



# Kofax SignDoc Standard Installation Guide

Version: 3.1.0

Date: 2022-02-01

© 2021 Kofax. All rights reserved.

Kofax is a trademark of Kofax, Inc., registered in the U.S. and/or other countries. All other trademarks are the property of their respective owners. No part of this publication may be reproduced, stored, or transmitted in any form without the prior written permission of Kofax.

# Table of Contents

<b>Preface</b> .....	<b>5</b>
Related documentation.....	5
Offline documentation.....	6
Training.....	6
Getting help with Kofax products.....	6
<b>Chapter 1: Introduction</b> .....	<b>8</b>
General overview.....	8
<b>Chapter 2: Base installation</b> .....	<b>10</b>
Installation as a Windows service.....	11
Definitions.....	11
General prerequisites.....	11
Quickstart.....	11
Content of the SignDoc Standard ZIP archive.....	12
Production setup.....	12
Advanced configuration.....	16
Configuration backup.....	18
Advanced information.....	18
Logging.....	19
Prepare Microsoft SQL database.....	21
Monitoring application using JMX.....	22
Installation on Tomcat without using the provided service installer.....	25
Installation on other JEE compliant application servers.....	26
Additional information.....	26
Install and configure Microsoft SQL Server.....	27
Database installation.....	27
Database engine configuration.....	31
Administration Center.....	40
Deployment on Linux.....	41
Installation in Docker environment.....	41
<b>Chapter 3: Advanced installation</b> .....	<b>43</b>
Hardening a SignDoc installation.....	43
Content Security Policy (CSP).....	44
Authentication LDAP.....	45
<b>Chapter 4: SignDoc Authentication Module (SAM)</b> .....	<b>48</b>

Purpose.....	48
Requirements.....	48
Installation.....	48
Usage with browser.....	48
Configuration of SAM.....	49
Service configuration.....	49
Identity provider related configuration.....	50
SDS SSO configuration options.....	53
Automatic user creation.....	54
Supported IDP user metadata.....	54
SSL configuration.....	55
Stop / Start / Uninstall the SAM Service.....	55
Starting the SAM service.....	55
Stopping the SAM service.....	55
Uninstalling the SAM service.....	55
Glossary.....	55
<b>Chapter 5: Uninstall SignDoc Standard.....</b>	<b>57</b>
<b>Chapter 6: Upgrade SignDoc Standard.....</b>	<b>58</b>
Upgrade from SignDoc Standard 2.2.1.....	58
Upgrade from SignDoc Standard 1.3.1 or earlier versions.....	59
Upgrade troubleshooting.....	59
<b>Chapter 7: Database migration.....</b>	<b>61</b>
Overview.....	61
Flyway.....	61
Flyway use in SignDoc.....	61
Integration and configuration.....	61
Version numbers.....	62
Classic deployment.....	63
Docker deployment.....	64
FAQ.....	64

# Preface

SignDoc Standard transforms customer experiences by streamlining the signing of documents. SignDoc Standard accelerates business workflows by removing steps such as printing, routing, and shipping documents back and forth. Constituents can sign electronically on any device anywhere resulting in significant operational cost reductions, productivity increases, and improved compliance.

## Related documentation

The full documentation set for SignDoc Standard is available at the following location:

<https://docshield.kofax.com/Portal/Products/SD/3.1.0-tp2w5bx8yi/SD.htm>

In addition to this guide, the documentation set includes the following items:

Release notes

- *Kofax SignDoc Release Notes*

Technical specifications

- *Kofax SignDoc Technical Specifications*

Guides

- *Kofax SignDoc Standard Administrator's Guide*
- *Kofax SignDoc Standard Developer's Guide*

Help

- *Kofax SignDoc Standard Help*
- *Kofax SignDoc Standard Administration Center Help*
- *Signing Documents with Kofax SignDoc Help*
- *Kofax SignDoc Assistant App Help*

Software development kit

- *Kofax SignDoc SDK API Documentation (C)*
- *Kofax SignDoc SDK API Documentation (C++)*
- *Kofax SignDoc SDK API Documentation (.NET with exceptions)*
- *Kofax SignDoc SDK API Documentation (.NET without exceptions)*
- *Kofax SignDoc SDK API Documentation (Java)*

## Offline documentation

Customers who require offline documentation can download the `KofaxSignDocDocumentation_3.1.0_EN.zip` from the [Kofax Fulfillment Site](#). The .zip file includes both `help` and `print` directories.

1. Download the documentation .zip file from the [Kofax Fulfillment Site](#).
2. Extract the contents of the compressed documentation file.
3. For online help navigate to the directory `help` and for all other documentation to the directory `print`.

## Training


Kofax offers both classroom and online training to help you make the most of your product. To learn more about training courses and schedules, visit the [Kofax Education Portal](#) on the Kofax website.

## Getting help with Kofax products

The [Kofax Knowledge Base](#) repository contains articles that are updated on a regular basis to keep you informed about Kofax products. We encourage you to use the Knowledge Base to obtain answers to your product questions.

To access the Kofax Knowledge Base:

1. Go to the [Kofax website](#) home page and select **Support**.
2. When the Support page appears, select **Customer Support > Knowledge Base**.

 The Kofax Knowledge Base is optimized for use with Google Chrome, Mozilla Firefox or Microsoft Edge.

The Kofax Knowledge Base provides:

- Powerful search capabilities to help you quickly locate the information you need.  
Type your search terms or phrase into the **Search** box, and then click the search icon.
- Product information, configuration details and documentation, including release news.  
Scroll through the Kofax Knowledge Base home page to locate a product family. Then click a product family name to view a list of related articles. Please note that some product families require a valid Kofax Portal login to view related articles.

From the Knowledge Base home page, you can:

- Access the Kofax Community (for all customers).  
Click the **Community** link at the top of the page.

- Access the Kofax Customer Portal (for eligible customers).  
Click the **Support** link at the top of the page. When the Customer & Partner Portals Overview appears, click **Log in to the Customer Portal**.
- Access the Kofax Partner Portal (for eligible partners).  
Click the **Support** link at the top of the page. When the Customer & Partner Portals Overview appears, click **Log in to the Partner Portal**.
- Access Kofax support commitments, lifecycle policies, electronic fulfillment details, and self-service tools.  
Go to the **General Support** section, click **Support Details**, and then select the appropriate tab.

## Chapter 1

# Introduction

### SignDoc Standard

- is designed to manage document-based transactions in a multi-channel environment.
- is the fastest way to get a signature from people in a convenient and secure process.
- is an end-to-end solution enabling preparation, execution and management of transactions in a digital environment across organizations and individuals.
- is compliant with federal e-signature legislation, which gives electronic documents and signatures the same legal standing as paper documents and ink signatures.
- provides a signed document containing all audit information if desired (self-contained document).
- supports multiple signature types such as click-to-sign, handwritten, and photo.
- can be operated behind the firewall (in-house) or in an enterprise cloud environment.

### The SignDoc Standard fundamentals are

- intuitive UI (self manageable process)
- customizable workflow (who, what, where, how & when, expiration, call back)
- signer authentication (options and API)
- sign anywhere and anytime - individually adapted to environment (click, mouse, photo, handwritten - be mobile and flexible)
- audit trail included in signed documents
- self-contained documents ("standard" PDF viewer also for audit trail)

For enterprise workflow applications SignDoc Standard provides integration interfaces via web services.

SignDoc Standard has been designed to be flexible - for SignDoc Standard users and recipients of signing packages.

**i** Recipients of SignDoc Standard do not need a SignDoc Standard account.

## General overview

### Layer 1 - Operating systems

SignDoc Standard can be installed on Windows and Linux operating systems with a 64-bit architecture.

### Layer 2 - SignDoc Standard application



The application consists of two WAR (web application archive) files which are deployed into a web application server. The sdweb.war file contains Native Libraries which are used for licensing and PDF handling. The configuration files are stored outside of the WAR files in directory referenced by environment variable SIGNDOC\_HOME.

### **Layer 3 - REST interface**

It is possible to interact with the system via REST API that supports almost all aspects of the application. Amongst many other things, it is possible to create and schedule signing packages with one REST request. A detailed API documentation is included.

### **Layer 4 - Web application server**

SignDoc Standard runs on Apache Tomcat, the standard web application server used within the industry. For supported versions and prerequisites, see *Kofax SignDoc Technical Specifications* document.

## Chapter 2

# Base installation

### General

SignDoc Standard can run on Windows and Linux operating systems. For a list of supported environments, see *Kofax SignDoc Technical Specifications* document.

This guide assists in setting up a standard installation on a Windows 64-bit. Screen shots might look different depending on the Windows version used. For an installation on Linux systems, see the provided and documented Dockerfile.

The installed system consists at least of these components:

- Database (Microsoft SQL Server)
- Application Server (Apache Tomcat and SignDoc Standard)

SignDoc Standard is executed as J2EE compatible application in the Application Server. Apache Tomcat and Microsoft SQL Server are usually installed on different computers (nodes) for various reasons, but they can also be installed on the same computer (node).

### Reverse Proxy / Load Balancing

This installation guide does not consider/discuss the setup of a Reverse Proxy or Load Balancing.

### SSL Setup

This installation guide does not consider/discuss an SSL configuration, since this usually depends on local IT regulations and is effectively transparent to SignDoc Standard.

### Software requirements for this guide

For version details, see *Kofax SignDoc Technical Specifications* document.

To be able to use SignDoc Standard 3.1.0, you have to install these Microsoft Visual C++ Redistributable Packages:

- [Visual Studio 2017](#) (depending on the Windows version, it might be necessary to install some updates via Windows Update before this setup can be successfully installed).
- Windows PowerShell (powershell.exe) must be in the system path. This should be the normal case for the Windows Server OS.
- If this is an upgrade of SignDoc Standard, check [Upgrade SignDoc Standard](#).
- Microsoft SQL Server is required (the express version is sufficient).

## Installation as a Windows service

This section provides information on installing SignDoc Standard as a Windows service.

### Definitions

- `INSTALLDIR` is the directory of the unpacked SignDoc\*-tomcat.zip file. See [Quickstart procedure](#).
- `CIRRUS_HOME` is the home directory of the web application (cirrus) that composes SignDoc Standard. Starting with SignDoc Standard 2.2.0 the `SDWEB_HOME` directory is no longer required. See [Content of the SignDoc Standard ZIP archive](#), section "Directories".

### General prerequisites

Before starting the installation it is required to install and check the following prerequisites.

- To be able to use SignDoc Standard 3.1.0, you have to install the [Microsoft Visual C++ Redistributable Package](#).  
Depending on the Windows version, it might be necessary to install some updates via Windows Update before this setup can be successfully installed.
- Windows PowerShell (powershell.exe) must be in the system path. This should be the normal case for the Windows Server OS.
- If this is an upgrade of SignDoc Standard, check [Upgrade SignDoc Standard](#).

**i** It is recommended to install SignDoc Standard behind a reverse proxy. If the reverse proxy is also used to load-balance requests, it can be done stateless (e.g. round-robin).

### Quickstart

Getting a simple local accessible SignDoc Standard installation running can be achieved in less than 5 minutes. It is not wasted time doing this, since it is a base for a production ready setup.

#### Quickstart goals

- Install SignDoc Standard as a Windows service.
- Database: preconfigured for a local file-based H2 database.

**i** The data stored in the local file-based H2 database cannot be migrated to a production database based on Microsoft SQL Server.

- SMTP configuration: preconfigured for localhost with port 1025 (no authentication or encryption).
  - Works with [MailHog](#) out of the box.
  - MailHog can be stopped and deleted at any time. After a real SMTP server is configured, there is no more need for it.

#### Quickstart prerequisites

- 8 GB RAM

- Download and run MailHog: <https://github.com/mailhog/MailHog>.  
If this is not possible or not wanted, a real SMTP server must be configured first. See [Configure SMTP server connection](#).
- Access MailHog: `http://localhost:8025` (if applicable).
- Enable startup email feature. See [Production setup](#), section "Startup email".

**i** MailHog is only needed for the quickstart scenario. When using startup email with a real SMTP server, make sure to use a real email address.

### Quickstart procedure

- Double-check that [General prerequisites](#) are fulfilled.
- Unpack the `signdoc-standard*-tomcat.zip` file in a new directory `INSTALLDIR`.

#### Example

```
C:\Program Files\signdoc-standard-3.1.0
```

- Double-click `INSTALLDIR\service_up.cmd`
- Wait approximately 1 minute on first start.
- A SignDoc Standard startup email should be sent to the specified startup email recipient while starting up.
- Open SignDoc Standard: `http://localhost:6611`.

## Content of the SignDoc Standard ZIP archive

The relevant and configurable content of SignDoc Standard consists basically of 3 files:

- a configuration file
- a script to install and configure the SignDoc Standard Windows service
- a script to deregister the SignDoc Standard Windows service

### Tools

- `INSTALLDIR\service_up.cmd` installs, applies configuration, and restarts the SignDoc Standard Windows service.
- `INSTALLDIR\service_remove.cmd` stops and deregisters the SignDoc Standard Windows service. No files are deleted.
- `INSTALLDIR\service_configuration.properties` is the configuration file of the SignDoc Standard Windows service. This file can be edited with a regular text editor. The syntax and usage is described in the file.

### Directories

- `INSTALLDIR\signdoc_home` is the consolidated default `CIRRUS_HOME` directory. Starting with SignDoc Standard 2.2.0 there is no need to maintain a SignDoc Web configuration directory.

## Production setup

The following sections describe basic tasks that should or must be completed for a production setup.

### Goals for production

- Configure SMTP server connection
- Configure database connection
- Configure network settings
- Configure reverse proxy setup (optional)
- Advanced configuration (optional)
- KTA integration (optional)
- LDAP integration (optional)

### Prerequisites for production setup

- Application server
  - Minimum 8 GB RAM. See [Advanced configuration](#), section "Tune Java memory settings".
  - Minimum 2 GB free disk space
- Database
  - Installed Microsoft SQL Server with a database for SignDoc Standard and a database user with database owner (dbo) credentials for this database. See [Prepare Microsoft SQL database](#).
- SignDoc Standard
  - Install SignDoc Standard as described in [Quickstart](#).

### Procedure for production

The following topics do not depend on each other and can be executed independently. What is common for all settings: The settings must be applied by executing (i.e. double-clicking) `INSTALLDIR\service_up.cmd`.

### Configure SMTP server connection

A valid and trustworthy SMTP connection is required to be able to send emails.

1. Open the `INSTALLDIR\service_configuration.properties` file in a text editor.
2. Navigate to # `[EMAIL CONFIGURATION]`.
3. Amend the settings with the configuration parameters of real SMTP server.  
See also the commented examples at the bottom of `service_configuration.properties`.

### SMTP TLS example

```
mail.smtp.host=email-smtp.us-east-1.amazonaws.com
mail.smtp.port=587
mail.smtp.user=<Access key ID>
mail.smtp.from=dont_reply@mydomain.com
mail.smtp.password=<Secret access key>
mail.smtp.starttls.enable=true
mail.smtp.starttls.required=true
mail.smtp.ssl.checkserveridentity=false
```

### Startup email

It can be useful to send a startup email to a predefined address whenever the SignDoc Standard server is starting. SignDoc Standard can be configured for this purpose. The startup email contains

information about configuration, the environment and start parameters of the SignDoc Standard application.

To enable the startup email follow these steps:

1. Open the `INSTALLDIR\service_configuration.properties` file in a text editor.
2. Navigate to # [EMAIL CONFIGURATION].
3. Uncomment the line starting with `#cirrus.startup.email`.
4. Set a valid email address.

Example

```
cirrus.startup.email=ksdadmin@localhost
```

### Configure JDBC Server connection



- For production purposes only Microsoft SQL Server is supported.
- Since SignDoc 2.2.1.2.0.64 the `jdbc.url` value MUST NOT be enclosed in single quotes anymore.

To configure JDBC Server connection follow these steps:

1. Open the `INSTALLDIR\service_configuration.properties` file in a text editor.
2. Navigate to # [DATABASE CONFIGURATION]
3. Amend the setting with the configuration parameters and credentials of the JDBC connection.  
See also the commented examples at the bottom of `service_configuration.properties`.

Microsoft SQL Server example

```
jdbc.url=jdbc:sqlserver://my-mssql-server:1433;databaseName=signdoc  
jdbc.username=signdoc  
jdbc.password=2beChanged!
```

### Configure network settings

The following network settings have to be configured.

- `SERVICE_HTTP_PORT`
- `SERVICE_EXTERNAL_HOST_URL`

To configure `SERVICE_HTTP_PORT` follow these steps:

1. Open the `INSTALLDIR\service_configuration.properties` file in a text editor.
2. Navigate to `SERVICE_HTTP_PORT`.
3. Set the preferred port number.

Example

```
SERVICE_HTTP_PORT=6611
```

For `SERVICE_EXTERNAL_HOST_URL` a production service must be accessible via official domain name, so it can be accessed from other computers. See section "Configure reverse proxy setup".

### HTTPS/TLS support

While it is possible to use TLS with SignDoc directly, it is generally recommended to use a reverse proxy to offload the TLS connections. This reduces the load and provides more flexibility for hosting and maintaining the SignDoc application.

To enable HTTPS/TLS the following configuration changes must be done:

1. Edit the `INSTALLDIR\service_configuration.properties` file.  
Use `https://` for the `SERVICE_EXTERNAL_HOST_URL` setting.  
Example  
`https://localhost:${SERVICE_HTTP_PORT}`
2. Edit the `INSTALLDIR\conf_templates\server.xml` file.
  - Comment the default `http` connector (as described in the documentation notes of the file).
  - Uncomment and configure the `https` connector (as described in the documentation notes of the file).
    - By default the `https` connector will use a self-signed certificate that can only be used for test purposes.
    - To use an individual and trustworthy certificate, at least `keystoreFile`, `keystorePass`, `keyAlias` must be adjusted.
    - It is recommended to use a PKCS#12 cert store (\*.pfx, \*.p12) that contains a private key as well as all required certificates.
3. Apply the configuration and use `service_up.cmd` to restart the service.

### Configure reverse proxy setup

In a reverse proxy scenario it is important to configure the application URLs correctly.

1. Open the `INSTALLDIR\service_configuration.properties` file in a text editor.
2. Navigate to `SERVICE_EXTERNAL_CONTEXT_URL`.  
This is the context URL that is used to access the application. This URL must be reachable from anywhere and is part of the signing links that are sent via email.
3. Change the values if required.

Example

```
SERVICE_EXTERNAL_CONTEXT_URL=https://signdoc.mydomain.com
```

### KTA integration

As of SignDoc Standard 2.1.0 the KTA (Kofax TotalAgility) connection is individually defined per account/tenant in the SignDoc Standard Manage Client or SignDoc Standard Administration Center. See [Related documentation](#):

- *SignDoc Standard Administration Center Help*, section "Plugins"
- *SignDoc Standard Administrator's Guide*, section "KTA state change plugin"

## LDAP integration

LDAP (or Active Directory) can be used to authenticate SignDoc Standard users in the Manage Client.

Configure LDAP settings in `INSTALLDIR\service_configuration.properties` (Section # LDAP integration) as described in chapter [Authentication LDAP](#).

**i** It's recommended to do the configuration in `INSTALLDIR\service_configuration.properties` instead of the `cirrus.properties` file.

See also [Advanced configuration - Option 2](#).

## Advanced configuration

### General

To configure more features of SignDoc Standard, there are 2 options:

**Option 1:** Configuration using `service_configuration.properties` (generally recommended)

1. Edit the `INSTALLDIR\service_configuration.properties` file.
2. Amend `service_configuration.properties` with additional settings.

**Option 2:** Configuration using `cirrus.properties` (backwards compatible and for special configurations)

1. Edit the `INSTALLDIR\_conf_templates\cirrus.properties` file .  
The default location is `INSTALLDIR\bin\signdoc_home\conf\cirrus.properties`.
2. Apply the configuration by executing `INSTALLDIR\service_up.cmd`.

**i** The configuration defined in `INSTALLDIR\service_configuration.properties` takes precedence over settings with the same name in `cirrus.properties`.

### Special configuration settings

The following configuration setting should always be configured in `cirrus.properties` and not in `service_configuration.properties`:

- `ldap.user.search.filter`

Example

```
ldap.user.search.filter=( &(objectClass=person) (| (cn=John Doe)
(mail=john.doe@example.com)) )
```

- `jdbc.password`

The `jdbc.password` setting can consist only of ASCII characters.

**i** If characters outside the ASCII character set are being used, the server cannot start up.

This restriction does not apply, when the password is declared as encrypted configuration property (see next chapter).



## Encrypted configuration properties

It is possible to create symmetrically encrypted configuration data. This is a pragmatic approach for SignDoc Standard to protect sensitive configuration data. By default the data will be encrypted using an AES-256 key. Configuration is applied using `service_up.cmd`.

1. Delete or comment the `property_name` from `service_configuration.properties`, otherwise the encrypted value will not be used.
2. Double-click the command `service/speh.cmd`.
3. In the input field enter the configuration to encrypt (e.g. the jdbc password).
4. The next dialog displays the encrypted data.
5. The next dialog displays the configuration entry that must be added to `INSTALLDIR\_conf_templates\cirrus.properties`. Must be replaced with the correct property name.
6. Restart the service with `service_up.cmd`.

Step by step example for configuration property `jdbc.password`.

1. Delete or comment the `jdbc.password` setting from `INSTALLDIR\_service_configuration.properties`.
2. Double-click the command `service/speh.cmd`.
3. Enter "1234" as password.
4. The next dialog shows `2682e444e7935f2af0b9e20a4266e29b`.
5. The next dialog shows `.encrypted_string_256=2682e444e7935f2af0b9e20a4266e29b`. Add `jdbc.password.encrypted_string_256=2682e444e7935f2af0b9e20a4266e29b` to `INSTALLDIR\_conf_templates\cirrus.properties`.
6. Restart the service with `service_up.cmd`.

## Control database migrations

If SignDoc Standard is run in a clustered environment, it makes sense to disable the automatic database migrations. See [Database migration](#).

1. Open the `INSTALLDIR\_service_configuration.properties` file in a text editor.
2. Navigate to section # Database migrations.
3. Set `cirrus.migrations.enabled` to `false`.

Example

```
# disable automatic migrations
cirrus.migrations.enabled=false
```

## Tune Java memory settings

1. Open the `INSTALLDIR\_conf_templates\SignDocStandard.xml` file in a text editor.
2. Look for the following lines and change the values to your needs:

```
<!-- minimum Java HEAP -->
<argument>-Xms1024m</argument>
<!-- maximum Java HEAP -->
```

```
<argument>-Xmx2048m</argument>
```

3. After having changed one of these values, `service_up.cmd` must be executed to apply the new values.

### Use SignDoc Standard in a clustered environment

SignDoc Standard can be used in a clustered environment. A typical use case is load balancing.

Starting with SignDoc 2.2.0 the application server is stateless and can therefore be used with simple round-robin load balancing.

If multiple instances should be installed on the same operating system, it must be ensured to use a different HTTP/TCP port for each instance. See [Production setup](#), section "Configure network settings". The default HTTP/TCP port for SignDoc Standard is 6611.

**i** If the `server.xml` file must be changed, it should be done in the file `INSTALLDIR\_conf_templates\server.xml`. Execute `service_up.cmd` to apply the change. See [Content of the SignDoc Standard ZIP archive](#), section "Tools".

## Configuration backup

For a backup of the SignDoc Standard instance configuration it is sufficient to backup the files and directories listed below. Such a backup set can be applied 1:1 to a new SignDoc Standard installation (e.g. for additional instances in a cluster). If a SignDoc Standard update is done make sure to check [Upgrade SignDoc Standard](#) first. It is also possible to simply backup the complete directory.

### Required

- `INSTALLDIR\service_configuration.properties`  
Is the main configuration file of the SignDoc Service.

### Optional

- `INSTALLDIR\_conf_templates`  
Contains the potentially modified configuration templates `cirrus.properties`, `logging.properties`, `server.xml`. If none of these files were modified, there is no need to backup them.
- `INSTALLDIR\signdoc_home\conf`  
Contains basic configuration. If this was not customized manually, there is no need to backup the files.
- `INSTALLDIR\signdoc_home\fonts`  
Contains the font configuration. If this was not customized manually, there is no need to backup the files.
- `INSTALLDIR\signdoc_home\bin\db`  
If existing, this directory contains the file based default database. If this database is not used, there is no need to backup the files.

## Advanced information

### View service details

To view a detailed service information double-click

```
INSTALLDIR\service\bin\SignDocStandard.exe
```

It is not recommended to change settings with this tool, since they are being overwritten whenever `service_up.cmd` is executed. See [Content of the SignDoc Standard ZIP archive](#), section "Tools" and [Advanced configuration](#).

## Logging

Since SignDoc 3.0.0, logging is configured in the SignDoc Administration Center.

There is also a configuration file (`INSTALLDIR\signdoc_home\conf\signdoc-logger.properties`), that is only considered, if configuration options in the Administration Center are not changed. It is recommended to use the configuration options of the Administration Center since configuration changes are applied without having to restart the service.

The SignDoc Standard Windows service uses the file `INSTALLDIR\signdoc_home\conf\tomcat-logging.properties` for the logging configuration file of the Tomcat application server. Consult the Tomcat configuration if changes should be made.

### Configuration options

The following configuration IDs described can be set:

- **signdoc.logger.handler.enabled**  
If 'on', logging is enabled. If 'off', logging is disabled and all other logging options are ignored.  
Default: on
- **signdoc.logger.level**  
Sets the system-wide logging level for all instances.

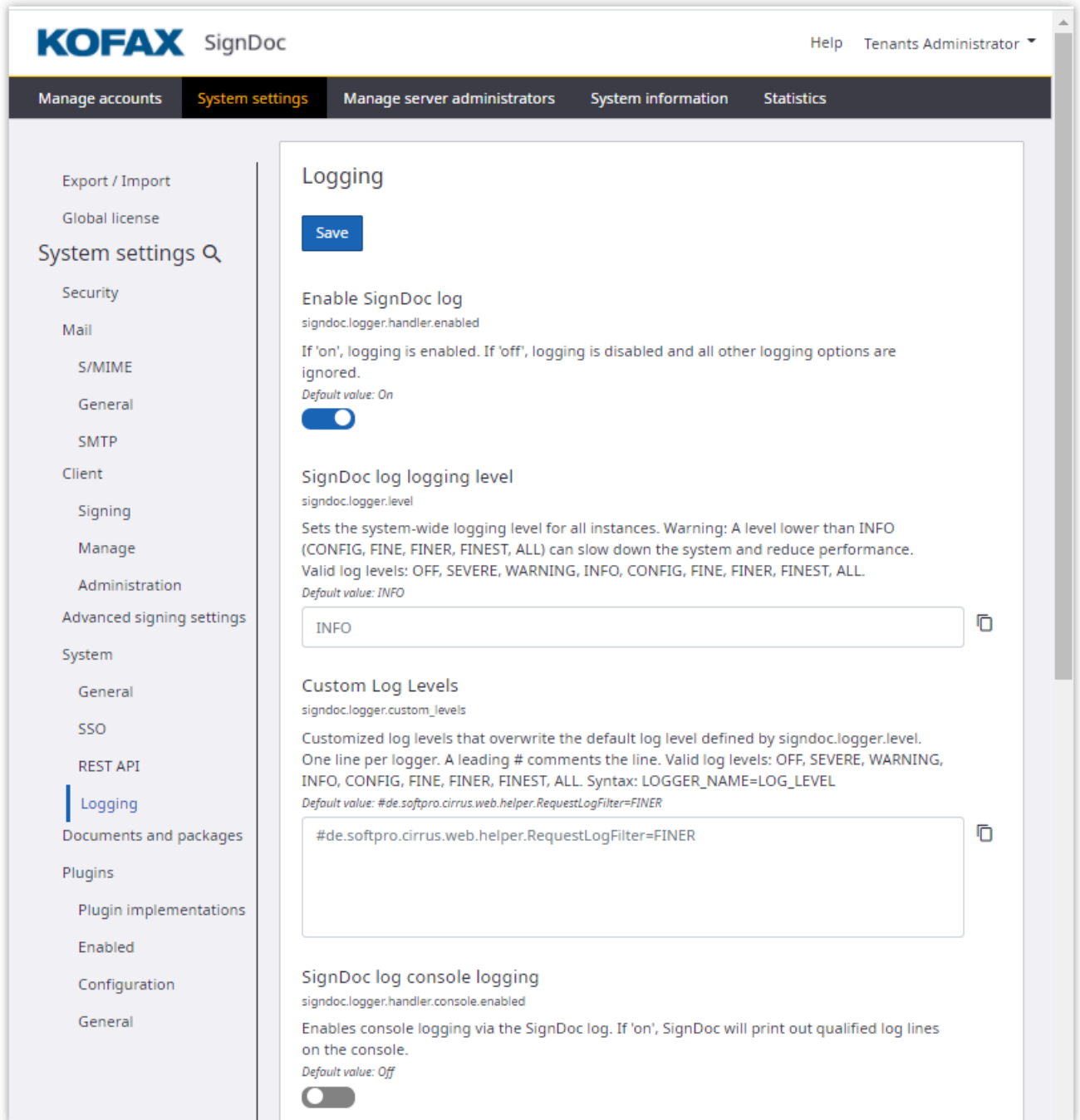
❗ A level lower than INFO (CONFIG, FINE, FINER, FINEST, ALL) can slow down the system and reduce performance.

Valid log levels: OFF, SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST, ALL

Default: INFO

- **signdoc.logger.custom\_levels**  
Customized log levels that overwrite the default log level defined by `signdoc.logger.level`. One line per logger. A leading # comments the line.  
Valid log levels: OFF, SEVERE, WARNING, INFO, CONFIG, FINE, FINER, FINEST, ALL  
Syntax: `LOGGER_NAME=LOG_LEVEL`  
Default: `#de.softpro.cirrus.web.helper.RequestLogFilter=FINE`
- **signdoc.logger.handler.console.enabled**  
Enables console logging via the SignDoc log. If 'on', SignDoc will print out qualified log lines on the console.  
Default: off

- **signdoc.logger.handler.file.enabled**  
Controls the SignDoc file logging. If 'on', SignDoc will print out qualified log lines in the specified log file. See signdoc.logger.handler.logfile.  
Default: on
- **signdoc.logger.handler.logfile**  
Defines the file used by the SignDoc log. The SignDoc process must be able to write to the file. If the file does not exist, the application will try to create the file and all needed parent directories.  
Default: SIGNDOC\_HOME/logs/signdoc/signdoc.log
- **signdoc.logger.handler.date.format**  
Sets the format of the timestamp used in the SignDoc log file.  
Default: yyyy-MM-dd\_HH:mm:ss.SSS
- **signdoc.logger.handler.logfile.maxsize**  
Defines the maximum size of a SignDoc log file in kilobytes. If the SignDoc log file size is more than the set limit, a new log file is created for future logs. The lowest possible allowed size is 10K kilobytes.  
Default: 1000000
- **signdoc.logger.handler.logfile.maxnumber**  
Defines the maximum number of SignDoc log files that can be stored or maintained. If the total number of SignDoc log files exceeds this set number, all the previous log files are deleted.  
Default: 20



## Prepare Microsoft SQL database

To be able to use SignDoc Standard for production usage, it is required to set up a database and database user that can be used by SignDoc Standard. This can either be achieved using the GUI tools provided by MS-SQL or with a T-SQL script.

### Example T-SQL script

```
CREATE DATABASE signdoc
GO
USE signdoc
GO
CREATE LOGIN signdoc WITHPASSWORD='2beChanged!'
GO
CREATE USER signdoc FOR LOGIN signdoc
GO
ALTER ROLE db_owner ADD MEMBER signdoc
GO
```

## Monitoring application using JMX

SignDoc can be monitored using the provided JMX metrics. Besides the standard metric provided by the JMV, SignDoc offers own metrics that are grouped below the SignDoc MBean node. By default, the JMX metrics can be queried by a local process running on the same server using the same user. If a remote access is required, the `jmxremote.port` must be secured for production use. See [Enabling the Ready-to-Use Management](#).

### Configuration

The JMX configuration options (find all options in the links below) are added without the `-D` prefix to `service_configuration.properties`. After restarting the service with `service_up.cmd` the settings are applied.

The basic configuration options are:

- **`com.sun.management.jmxremote=[true | false]`**  
Must be set to true to enable the remote JMX management functionality.
- **`com.sun.management.jmxremote.port=<port_number>`**  
Defines the port JMX uses.  
If a firewall is used, make sure that this port must be accessible.
- **`com.sun.management.jmxremote.ssl=[true | false]`**  
Specifies if ssl should be used to encrypt the data.  
If this setting is set to true, further configuration options are required. Please check the links in "Related information" section.
- **`com.sun.management.jmxremote.authenticate=[true | false]`**  
Specifies, if the user must authenticate to get authorization.  
If this setting is set to true, further configuration options are required. Please check the links in "Related information" section.
- **`java.rmi.server.hostname=<IP or hostname>`**  
The setting `java.rmi.server.hostname` restricts the access to the specified IP Address or hostname:
  - `java.rmi.server.hostname=127.0.0.1`  
Will make the JMX port only accessible from localhost and not from a remote computer.
  - `java.rmi.server.hostname=<HOSTNAME>`  
Will make the JMX port accessible from any remote computer that can resolve and access the HOSTNAME.

## Example

The section below will open an unsecured JMX port that is accessible from 127.0.0.1 (localhost) only. To secure the port, see [Enabling the Ready-to-Use Management](#).

```
com.sun.management.jmxremote=true
com.sun.management.jmxremote.port=1099
com.sun.management.jmxremote.ssl=false
com.sun.management.jmxremote.authenticate=false
java.rmi.server.hostname=127.0.0.1
```

## SignDoc Metrics

SignDoc offers metrics and operations below the domain. Supported SignDoc metrics can be found under:

### SignDoc > Status > Global

#### Attributes

- **Accounts:** List of all account IDs
- **ActiveSigningSessions:** The number of all started, but unfinished signing sessions across all accounts.
- **SysAdmins:** List all system administrators
- **TotalAccounts:** The number of all accounts (regardless of status)
- **TotalDocuments:** The number of all documents in the system
- **TotalPackages:** The number of all signing packages in the system (regardless of status)
- **TotalUsers:** The number of all users in the system across all accounts.

#### Operations (apart from trivial derivation of the attributes)

```
getSignDocAccountStatus(String accountID)
```

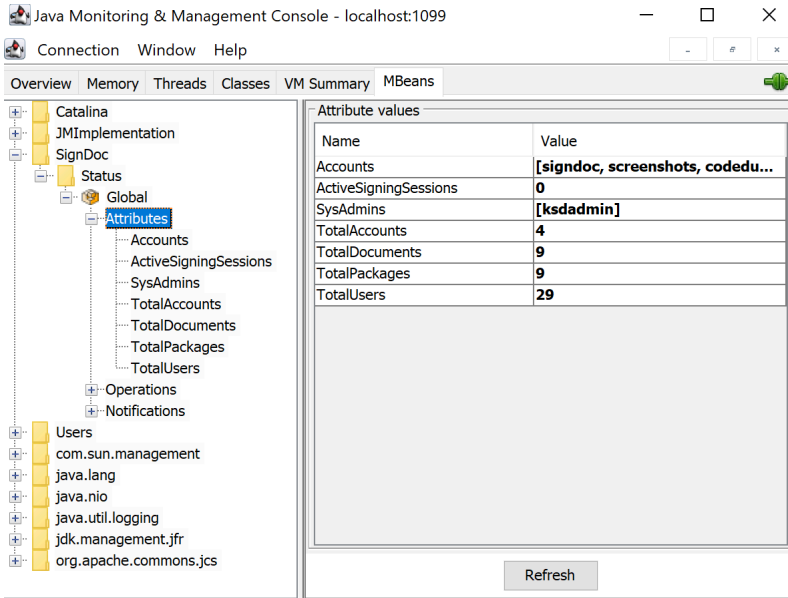
This operation returns a map with account-specific information.

Returned account information:

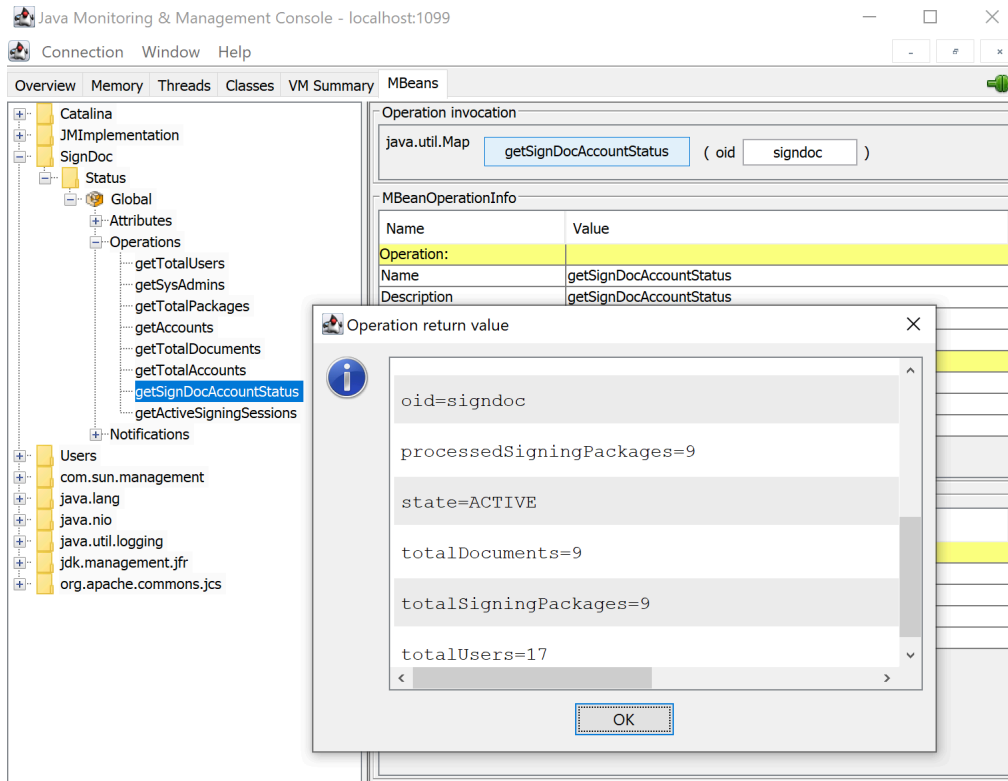
Attribute	Description
activeSigningSessions	The number of all started, but unfinished signing sessions of this account
admins	List of all account administrators
contactInformation	Contact information of the account. Empty, if not set
lastRenewalTimestamp	Last time the license was renewed
maxSigningPackages	Maximum number of allowed signing packages. -1 means unlimited
maxUsers	Maximum number of allowed users. -1 means unlimited
name	The account name
oid	The account ID
processedSigningPackages	Number of already processed signing packages
state	The account state. Must be ACTIVE to be usable

Attribute	Description
totalDocuments	The number of all documents in the account
totalSigningPackages	The number of all signing packages in the system (regardless of status)
totalUsers	The number of all users in the system across all accounts

If JConsole is used, it looks typically like this:







### Related information

- Monitoring and Management Using JMX Technology:  
<https://docs.oracle.com/en/java/javase/11/management/monitoring-and-management-using-jmx-technology.html#GUID-805517EC-2D33-4D61-81D8-4D0FA770D1B8>
- Enabling the Ready-to-Use Management:  
<https://docs.oracle.com/en/java/javase/11/management/monitoring-and-management-using-jmx-technology.html#GUID-E17343B0-FED2-4050-88D7-E4793E704ED5>
- Using JConsole:  
<https://docs.oracle.com/en/java/javase/11/management/using-jconsole.html#GUID-77416B38-7F15-4E35-B3D1-34BFD88350B5>
- JConsole Monitoring application is part of OpenJDK:  
<https://adoptium.net/releases.html?variant=openjdk11>

## Installation on Tomcat without using the provided service installer

SignDoc can be installed on an already existing Tomcat server.

See the *Kofax SignDoc Technical Specifications* document available on the [Kofax SignDoc 3.1.0](#) product documentation page for information about supported versions of Tomcat server and Java Runtime server you must use for the installation.

Follow these steps:

1. Unpack the `cirrus.war` file manually in `%CATALINA_HOME%\webapps` so that there is a `cirrus` context directory.
2. Copy the following `.jar` files from the `service\lib` directory to the `lib` directory of the Tomcat directory (usually `CATALINA_HOME`):
  - `splm2jni-*.jar`
  - `SPSignDoc-*.jar`
3. Make sure that the directory with the native libraries `service\lib\native\Win64` is in the `PATH` of the Tomcat process.
4. Make sure that there are no older SignDoc native libraries in the `PATH` of the Tomcat process.
5. Copy the `signdoc_home` directory to the preferred location. This location must be readable and writable for the Tomcat process.
6. Copy `_conf_templates\cirrus.properties` to the `signdoc_home/conf` directory and configure it as desired.
7. To allow big file uploads it might be required to adjust the `maxPostSize` attribute of the `<Connector>` element of Tomcat's `server.xml`.
8. Start the Tomcat service with the following system properties.

Required properties:

```
-DCIRRUS_HOME=<path_to_signdoc_home_directory>  
-DSERVICE_EXTERNAL_HOST_URL=<generally_accessible_url_to_root_context>
```

Example:

```
-DCIRRUS_HOME="c:/signdoc_home"  
-DSERVICE_EXTERNAL_HOST_URL=http://mysigndocserver.example.com
```

## Installation on other JEE compliant application servers

Installation on other JEE compliant servers is not supported.

## Additional information

### Device Connector - Certificate provider plugin

To be able to use a certificate provider plugin with SignDoc Standard, it must be recompiled due to improved encryption standards.

In SignDoc Standard 2.0.x it was required to use these RSA parameters:

- Padding algorithm: PKCS1

Since SignDoc Standard 2.1.0 it is required to use these RSA parameters:

- Padding algorithm: OAEP
- Message digest: SHA-1
- Mask generation function: MGF1

## Install and configure Microsoft SQL Server

For production purposes, SignDoc Standard requires a database server to be able to store application data. Currently Microsoft SQL Server is supported. While installing the database server, use the suggested defaults unless noted otherwise.

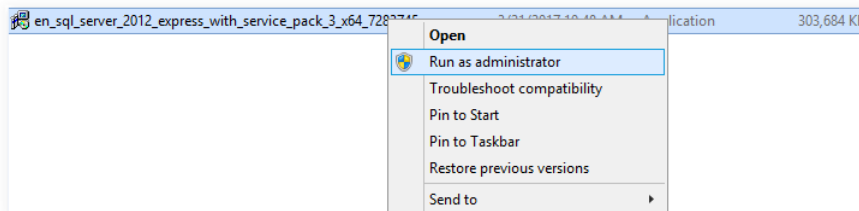
For this guide Microsoft SQL Server 2012 will be used as database service.

### Database installation

Example

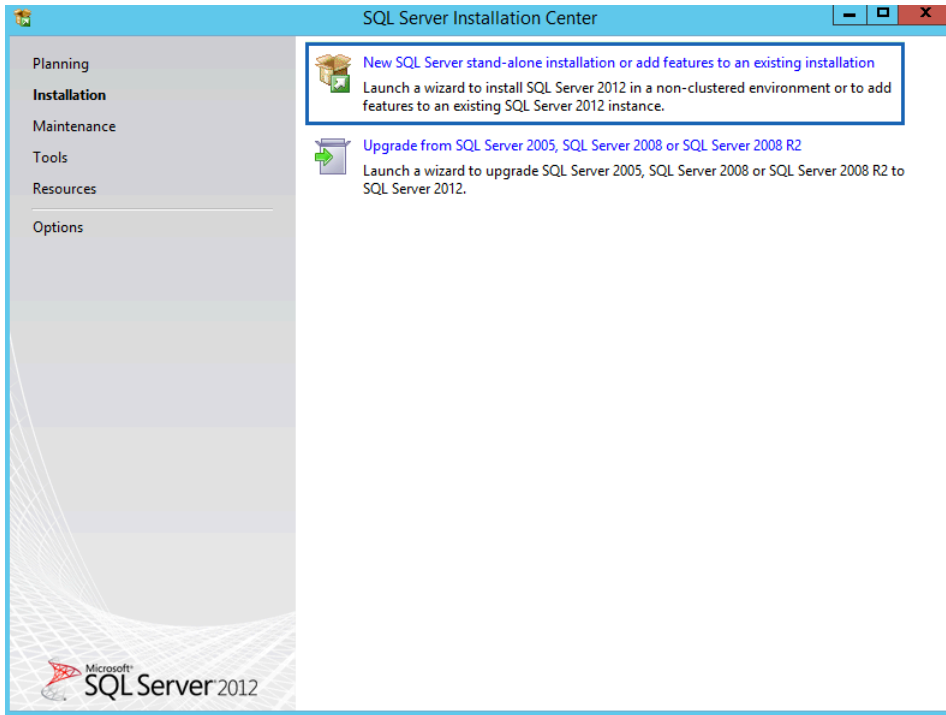
Microsoft SQL Server 2012 Express  
(en\_sql\_server\_2012\_express\_with\_service\_pack\_3\_x64\_7283745.exe)

1. Run the installation with administrator rights (if required).



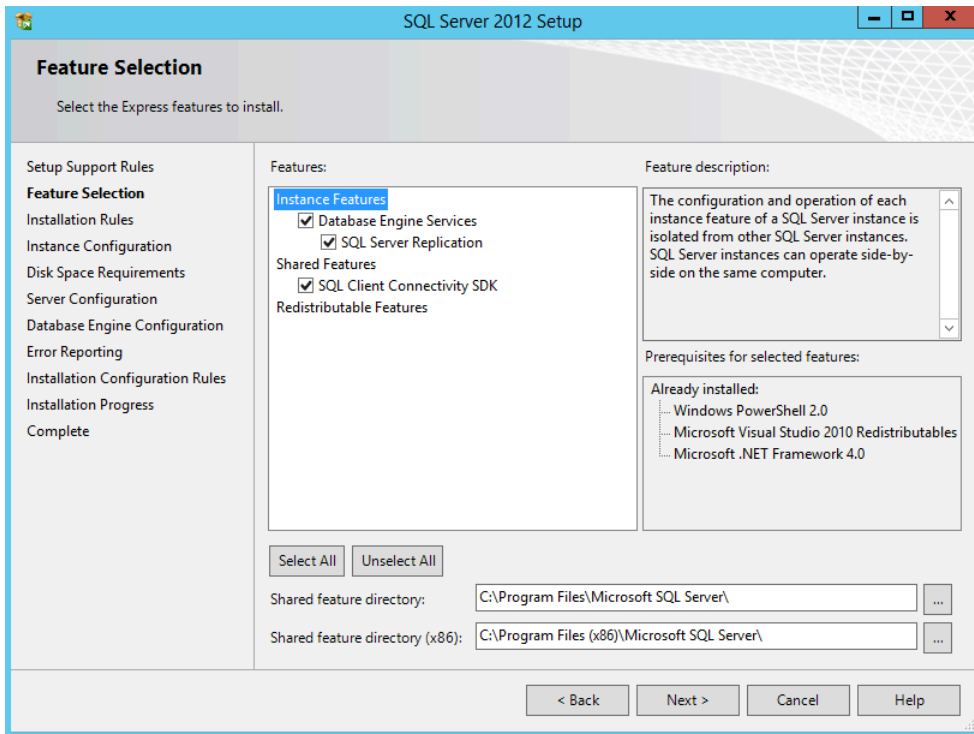
The **SQL Server Installation Center** window appears.

2. From the **Installation** menu select **New SQL Server stand-alone installation or add features to an existing installation.**

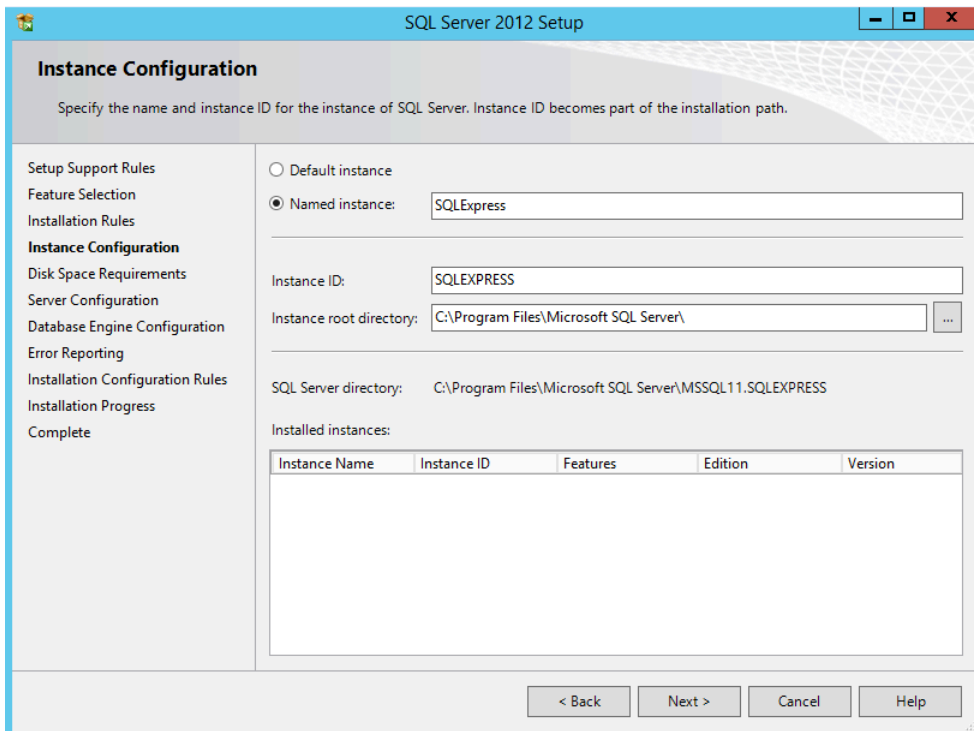


After following the installation steps the **SQL Server 2012 Setup** window appears.

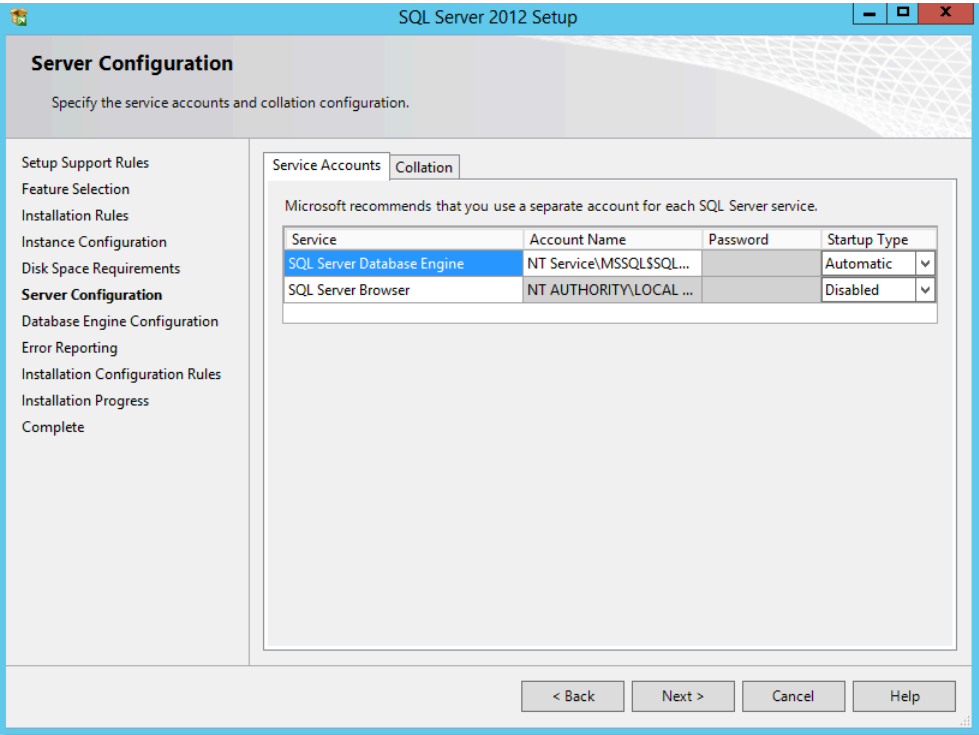
3. Click **Feature Selection** and select all available features.



4. Click **Instance Configuration**.

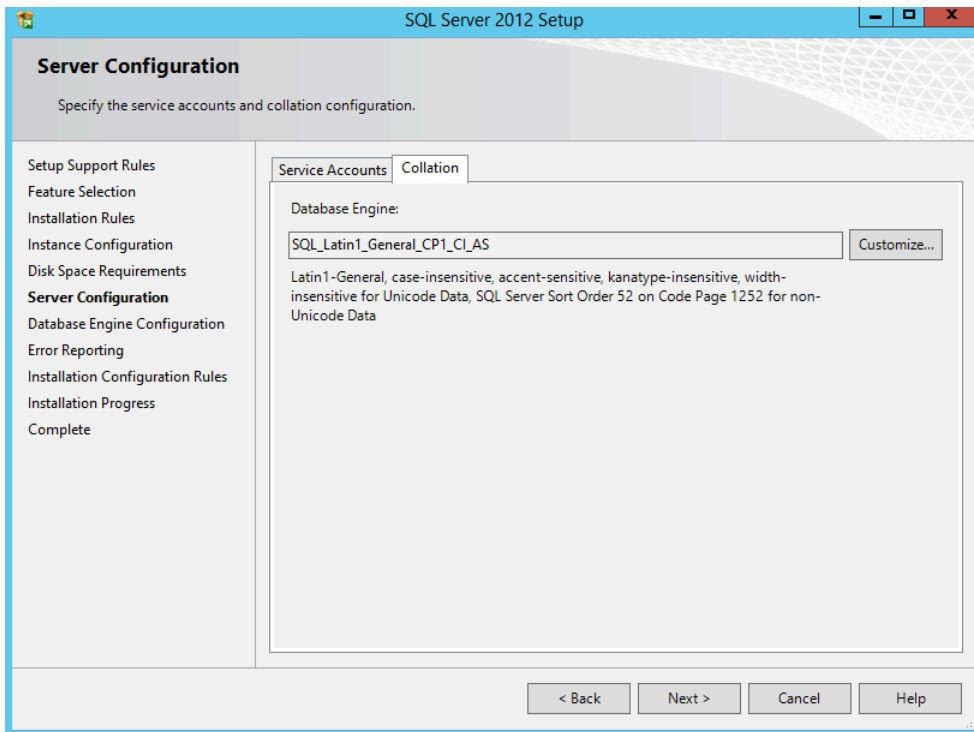


5. Then click **Server Configuration**.



6. Click **Collation** tab.

⚠ Choose a collation which is case-insensitive and also accent-sensitive (e.g. Latin1\_General\_CI\_AS, which is the default for Microsoft SQL Server). This ensures that no duplicates are stored for particular data like email addresses and object identifiers.



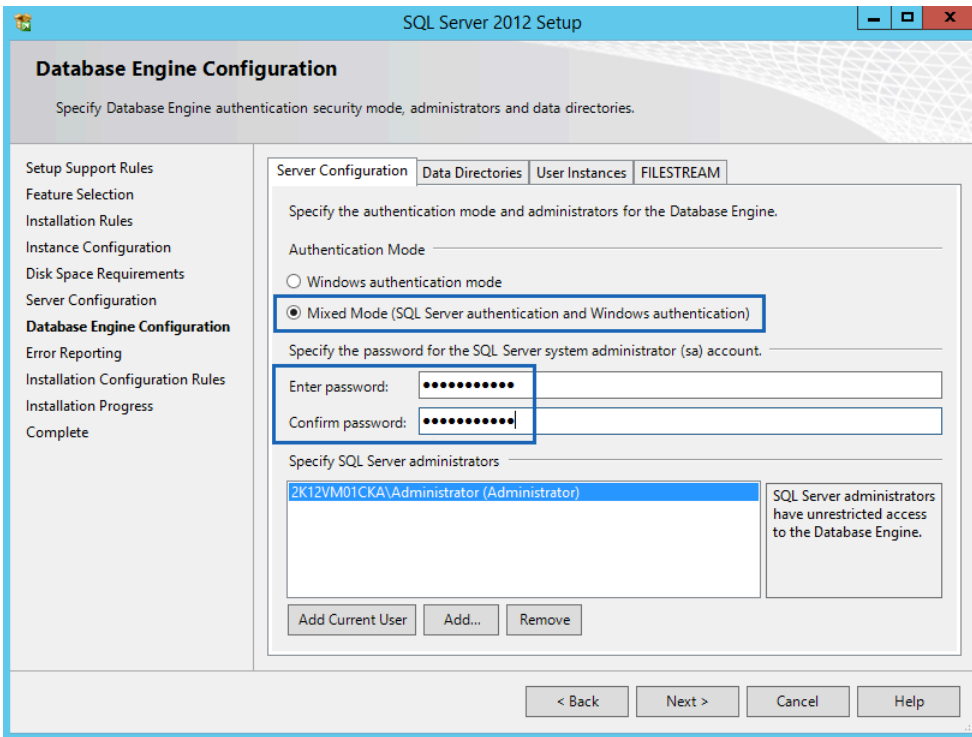
ⓘ The application will not startup if the collation is not suitable. An `InvalidOperationException("The database collation '<collationName>' is not suitable for the application.")` is written to the log file in that case.

## Database engine configuration

Example

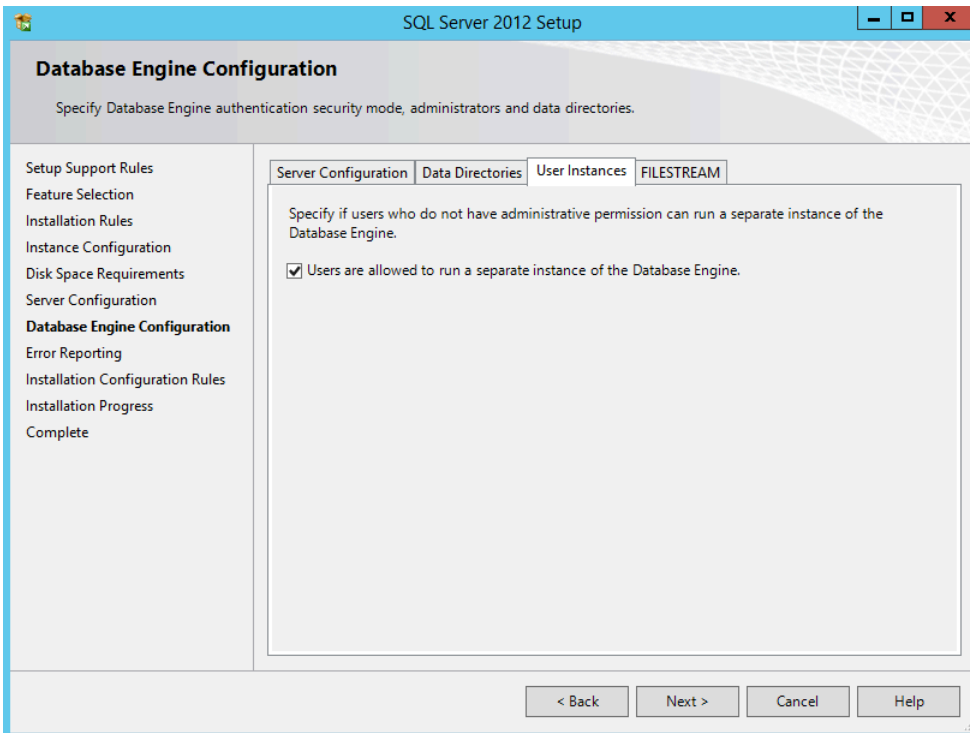
Microsoft SQL Server 2012 Express  
(en\_sql\_server\_2012\_express\_with\_service\_pack\_3\_x64\_7283745.exe)

1. In the **SQL Server 2012 Setup** window, click **Database Engine Configuration**.  
With the **Server Configuration** tab opened, select the **Mixed Mode** option button.  
Then define the password for the system administrator (sa) user. For this guide **2beChanged!** will be used.





2. Click the **User Instances** tab .



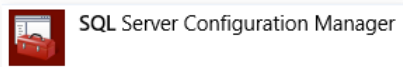
3. Finish the installation.

### Set the preferred TCP port for connections

Make sure that the server listens on the preferred TCP port for connections. For this guide port 1433 will be used.

Example

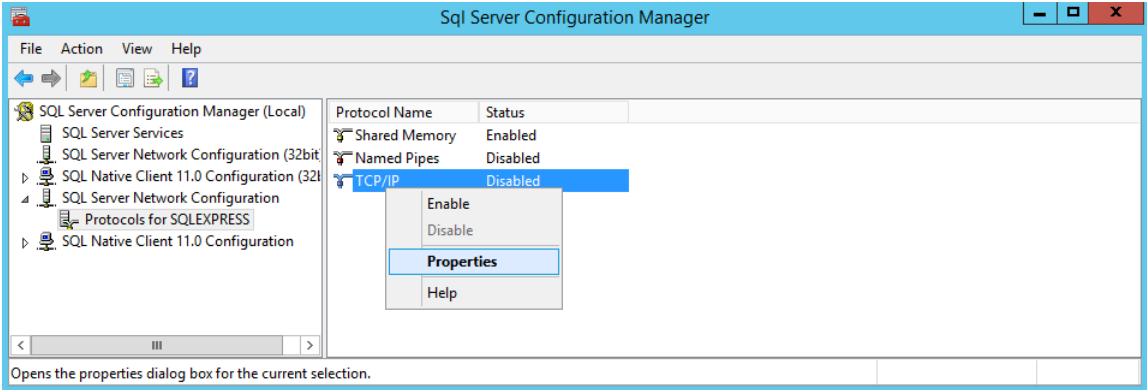
1. Start the **SQL Server Configuration Manager**



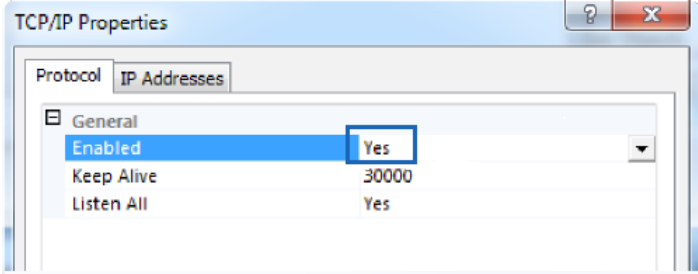
and run the program as administrator.

2. Select **SQL Server Network Configuration > Protocols for SQLEXPRESS.**

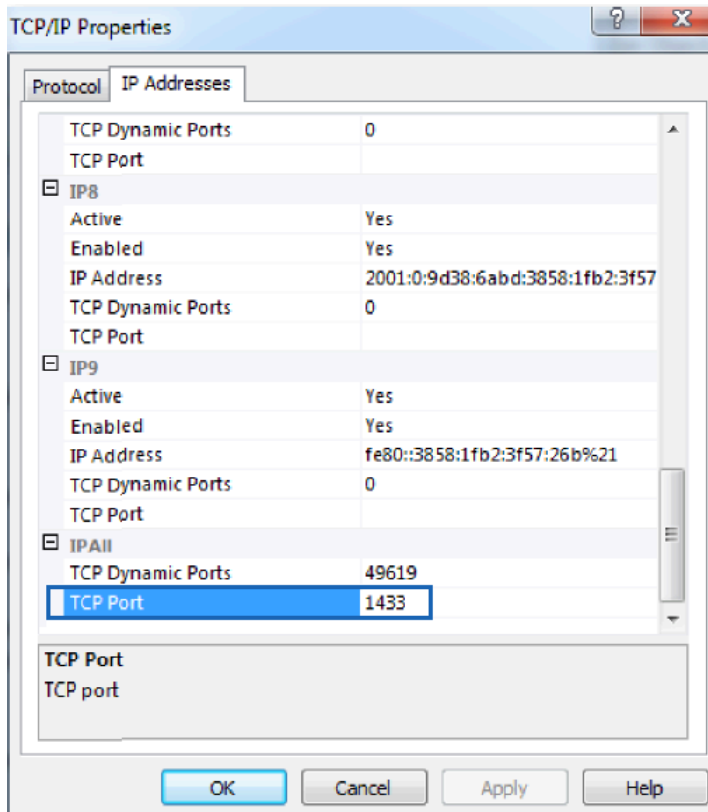
3. Right-click **TCP/IP** and select **Properties**.



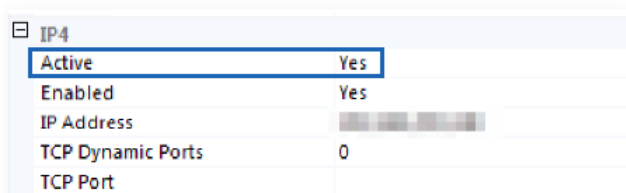
4. Select the **Protocol** tab, set the property **Enabled** to **Yes** and confirm the changes.



5. Select the **IP Addresses** tab, scroll down to section **IPALL** and enter the value 1433 for the **TCP Port**.



6. Make sure to activate the IP Addresses the server should listen on by setting **Active** to **Yes**: It's possible to activate all first and limit them later to the real needs.



Apply changes with **OK**.

7. Stop and start service.

### Create an empty or new database instance

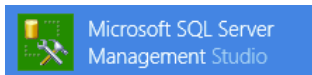
For this guide **signdoc** will be used.

A new database instance can be created using the SQL Server Management Studio (SSMS).

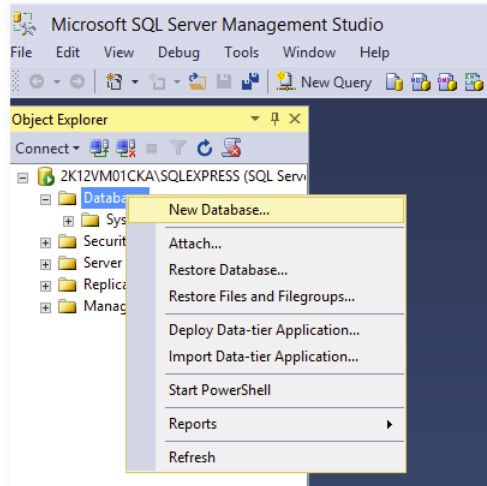
Example

1. Download SSMS from Microsoft and install the program with administrator rights.

2. Start the SSMS.

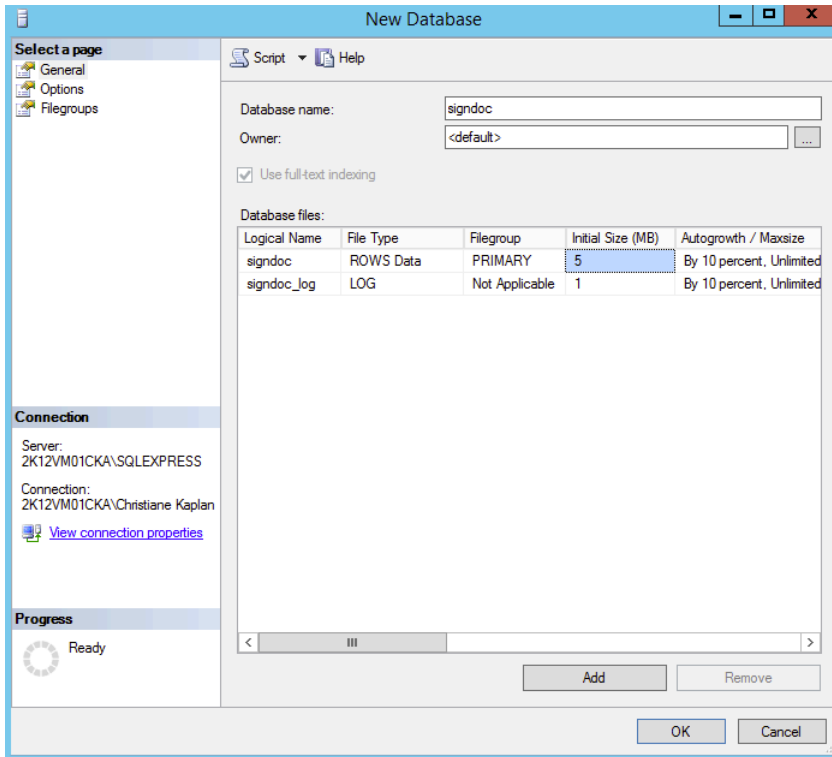


3. Connect to the server.
4. To create a new database, right-click **Database** and select **New Database**.

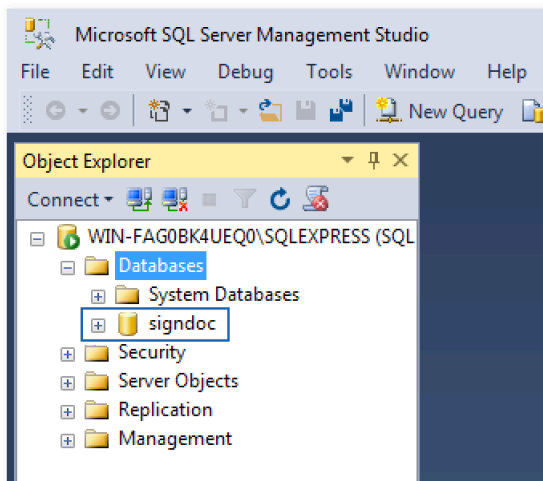


The **New Database** window appears.

- On the **General** page, enter a name for the database. For this guide **signdoc** is used as database name.



- After confirming the input, the new database **signdoc** is listed under **Databases**.



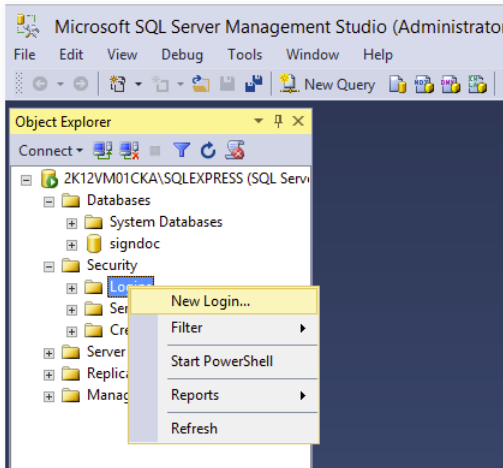
### Create a new database user

The new database user is a member of the role **db\_owner** of the **signdoc** database and uses "SQL Server authentication".

For this guide **signdoc** will be used also as user name. The new user requires also a password – make sure to unselect "".

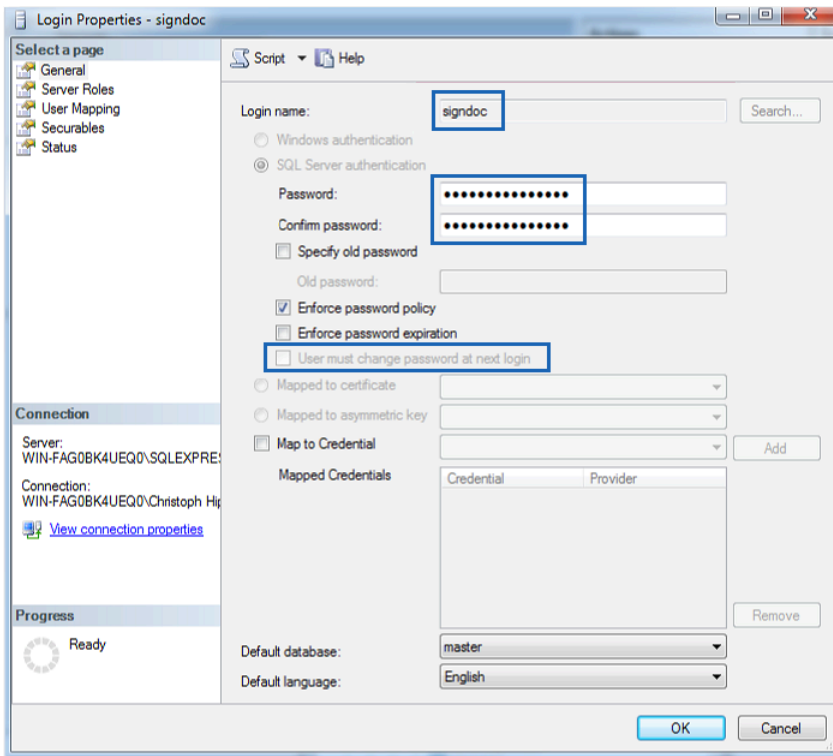
For this guide **2beChanged!** will be used.

1. To create a new login, click **Security**, then right-click **Logins** and select **New Login**.



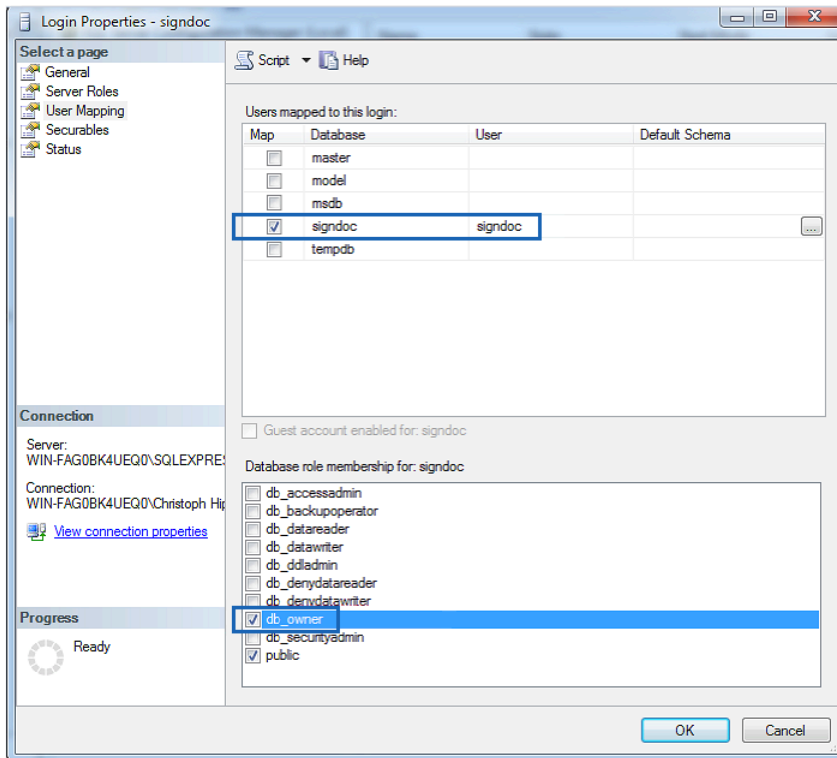
The **Login Properties** window appears.

2. Go to **General** pane and enter the according information.



**i** Ensure that the check box is not selected for "User must change password at next login".

3. Go to **User Mapping** pane and select database **signdoc** and role membership.



## Administration Center

Open the **Administration Center** to create accounts and users:

`http[s]://<server>/cirrus/admin-center`

For this guide this is (see `service_configuration.properties` in chapter [Production setup](#)):

`http://<SERVICE_EXTERNAL_HOST_URL><SERVICE_HTTP_PORT>/cirrus/admin-center`

**Example**

When `<SERVICE_EXTERNAL_HOST_URL>` is `localhost` and `SERVICE_HTTP_PORT` is `6611` then the URL is

`http://localhost:6611/cirrus/admin-center`



Kofax SignDoc®  
e-sign with Kofax

Sign in to Kofax SignDoc Administration Center

User id or email address \* ⓘ  
ksdadmin

Password \* [Forgot password?](#)  
.....  Show

**Sign in**

[Sign in to Kofax SignDoc](#)

To create accounts and users see [Related documentation](#), *Kofax SignDoc Standard Administration Center Help*.

## Deployment on Linux

If SignDoc is to be deployed under Linux, it is recommended to use Docker. SignDoc provides several sample Dockerfiles (in the `docker` directory). These Dockerfiles contain documentation and describe the basic installation procedure for a Linux environment. These files can also be used and amended/adapted for deployment, if required/desired.

## Installation in Docker environment

SignDoc Standard can be run in Docker Linux or Windows container.

Additional information can be found in the `docker` directory of the SignDoc Standard ZIP archive.

### Content of the docker directory

The `docker` directory in the SignDoc Standard ZIP archive consists of tools and sub directories with example configurations.

#### Tools

- **build\_images.cmd** builds the required Windows Docker images.
- **build\_images.sh** builds the required Linux Docker images.
- **start\_server.ps1** starts SignDoc Standard as service inside Docker Windows container.
- **start\_server.sh** starts SignDoc Standard inside Docker Linux container.

## Directories

- **01\_base\_image** is an example of base configuration for running SignDoc Standard. Also contains SignDoc Standard Docker files that are used in other samples.
- **02\_high\_availability** is an example of how to run SignDoc Standard behind a load balancing reverse proxy. It is possible to scale SignDoc Standard instances transparently and independently of other services without breaking the client sessions.
- **03\_high\_availability\_ssl** is an example of how to run SignDoc Standard in HTTPS configuration. This is an extension of 02\_high\_availability example.
- **04\_env\_variables** is an example of SignDoc Standard configuration using environment variables. There is no need to rebuild the Docker image to apply new SignDoc Standard service configuration properties. Windows Docker containers only.
- **mssql\_database** contains everything you need to build Docker image of Microsoft SQL Server with prepared SignDoc Standard database.
- **plugins** contains extension files. You can implement your own logic for starting SignDoc Standard inside the Docker container. In this case, all additional files must be put in this directory. For instance, 04\_env\_variables example has additional configuration checks for Mail and SQL servers. The PowerShell scripts used are located in this directory. Windows Docker containers only.

## How to run

There are different possibilities to run SignDoc Standard in Docker container. Follow the recommendation on the official Docker website and your preferred orchestration tools.

As a basic example, it can be run with Docker CLI:

```
docker run -init-d -p 8080:8080 -name=signdoc_standard_basicsigndoc_standard
```

**i** Due to a bug in Docker Build Kit, it might be required to do 'docker login' before building the SignDoc images. See also: <https://github.com/moby/buildkit/issues/1271>

## Chapter 3

# Advanced installation

## Hardening a SignDoc installation

Hardening a SignDoc Setup means to apply best practices and security measures to a SignDoc installation for production usage.

### Reverse Proxy

It is recommended to run SignDoc behind a reverse proxy, since this provides an additional abstraction layer between the application server and the users. The reverse proxy

- can act as TLS/SSL endpoint for the system, what simplifies deployment and maintenance.
- is usually capable of load-balancing requests to multiple SignDoc installations what improves the high-availability of an installation.
- can be configured to set specific HTTP header attributes to minimize common known attack vectors like e.g. XSS.

### HTTP Headers

Additional HTTP response headers can be set or added using the configuration options `security.http.response.headers.set` or `security.http.response.headers.add` of the Administration Center.

Make sure to set these HTTP headers concerning security:

```
X-Frame-Options "SAMEORIGIN";  
X-Content-Type-Options "nosniff";  
X-XSS-Protection "1; mode=block";
```

### Turn off server tokens

Application servers and reverse proxies often announce their identity and version via server tokens in the HTTP response. This information is superfluous and should be avoided.

### Block too large uploads

The reverse proxy should reject upload requests that are too big. But it must also allow uploads that are justified. Some reverse proxies have a too small upload limit that must be increased. The limit must be at least 33%, better 50%, larger than a single to be uploaded document can be in size.

Example: If the largest acceptable document size is 60 MByte, the upload limit (i.e. maximum body size) 90 MByte (+50%) would be a safe limit.

### TLS/SSL

When the reverse proxy acts as TLS endpoint, it must reject unsafe or outdated SLL protocols or cipher versions. Besides this, the HTTP protocol should be disabled.

### Block access to URIs

Some resources are not needed for a typical production environment and can be safely blocked:

- /cirrus/swagger
- /cirrus/api-docs
- /cirrus/static/swagger-ui

### Example configurations

The following files show an example configuration for a Nginx web server.

File: my\_proxy.conf

```
server_tokens off;
client_max_body_size 100m;
```

File: default

```
add_header X-Frame-Options "SAMEORIGIN";
add_header X-Content-Type-Options "nosniff";
add_header X-XSS-Protection "1; mode=block";

location cirrus/swagger {
    return 404;
}
location cirrus/api-docs {
    return 404;
}
location cirrus/static/swagger-ui {
    return 404;
}
location /cirrus/swagger {
    return 404;
}
location /cirrus/api-docs {
    return 404;
}
location /cirrus/static/swagger-ui {
    return 404;
}
```

## Content Security Policy (CSP)

The following CSP related HTTP headers can be set in a reverse proxy to enable Content Security Policy for the Manage and Signing Client.

**Manage Client and Signing Client combined** (URI: <SignDoc Standard content, usually /cirrus>)

If you want to tailor the CSP for the specific clients one can do it like this:

```
Content-Security-Policy: default-src 'none'; style-src 'self' 'unsafe-inline'; script-
src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data: blob:; frame-src 'self';
connect-src 'self' localhost:6613; font-src 'self'; media-src 'self'; object-src
'none'; form-action 'self'
```

### Manage Client (URI: /cirrus/static/mc)

```
Content-Security-Policy: default-src 'none'; style-src 'self' 'unsafe-inline'; script-  
src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data: blob:; frame-src 'self';  
connect-src 'self'; font-src 'self'; media-src 'self'; object-src 'none'; form-action  
'self'
```

### Signing Client (URI: /cirrus/static/sc)

```
Content-Security-Policy: default-src 'none'; style-src 'self' 'unsafe-inline'; script-  
src 'self' 'unsafe-inline' 'unsafe-eval'; img-src 'self' data: blob:; frame-src 'self';  
connect-src 'self' localhost:6613; font-src 'self'; media-src 'self'; object-src  
'none'; form-action 'self'
```

## Authentication LDAP

❗ SignDoc Standard before version 2.1.0 was mainly configured with the configuration file `cirrus.properties`. This file moved to `INSTALLDIR\_conf_templates\cirrus.properties` with version 2.1.0.

Since SignDoc Standard 2.1.0, it is highly recommended to use the file `INSTALLDIR\_service_configuration.properties` (instead of `cirrus.properties`) whenever it is required to configure SignDoc with a configuration file. Configurations set in this file are applied as Java system property and have therefore highest precedence.

### General

This section describes the specification of the basic authentication via LDAP provided with SignDoc Standard 3.0.0 or newer.

### Prerequisites

The LDAP support in SignDoc Standard is not usable and not supported in a multi-tenant environment. SignDoc Standard maps LDAP user entries to SignDoc Standard users by the unique email address. A SignDoc Standard multi-tenant installation requires email addresses only to be unique within a single account.

The user's id is defined by the setting `ldap.user.mail.attr`. It must represent the email address of the user.

❗ This is not a standard LDAP attribute and may have to be added by a system administrator.

### Activating LDAP

LDAP support is activated by setting the property `authentication.provider` to the value `LDAP,CIRRUS`.

❗ Activate LDAP only after you have created the single account.

### Auto creating a user

If a user logs in and a user with the mail address received from LDAP does not exist, a user is created automatically in SignDoc Standard. SignDoc Standard maps LDAP attributes to SignDoc Standard user attributes. Each name of an LDAP attribute has a default value but can be customized by a SignDoc Standard property:

SignDoc Standard property	Default value	Mapped to this SignDoc Standard user attribute	Constraints
ldap.user.name.attr	cn	user name	
ldap.user.mail.attr	mail	OID	(mandatory setting) Must not already exist as a SignDoc Standard user. The email address must match this regular expression: <code>^[A-z0-9\.\_%+\-]+@[A-z0-9\.\-]+</code>
ldap.user.uid.attr	uid		(optional setting) Must not already exist and match the SignDoc Standard validation rule for OID (regex <code>^[a-zA-Z0-9_\-]+</code> )

**i** If one of the above constraints are violated SignDoc Standard will report an error and LDAP integration will not work reliably.

All values can be customized by the `INSTALLDIR\service_configuration.properties` file:

- **authentication.provider** (string): Activates LDAP support. Must be 'LDAP,CIRRUS'.
- **ldap.url** (string): The URL to connect to an LDAP server.

**!** It is strongly recommended to use the ldaps protocol because passwords are send in plain text over the network. A suitable certificate must be installed at the server in that case.

Example

```
ldap://ad.kofax.com:389/dc=kofax,dc=de
```

- **ldap.manager.dn** (string): The manager DN. If your LDAP implementation does not allow anonymous access a suitable user and password must be defined here.

Example

```
uid=admin,ou=system
```

- **ldap.manager.password** (string): The manager password.

Example

```
ldap.manager.password=secret
```

Omit manager dn and password for anonymous access.

- **ldap.userdn.patterns** (string): The value is a list of distinguished names (DN) separated by a colon.

**i** Because the field delimiter is the colon (':'), a DN containing colon(s) must be double-quoted. And a double-quoted DN must escape any double-quote sign with the escape character '\', should it be present in the DN.

Example

```
uid={0},ou=Users
```

The key '{0}' will be substituted with the login name.

- **ldap.user.search.base** (string): The base DN for starting a search.

Example

```
dc=kofax,dc=de
```

- **ldap.user.search.filter** (string): A filter for the search (see RFC 2254)

Example

```
(cn=Babs Jensen)
```

- **ldap.user.name.attr** (string): The LDAP attribute which maps to a SignDoc Standard user name. Default: cn

Example

```
ldap.user.name.attr=cn
```

- **ldap.user.mail.attr** (string):

The LDAP attribute which maps to a SignDoc Standard user email. Default: mail

Example

```
ldap.user.mail.attr=mail
```

**i** Additional "Brute Force Authentication Prevention" is not implemented if LDAP Authentication is configured.

## Chapter 4

# SignDoc Authentication Module (SAM)

## Purpose

The SignDoc Authentication Module (SAM) is required to support Single Sign-on (SSO) scenarios with SignDoc Standard (SDS). This version implements authentication using SAML2. Supported IDPs are okta (<http://www.okta.com>) and Active Directory Federation Services (ADFS).

## Requirements

- From the IDP: IDP\_METATDATA can be a URL or a local file
- From the SP: SP\_ENTITY\_ID and SSO\_URL

See also [Identity provider related configuration](#).

## Installation

- Unpack the SDS zip and go to the `modules` directory.
- Unpack the embedded `signdoc-auth-module-service-*-windows-zip` in a separate directory (`SAM_DIR`).
- Configure the service. See [Configuration of SAM](#).
- Doubleclick on `SAM_DIR\service_up.cmd`  
After a few seconds the Windows Service "SignDoc Authentication Module" should appear in the Windows Service Panel.
- Configure SSO configuration options in SDS (optional) [SDS SSO configuration options](#).

## Usage with browser

The SP endpoint of SAM is the / (root) context

For example



`http://localhost:6612`

If a browser opens this URL, SAM will try to authenticate the user as defined by the configuration.

Query parameters (optional)

- `return_url`: SAM will return to this URL, after a successful user authentication.

## Configuration of SAM

### Service configuration

The SAM Windows service is configured using the XML file `SAML_DIR\SignDocAuthModule.xml`

**i** Because of XML restrictions, it is not possible to put 2 consecutive hyphen ("-") characters in an XML comment, we are using the - escape sequence to substitute a "-" character. This applies especially to the content of the argument elements that usually start with a double hyphen. This escape character can optionally be replaced for better readability with a simple hyphen ("-") character when uncommented.

#### General service configuration options

- **service.context.url** (required)  
`<argument>--service.context.url=http://localhost:6612</argument>`  
Set the value to the URL the SAM service can be accessed externally. One must be especially careful, if the SAM service is hosted behind a reverse proxy that does the SSL offloading.  
Default value: `http://localhost:6612`
- **server.port** (required)  
`<argument>--server.port=6612</argument>`  
Defines the TCP port the Service will use to listen on requests. Make sure that the port is not used by other services on the same system.  
Default value: `6612`
- **cirrus.url** (required)  
`<argument>--cirrus.url=http://localhost:6611/cirrus</argument>`  
The URL to the SDS cirrus context. This is usually `SERVICE_EXTERNAL_HOST_URL/cirrus`. `SERVICE_EXTERNAL_HOST_URL` is the SDS config property of `service_configuration.properties`. See also SDS configuration documentation.  
Default value: `http://localhost:6611/cirrus`
- **cirrus.sso.default.account** (optional)  
`<argument>--cirrus.sso.default.account=REPLACE_WITH_REAL_VALUE</argument>`  
SAM can be configured to use a default account if the IDP user has no account attribute set - what is probably the normal case. It should be noted that this setting is optional, since the SDS server setting `cirrus.sso.create.user.account` fulfills the same purpose.  
Default value: `REPLACE_WITH_REAL_VALUE`

## Identity provider related configuration

SignDoc users are identified via email address and SignDoc account. Therefore, the IDP SAML Name ID Format should be set to EmailAddress.

**Disclaimer:** This section describes the configurations that have to be done in the IDP application. The way of how this is to be done, might be changed by the IDP. Especially the screen shots might be out of date, if the IDP changes its user interface.

### SAML2 specific configuration options

This section describes the common IDP SAML2 settings that are required to enable the functionality.

- **saml.idp.metadata.urlOrFile** (required)

```
<argument>--saml.idp.metadata.urlOrFile=%BASE%\idp-metadata.xml</argument>
```

This property provides access to the IPD\_METADATA. There are 2 options to set the IPD\_METADATA: URL or local file.

- URL: The IDP\_METADATA must be accessible via simple GET request. Advantage: The IDP\_METADATA can be changed by the IDP without having to reconfigure SAM.
- Local file: The IDP\_METADATA (usually XML) is provided in a local file. The absolute filepath must be used. Example: C:\

Default value: %BASE%\idp-metadata.xml (i.e. SAML\_DIR\idp-metadata.xml)

**i** The default value is a configuration for an internal SignDoc Demo Service at okta. This service can not be used for any practical purposes, but can provide a quick check that the basic installation works.

- **saml.spEntityId** (required)

```
<argument>--saml.spEntityId=signdoc-demo-saml</argument>
```

This property defines the unique ID of the SAML2 service provider. It can be freely chosen, but must be compatible with the URI specification (<https://tools.ietf.org/html/rfc3986>).

Default value: signdoc-demo-saml

**i** The default value is a configuration for an internal SignDoc Demo Service at okta. This service can not be used for any practical purposes, but can provide a quick check that the basic installation works.

### User attribute mapping

It is possible to map SAML user attributes to SignDoc user attributes. This is especially useful for the user name when a user is automatically created in SignDoc.

#### saml.user.attribute.key.display\_name

Optional setting. Is useful when users should be auto created. display\_name (i.e. the full name) has precedence over first and last name. Default: signdoc\_name or name (signdoc\_name has precedence)

Azure AD Example:

```
<argument>--saml.user.attribute.key.display_name=http://schemas.microsoft.com/identity/claims/displayname</argument>
```

**saml.user.attribute.key.first\_name & saml.user.attribute.key.last\_name**

Optional setting. Only useful when `saml.user.attribute.key.display_name` is not set. `first_name` and `last_name` - will only be considered, if `saml.user.attribute.key.display_name` is not set.

Defaults...

first name: `signdoc_firstname` or `firstname` (`signdoc_firstname` has precedence) last name: `signdoc_lastname` or `lastname` (`signdoc_lastname` has precedence)

Azure AD Example:

```
<argument>--saml.user.attribute.key.first_name=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname</argument>
```

```
<argument>--saml.user.attribute.key.last_name=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname</argument>
```

**saml.user.attribute.key.email**

Optional setting. Usually not needed. The user's email attribute. Usually this is not required, since the principal is usually the users email but pointing this setting to a custom attribute would allow the user to use different email in SignDoc than in the IDP.

Default: `signdoc_email` or `email` (`signdoc_email` has precedence)

Azure AD Example:

```
<argument>--saml.user.attribute.key.email=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name</argument>
```

**saml.user.attribute.key.login**

Optional setting. Usually not needed. If the user wants to login with the SignDoc user id, the attribute name can be specified here.

Default: `signdoc_login` or `login` (`signdoc_login` has precedence)

Azure AD Example:

```
<argument>--saml.user.attribute.key.login=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/signdoc_login</argument>
```

**saml.user.attribute.key.account**

Optional setting. Usually not needed. If the user wants to login always into a specific account, it can be specified here.

Default: `signdoc_account` or `account` (`signdoc_account` has precedence)

Azure AD Example:

```
<argument>--saml.user.attribute.key.account=http://schemas.xmlsoap.org/ws/2005/05/identity/claims/signdoc_account</argument>
```

## ADFS

To configure ADFS to work with SAM, the MMC snap-in for ADFS should be added. Add a Relying Party Trust with the following properties:

- In the Identifiers tab, add a Relying Party Identifier that will match the SP\_ENTITY\_ID.
- In the Endpoints tab, add the SSO\_URL that will process SAML responses to the list, using POST for the Binding value.

To obtain the metadata provider XML, load this url in your browser:

```
https://myserver.domain.com/FederationMetadata/2007-06/FederationMetadata.xml
```

## Okta

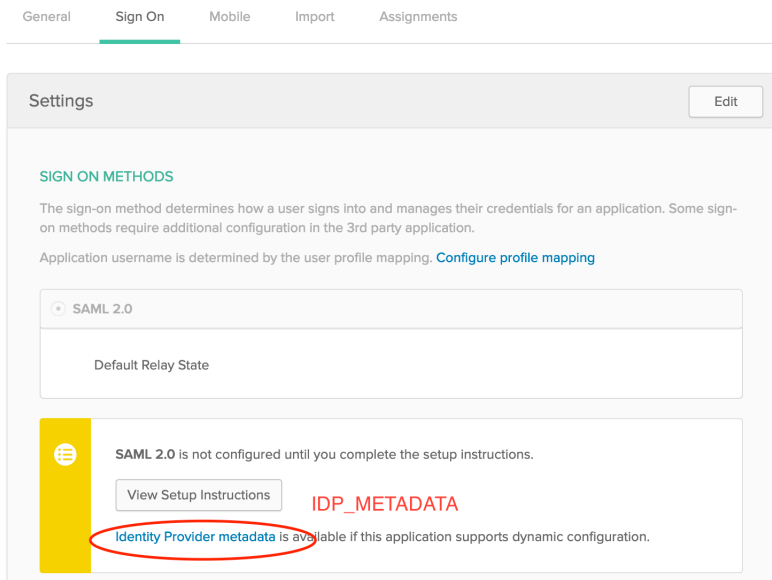
To configure Okta to work with SAM, create an SAML 2.0 application (one must currently use the "Classic UI") with the following settings:

- The Single sign on URL should be the URL that processes SAML responses (e.g. assertions).
- The Audience URI is set to the SP\_ENTITY\_ID.

SSO\_URL and SP\_ENTITY\_ID in okta



IDP\_METADATA in okta



## SDS SSO configuration options

There are several configuration options in SDS that can modify the SSO behavior of SDS. See also the SDS documentation for further explanation.

- **cirrus.sso.auth.module.url**

This is the same as `service.context.url`. Setting this config option enables `SP_INITIATED_LOGIN` from SDS. SDS will display a SSO login button on the login screen, if this information is set.

Examples: <http://localhost:6612>, <https://sso.mysigndocserver.com>

Default value: not set

- **cirrus.sso.autologin**

If this setting is ON, SDS will try to do an SSO login automatically, whenever possible.

Default value: OFF

- **cirrus.sso.create.user**

If this setting is ON, SDS will try to create new users in SDS if a matching user account is missing for an authenticated IDP user in SDS. See also `cirrus.sso.create.user.account`.

Default value: ON

- **cirrus.sso.create.user.account**

If this setting should be set, to let SDS know in what account new user profiles should be added in SDS. See also `cirrus.sso.create.user`. Alternatively, it is possible to set the SAM configuration `cirrus.sso.default.account` (optional). See [Service configuration](#).

Default value: not set

- **cirrus.sso.sanitize.userid**

If this setting is ON, SDS will sanitize unusable user ids (e.g. email addresses) that are provided via IDP metadata. See [Supported IDP user metadata](#).

## Automatic user creation

New SDS users can be created by SAM, if the feature is enabled and a SignDoc user is missing for the corresponding authenticated IDP user. SignDoc users will always be searched via their email address not user id. The IDP SAML Name ID Format should be set to EmailAddress.

The account information can be provided by different means, as explained in:

- `cirrus.sso.default.account` (optional). See [Configuration of SAM](#).
- `cirrus.sso.create.user.account`. See [SDS SSO configuration options](#).
- [Supported IDP user metadata](#)



- Existing SignDoc users will never be modified by SAM
- To create a user, SAM needs at least the email address of the user.
- If only the email address is available, the SignDoc user will be created with predefined attributes
  - name: "unknown"
  - userid: random UUID

## Supported IDP user metadata

Mapping of IDP user attributes to SDS user attributes.

IDP user attribute	SDS user attribute	Remarks
implicit userid or signdoc_email or email	email	Provided by IDP.
signdoc_name or name	name	Alternatively first- and lastname can be used. Has priority over first- and lastname settings. If not set the name "unknown" is used.
signdoc_firstname, firstname, signdoc_lastname, lastname	name	First- and lastname will be concatenated with a whitespace
signdoc_login, login	userid	Can be automatically sanitized. If not set, a random value is used.
signdoc_account, account	accountid	This information defines the SDS account the user belongs to. Alternatively this can be defined by the corresponding SAM ( <code>cirrus.sso.default.account</code> (optional) SDS ( <code>cirrus.sso.create.user.account</code> ) configurations. See <a href="#">Automatic user creation</a> .

## SSL configuration

The SAM Service can be also be run with an SSL configuration by setting some configuration values. To activate the setting they must be specified like the other configuration options above.

Example settings for using a PKCS#12 cert stores:

```
<argument>--server.ssl.key-store-type=PKCS12</argument>
```

```
<argument>--server.ssl.key-store=file:path_to_my_keystore.pfx</argument>
```

```
<argument>--server.ssl.key-store-password=2beChanged!</argument> <argument>--server.ssl.key-alias=1</argument>
```

## Stop / Start / Uninstall the SAM Service

SAM is provided as a standard Windows Service.

### Starting the SAM service

Adds the Windows Service "SignDoc Authentication Module" from the Windows Service panel.

- Doubleclick on: SAM\_DIR\service\_up.cmd

### Stopping the SAM service

Removes the Windows Service "SignDoc Authentication Module" from the Windows Service panel.

- Doubleclick on: SAM\_DIR\service\_remove.cmd

### Uninstalling the SAM service

- Stop the service. See [Stopping the SAM service](#).
- Delete SAM\_DIR.

## Glossary

### **ADFS**

Active Directory Federation Services (from Microsoft)

### **IDP**

Identity provider. A software service that can authenticate users.

**IDP\_METADATA**

Metadata that establishes the bond between SP and IDP. SAM accepts local files or a remote URL to retrieve the IDP Metadata.

Related SAM setting: saml.spEntityId

**SAM**

SignDoc Authentication Module (this Software package)

**SAM-DIR**

The directory where SAM is installed. Example: C:\Program Files\SignDoc Authentication Module

Example: C:\Program Files\SignDoc Authentication Module

**SDS**

SignDoc Standard (2.2.1 or newer)

**SP**

Service provider. A service that grants IDP authenticated users access to a system. SAM is a service provider.

**SP\_ENTITY\_ID**

A unique ID (URI syntax) that identified the SAM service.

Related SAM setting: saml.idp.metadata.urlOrFile

**SP\_INITIATED\_LOGIN**

The IDP authentication session is started by the SP

**SSO**

Single Sign-on. An environment that allows a user to authenticate against a central instance (IDP). By doing this the user gets implicit authorization services supporting Single Sign-on.

**SSO\_URL**

This is the URL that SAM uses to process SAML2 assertions.

Definition: \${service.context.url}/saml/SSO

Example: <http://localhost:6612/saml/SSO>



## Chapter 5

# Uninstall SignDoc Standard

To uninstall SignDoc Standard follow these steps:

1. Double-click `INSTALLDIR\service_remove.cmd`. Confirm the dialog boxes and wait until the Windows service "SignDoc Standard" is stopped and deregistered.
2. Delete the installation directory `INSTALLDIR`.

## Chapter 6

# Upgrade SignDoc Standard

SignDoc can be upgraded from any previous version. If nothing else is specified in the version list below, the following generic upgrade procedure can be applied:

- Stop SignDoc using `service_remove.cmd`. See [Content of the SignDoc Standard ZIP archive](#), section "Tools".
- Make a backup of the database or create a snapshot of the database that can be restored, in case the upgrade fails.
- Install the new SignDoc version as described in [Quickstart](#).
- Apply the existing old SignDoc configuration to the new SignDoc installation. I.e. apply any existing configuration from `service_configuration.properties` (and possibly other configuration files) to the new installation. This usually concerns, among other things: database connection, SMTP server, `SERVICE_EXTERNAL_HOST_URL`, and so on.
- Make sure database migrations are enabled (this is the default). See [Advanced configuration](#), section "Control database migrations".
- Start the new SignDoc version using `service_up.cmd`. See [Content of the SignDoc Standard ZIP archive](#), section "Tools".

**i** After the system is upgraded, it is no longer possible or supported to connect the old SignDoc installation to the upgraded/migrated database.

## Upgrade from SignDoc Standard 2.2.1

**i** Configuration changes are required for SignDoc 3.0.0 or newer.

If you are upgrading an existing SignDoc 2.2.1 installation (version  $\leq 2.2.1.2.0.63$ ) there are a few configuration options that must be slightly changed to work again.

File: `service_configuration.properties`

- The `jdbc.password` configuration setting must not contain any of these 3 characters: `<` `>` `&`
- The `jdbc.url` configuration setting **MUST NO LONGER** be enclosed in quotes as it was true for SignDoc up to 2.2.1.2.0.63. Until now, the `jdbc.url` value had to be enclosed in single quotes, if there were spaces in `jdbc.url`. If this should be the case for the installation to upgrade, the single quotes must now be removed.

Example

Invalid with SignDoc 3.0.0 or newer (was valid up to version 2.2.1.2.0.63):

```
jdbc.url='jdbc:h2:${SIGNDOC_HOME}/db/  
signdoc_database;MVCC=TRUE;DB_CLOSE_DELAY=-1;INIT=SET COLLATION ENGLISH  
STRENGTH SECONDARY'
```

Valid with SignDoc 3.0.0 or newer:

```
jdbc.url=jdbc:h2:${SIGNDOC_HOME}/db/  
signdoc_database;MVCC=TRUE;DB_CLOSE_DELAY=-1;INIT=SET COLLATION ENGLISH  
STRENGTH SECONDARY
```

## Upgrade from SignDoc Standard 1.3.1 or earlier versions

To upgrade an existing SignDoc Standard 1.3.1 or earlier version, the following steps need to be performed:

1. Stop and disable automatic restart for all existing SignDoc Standard and/or SignDoc Web instances older than SignDoc 2.1.0.  
Don't stop the SignDoc Standard database.  
If applicable: Remove any global or system PATH setting, that contains the existing SignDoc Standard instance.
2. Install SignDoc 2.1.0 as described in [Quickstart](#) above.
3. Make sure database migrations are enabled (this is the default). See [Control database migrations](#).
4. Configure SignDoc 2.1.0 for production as described in [Production setup](#). It is recommended to apply the configuration fresh.
  - If **KTA integration** is used, the existing configuration section of `cirrus.properties` can be copied to the new installation. See [KTA integration](#).
  - If **LDAP authentication** is used, the existing configuration section of `cirrus.properties` can be copied to the new installation. See [LDAP integration](#).
5. Start the new SignDoc version using `service_up.cmd`. See [Content of the SignDoc Standard ZIP archive](#), section "Tools".
6. Open `http://<your_server>:<port>/cirrus/client` and log in.

## Upgrade troubleshooting

It should be sufficient to make sure that there are no PATH entries that point to an old SignDoc Standard installation. If an installation fails, the following should be double-checked:

- Double-check [General prerequisites](#).
- The installation should be done on a supported Windows Server operating system such as Windows Server 2012 R2.
- There should be no Windows PATH (system or user) entry (i.e. Environment variable) pointing to an old SignDoc Standard installation.
- There should be no CIRRUS\_HOME Environment variable set (system or user).

- There should be no SDWEB\_HOME Environment variable set (system or user).

## Chapter 7

# Database migration

This chapter describes the database migration mechanism used by SignDoc starting with version 1.1.0.1.

## Overview

Any product that uses a schema based database and gets past its first version faces the problem of tackling database changes while migrating from one version to another. This includes changes to the database schema, like adding a new column, moving data from a location to another etc. Not only the database has to be adapted to the new schema, also the existing data has to be migrated to fit it.

Database migrations standardize the way this is done, keeping track of the versions that have been applied to the data.

## Flyway

SignDoc uses Flyway to standardize database migration scripts. You can read about flyway at <https://flywaydb.org>. In short (check the Flyway documentation at the website), Flyway uses migrations that are named to a specific schema, containing the version number and description in the file name. It also keeps track of the version the database currently has and all applied changes by creating a database table named `schema_version` and recording all migrations it has done.

Since the migration scripts (or migration Java classes) are part of the product, one can always tell what state the database is in and what changes still need to be applied.

Flyway migrations can either be run from the command line, or be integrated into the product itself. When the application starts, it checks the database version and executes any outstanding migrations in the order of their version number, thus bringing the database up to date.

## Flyway use in SignDoc

### Integration and configuration

Flyway is built into SignDoc, starting with release 1.1.0.1. Each time SignDoc is started, it will check if the database is up to date and can run any outstanding database migrations.

Migrations can be configured to run either automatically or manually. Running migrations automatically can be convenient for a classic deployment with a single server, or a test environment

with frequent changes. In a cloud environment running multiple servers a manual invocation of the database migration is recommended. This allows for a better control of the process, including the necessary backup and QA steps.

The way migrations are run is controlled by the `cirrus.migrations.enabled` property:

- **true**  
Enables automatic migrations. The application will compare the version currently stored in the database and attempt to migrate it to the one used by the application. It will apply all necessary steps in sequence, without needing confirmation.
- **false**  
Disables automatic migrations. The application will compare the version currently stored in the database and refuse to start if it does not match the one used by the application (will throw an exception). The migration step has to be run manually using the command line tool or Docker container.

❗ Regardless of the setting used it is strongly recommended to perform a database backup before attempting to migrate the database. Due to the nature of some migration steps and the fact that multiple migration steps are applied during a version update, a database rollback is not possible.

## Version numbers

Following schema is used for SignDoc migration version numbers:

```
<ver major>.<ver minor>.<release>.<bugfix>.<hotfix>_<migration sequence>
```

The migration also includes a short textual description. A sample output of the version information is shown below:

Version	Description	Installed on	State
1.1.0.0.0.0	Baseline	2015-11-24 17:56:54	Success
1.1.0.1.0.1	Upgrade	2015-11-24 17:56:54	Success
1.2.0.0.0.1	Release upgrade	2015-11-24 17:56:54	Success
1.2.0.0.0.2	NewAccountLicenseHandling	2015-11-24 17:56:54	Success
1.2.0.0.0.3	Add package counter	2015-11-24 17:56:54	Success
1.2.0.0.0.4	AddDnsLabel	2015-11-24 17:56:54	Success
1.2.0.0.0.5	AddTimezoneToAccount	2015-11-24 17:56:54	Success
1.2.0.0.0.6	RemoveUserStateINACTIVE	2015-11-24 17:56:54	Success
1.2.0.0.0.7	AddContactInfoToAccount	2015-11-24 17:56:54	Success
1.2.0.0.0.8	HandleKeysTable	2015-11-24 17:56:55	Success
1.2.0.0.0.9	AddSignatureSettings	2015-11-24 17:56:55	Success
1.2.0.0.0.10	AddAccountPersonalization	2015-11-24 17:56:55	Success
1.2.0.0.0.11	UserRolesNotNullable	2015-11-24 17:56:55	Success
1.2.0.0.0.12	DropObsoleteTimestamps	2015-11-24 17:56:55	Success
1.2.0.0.0.13	Add document counter	2015-11-24 17:56:55	Success
1.2.0.0.0.14	RemoveUnusedUserStates	2015-11-24 17:56:55	Success
1.2.0.0.0.15	NotDeleteAuditTrails	2015-11-24 17:56:55	Success

## Classic deployment

The classic deployment describes the installation of SignDoc in a servlet container (Tomcat), without the use of Docker containers.

### Automatic migration

To enable automatic migrations you have to set `cirrus.migrations.enabled` to `true` in `cirrus.properties`. SignDoc will check the database version and run pending migrations automatically during system start.

### Manual migration

If is set to false, SignDoc will only verify if the database has been updated to the current version. Migrations have to be run manually using the Flyway command line tool.

### Command line tool

The Flyway command line tool can be used to query the database version information, check outstanding migrations, perform migrations and clean or repair the database. All necessary information, like database driver, URL, login info, etc. can be given as arguments. A more convenient way is to store them in a configuration file. A sample configuration (`flyway.conf`) is shown below:

```
# Database URL
flyway.url=jdbc:jtds:sqlserver:// servername/database_name
# User to use to connect to the database (default: <<null>>)
flyway.user= username
# Password to use to connect to the database (default: <<null>>)
flyway.password= password

# Locations starting with filesystem: point to a directory on the filesystem and may
# only contain sql migrations.
flyway.locations=classpath:sql/migration/
net_sourceforge_jtds_jdbc_Driver,classpath:sql/migration/common,classpath:de/softpro/
cirrus/db/migrations

# Comma-separated list of directories containing JDBC drivers and Java-based
# migrations. (default: <INSTALL-DIR>/jars)
# flyway.jarDirs= <path to flyway> /flyway-3.2.1/jars, <path to cirrus-db-
<versionnumber> directory>
flyway.jarDirs= /flyway/jars,/tomcat/webapps/cirrus/WEB-INF/lib

# The version to tag an existing schema with when executing baseline. (default: 1)
flyway.baselineVersion=1.1.0.0_0
# Whether to automatically call baseline when migrate is executed against a non-empty
# schema with no metadata table.
flyway.baselineOnMigrate=true
```

Fields that have to be configured are marked in bold.

You can check the status of the database using the command:

```
flyway info
```

Migrations can be applied using:

```
flyway migrate
```

The Flyway command line and the Flyway web page provide a reference on the available commands and options.

If Flyway is not included in the deliverables, it can be downloaded and installed from <https://flywaydb.org>.

## Docker deployment

In case of a Docker deployment, the complete application environment is packaged inside a Docker container. This includes the migration tool. The default migration setting for a Docker container is false.

### Automatic migration

If automatic migrations are desired the migration property can be set to true by providing the environment variable

```
SPEC_CIRRUS_MIGRATIONS=true
```

at container start:

```
docker run -e SPEC_CIRRUS_MIGRATIONS=true ... softpro/signdoc-standard:<version>
```

### Manual migration

Running migrations manually is the default setting for the Docker container. Migrations can be run by overriding the `flyway.conf` and invoking Flyway migrations. It is generally recommended to add a Docker layer to the container that copies the `flyway.conf` file into a configured container version (to set the database URL and credentials). The sample shown above lists the configuration needed for the Docker container (notably the `flyway.jdbcUrl` setting). A configured container can be run using:

```
docker run -ti softpro/signdoc-standard:<version> flyway <command>
```

## FAQ

### General

- Useful log files for support requests
  - `INSTALLDIR\signdoc_home\logs\` (all files)
  - `INSTALLDIR\service\logs\` (all files)
  - Windows event log
  - Firewall logs
  - Browser console log
- Useful configuration files for support requests
  - `INSTALLDIR\signdoc_home\conf\` (all files, make sure to delete sensitive information)



### **Email settings**

- An error occurred sending an email
  - You specified the wrong server. The server you specified exists, but it is not an SMTP server.
  - You specified the wrong port number. Ask whoever runs the SMTP server what the correct port number is.
  - The server is down. This is usually temporary. If it persists, contact whoever administers the server.
  - Your firewall is blocking the port.
  - Your ISP is blocking the port. This usually affects port 25, and you can often work around it by using port 587, but details depend on your ISP and on the SMTP server's configuration.
  - You specified TLS, but the server does not support it.

### **Apache Tomcat**

- Security
  - For a production environment restrict the communication between cirrus and sdweb to local interfaces.
- Performance
  - The connector default size in bytes for POST requests is limited to 2 Megabytes, increase the size to handle larger documents by adding the attribute `maxPostSize="52428800"` to the tomcat connector definition.