SSL Configuration

Contact:
Method Park
Wetterkreuz 19a
91058 Erlangen
Germany
stages-support@methodpark.de
Internet
www.methodpark.de

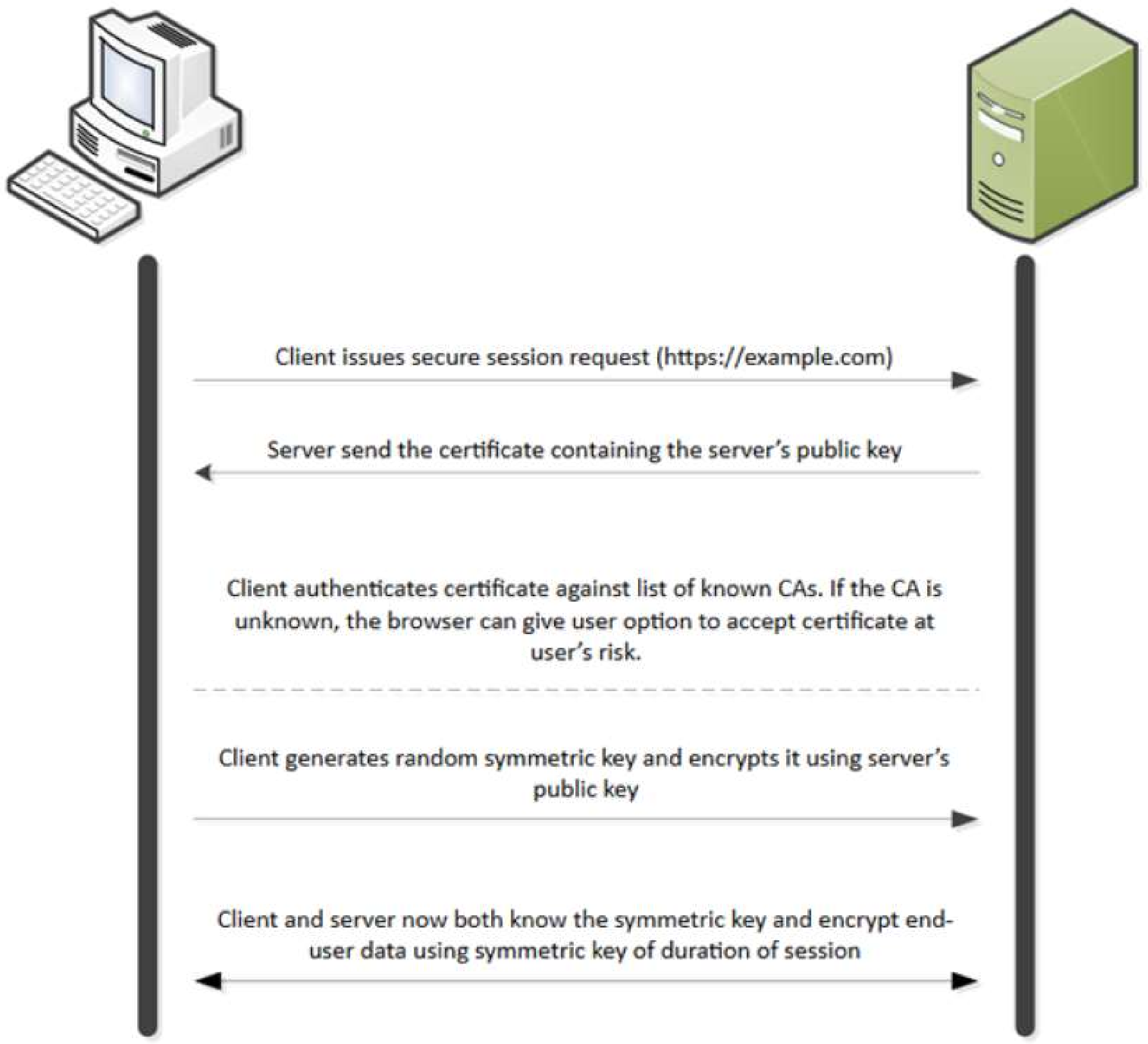
Contents

1 General information

1.1 SSL and how it works

A HTTP-based SSL connection is always initiated by the client using a URL starting with https:// instead of http://. At the beginning of an SSL session, an SSL handshake is performed. This handshake produces the cryptographic parameters of the session. A simplified overview of how the SSL handshake is processed is shown in the diagram below.

- A browser requests a secure page (https://).
- The web server sends its public key with its certificate.
- The browser checks that the certificate was issued by a trusted party (usually a trusted root CA), that the certificate is still valid, and that the certificate is related to the site contacted.
- The browser then uses the public key to encrypt a random symmetric encryption key and sends it to the server, along with the encrypted URL required as well as other encrypted http data.
- The web server decrypts the symmetric encryption key using its private key and uses the symmetric key to decrypt the URL and http data.
- The web server sends back the requested html document and http data, encrypted with the symmetric key.
- The browser decrypts the http data and html document using the symmetric key and displays the information.


Client issues secure session request (https://example.com)
Server send the certificate containing the server's public key
Client authenticates certificate against list of known CAs. If the CA is unknown, the browser can give user option to accept certificate at user's risk.
Client generates random symmetric key and encrypts it using server's public key
Client and server now both know the symmetric key and encrypt end-user data using symmetric key of duration of session

There are three essential types of certificates: private keys, public keys (also called public certificates), and root certificates.

1.2 The Private Key

The private key contains the identity information of the server, along with a key value. It should keep this key safe and protected by a password because it is used to negotiate the hash during the handshake. It can be used by someone to decrypt the traffic and get your personal information.

1.3 The Public Key

The public certificate (public key) is the portion that is presented to a client. The public certificate, tightly associated to the private key, is created from the private key using a Certificate Signing Request (CSR). After you create a private key, you create a CSR, which is sent to your Certificate Authority (CA). The CA returns a signed certificate, which has information about the server identity and about the CA.

1.4 The Key Pair

Every digital certificate has a pair of associated cryptographic keys. This pair of keys consists of a private key and a public key. Public/private key pairs are used for asymmetric encryption. Asymmetric encryption is used mainly to encrypt and decrypt session keys and digital signatures.

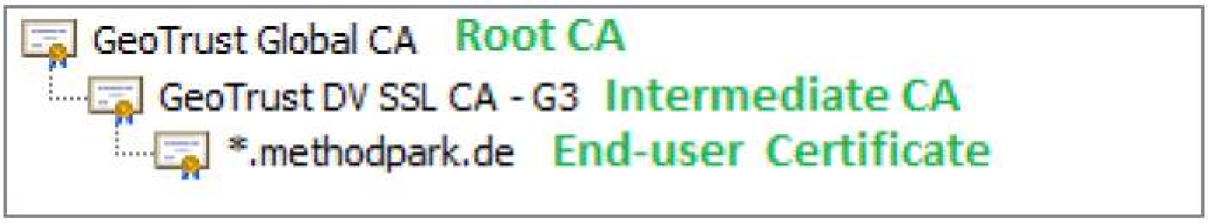
1.5 The Root Certificate

A Root CA Certificate is a CA Certificate which is simply a Self-signed Certificate. This certificate represents an entity which issues certificates and is known as Certificate Authority or the CA, such as VeriSign, Thawte, etc.

1.6 The Certificate Chain

When a server and client establish an SSL connection, a certificate is presented to the client. The client should determine whether to trust this certificate, a process called the certificate chain. The client examines the issuer of a certificate, searches its list of trusted root certificates, and compares the issuer on the presented certificate to the subjects of the trusted certificates. If a match is found, the connection proceeds. If not, the Web browser may pop up a dialog box, warning you that it cannot trust the certificate and offering the option to trust the certificate.

The list of SSL certificates, from the root certificate to the end-user certificate, represents the SSL certificate chain.



1.7 The Certificate Authority

Companies who will sign certificates for you are, for instance, VeriSign, Thawte, Commodo, GetTrust. Also, many companies and institutions act as their own CA, either by building a complete implementation from scratch, or by using an open source option, such as OpenSSL.

2 Introduction to SSL Configuration for Stages

Note: All described steps will be performed on the server where Stages is installed.

Before you start, please make sure that you have the following:

• Root Certificate of your Company's Certificate Authority

Necessary if existing:

• All Certificates of your Company's Intermediate Certificate Authority
• If you are not allowed to use or generate a Key Pair yourself you need a Key Pair issued and signed by your Company's Certificate Authority

2.1 Install KeyStore Explorer

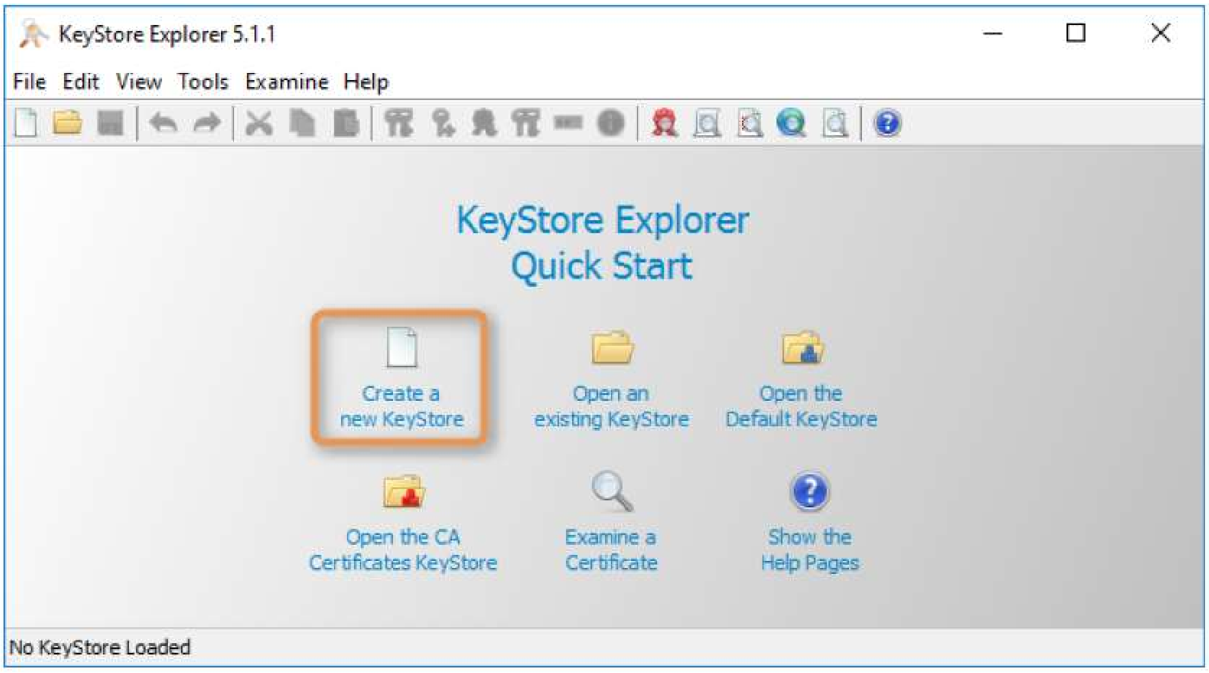
To configure SSL for your Stages server a Key Store is required. For the creation of a Key Store the tool KeyStore Explorer is necessary.

• Download KeyStore Explorer from: http://www.keystore-explorer.org/downloads.html
• Follow the on-screen instructions to complete the install.
• Start KeyStore Explorer

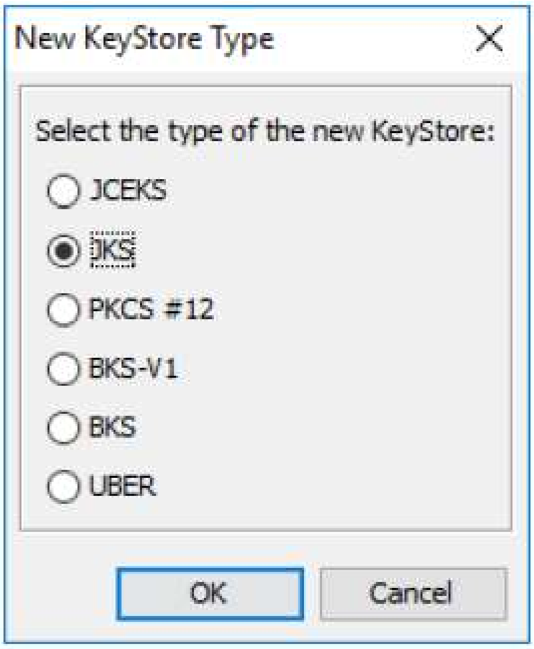
2.2 Create a new Key Store (JKS)

The keys Tomcat will use for SSL transactions are stored in a password-protected file called “Keystore".

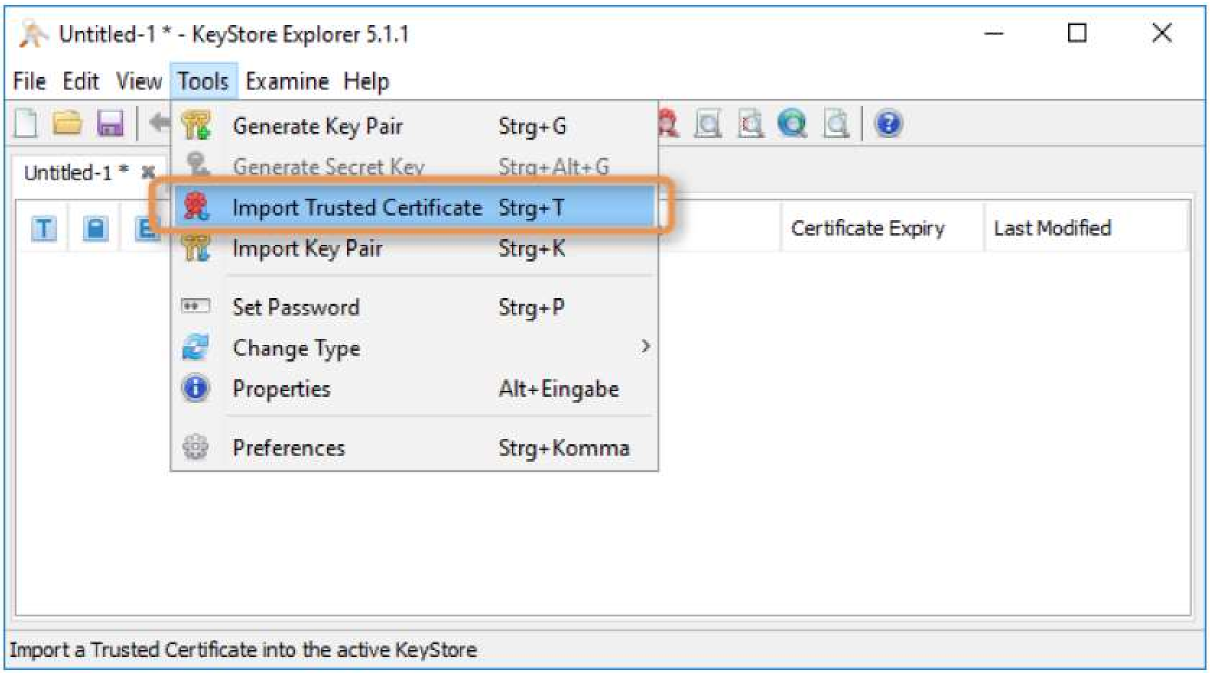
• Start KeyStore Explorer
• **Click on:** Create a new KeyStore



• **Select:** JKS



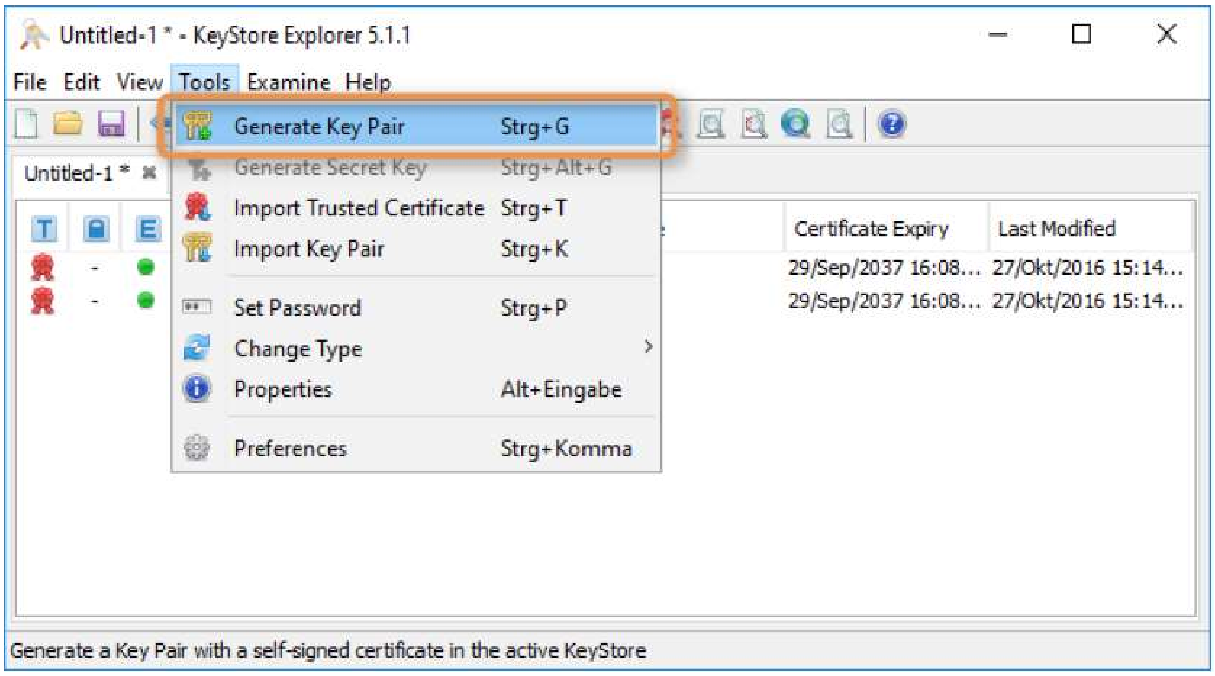
• Import the full certificate chain into the JKS. **Click on:** Tools > Import Trusted Certificate


Untitled-1 * - KeyStore Explorer 5.1.1
File Edit View Tools Examine Help
Untitled-1 *
Generate Key Pair Strg+G
Generate Secret Key Strg+Alt+G
Import Trusted Certificate Strg+T
Import Key Pair Strg+K
Set Password Strg+P
Change Type
Properties Alt+Eingabe
Preferences Strg+Komma
Certificate Expiry
Last Modified
Import a Trusted Certificate into the active KeyStore

2.3 Generate a new Key Pair

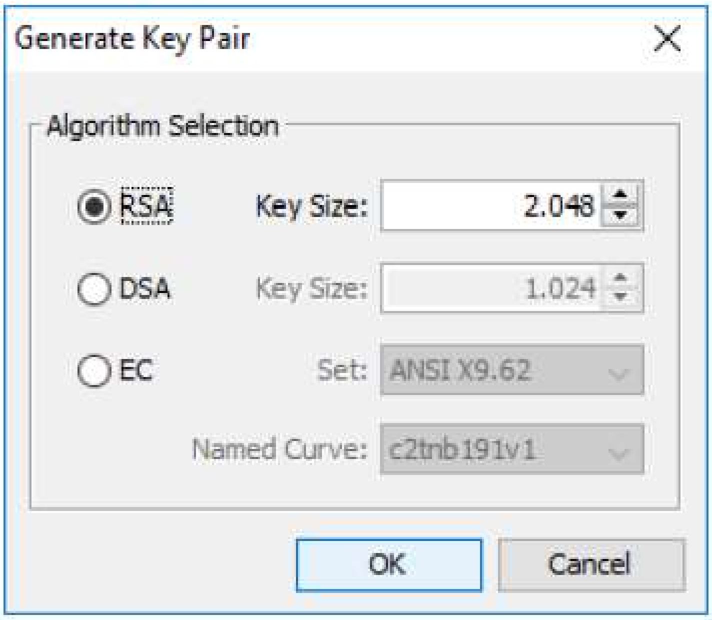
Important: If you already have a Key Pair which is signed by your Company's CA, then you can skip this part go to *Import an existing Key Pair*.

• Click on: **Tools→ Generate Key Pair**


Untitled-1 * - KeyStore Explorer 5.1.1
File Edit View Tools Examine Help
Generate Key Pair Strg+G
Generate Secret Key Strg+Alt+G
Import Trusted Certificate Strg+T
Import Key Pair Strg+K
Set Password Strg+P
Change Type
Properties Alt+Eingabe
Preferences Strg+Komma
Untitled-1 *
Certificate Expiry
Last Modified
29/Sep/2037 16:08... 27/Okt/2016 15:14...
29/Sep/2037 16:08... 27/Okt/2016 15:14...

Generate a Key Pair with a self-signed certificate in the active KeyStore

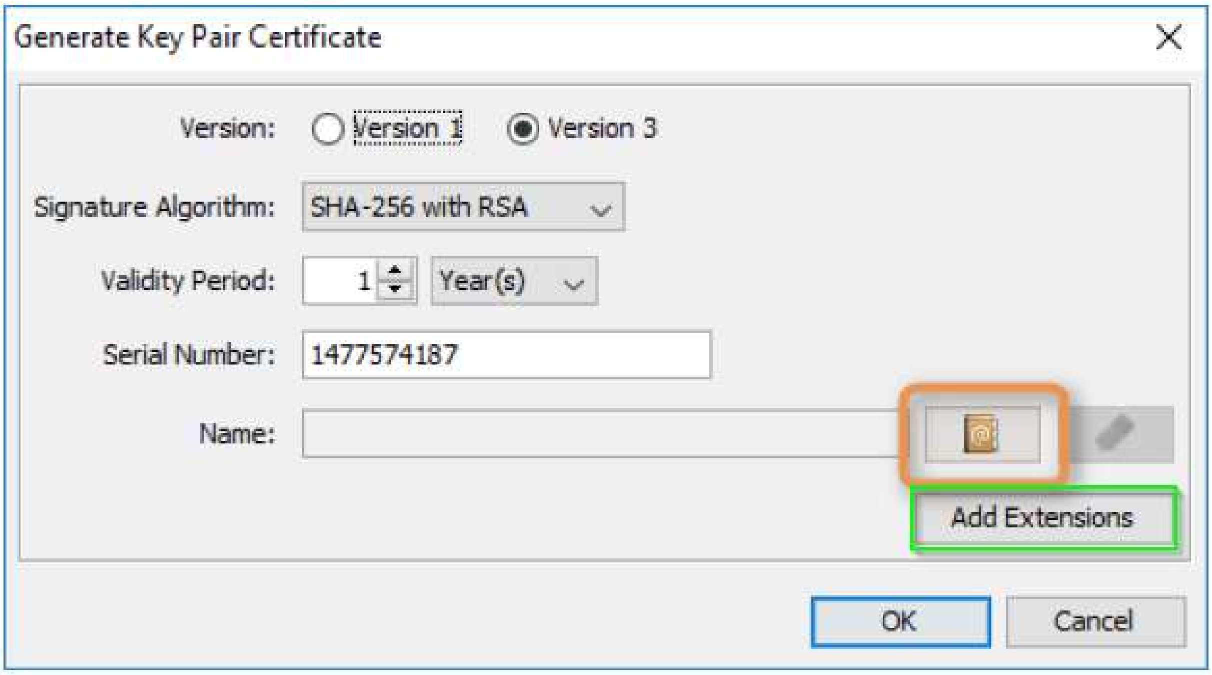
• Set Algorithm to: **RSA**
• Set Key Size to: **2048**



[✓ RSA] Key Size: 2.048
DSA Key Size: 1.024
EC Set: ANSI X9.62
Named Curve: c2tnb191v1

• Click on: **OK**

• Click on the Address Book Icon


Generate Key Pair Certificate

Version: ( ) Version 1 (●) Version 3

Signature Algorithm: SHA-256 with RSA

Validity Period: 1 Year(s)

Serial Number: 1477574187

Name:

• Enter the specific data.

Question	Answer
Common Name (CN)	FQDN (e.g. mystagesserver.my.domain.com)
Organization Unit (OU)	IT Server Administration
Organization Name (O)	My Company
Locality Name (L)	LA
State Name (ST)	CA
Country (C)	US

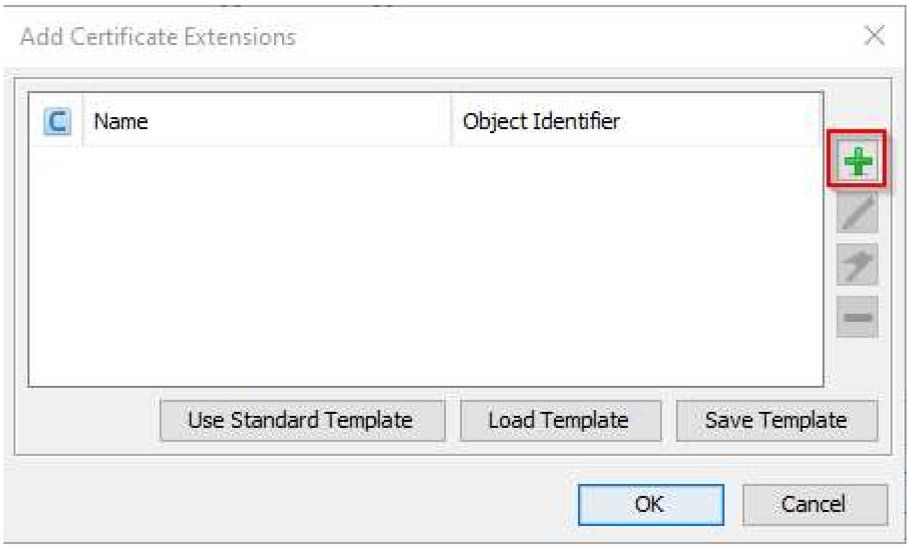
• Click on: **OK**

2.3.A Add Subject Alternative Names

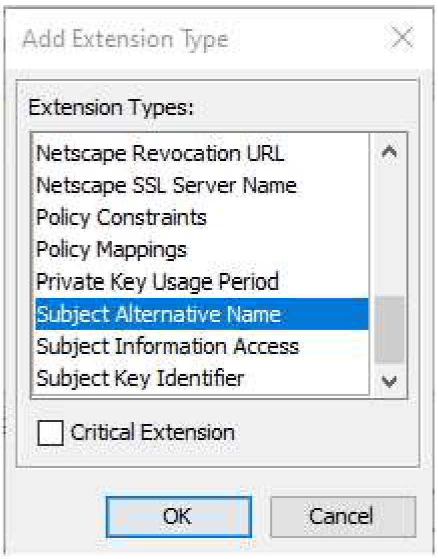
The button "**Add Extensions**" allows you to add **Subject Alternative Names (SAN)**. This means that you can, in addition to the common name (CN) you just entered, specify other common names, hostnames and IP addresses (although the latter could pose a security risk by showing the IP address of the server). This feature comes in handy if you need to use your keystore and certificate on more than one server, to include the name of a load balancer, or to have several different names pointing to your server. If you use SANs make sure you also include the original CN as a SAN as otherwise many browsers will ignore it in a certificate containing SANs.

If you do not want to add SANs please skip to step 2.3.B on Page 12. Otherwise proceed here first.

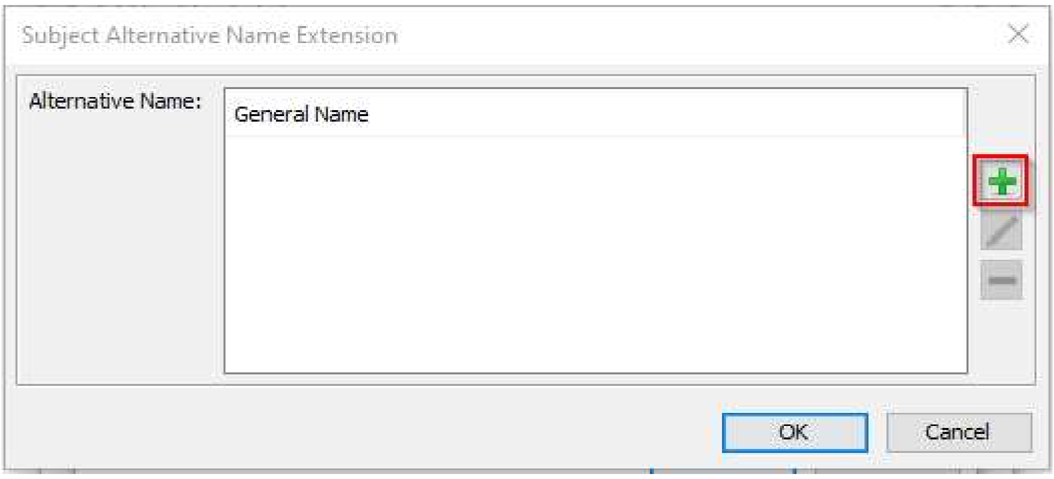
• Click on: **+**



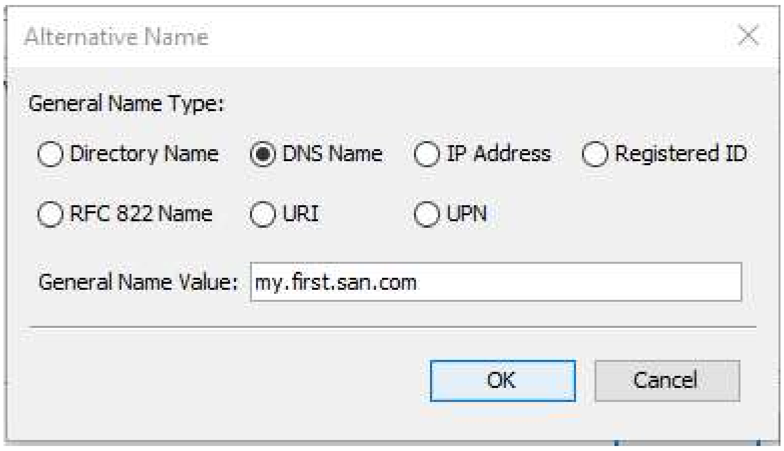
• Click on: Subject Alternative Name
• Click on: **OK**



• Click on: **+**



• Enter your desired SAN


Alternative Name

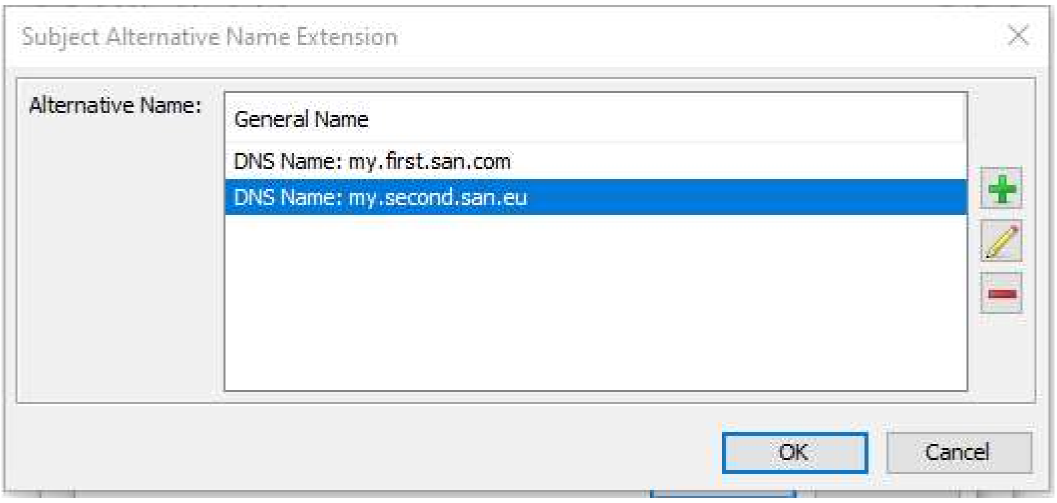
General Name Type:

○ Directory Name [✓ DNS Name] ○ IP Address ○ Registered ID

○ RFC 822 Name ○ URI ○ UPN

General Name Value: my.first.san.com

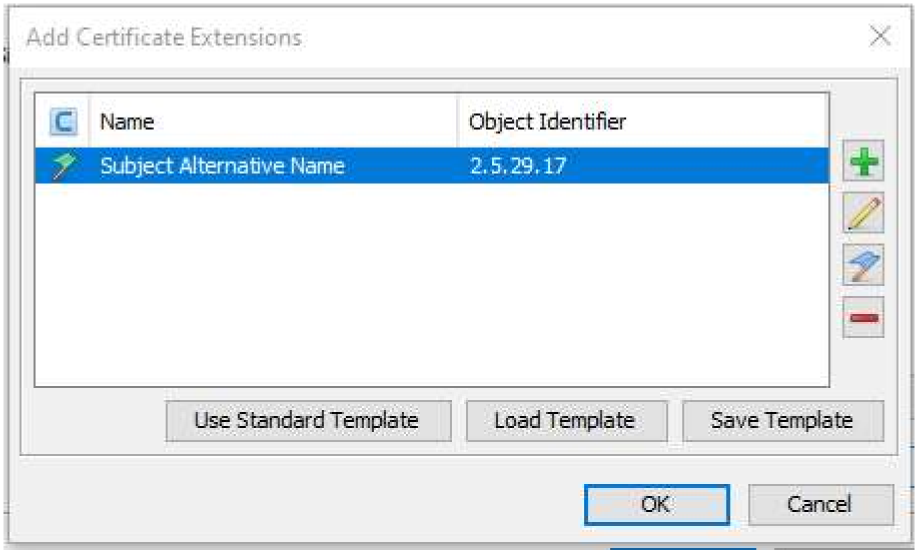
• Click **+** to add another SAN, repeat as needed
• Click on: **OK**


Subject Alternative Name Extension

Alternative Name:

General Name
DNS Name: my.first.san.com
DNS Name: my.second.san.eu

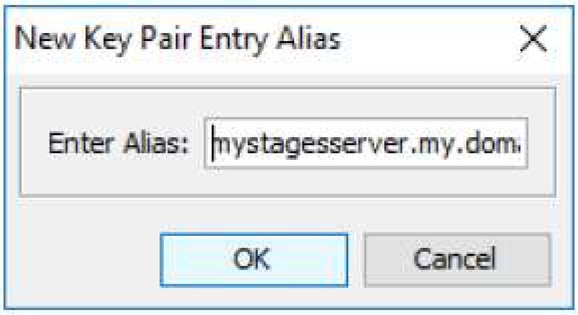
• Click on: **OK**


Add Certificate Extensions

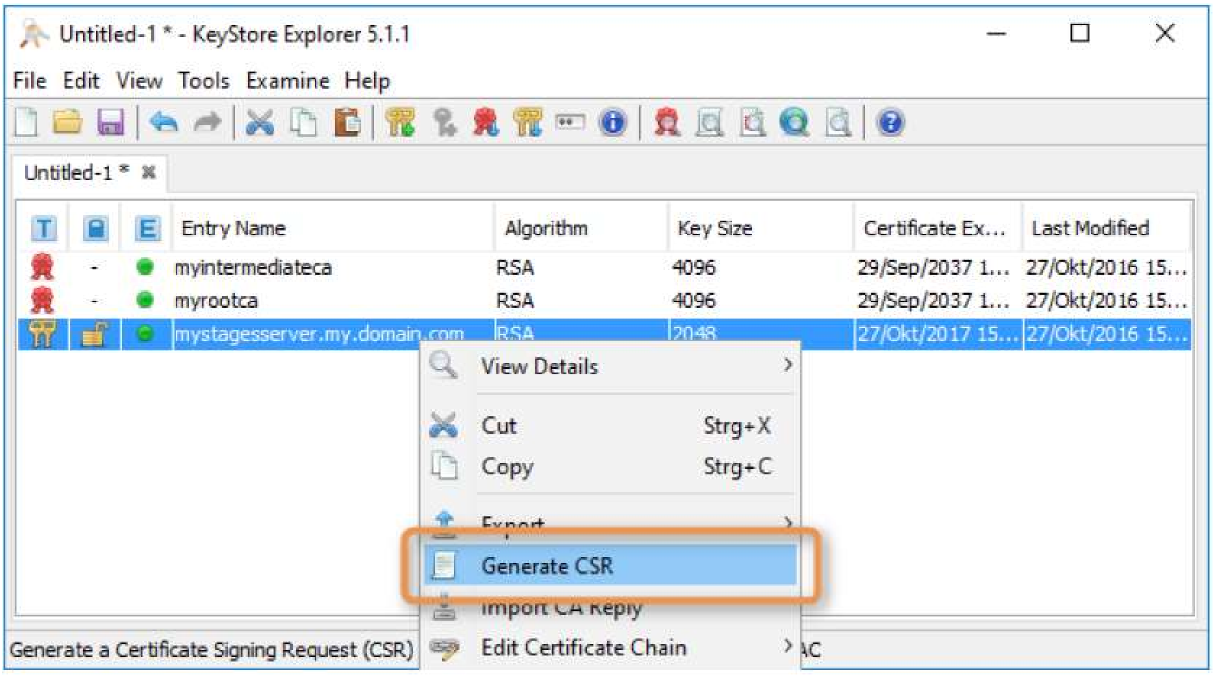
Name	Object Identifier
Subject Alternative Name	2.5.29.17

2.3.B Enter a Password

• Click on: **OK**

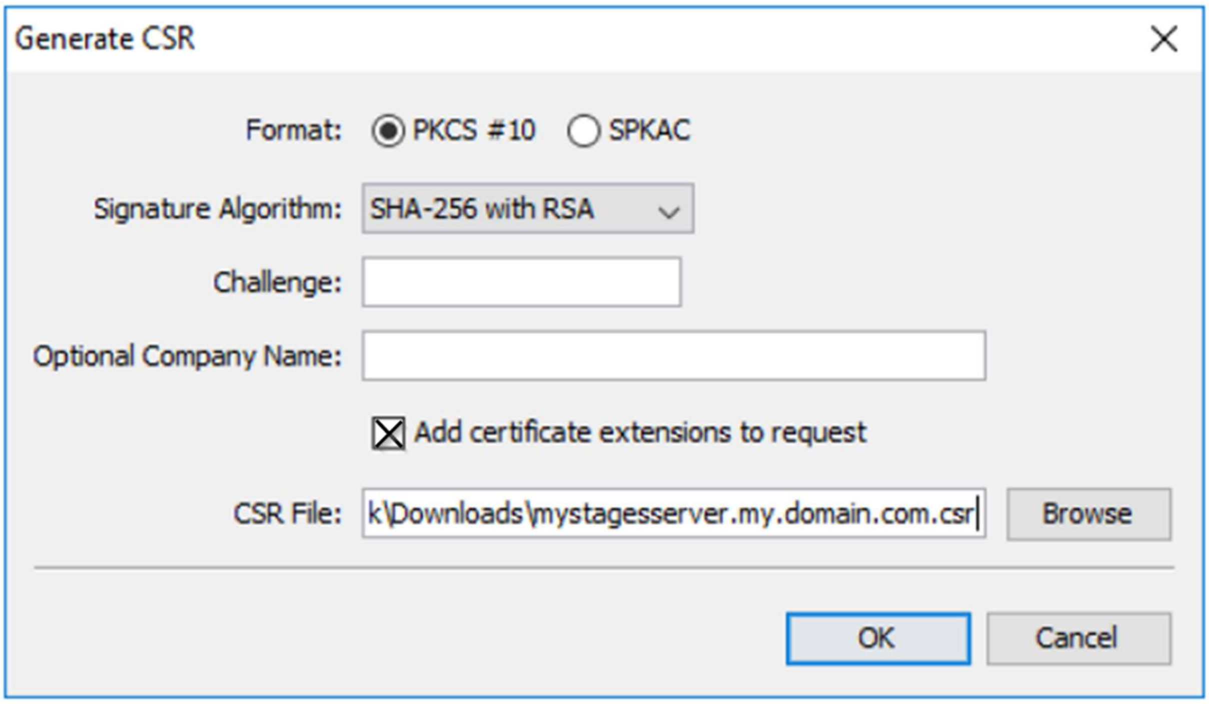


• Enter a password (use the same one that is used for the Key Store (.jks) file)
• Click on: **OK**
• Right Click on the new Certificate and select: **Generate CSR**


Untitled-1 * - KeyStore Explorer 5.1.1
File Edit View Tools Examine Help
Untitled-1 *
View Details
Cut Strg+X
Copy Strg+C
Export
Generate CSR
Import CA Reply
Edit Certificate Chain
Generate a Certificate Signing Request (CSR)

T		E	Entry Name	Algorithm	Key Size	Certificate Ex...	Last Modified
	-	●	myintermediateca	RSA	4096	29/Sep/2037 1...	27/Okt/2016 15...
	-	●	myrootca	RSA	4096	29/Sep/2037 1...	27/Okt/2016 15...
		●	mystagesserver.my.domain.com	RSA	2048	27/Okt/2017 15...	27/Okt/2016 15...

• Click on browse and set the filename to: **<filename>.csr** (e.g. mystagesserver.my.company.com.csr)


Generate CSR

Format: [✓ PKCS #10] SPKAC

Signature Algorithm: SHA-256 with RSA

Challenge:

Optional Company Name:

[✓ Add certificate extensions to request]

CSR File: k\Downloads\mystagesserver.my.domain.com.csr

• Click on: **OK**
• Click on: **File→ Save**
• Enter a password for the Key Store
• Add to the filename the `.jks` file extension
• Close KeyStore Explorer
• Send the CSR file to your Certificate Authority to sign the certificate signing request.
• After you have got the reply from your Certificate Authority, start KeyStore Explorer and open the JKS file.
• Right Click on the certificate and click on: **Import CA Reply**
• Select the certificate you got from your Certificate Authority.
• Click on: **Import**
• Click on: **File→ Save**

2.4 Import an existing Key Pair

Important: If you have created a new Key Pair in the last chapter, then skip this chapter and go to *Configure SSL for Tomcat*.

• Import your existing Key Pair: **Tools→ Import KeyPair**
• Select the type of your Key Pair
• Enter the specific information for your Key Pair
• Click on: **Import**
• Click on: **File→ Save**
• Enter a password for the Key Store
• Add to the filename the `.jks` file extension

2.5 Configure SSL for Tomcat

Now that you have created a functional keystore with valid certificates it is time to configure Tomcat to use SSL.

- Copy the JKS into the folder called `methodpark` which is usually the parent folder of your Stages installation directory
- Open the file: `<Stages installation directory>\conf\server.xml`
- Right after the connector with port 8080 (search for Connector port="8080") enter the following connector:

```
<Connector port="8443"
    maxHttpHeaderSize="8192"
    maxThreads="150"
    minSpareThreads="25"
    maxSpareThreads="75"
    enableLookups="false"
    disableUploadTimeout="true"
    acceptCount="100"
    scheme="https"
    secure="true"
    clientAuth="false"
    sslProtocol="TLS"
    SSLEnabled="true"
    keystoreFile="Path/To/Your/JKS/KeyStoreName.jks"
    keystorePass="EnterHereYourCertPassword"/>
```

- Save the file.
- Navigate to the Stages installation directory and enter: **bin\update.bat**
- Restart Stages and check if Stages is available under the following address: https://**<FQDN>**:8443 e.g. https://mystagesserver.my.company.com:8443

Note: If Stages is not available via https, please make sure that the server's firewall does not drop incoming connections from port 8443.

Note: >Instead of port 8443 you can use the standard SSL port 443. If this port is not in use, you can set the port for your SSL Connector to 443 e.g. Connector port="443".

2.6 Configure redirect from HTTP to HTTPS

Stages is now available via http and https. To make the connections more secure we will configure a redirect from http to https, if the user tries to connect via http.

- Open the file: `<Stages installation directory>\conf\web-customer.xml`
- The security constraint is actually a comment, you need to activate the security constraint. This is what the file should look like:

```
<!-- Define a Security Constraint on this Application -->
<!-- This requires all accesses to the application to be encrypted.
<security-constraint>
<web-resource-collection>
<web-resource-name>Entire Application</web-resource-name>
<url-pattern>/*</url-pattern>
</web-resource-collection>
<user-data-constraint>
<transport-guarantee>CONFIDENTIAL</transport-guarantee>
</user-data-constraint>
</security-constraint>
```

- Save the file.

• Open a command line with administrative privileges
• Navigate to the Stages installation directory and enter: **bin\update.bat**
• Make sure, that your http Connector (usually Connector with port 8080) which is located at `<Stages installation directory>\conf\server.xml` has set the attribute `redirectPort="8443"` or `redirectPort="443"` if you have set the standard SSL port 443 for your SSL Connector.
• Restart Stages
• Your http (http://**<FQDN>**:8080) requests will be redirected to https (https://**<FQDN>**:8443)

2.7 Add CAs to your TrustStore

If you are using reports in Stages you must add your company's Root Certificate and all Intermediate Certificates (if existing) to the server's Java Trust Store. The Java Trust Store is a store which includes all trusted root certificates. If you do not add these certificates to it then when a report tries to connect via SSL to your Stages server it will not be executed correctly because the installed Java does not trust the SSL Certificate, because the SSL Certificate was not issued by a trusted Certificate Authority.

• Start KeyStore Explorer
• Navigate within the Windows Explorer to: `<Java Development Kit Installation Directory>\jre\lib\security`
• Drag and drop the file called `cacerts` into the KeyStore Explorer
• Enter the password: `changeit`
• Import your Company's Root Certificate and all Intermediate Certificates (if existing). **Click on:** Tools >Import Trusted Certificate
• Click on: **File→ Save**
• Restart Stages