServer archive contains the file **printserver.war**, and a sub folder **shared/classes**.
2. Copy the file **printserver.war** into the folder **<TOMCAT_HOME>/webapps**.
3. Copy the entire content of the **shared/classes** folder of the archive into the folder **<TOMCAT_HOME>/shared/classes**.

Congratulations, you installed the Hmedia PrintServer! Now, you need to configure the PrintServer as described in next section.

PrintServer Configuration

All configuration files of the PrintServer exist inside the Tomcat installation folder. Like above, that is referred to as **<TOMCAT_HOME>**. You can find default versions of the config files in the extracted PrintServer archive.

iNEWS Configuration

Set up the connection from PrintServer into one or multiple Newsroom Management systems with the file **inewsconf.xml**. That needs to be in the folder **<TOMCAT_HOME>/shared/classes/printserver**. The content of the file looks like this:

```
<?xml version="1.0" encoding="UTF-8"?>

<!--
The connectionTimeout variable is the maximum time in milliseconds this program waits for an
iNEWS Server response when connecting. If an iNEWS server does not respond in the given time,
this program will assume the server to be down.
-->

<iNEWSConfiguration>
    <system name="INEWS" maxConnections="6" loadBalance="2">
        <server host="inews-a" maxConnections="6"
                    defaultUser="avstar" defaultPassword="avstar" connectionTimeout="3000"/>
        <server host="inews-b" maxConnections="6"
                    defaultUser="avstar" defaultPassword="avstar" connectionTimeout="3000"/>
    </system>
    <system name="NRCS" maxConnections="6" loadBalance="2">
        <server host="172.21.10.31" maxConnections="6"
                    defaultUser="avid" defaultPassword="inews" connectionTimeout="3000"/>
    </system>
</iNEWSConfiguration>
```

In this example, the PrintServer works for two iNEWS systems INEWS und NRCS. INEWS is a dual-server configuration, and NRCS is a single-server deployment. Each server machine must be configured with a network name and credentials (username and password). Those credentials are used to authenticate for the FTP(S) connection (RXNET).

The PrintServer can handle any number of iNEWS systems, each with one up to four servers.

Elements of the **inewsconf.xml**:

Element name: iNEWSConfiguration

Subelements: system (any number)

Attributes: none

Meaning: The root element of the xml document.

Element name: system

Subelements: server (1 – 4)

Attributes:

- **nam**e: The name of the iNEWS system. Must match the configuration in the PrintClient (see chapter The PrintClient – Configuration). Typically it is the same name as configured in the iNEWS database.
- **maxConnections:** The max number of FTP(S) connections that the PrintServer should establish to that iNEWS system.
- **loadBalance:** If the iNEWS system has multiple servers online, the PrintServer distributes the FTP(S) connections so that the number of connections to one server is not bigger than the loadBalance value, compared to any other server in that particular system.

Meaning: Represents an iNEWS system.

Element name: server

Subelements: none

Attributes:

- **host:** The network name oft he server (IP address, host name, or FQDN)
- **maxConnections:** The max number of FTP(S) connections that the PrintServer should establish to that iNEWS system. That value can't get bigger than the total number of maxConnection for the entire system.
- **defaultUser:** The username to authenticate on the iNEWS server.
- **defaultPassword:** The password for the defaultUser.
- **connectionTimeout:** Timeout for connection attempts in milliseconds. If a connection can't be established within this time, the PrintServer assumes that this server is offline.

Meaning: Represents a Newsroom Management server.

PrintStyle Configuration

All printstyle files life in the folder **<TOMCAT_HOME>/shared/classes/printserver/stylesheets**. To enable printstyles for the users of the PrintClient, you need to register them in the configuration file **<TOMCAT_HOME>/shared/classes/printserver/iNEWSSystems.txt**. With that configuration, you can:

1. Give each printstyle files a friendly name which is then displayed in the PrintClient. That name can contain spaces and special characters.
2. Assign printstyles to individual iNEWS systems.
3. Assign printstyles to individual iNEWS queues.

In the configuration file, each line stands for one printstyle file. Each entry starts with the filename of the printstyle file. Then follows a whitespace (space or tab) as a delimiter, followed by the display name.

A line starting with a square bracket **[** is treated as an iNEWS system, like [INEWS] oder [NRCS]. The PrintClient shows all printstyles that follow such a line (before the next iNEWS system line) only when working against that iNEWS system.

A line that starts with an @, followed by a regular expression, indicates an iNEWS queue name. The PrintClient will show all following printstyles (before the next system or queue line) only if the queue to print matches the regular expression.

A line starting with **@ nomatch** can be followed by printstyles that are displayed when no above queue entry matches the queue to print.

Example 1

The following iNEWSSystems.txt sample contains three printstyles. The first file is **rundown_body.xsl** and the friendly name is **Rundown with Text**, and so on. PrintClients connecting to that PrintServer show all three styles independent of the iNEWS system and queue.

```
rundown_body.xsl Rundown with Text
rundown_cue.xsl Rundown with cues
production_cues.xsl Stories with text and cues
```

Example 2

Next shows an iNEWSSystems.txt of a PrintServer that is working with two iNEWS systems. The first printstyle (rundown_body.xsl) is available for both iNEWS systems. The second printstyle (rundown_cue.xsl) only shows up when printing from the system NRCS. Similarly, the third style (production_cues.xsl) is displayed as an available printstyle only when the source system is INEWS.

```
rundown_body.xsl Rundown with Text
[NRCS]
rundown_cue.xsl Rundown with cues
[INEWS]
production_cues.xsl Stories with text and cues
```

Example 3

The final sample configuration comes with a more complex setup:

The first iNEWS system **NRCS** has three styles available.

For the second system **NRCS26**, the PrintClient shows different printstyles, depending on the source queue.

- For a queue containing the term RUNDOWN anywhere in its name/path, the PrintClient displays Presenter, Production Cues, RAI Cues, Rundown Cue und Rundown Cue Body.
- For a queue containing the term TRAINING, the PrintClient displays Presenter, Production Cues, RAI Cues und Training Stylesheet.
- For any other queue, neither containing RUNDOWN nor TRAINING, the PrintClient displays Presenter, Production Cues, RAI Cues, RAI Title und RAI Text.

```
[NRCS]
production_cues.xsl Production_Cues
rundown_cue.xsl Rundown_Cue
rundown_cue_body.xsl Rundown_Cue_Body

[NRCS26]
presenter.xsl         Presenter
production_cues.xsl   Production Cues
rai_cues.xsl          RAI Cues

@ .*RUNDOWN.*
rundown_cue.xsl       Rundown Cue
rundown_cue_body.xsl  Rundown Cue Body

@ .*TRAINING.*
rai_training.xsl      Training stylesheet

@ nomatch
rai_title.xsl         RAI Title
rai_text.xsl          RAI Text
```

STATUS PAGE

After installation and configuration, the Tomcat should be restarted. The related command depends on your operating system and the Tomcat installation.

To check the status of the PrintServer after the start, you can utilize any web browser on any computer, that has network access to the following URL:

http://<printserver_host>:8080/printserver.

Here is **<printserver_host>** the network name of the PrintServer computer.

The output in the browser will then look like:

AFTER STARTING THE PRINTSERVER, THE STATUS PAGE IS AVAILABLE ONLY AFTER DETERMINING THE STATUS OF ALL CONFIGURED INEWS SYSTEMS. IF ONE OR MORE INEWS SYSTEMS ARE OFFLINE, THE PRINTSERVER TRIES TO CONNECT UNTIL THE CONFIGURED CONNECTION TIMEOUT HAS BEEN REACHED.

In this example, the iNEWS system NRCS is not available while the system with name INEWS is ready.

The PrintServer in the above example can't be used because it's not licensed.

LICENSING

To fully operate, the PrintServer requires a license file named **license.lic**. Hmedia delivers that file separately from the software package. You need to copy the license file into the folder **<TOMCAT_HOME>/shared/classes/printserver**.

At the next Tomcat start, the PrintServer will validate the license and updates the status:

There is no online activation required.

With validating the license on the PrintServer, the entire PrintManager system is licensed and fully operational. There is no separate license for the PrintClient. The number of connected PrintClients, iNEWS systems, or printstyles is not limited.

The license is not shrunk with the hard- or software of the PrintServer computer and can be reused for new or further installations.

However, the license is specified for the customer and must not be given to third parties or other customers.

The license is related to a particular PrintServer version or version range and can be validated only from matching software.

There are time-limited and perpetual licenses.

Operation

There is no dedicated operation for the PrintServer itself. Starts, stops, restarts are performed on the Tomcat level.

If you reconfigure the iNEWS systems (**inewsconf.xml**), restart the Tomcat.

If you modify/debug existing printstyles, you don't need to restart Tomcat. The PrintServer reads a particular printstyle file at runtime on request and doesn't cache it.

However, if you change the **iNEWSSystems.txt** you need to restart the Tomcat afterwards.

Log Files

The log files of Tomcat are located in the folder **<TOMCAT_HOME>/logs**.

You can find specific messages of the PrintServers in the file **<TOMCAT_HOME>/Hmedia.log**.

That log file will be rotated as usual.

THERE IS NO AUTOMATIC LOG CLEANING FOR TOMCAT/PRINTSERVER. IF NEEDED, THE ADMINISTRATOR MUST DELETE OLD LOG FILES MANUALLY.

HTTPS AND AUTHENTICATION

With version 4.0, the PrintClient can access the REST API via HTTPS. It also supports Basic HTTP Authentication. See chapter The PrintClient - Configuration, on page 31 for further details.

On the PrintServer side, those features are not provided by the PrintServer itself. They could be configured either in the Servlet container software (Tomcat) or, much simpler, by fronting the PrintServer REST API with a reverse proxy. A common solution for such a reverse proxy is the Apache Nginx.

If you need help installing and configuring the Nginx for TLS and authentication, please contact us at mail@hmedia.de.

THE PRINTSERVER – A CONTAINERIZED APPLICATION

OVERVIEW

With version 4.0, PrintServer is available as a containerized application. This offers a much more robust and stable deployment, easy to scale, and configurable for fail-over functionality.

There are multiple options for installing the PrintServer as a containerized application, giving you various degrees of flexibility. This way, you can best fit the PrintServer into your infrastructure.

The available deployment options are:

- As a plain Docker container in a Docker-compliant container runtime. In this case, you can get more details on https://hub.docker.com/r/hmediade/printserver, and pull the image simply with `docker pull hmediade/printserver`.

  Additionally to the PrintServer container, a reverse proxy container (like Nginx) must be configured and executed to support encrypted and authentication-protected access to the PrintServer REST API.

  Data injection (configuration and printstyles) is completely of your choice.

- Deployment in a Kubernetes cluster can be best controlled via the Helm Chart that we at Hmedia have created for you. It is available including certain instructions on https://artifacthub.io/packages/helm/hmediade/printserver. With that chart, you get a complete, scalable deployment, optionally including encryption and authentication for the PrintServer REST API. Configuration and installation is described in detail on that referred page.

- By far the simplest and most straightforward method to install the PrintServer: If you don't want to prepare the Helm Chart manually, we offer a deployment script that guides you through an interactive setup routine to enter all required data, and install, upgrade, remove the PrintServer software in a seamless way. You can download that script (printserverctl) from the PrintServer product page.

  Simply put it on the control client for your Kubernetes cluster (where you run kubectl). Further help comes with the script and this document.

This manual only focuses on the installation with our printserverctl script. If you would like to customize your environment utilizing one of the other deployment scenarios, please contact us at mail@hmedia.de for further help.

CONCEPTS AND ARCHITECTURE

This section describes the architecture of a fully featured PrintServer environment.

The core component of any containerized app is an image. Hmedia uses Docker to create the image for the PrintServer. That image contains a basic Linux layer, Java, Tomcat, and the PrintServer software itself. To fully feature that unit, you need to inject all site-specific data into the container, and probably pair it with other components (like a reverse proxy).

Data Injection

The PrintServer requires two sorts of configurations and settings, which are site-specific and can't be delivered with the software itself:

- PrintServer Configuration
- PrintStyles and additional files (like graphics)

Basically, data can be injected into a container via file mounts or via environment variables. The PrintServer container itself works completely with a file-based provision, because all required data are files.

If the PrintServer container is deployed individually (not via Kubernetes, Helm, or the printserverctl script), one can simply mount two directories in the container, one for the config files, and one for the printstyles. The procedure is explained in detail on https://hub.docker.com/r/hmediade/printserver.

If the PrintServer is deployed inside a Kubernetes cluster, the PrintServer pod typically consists of a printserver-init container and the main printserver container. Figure 1 depicts the way how the pod gets the required data.

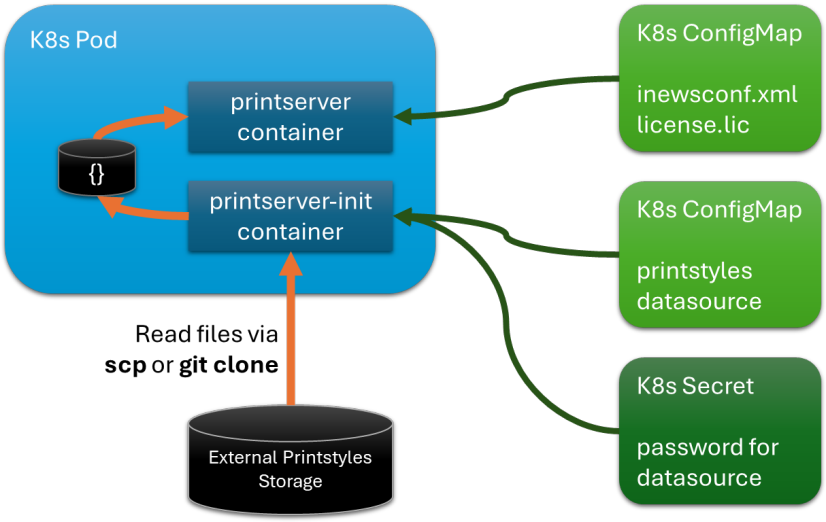


Figure 1: Data Injection into the PrintServer Container

When creating the pod, Kubernetes first executes the init container. It reads the printstyle files from an external source and then terminates. This version of the PrintServer can read the files via scp or git. The printserver-init container gets the protocol configuration and connection user injected as environment variables, and the password injected as a file. All details are described on

https://hub.docker.com/r/hmediade/printserver-init. In a Kubernetes deployment, the data are provided through the common artifacts (ConfigMaps and a Secret)

The init-container stores printstyles and subsequent files in an ephemeral storage inside the Kubernetes pod.

Once, the init container has finished, the main container starts and can serve the PrintServer REST API.

Staging the PrintServer REST API

If the PrintServer is deployed as a standalone container (not in Kubernetes), the container runtime maps the desired host port to the exposed port of the PrintServer container (8080). The communication is pure http – no encryption, no authentication.

In a fully featured Kubernetes deployment, the PrintServer pod will be reached through multiple artifacts. The order from in to out is:

1. Pod – providing the core functionality, offering the REST API on an internal pod network. Pods are mortal/volatile. They can be destroyed and recreated at any time. A new pod is a different object and will have a different IP address in the pod network than the previous one. The workload can be scaled by running multiple instances.
2. Service – A service has a fixed IP in the internal service network and provides an access point to the API through a stable address/DNS name. Though, that access is still purely Kubernetes-internal.
3. Ingress – A reverse proxy (Nginx) has to be deployed in the Kubernetes cluster as an ingress controller. An ingress object then routes the outer network to the internal service. Additionally, the ingress can enrich the communication with encryption (HTTPS/TLS) and authentication (Basic Http Authentication)

Figure 2 shows the full setup in a Kubernetes cluster.

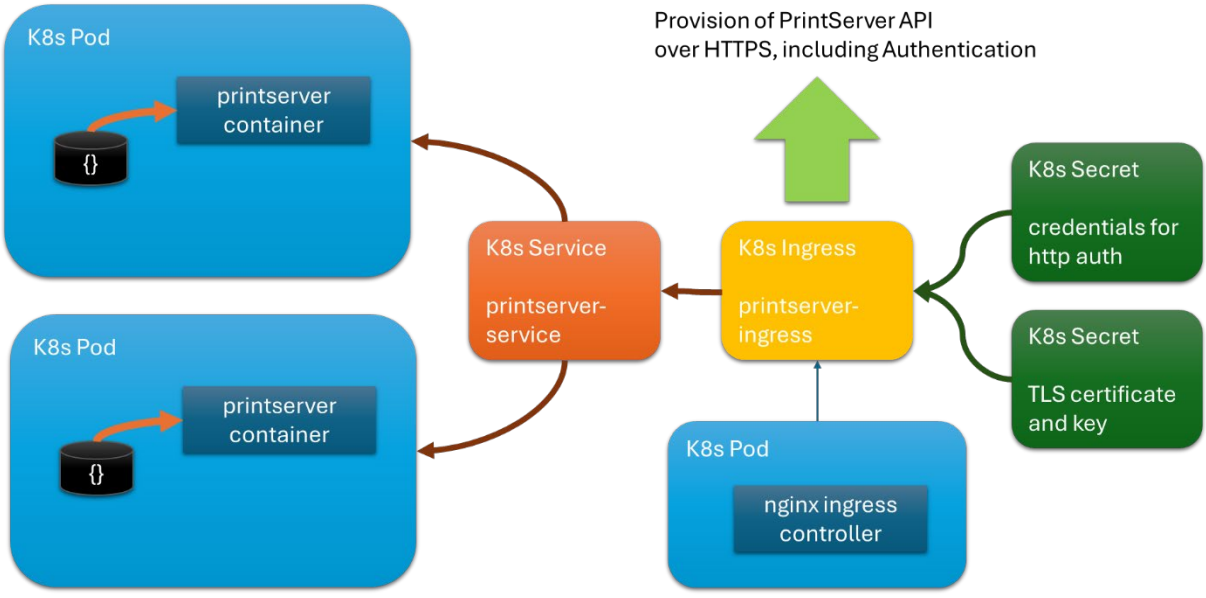


Figure 2: Kubernetes Objects for a PrintServer Deployment

In the Appendix, you can find a sample declaration of all required artifacts. You can deploy the entire PrintServer with a single command `kubectl apply -f <declaration.yaml>`, where declaration.yaml is the description of all required objects in the Kubernetes-specific declaration language. However, Hmedia also offers an interactive, dialogue-driven deployment. See chapter Deployment, on page 21 for more information.

SOFTWARE SPECIFICATIONS

PrintServer Container

The image for the printserver container is built on top of the image `tomcat:9`.

As a base layer, that image uses an Ubuntu 22.04 LTS system. The Java is of the latest version – JDK 21, built by the Eclipse Temurin project. The Tomcat itself is of version 9.0.87, at the time of the image build.

PrintServer-Init Container

The image for the printserver-init container is built on `ubuntu:latest`. That image sports a base version of Ubuntu 22.04 LTS, at the time of the image build. Additional software used in the container is:

- openssh-client
- sshpass
- git

DEPLOYMENT WITH THE HMEDIA PRINTSERVERCTL

Overview

The printserverctl script simplifies the deployment of the Hmedia PrintServer. It offers various commands for different operations:

- prepare
- clean
- review
- dryrun
- deploy
- remove
- version
- help

Eventually, it will control Helm to deploy or remove the PrintServer into the current Kubernetes.

The printserverctl script prepares the configuration and initiates jobs with the helm tool. It stores all configuration data in the folder `/etc/hmedia/printserver`. All files in this folder are only

required for the deployment; and not used by a running instance of the PrintServer. The printserverctl will create that folder if not present.

If you like, you can review and manually adjust data in the settings folder `/etc/hmedia/printserver`. This will not affect a running instance of the PrintServer until you execute `printserverctl deploy`.

Prerequisites

Required Tools

Following Linux tools and programs are used by the deployment script:

- mkdir
- sed
- awk
- less
- base64
- helm
- kubectl

The printserverctl script checks all those tools and programs at each execution.

Kubernetes must be up and running and reachable with `kubectl` command on the same machine. Kubernetes must be of version 1.28 or higher.

Helm must be installed and available. Helm version 3.0 and higher is supported.

Network Access

It is highly recommended to enable Internet access during installation. The helm command will retrieve the PrintServer Helm chart from the web, and Kubernetes will pull, at least for the first run, the container images from Docker Hub web site.

The installation of the PrintServer is also possible in a complete offline fashion. If you are interested in such a deployment scenario, please contact us on mail@hmedia.de for further information.

Configuration Data and Files

To successfully collect all required information, the interactive setup will ask for the following parameters. Please have your settings ready, and the requested files copied to the installation machine, before you run the `printserverctl prepare` command:

Parameter	Description
deployment.name	Name of the Helm object. Default is **printserver**.
deployment.namespace	Target Namespace in Kubernetes. Default is **default**.

	IF YOU HAVE OTHER SOFTWARE RUNNING IN THAT KUBERNETES AND WANT TO SEPARATE THE PRINTSERVER OBJECTS, USE A SPECIFIC NAMESPACE LIKE PRINTSERVER.
ingressNginx.install	Do you want the NGINX Ingress chart automatically installed? Answer: true or false. Default is **true**. THE PRINTSERVER DEPENDS ON THE INGRESS CONTROLLER. ONLY SET TO FALSE IF THE INGRESS CONTROLLER IS ALREADY DEPLOYED IN THIS PARTICULAR KUBERNETES CLUSTER.
ingressNginx.className	Name of the nginx ingress class. Default is **nginx**.
ingressNginx.auth.enabled	Do you want the connection to PrintServer protected with HTTP Basic Authentication? Answer true or false. Default is **true**.
ingressNginx.auth.username	Username for HTTP Basic Authentication. Default is **printclient_user**. Not used when **ingressNginx.auth.enabled** is set to **false**.
ingressNginx.auth.password	Password for HTTP Basic Authentication. Default is **printclient_passwd**. Not used when **ingressNginx.auth.enabled** is set to **false**.
ingressNginx.tls.enabled	Do you want the connection to PrintServer encrypted via TLS 1.3? Answer: true or false. Default is **true**.
ingressNginx.tls.secretName	Name of the Secret object to store the TLS certificate. Default is **printserver-tls**.
Filepath to TLS certificate	Absolute path and filename of the TLS certificate. printserverctl will copy the file to /etc/hmedia/printserver and process it further. No default value. Not used when **ingressNginx.tls.enabled** is set to **false**.
Filepath to TLS key	Absolute path and filename of the TLS key. The key file must be decrypted and NOT protected by a password. printserverctl will copy the file to /etc/hmedia/printserver and process it further. No default value.

	Not used when **ingressNginx.tls.enabled** is set to **false**.

For the TLS certificate: In this installation, we provide the cert data as files. As they have an expiration date, they need to be renewed regularly (max lifetime should be 397 days). The PrintServer installation could also be equipped with CertManager to automatically renew the certificates through a Root CA. Because of the many options for a PKI infrastructure, that kind of implementation is beyond the scope of this document. If you need help to establish an automatic certificate management in your Kubernetes, please contact us on mail@hmedia.de.

Parameter	Description
printserver.pullPolicy	When should Docker retrieve the printserver image from the Docker Hub registry? Answers: Always - Helm retrieves the image each time when installing that chart. IfNotPresent - Helm retrieves the image from the registry when it is not locally available. Never - Helm doesn't retrieve the image at all. It must be already present locally. Default is **Always**.
printserver.dockerTag	Which Docker Tag of the PrintServer container shall be used for the installation. Docker tags usually refer to the software version or variant. For available tags refer to https://hub.docker.com/r/hmediade/printserver/tags. Default is **4.0.0**.
printserver.numberOfInstances	How many instances of the PrintServer shall be deployed in the Kubernetes cluster? Answer is an Integer. Default is **1**.
Filepath to the iNEWS config file	Absolute path and filename of PrintServer inewsconf.xml. printserverctl will copy the file to /etc/hmedia/printserver and process it further. No default value. WITHOUT THAT CONFIG FILE, PRINTSERVER WILL NOT OPERATE. TO CREATE THE CORRECT INEWSCONF.XML, PLEASE REFER TO CHAPTER PRINTSERVER CONFIGURATION, ON PAGE 11.

Filepath to your PrintServer license file	Absolute path and filename of the PrintServer license.lic. printserverctl will copy the file to /etc/hmedia/printserver and process it further. No default value. WITHOUT THAT LICENSE, PRINTSERVER WILL NOT OPERATE FULLY FUNCTIONAL. YOU CAN OBTAIN SUCH A LICENSE FROM HMEDIA.

Parameter	Description
printserverInit.pullPolicy	When should Docker retrieve the printserver-init image from the Docker Hub registry? Answers: Always - Helm retrieves the image each time when installing that chart. IfNotPresent - Helm retrieves the image from the registry when it is not loacally available. Never - Helm doesn't retrieve the image at all. It must be already present locally. Default is **Always**.
printserverInit.dockerTag	Which Docker Tag of the PrintServer Init container shall be used for the installation. Docker tags usually refer to the software version or variant. For available tags refer to https://hub.docker.com/r/hmediade/printserver-init/tags. Default is **1.0.0**.
printstyleSource.connectionType	How shall the PrintServer retrieve its printstyle files during startup? Answers: scp - PrintServer will connect into a Linux machine via SCP. git - PrintServer checks out the required files from a Git repository via HTTPS. Default is **scp**.
printstyleSource.username	Username for printstyles retrieval. Default is **printstyles_user**.
printstyleSource.password	Password for printstyles retrieval. Default is **printstyles_passwd**.

printstyleSource.host	IP address, hostname, or FQDN of the printstyle source. Default is **printstyles_host**.
printstyleSource.path	The local path to read the printstyles from. Default is **/path/to/printstyles**. In case of SCP connection, the path should be an absolute path in the file system. In case of git clone, the path is the URL part after the server name. In total, the URL will look like: https://username:password@host/path.

Parameter	Description
cluster.fqdn	FQDN of the Kubernetes cluster. The PrintServer web page and REST API will be available on the URL https://<cluster.fqdn>/printserver. Default is **k8s.cluster.fqdn**.
service.port	The network port for the Service object (connection from the ingress to the service). Default is **8080**.

Initial Procedure

1. Ensure that all prerequisites are matched, that all data is available, and that all config files are created/copied onto the installation machine.
2. Retrieve the printserverctl script from https://hmedia.de/helm/printserverctl. Please save the script with the name `printserverctl` into an arbitrary folder of your Kubernetes Control machine (from where you execute kubectl and helm), and make the file executable with the command
   `chmod a+x <path>/printserverctl`.

   THE PRINTSERVERCTL SCRIPT MUST HAVE ACCESS TO THE CORRECT ENVIRONMENT. USUALLY THAT IS, THE SAME AS THE CURRENT USER. IF THE SCRIPT CAN'T REACH KUBERNETES, EXECUTE IT WITH `sudo -E printserverctl`.

3. Execute the script with
   `sudo -E <path>/printserverctl prepare`,
   and enter all requested data. You can safely interrupt the script at any time with `Ctrl+C`.
4. (Optional) Review/test the configuration with the commands
   `sudo -E <path>/printserverctl review`, or
   `sudo -E <path>/printserverctl dryrun`.

5. Deploy the PrintServer with the given configuration by
   `sudo -E <path>/printserverctl deploy`.

Modification

You can rerun the deployment script at any time to adjust or change the configuration, such as when updating the TLS certificate files or changing the inewsconf.xml.

WHEN YOU ONLY UPDATE THE SETTING FILES ON THE LINUX HOST, THE RUNNING PRINTSERVER INSTANCE WILL NOT BE AFFECTED. YOU NEED TO EXECUTE THE COMMAND `printserverctl deploy` AGAIN.

1. Execute the script with
   `sudo -E <path>/printserverctl prepare`,
   and update/change all requested data. The script will present all current settings as default values. You can skip those values with `Enter`, and don't need to retype each setting.
   You can safely interrupt the script at any time with `Ctrl+C`.
2. (Optional) Review/test the configuration with the commands
   `sudo -E <path>/printserverctl review`, or
   `sudo -E <path>/printserverctl dryrun`.
3. Deploy the PrintServer with the given configuration by
   `sudo -E <path>/printserverctl deploy`.
   The script will inform that it found a running instance and is going to update that deployment.

Remove

If you want to uninstall a running instance of the PrintServer, execute the command
`sudo -E <path>/printserverctl remove`,
and answer the according confirmation questions for the software itself and the namespace.

As a result, all artifacts of the PrintServer deployment will be removed from your Kubernetes cluster.

Changing Printstyles or iNEWSSystems.txt

The PrintServer as a containerized application needs access to one folder containing all printstyles. That same folder needs to contain the `iNEWSSystems.txt` file. Additionally, it can include files or subfolders/files, that are referenced by the printstyles (like icons or other graphics).

The format of the printstyles and iNewsSystems.txt is exactly the same as for the native installation and described in detail over there (chapter PrintStyle Configuration, on page 12).

When you modify printstyle files or the iNEWSSystems.txt, you need to do this on the printstyle source – either an external Linux machine or a Git repository.

USING A GIT REPOSITORY FOR THE PRINTSTYLE FILES COMES WITH THE ADDITIONAL BENEFIT OF A VERSION CONTROL SYSTEM. SO YOU CAN REVIEW OLDER VERSIONS OR ROLLBACK UNWANTED CHANGES SEAMLESSLY.

To activate the modification, simply delete the printserver pod with the command
`kubectl delete pod printserver -n <namespace>`,

where namespace is the value of the **deployment.namespace** setting. If the namespace is default, you can leave that `-n` parameter away.

Kubernetes will instantly create a new pod which initializes with the new set of printstyle files and/or iNewsSystems.txt.

OPERATION

There is no dedicated operation for the PrintServer. You can see the status page of the PrintServer via https://<cluster.fqdn>/printserver or http://<cluster.fqdn>/printserver, depending on whether TLS is enabled or not.

Refer to chapter Status Page, on page 14 for further information.

The PrintServer is equipped with a Lifeness probe. That means, Kubernetes frequently checks (every 10 seconds) whether the PrintServer REST API is still accessible. If that check fails for three times, Kubernetes will delete and recreate the PrintServer pod.

A PrintServer pod can be deleted at any time. Kubernetes will instantly create a new pod automatically.

BE AWARE THAT ALL LOG INFORMATION OF THE CURRENT POD ARE LOST WHEN THAT POD IS TERMINATED.

LOG DATA

There are the same log files inside a PrintServer pod as in the native installation. You can view the entire pod logs with the command
`kubectl logs printserver -n <namespace>`,
where namespace is the value of the **deployment.namespace** setting. If the namespace is default, you can leave that parameter away.

Alternatively, you can equip the Kubernetes cluster with a graphical admin tool like the Kubernetes Dashboard, or with metrics aggregation and a graphical dashboard solution (Prometheus and Grafana). Currently, the PrintServer doesn't export metrics on its own. The Nginx ingress controller, on the other hand, does.

If you are interested in further extensions for your Kubernetes cluster, please contact us on mail@hmedia.de.

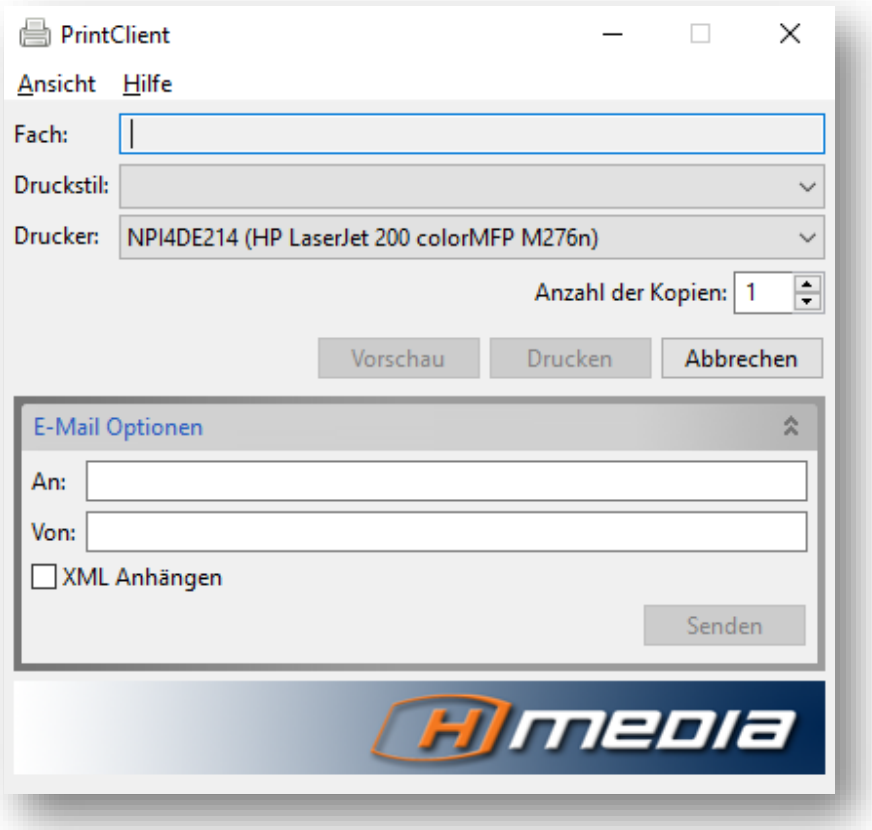
THE PRINTCLIENT

GENERAL

The PrintClient is built as a Java program. For better usability, a .NET wrapper turns the software into a common .EXE program.

Hence, the PrintClient needs both runtime environments - .NET und JRE.

The PrintClient doesn't have specific hardware requirements. It can be operated with a graphical UI, but also purely through command line parameters.



PREREQUISITES

Operating System

The PrintClient runs along with the iNEWS Workstation software on a Windows Desktop operating system.

Both Windows 10 und Windows 11 have been tested and qualified.

.NET Framework

Die .NET CLR version 4.0 or above is required. The .NET framework version 4.5 or above is required. You can download the .NET framework from https://www.microsoft.com/downloads. On that web page you can also find further details for installation of the .NET environment.

THE PRINTCLIENT INSTALLER IS ABLE TO RETRIEVE AND INSTALL THE NEEDED .NET FRAMEWORK DURING THE INSTALLATION PROCEDURE AS A PREREQUISITE.

Java

For the PrintClient, the same rules apply as for the PrintServer (see above) when it comes to the Java installation. The minimal supported version is Java 11. Hmedia recommends to use the OpenJDK, that is available on https://jdk.java.net for download.

Embedded JRE

In this version, the PrintClient comes with an embedded OpenJDK. So, the Hmedia PrintClient is independent from an otherwise installed JRE in the OS.

THE EMBEDDED JAVA RUNTIME DOESN'T RETRIEVE ANY SECURITY UPDATES AND NEED TO GET PATCHED MANUALLY.

PrintClient version 4.0 utilizes the OpenJDK 21.0.2. It is allocated in the sub folder **Java** in the PrintClient home directory (typically **C:\Program Files\Hmedia\PrintClient**).

System Default JRE

Deleting the sub folder **Java** in the PrintClient home folder forces the PrintClient to work with the system default JRE. In this case, ensure, that the java command is executable by any user that has to run the PrintClient.

ENSURE THAT THE **BIN** FOLDER OF YOUR JAVA INSTALLATION IS INCLUDED IN THE PATH ENVIRONMENT VARIABLE. THE PRINTCLIENT USER MUST BE ABLE TO EXECUTE THE JAVA COMMAND AT THE COMMAND LINE.

With **java -version** can you check if java is generally available on the computer and which version of the JRE is installed.

Adobe Acrobat Reader

The PrintClient utilizes Adobe Acrobat for the preview of rendered outputs. The installation of Acrobat Reader is not part of the PrintClient installation procedure and need to be accomplished separately. You can download/install the Acrobat Reader from the web page https://www.adobe.com/acrobat/pdf-reader.html.

The PrintClient provides a configuration parameter in the file **<PRINTCLIENT_HOME>/printclient.properties**. (See chapter PrintClient Properties, on page 33)

```
viewer.acrobat.exe= C:\\Program Files (x86)\\Adobe\\Acrobat DC\\Acrobat\\Acrobat.exe
```

That value keeps the program path of the Acrobat Reader. You don't need to quote the path if it contains white spaces. However, any backslash needs to escaped with another backslash.

If that config value is not available or empty, the PrintClient evaluates the allocation of the Adobe Acrobat Reader executable out of the Windows Registry key

`HKLM\Software\Classes\acrobat\shell\open\command`.

INSTALLATION

Ensure that all prerequisites (previous chapter) are met and available.

The PrintManager installation media contains the setup file **PrintClient Java Setup.msi** in the sub folder **PrintClient**.

Execute that program with administrative privileges. Follow the instructions. The setup tool installs the PrintClient in the chosen folder and creates links in the start menu and on the desktop.

The folder, where the PrintClient software is installed in, will be referred to as the **<PRINTCLIENT_HOME>**.

CONFIGURATION

PrintServer

You set up all PrintServers that are used by the PrintClient in the file **<PRINTCLIENT_HOME>\printserver.xml**. This XML file is a default Java Bean configuration.

Server Spec

Each PrintServer is defined as a bean of the class **de.hmedia.printmanager.printclient.inews.INewsPrintServer**. It specifies the RESTful URL of that particular server as **constructor-arg** sub element, as in the following examples.

The PrintServer is identified via the XML attribute **name**. Its content is arbitrary but must be unique in the entire configuration – in the samples below, **server1** and **server2**.

Additionally, the the PrintServers with their names must be included into the **serverList** element.

Encryption

The PrintManager 4.0 allows secure and encrypted communication between PrintClient and PrintServer. On the server side, this feature is usually provided by an additional reverse proxy (like Nginx).

The PrintClient utilizes the same keystore as the OS. Depending on the available PKI infrastructure, the key chain of the Root CA that signed the PrintServer certificate must be stored in the “Trusted Root Certification Authorities” store in the Internet Options of the Windows computer.

Authentication

If the PrintServer REST API is protected with Basic Http Authentication, the bean XML must additionally contain two property values for user and password. The user value is cleartype, and the password value needs to be base64 encoded.

In summary, the PrintServer configuration file will look like:

Example 1

A simple PrintServer configuration with only one PrintServer specified. The PrintServer API will be accessed through HTTPS. Authentication is enabled.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:util="http://www.springframework.org/schema/util"
       xsi:schemaLocation="
          http://www.springframework.org/schema/beans
          http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
          http://www.springframework.org/schema/util
          http://www.springframework.org/schema/util/spring-util-2.0.xsd">

    <bean name="server1" class="de.hmedia.printmanager.printclient.inews.INewsPrintServer">
       <constructor-arg value="https://toolserver/printserver"/>
        <property name="user" value="printclient">
        <property name="password" value="cHJpbnRjbGllbnQ=">
    </bean>

    <util:list id="serverList" list-class="java.util.ArrayList">
        <ref bean="server1"/>
    </util:list>
</beans>
```

Example 2

In this configuration, the PrintClient can connect to two PrintServers. They could either operate for different iNEWS systems, provide redundancy for the same system, or both. Encryption and authentication is only enabled for server2.

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:util="http://www.springframework.org/schema/util"
       xsi:schemaLocation="
          http://www.springframework.org/schema/beans
          http://www.springframework.org/schema/beans/spring-beans-2.0.xsd
          http://www.springframework.org/schema/util
          http://www.springframework.org/schema/util/spring-util-2.0.xsd">

    <bean name="server1" class="de.hmedia.printmanager.printclient.inews.INewsPrintServer">
        <constructor-arg value="http://toolserver:8080/printserver"/>
    </bean>

    <bean name="server2" class="de.hmedia.printmanager.printclient.inews.INewsPrintServer">
```

```
        <constructor-arg value="https://toolserver2.buero.hmedia.de/printserver"/>
        <property name="user" value="printclient">
        <property name="password" value="cHJpbnRjbGllbnQ=">
    </bean>

    <util:list id="serverList" list-class="java.util.ArrayList">
        <ref bean="server1"/>
        <ref bean="server2"/>
    </util:list>
</beans>
```

PrintClient Properties

The file **<PRINTCLIENT_HOME>/printclient.properties** hosts additional parameters to control the PrintClient behavior:

- Localdb.encoding=UTF-16LE – Encoding of the local iENWS database. Version 4.0 of the PrintManagers doesn't support printing of the local database. Hence, that setting is ignored.

- SMTP Parameters – The PrintClient is able to send out the rendered PDF straight as email. Three essential settings are required for that feature:
    - mail.smtp.host – Network name of the SMTP server.
    - mail.smtp.user – User to authenticate against the SMTP server.
    - mail.smtp.password – Password for the SMTP user.

THE SMTP CONFIGURATION CAN BE EXTENDED WITH FURTHER PARAMETERS. REFER TO THE WEB PAGE HTTPS://JAVAEE.GITHUB.IO/JAVAMAIL/DOCS/API/COM/SUN/MAIL/SMTP/PACKAGE-SUMMARY.HTML FOR FURTHER DETAILS.

In the PrintClient itself, the user can specify a sender address. The SMTP server must be configured to relay such emails that have a different from address compared to the authentication.

Example

```
localdb.encoding=UTF-16LE

# SMTP settings used by the PrintClient email function.
mail.smtp.host=smtp.hmedia.de
mail.smtp.user=mail@hmedia.de
mail.smtp.password=secret
```

OPERATION

PrintClient in Graphical Mode

If the PrintClient desktop icon is clicked or the PrintClient.exe command is issued without any parameters then the Graphical User Interface (GUI) of the PrintClient will open.

The PrintClient windows stays in foreground as long as it waits for user inputs.

A typical procedure includes following steps:

1. The user drag&drops the print scope (selected stories or a particular iNEWS queue) onto the PrintClient window. Drop target is the entire PrintClient app.
2. Next, the PrintClient retrieves the list of available printstyles from the PrintServer and filters that list depending on the selected print scope.
3. The user selects the print style and specifies what happens next:
    a. The Preview button opens the rendered PDF in Acrobat Reader.
    b. The Print button prints the document on the selected printer.
    c. The Cancel button is always available. It stops the current operation and closes the PrintClient.
    d. In the Email panel, the user can set sender and recipient and send out the email. The Email panel is only available if email is configured for the PrintClient.

As long as the PrintManager processes data and when the PrintClient shows a document in Acrobat Reader, the PrintClient window is minimized and not visible.

The PrintClient terminates after successfully sending an email or printing a document on a printer.

PrintClient Command Line Parameters

The **`PrintClient Java.exe`** can be called with parameters, to hand over certain data to the PrintClient.

If all necessary data (iNEWS system, iNEWS queue, print style) are included in the parameters, the PrintClient will start the process instantly without showing the graphical UI.

The PrintClient starts the graphical UI wiht given parameters as a pre-selection, if the parameters are not sufficient to complete the print operation..

Parameter	Description
-s [iNEWS System Name]	The iNEWS system, as configured in the PrintServer (typically the iNEWS DB name). Example: **`-s NRCS`**
-q [Path of a Queue]	The path for the queue to print. Example: **`-q NEWS.SHOW.1400.RUNDOWN`**
-t [Printstyle]	The print style, how to render the source data. Here, it is the file name of the print style file, not the display name in the PrintClient! Example: **`-t rundown_body.xsl`**
-c	With this parameter set, the PrintClient tries to read the iNEWS system and queue path out of the Windows Clipboard. In this case, the parameters -s and -q are ignored. The data in the clipboard must match following pattern (including the square brackets): **`-c [iNEWS System Name]Queue Name`**
-p [Drucker]	Specifies the printer device. It must be enclosed in double quotes if the printer name contains spaces. Without that parameter, the PrintClient

	automatically takes the Windows standard printer. Example: **-p "Microsoft XPS Document Writer"**
-show	Show the graphical UI, even if the parameters are sufficient to complete a print operation.
-o	With this parameter set, the PrintClient will open the rendered document in Adobe Acrobat and terminates.
-from [Email-Adresse]	Specifies the sender email address. Example: **-from andreas@hmedia.de**
-to [Email-Adresse]	Specifies the recipient email address. Example: **-to andreas@hmedia.de**
-type [Datentyp]	This parameter controls which data are sent via email: • **PDF_REQUEST** – the PDF will be sent. That is the default setting. • **PDF_AND_XML_REQUEST** – the PDF and additional the raw data XML will be sent. That is useful for technical implementations or trouble shooting.

Example 1

You want to print the queue NEWS.1200.RUNDOWN of the iNEWS System NRCS with the print style rundown.xsl. Then, your command line must look like:

```
PrintClient Java.exe -s NRCS – q NEWS.1200.RUDOWN -t rundown.xsl
```

Example 2

You want to print with the same source data as in example 1 but printing via the network printer \\buero.hmedia.de\hp_laser:

```
PrintClient Java.exe -s NRCS – q NEWS.1200.RUDOWN -t rundown.xsl -p \\buero.hmedia.de\hp_laser
```

Example 3

Again, you want to print with the same data as in example 1, but this time, the source shall be taken out of the Windows Clipboard. You create the clipboard data as normal text with the pattern:

```
[iNEWS System Name]Queue Name
```

Using the source data of example 1, the clipboard data then looks like:

```
[NRCS]NEWS.1200.RUNDOWN
```

Finally, the command line will then look like:

```
PrintClient Java.exe -c -t rundown.xsl
```

TO COPY THE INEWS SYSTEM NAME AND THE SOURCE QUEUE INTO THE CLIPBOARD, YOU CAN ALSO USE AN INEWS MACRO. WITHIN INEWS, THAT MACRO CAN BE ASSIGNED TO A TOOLBAR BUTTON OR A KEYBOARD SHORTCUT.

One-Click Mode

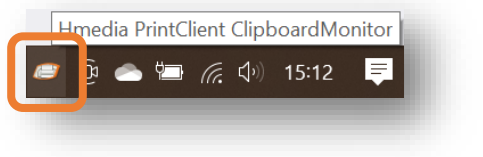
For a generic operation, the user typically has to execute two steps, even using the command line parameters. That are:

1. Copy the source queue into the clipboard. For example, this operation could be done by an iNEWS keyboard shortcut.
2. Start the render/print with a particular print style. This operation is typically done via an iNEWS toolbar button.

With the PrintClientClipboardMonitor – a Windows System Tray application – those steps can be consolidated into a single operation.

The PrintClientClipboardMonitor runs permanently in the System Tray for this purpose. If data in a specific format arrive in the Windows Clipboard, the PrintClientClipboardMonitor starts the PrintClient with all needed parameters.

The PrintClientClipboardMonitor will be automatically installed into the Autostart folder of the users.



The in the clipboard need to follow the pattern:

```
HMEDIA_PRINT[iNEWS System Name]Queue Name <Optionen>
```

To initiate a print process for the above example, following text must be copied into the clipboard:

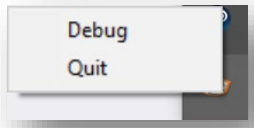
```
HMEDIA_PRINT[NRCS]NEWS.1200.RUNDOWN -t rundown.xsl
```

For example, you can bring that text into the clipboard with following iNEWS macro (English iNEWS Client). The user needs to be in the queue **NEWS.1200.RUNDOWN**. (the current queue gest printed with that macro)

```
{alt {right}{right}{left}}{alt wo}{home}HMEDIA_PRINT[NRCS]{end}{space}-t rundown.xsl {shift {home}} {ctrl c} {esc}
```

Options

The program in the system tray offers two options via the context menu (right mouse click).



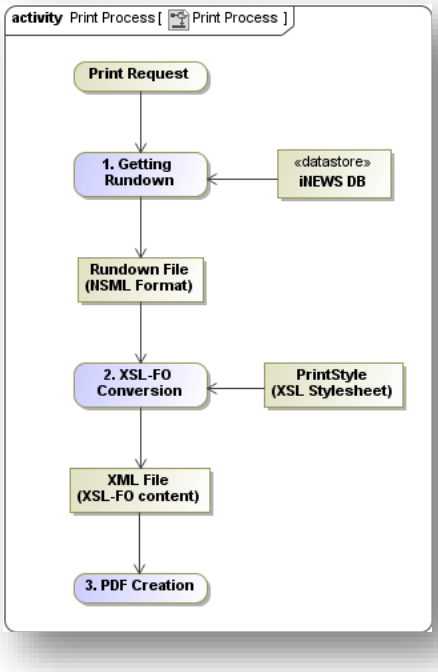
With **Debug**, the ClipboardMonitor activates the debug mode and shows the resulting executable for an identified print request. You can disable the debug mode with another click on that menu entry. The print command will be displayed as a Windows notification.

With **Quit**, the user can disable the PrintClientClipboardMonitor for that Windows session.

PRINTSTYLES

PDF CREATION PROCESS

To create own PrintStyles it is necessary to understand how data from an iNEWS queue is converted into a printable document. Figure 1 shows this process, which consists of three steps:



1. After getting a print request, the PrintServer pulls the complete rundown data out of the iNEWS database and writes it into an NSML file. While previous PrintManager versions used the NSML version 2, the PrintManager 3.2 works with the XML-conform NSML 3.
2. An XSLT processor can use this XML file and an XSL style sheet (the PrintStyle) to create an XSL-FO file. The XSL-FO file contains the rundown data together with formatting information, the so-called formatting objects (FO).
3. The Apache Formatting Objects Processor (FOP) is then used to convert the XSL-FO file into a PDF file which is finally downloaded and printed or displayed by the PrintClient/Adobe Acrobat Reader.

CREATING PRINTSTYLES

Hence creating own printstyles means to write an XSL style sheet file that can be used by the XSLT processor in step 2 (see above).

Therefore profound knowledge about XML, XSL and XSL-FO is needed to develop own printstyles. These topics are not covered in this documentation because they are not PrintManager specific and there are many websites and books devoted to. For additional information on the three topics you can consult one of the websites with tutorials by the W3C.

- XML - https://www.w3schools.com/xml/

- XSL - https://www.w3schools.com/xml/xsl_intro.asp
- XPath - https://www.w3schools.com/xml/xpath_intro.asp
- XSL-FO - https://w3schools.sinsixx.com/xslfo/default.asp.htm

The PrintManager is delivered with some default printstyles. They can be greatly used for own extensions and implementations. The default printstyles folder consists of:

- hmedia_scripts.xsl – Not a printstyle but various functions for time calculations, etc.
- production_cues.xsl – Prints stories with two columns for production cues and story text.
- rundown_body.xsl – Prints a queue as a rundown table, including story body.
- rundown_cue.xsl – Prints a queue as a rundown table including the production cues of the stories.

You find the content of those three printstyles in the Appendix below.

APPENDIX

SOURCE CODE OF DEFAULT PRINTSTYLES

production_cues.xsl

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fo="http://www.w3.org/1999/XSL/Format"
    xmlns:hmedia="xalan://hmedia.extensions"
    xmlns:java="http://xml.apache.org/xalan/java"
    exclude-result-prefixes="java hmedia" version="1.0">
    <xsl:import href="hmedia_scripts.xsl" />

    <xsl:param name="SYSTEM"/>
    <xsl:param name="QUEUE"/>

    <xsl:template match="/NSMLRoot">
        <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
            <fo:layout-master-set>
                <fo:simple-page-master master-name="alleSeiten"
                    page-height="29.7cm" page-width="21cm" margin-top="1cm"
                    margin-bottom="1cm" margin-left="1cm" margin-right="1cm">
                    <fo:region-body margin-top="0.8cm"
                        margin-bottom="1.5cm" />
                    <fo:region-before extent="0.8cm" />
                    <fo:region-after extent="0.8cm" />
                </fo:simple-page-master>
            </fo:layout-master-set>
            <fo:page-sequence master-reference="alleSeiten">
                <fo:static-content flow-name="xsl-region-after">
                    <fo:block font-size="8pt">
                        <fo:table table-layout="fixed">
                            <fo:table-column column-width="5cm"
                                column-number="1" />
                            <fo:table-column column-width="9cm"
                                column-number="2" />
                            <fo:table-column column-width="5cm"
                                column-number="3" />
                            <fo:table-body>
                                <fo:table-row>
                                    <fo:table-cell text-align="left" padding-top="5pt">
                                        <fo:block>
                                            Printed
                                            <xsl:value-of

select="java:format(java:java.text.SimpleDateFormat.new('d-MM-yyyy, HH:mm:ss'),
java:java.util.Date.new())" />
                                        </fo:block>
                                    </fo:table-cell>
                                    <fo:table-cell text-align="center">
                                        <fo:block>
                                            <fo:external-graphic
                                                src="stylesheets/icons/hmedia.jpg" width="3cm"
/>
                                        </fo:block>
                                    </fo:table-cell>
                                    <fo:table-cell text-align="right" padding-top="5pt">
                                        <fo:block>
                                            Page
                                            <fo:page-number />
                                             of
                                            <fo:page-number-citation ref-id="last-page"/>
                                        </fo:block>
                                    </fo:table-cell>
                                </fo:table-row>
                            </fo:table-body>
                        </fo:table>
                    </fo:block>
                </fo:static-content>
                <fo:flow flow-name="xsl-region-body" font-size="10pt">
```

```
                <xsl:variable name="t2" select="java:put($props, 'end', -1)"/>
                <xsl:apply-templates select="nsml" mode="endtime" />
                <xsl:call-template name="banner"/>
                <fo:table table-layout="fixed" start-indent="2pt"
                    end-indent="2pt" space-before="0.5cm" hyphenate="false">
                    <fo:table-column column-number="1"
                        column-width="9cm" />
                    <fo:table-column column-number="2"
                        column-width="9cm" />
                    <fo:table-header>
                        <fo:table-row text-align="center"
                            font-weight="bold" background-color="gray">
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Text
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Cues
                                </fo:block>
                            </fo:table-cell>
                        </fo:table-row>
                    </fo:table-header>
                    <fo:table-body>
                        <xsl:apply-templates select="nsml" />
                    </fo:table-body>
                </fo:table>
                <fo:block id="last-page"></fo:block>
            </fo:flow>

        </fo:page-sequence>
    </fo:root>
</xsl:template>

<xsl:template match="nsml">
    <xsl:variable name="video-id" select="fields/string[@id='video-id']" />
    <xsl:choose>
        <xsl:when test="head/meta/@break = 1">
            <xsl:call-template name="inhalt">
                <xsl:with-param name="rowbackground" select="'lightgray'" />
                <xsl:with-param name="formatbackground" select="'lightgray'" />
            </xsl:call-template>
        </xsl:when>
        <xsl:when
            test="string-length(normalize-space($video-id) ) > 0">
            <xsl:call-template name="inhalt">
                <xsl:with-param name="rowbackground" select="'white'" />
                <xsl:with-param name="formatbackground" select="'#FFFF99'" />
            </xsl:call-template>
        </xsl:when>
        <xsl:otherwise>
            <xsl:call-template name="inhalt">
                <xsl:with-param name="rowbackground" select="'white'" />
                <xsl:with-param name="formatbackground" select="'white'" />
            </xsl:call-template>
        </xsl:otherwise>
    </xsl:choose>
</xsl:template>

<xsl:template name="inhalt">
    <xsl:param name="rowbackground" />
    <xsl:param name="formatbackground" />
    <xsl:variable name="total-time" select="fields/duration[@id='total-time']" />
    <fo:table-row background-color="{$rowbackground}">
        <fo:table-cell border="solid thin black">
            <fo:block space-after="15pt" space-before="2pt">
                <xsl:apply-templates select="body">
                </xsl:apply-templates>
            </fo:block>
        </fo:table-cell>
        <fo:table-cell border="solid thin black">
            <fo:block space-after="15pt" space-before="2pt" font-weight="bold">
                <xsl:call-template name="inserts">
```

```
                        <!-- uncomment the next line if you don't want to print the name of
the insert -->
                        <!-- <xsl:with-param name="printInsertName" select="false()"/> -->
                        <!-- uncomment the next line if you don't want to print production
cues between inserts -->
                        <!-- <xsl:with-param name="printProductionCues" select="false()"/> -->
                    </xsl:call-template>
                </fo:block>
            </fo:table-cell>
        </fo:table-row>
    </xsl:template>

    <xsl:template name="banner">
        <fo:table table-layout="fixed" start-indent="4pt" end-indent="4pt">
            <fo:table-column column-number="1" column-width="19cm" />
            <fo:table-body font-size="14pt">
                <fo:table-row>
                    <fo:table-cell text-align="center">
                        <fo:block space-before="10pt">
                            <xsl:text>Text and Production Cues</xsl:text>
                        </fo:block>
                        <fo:block font-weight="bold" space-before="5pt" space-after="10pt">
                            <xsl:value-of select="$SYSTEM"/> - <xsl:value-of select="$QUEUE"/>
                        </fo:block>
                    </fo:table-cell>
                </fo:table-row>
            </fo:table-body>
        </fo:table>
    </xsl:template>
</xsl:stylesheet>
```

rundown_body.xsl

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fo="http://www.w3.org/1999/XSL/Format"
    xmlns:hmedia="xalan://hmedia.extensions"
    xmlns:java="http://xml.apache.org/xalan/java"
    exclude-result-prefixes="java hmedia" version="1.0">
    <xsl:import href="hmedia_scripts.xsl" />

    <xsl:param name="SYSTEM"/>
    <xsl:param name="QUEUE"/>

    <xsl:template match="/NSMLRoot">
        <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
            <fo:layout-master-set>
                <fo:simple-page-master master-name="alleSeiten"
                    page-height="29.7cm" page-width="21cm" margin-top="1cm"
                    margin-bottom="1cm" margin-left="1cm" margin-right="1cm">
                    <fo:region-body margin-top="0.8cm"
                        margin-bottom="1.5cm" />
                    <fo:region-before extent="0.8cm" />
                    <fo:region-after extent="0.8cm" />
                </fo:simple-page-master>
            </fo:layout-master-set>
            <fo:page-sequence master-reference="alleSeiten">
                <fo:static-content flow-name="xsl-region-after">
                    <fo:block font-size="8pt">
                        <fo:table table-layout="fixed">
                            <fo:table-column column-width="5cm"
                                column-number="1" />
                            <fo:table-column column-width="9cm"
                                column-number="2" />
                            <fo:table-column column-width="5cm"
                                column-number="3" />
                            <fo:table-body>
                                <fo:table-row>
                                    <fo:table-cell text-align="left" padding-top="5pt">
                                        <fo:block>
                                            Printed
                                            <xsl:value-of

select="java:format(java:java.text.SimpleDateFormat.new('d-MM-yyyy, HH:mm:ss'),
java:java.util.Date.new())" />
```

```
                                    </fo:block>
                                </fo:table-cell>
                                <fo:table-cell text-align="center">
                                    <fo:block>
                                        <fo:external-graphic
                                            src="stylesheets/icons/hmedia.jpg" width="3cm"
/>
                                    </fo:block>
                                </fo:table-cell>
                                <fo:table-cell text-align="right" padding-top="5pt">
                                    <fo:block>
                                        Page
                                        <fo:page-number />
                                         of
                                        <fo:page-number-citation ref-id="last-page"/>
                                    </fo:block>
                                </fo:table-cell>
                            </fo:table-row>
                        </fo:table-body>
                    </fo:table>
                </fo:block>
            </fo:static-content>
            <fo:flow flow-name="xsl-region-body" font-size="10pt">
                <xsl:call-template name="banner"/>
                <fo:table table-layout="fixed" start-indent="2pt"
                    end-indent="2pt" space-before="0.5cm" hyphenate="false">
                    <fo:table-column column-number="1"
                        column-width="0.8cm" />
                    <fo:table-column column-number="2"
                        column-width="2cm" />
                    <fo:table-column column-number="3"
                        column-width="7cm" />
                    <fo:table-column column-number="4"
                        column-width="7cm" />
                        <fo:table-column column-number="5"
                        column-width="2cm" />
                    <fo:table-header>
                        <fo:table-row text-align="center"
                            font-weight="bold" background-color="gray">
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Pos.
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Backtime
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Title
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Text
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Total
                                </fo:block>
                            </fo:table-cell>
                        </fo:table-row>
                    </fo:table-header>
                    <fo:table-body>
                        <xsl:apply-templates select="nsml" />
```

```
                    </fo:table-body>
                </fo:table>
                <fo:block id="last-page"></fo:block>
            </fo:flow>

        </fo:page-sequence>
    </fo:root>
</xsl:template>

<xsl:template match="nsml">
    <xsl:variable name="video-id" select="fields/string[@id='video-id']" />
    <xsl:choose>
        <xsl:when test="head/meta/@break = 1">
            <xsl:call-template name="inhalt">
                <xsl:with-param name="rowbackground" select="'lightgray'" />
                <xsl:with-param name="formatbackground" select="'lightgray'" />
            </xsl:call-template>
        </xsl:when>
        <xsl:when
            test="string-length(normalize-space($video-id) ) > 0">
            <xsl:call-template name="inhalt">
                <xsl:with-param name="rowbackground" select="'white'" />
                <xsl:with-param name="formatbackground" select="'#FFFF99'" />
            </xsl:call-template>
        </xsl:when>
        <xsl:otherwise>
            <xsl:call-template name="inhalt">
                <xsl:with-param name="rowbackground" select="'white'" />
                <xsl:with-param name="formatbackground" select="'white'" />
            </xsl:call-template>
        </xsl:otherwise>
    </xsl:choose>
</xsl:template>

<xsl:template name="inhalt">
    <xsl:param name="rowbackground" />
    <xsl:param name="formatbackground" />
    <xsl:variable name="total-time" select="fields/duration[@id='total-time']" />
    <fo:table-row background-color="{$rowbackground}" keep-together.within-page="auto"
keep-with-next="always">
        <fo:table-cell border="solid thin black">
            <fo:block space-after="15pt" space-before="2pt"
                text-align="center" font-weight="bold">
                <xsl:value-of select="fields/string[@id='page-number']" />
            </fo:block>
        </fo:table-cell>
        <fo:table-cell border="solid thin black">
            <fo:block space-after="15pt" space-before="2pt"
                text-align="right" font-weight="bold">
                <!-- calculating backtime -->
                <xsl:variable name="sum">
                    <xsl:variable name="back-time" select="substring-
after(fields/duration[@id='back-time'], '@')" />
                    <xsl:choose>
                        <xsl:when
                            test="string-length($back-time) > 0">
                            <xsl:value-of select="$back-time" />
                        </xsl:when>
                        <xsl:when
                            test="string-length($back-time) = 0">
                            <xsl:variable name="sum">
                                <xsl:call-template name="calculateBacktime">
                                    <xsl:with-param name="nsmlNodes"
                                        select="following-sibling::*" />
                                </xsl:call-template>
                            </xsl:variable>
                            <xsl:value-of
                                select="$sum - fields/duration[@id='total-time']" />
                        </xsl:when>
                    </xsl:choose>
                </xsl:variable>
                <xsl:if test="not(head/meta/@float = '1') and not($sum &lt; 0)">
                    <xsl:value-of select="hmedia:DateTime.getTime('H:mm:ss',
number($sum))" />
                </xsl:if>
            </fo:block>
        </fo:table-cell>
        <fo:table-cell border="solid thin black">
```

```
                <fo:block space-after="15pt" space-before="2pt"
                    font-weight="bold">
                    <xsl:value-of select="fields/string[@id='title']" />
                </fo:block>
            </fo:table-cell>
            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt">
                    <!-- <xsl:call-template name="inserts" /> -->
                </fo:block>
            </fo:table-cell>
            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt"
                    text-align="right" font-weight="bold">
                    <xsl:value-of
                        select="hmedia:DateTime.getTime('mm:ss', number($total-time))" />
                </fo:block>
            </fo:table-cell>
        </fo:table-row>
        <fo:table-row keep-together="auto">
            <fo:table-cell border="solid thin black">
                <fo:block/>
            </fo:table-cell>
            <fo:table-cell border="solid thin black" number-columns-spanned="4">
                <fo:block space-after="15pt" space-before="2pt">
                    <xsl:apply-templates select="body">
                    </xsl:apply-templates>
                </fo:block>
            </fo:table-cell>

        </fo:table-row>
    </xsl:template>

    <xsl:template name="banner">
        <fo:table table-layout="fixed" start-indent="4pt" end-indent="4pt">
            <fo:table-column column-number="1" column-width="19cm" />
            <fo:table-body font-size="14pt">
                <fo:table-row>
                    <fo:table-cell text-align="center">
                        <fo:block space-before="10pt">
                            <xsl:text>Rundown with text</xsl:text>
                        </fo:block>
                        <fo:block font-weight="bold" space-before="5pt" space-after="10pt">
                            <xsl:value-of select="$SYSTEM"/> - <xsl:value-of select="$QUEUE"/>
                        </fo:block>
                    </fo:table-cell>
                </fo:table-row>
            </fo:table-body>
        </fo:table>
    </xsl:template>
</xsl:stylesheet>
```

rundown_cues.xsl

```
<?xml version="1.0" encoding="utf-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
    xmlns:fo="http://www.w3.org/1999/XSL/Format"
    xmlns:hmedia="xalan://hmedia.extensions"
    xmlns:java="http://xml.apache.org/xalan/java"
    exclude-result-prefixes="java hmedia" version="1.0">

    <xsl:import href="hmedia_scripts.xsl" />

    <xsl:param name="SYSTEM"/>
    <xsl:param name="QUEUE"/>

    <xsl:template match="/NSMLRoot">
        <fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">
            <fo:layout-master-set>
                <fo:simple-page-master master-name="alleSeiten"
                    page-height="29.7cm" page-width="21cm" margin-top="1cm"
                    margin-bottom="1cm" margin-left="1cm" margin-right="1cm">
                    <fo:region-body margin-top="0.8cm"
                        margin-bottom="1.5cm" />
                    <fo:region-before extent="0.8cm" />
                    <fo:region-after extent="0.8cm" />
```

```
            </fo:simple-page-master>
        </fo:layout-master-set>
        <fo:page-sequence master-reference="alleSeiten">
            <fo:static-content flow-name="xsl-region-after">
                <fo:block font-size="8pt">
                    <fo:table table-layout="fixed">
                        <fo:table-column column-width="9.5cm"
                            column-number="1" />
                        <fo:table-column column-width="9.5cm"
                            column-number="2" />
                        <fo:table-body>
                            <fo:table-row>
                                <fo:table-cell text-align="left">
                                    <fo:block>
                                        Printed
                                        <xsl:value-of

select="java:format(java:java.text.SimpleDateFormat.new('d-MM-yyyy, HH:mm:ss'),
java:java.util.Date.new())" />
                                    </fo:block>
                                </fo:table-cell>
                                <fo:table-cell text-align="right">
                                    <fo:block>
                                        Page
                                        <fo:page-number />
                                         of
                                        <fo:page-number-citation ref-id="last-page"/>
                                    </fo:block>
                                </fo:table-cell>
                            </fo:table-row>
                        </fo:table-body>
                    </fo:table>
                </fo:block>
            </fo:static-content>
            <fo:flow flow-name="xsl-region-body" font-size="10pt">
                <xsl:call-template name="banner"/>
                <fo:table table-layout="fixed" start-indent="2pt"
                    end-indent="2pt" space-before="0.5cm" hyphenate="false">
                    <fo:table-column column-number="1"
                        column-width="1.5cm" />
                    <fo:table-column column-number="2"
                        column-width="7cm" />
                    <fo:table-column column-number="3"
                        column-width="6.6cm" />
                    <fo:table-column column-number="4"
                        column-width="2.0cm" />
                    <fo:table-column column-number="5"
                        column-width="1.7cm" />
                    <fo:table-header>
                        <fo:table-row text-align="center"
                            font-weight="bold" background-color="gray">
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Pos.
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Title
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Cue
                                </fo:block>
                            </fo:table-cell>
                            <fo:table-cell
                                border="solid thin black">
                                <fo:block space-after="2pt"
                                    space-before="2pt">
                                    Total
                                </fo:block>
```

```
                                </fo:table-cell>
                                <fo:table-cell
                                    border="solid thin black">
                                    <fo:block space-after="2pt"
                                        space-before="2pt">
                                        Backtime
                                    </fo:block>
                                </fo:table-cell>
                            </fo:table-row>
                        </fo:table-header>
                        <fo:table-body>
                            <xsl:apply-templates select="nsml" />
                        </fo:table-body>
                    </fo:table>
                    <fo:block id="last-page"></fo:block>
                </fo:flow>

            </fo:page-sequence>
        </fo:root>
    </xsl:template>

    <xsl:template match="nsml">
        <xsl:variable name="video-id" select="fields/string[@id='video-id']" />
        <xsl:variable name="break" select="head/meta/@break" />
        <xsl:choose>
            <xsl:when test="head/meta/@break = 1 or head/meta/@break = 'true'">
                <xsl:call-template name="inhalt">
                    <xsl:with-param name="rowbackground" select="'lightgray'" />
                    <xsl:with-param name="formatbackground" select="'lightgray'" />
                </xsl:call-template>
            </xsl:when>
            <xsl:when
                test="string-length(normalize-space($video-id) ) > 0">
                <xsl:call-template name="inhalt">
                    <xsl:with-param name="rowbackground" select="'white'" />
                    <xsl:with-param name="formatbackground" select="'#FFFF99'" />
                </xsl:call-template>
            </xsl:when>
            <xsl:otherwise>
                <xsl:call-template name="inhalt">
                    <xsl:with-param name="rowbackground" select="'white'" />
                    <xsl:with-param name="formatbackground" select="'white'" />
                </xsl:call-template>
            </xsl:otherwise>
        </xsl:choose>
    </xsl:template>

    <xsl:template name="inhalt">
        <xsl:param name="rowbackground" />
        <xsl:param name="formatbackground" />
        <xsl:variable name="float" select="head/meta/@float" />
        <xsl:variable name="total-time" select="fields/duration[@id='total-time']" />
        <xsl:variable name="back-time" select="substring-after(fields/duration[@id='back-
time'], '@')" />

        <xsl:variable name="foreground">
            <xsl:choose>
                <xsl:when test="$float = 1 or $float = 'true'">
                    <xsl:value-of select="'blue'"/>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="'black'"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:variable>

        <xsl:variable name="backtime-foreground">
            <xsl:choose>
                <xsl:when test="string-length($back-time) > 0">
                    <xsl:value-of select="'green'"/>
                </xsl:when>
                <xsl:otherwise>
                    <xsl:value-of select="'black'"/>
                </xsl:otherwise>
            </xsl:choose>
        </xsl:variable>

        <fo:table-row keep-together="always" background-color="{$rowbackground}">
```

```
            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt"
                    text-align="center" font-weight="bold" color="{$foreground}">
                    <xsl:value-of select="fields/string[@id='page-number']" />
                </fo:block>
            </fo:table-cell>

            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt"
                    font-weight="bold" color="{$foreground}">
                    <xsl:value-of select="fields/string[@id='title']" />
                </fo:block>
            </fo:table-cell>
            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt">
                    <xsl:call-template name="inserts" />
                </fo:block>
            </fo:table-cell>
            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt"
                    text-align="right" font-weight="bold" color="{$foreground}">
                    <xsl:value-of
                        select="hmedia:DateTime.getTime('mm:ss', number($total-time))" />
                </fo:block>
            </fo:table-cell>
            <fo:table-cell border="solid thin black">
                <fo:block space-after="15pt" space-before="2pt"
                    text-align="right" font-weight="bold" color="{$backtime-foreground}">
                    <xsl:if test="not($float = 1 or $float = 'true')">
                        <!-- calculating backtime -->
                        <xsl:variable name="sum">
                             <xsl:choose>
                                 <xsl:when test="string-length($back-time) > 0">
                                     <xsl:value-of select="$back-time" />
                                 </xsl:when>
                                 <xsl:when
                                     test="string-length($back-time) = 0">
                                     <xsl:variable name="sum">
                                         <xsl:call-template name="calculateBacktime">
                                             <xsl:with-param name="nsmlNodes"
                                                 select="following-sibling::*" />
                                         </xsl:call-template>
                                     </xsl:variable>
                                     <xsl:value-of
                                         select="$sum - fields/duration[@id='total-time']" />
                                 </xsl:when>
                             </xsl:choose>
                        </xsl:variable>
                        <xsl:if test="not(head/meta/@float = '1') and not($sum &lt; 0)">
                            <xsl:value-of select="hmedia:DateTime.getTime('H:mm:ss',
number($sum))" />
                        </xsl:if>
                    </xsl:if>
                </fo:block>
            </fo:table-cell>
        </fo:table-row>
    </xsl:template>

    <xsl:template name="banner">
        <fo:table table-layout="fixed" start-indent="4pt" end-indent="4pt">
            <fo:table-column column-number="1" column-width="13cm" />
            <fo:table-column column-number="2" column-width="6cm" />
            <fo:table-body font-size="14pt">
                <fo:table-row>
                    <fo:table-cell text-align="center">
                        <fo:block space-before="10pt">
                            <xsl:text>Rundown</xsl:text>
                        </fo:block>
                        <fo:block font-weight="bold" space-before="5pt" space-after="10pt">
                            <xsl:value-of select="$SYSTEM"/> - <xsl:value-of select="$QUEUE"/>
                        </fo:block>
                    </fo:table-cell>
                    <fo:table-cell text-align="right">
                        <fo:block>
                            <fo:external-graphic
                                src="stylesheets/icons/hmedia.jpg" width="6cm" />
                        </fo:block>
                    </fo:table-cell>
```

```
                </fo:table-row>
            </fo:table-body>
        </fo:table>
    </xsl:template>
</xsl:stylesheet>
```