Understanding Java Keytool Keystore Commands

This great compilation of Java Keytool Keystore commands will make sure you're ready to handle your private keys, signing requests, and certificates.

Jim Aron, Apr. 23, 18 · Java Zone · Tutorial

This guide will help you with the Java Keytool Keystore platform. We will show you which Java Keytool Keystore commands work for which process for certificate management.

This blog is a comprehensive guide on how Java Keytool Keystore commands are used to manage your digital certificate in Keystore. And, ultimately, it becomes a time saver for busy developers.

Before we start the main conversation on how each of the Java Keytool commands work, we want to give you a little overview on Java Keytool Keystore.

What Is Java Keytool Keystore?

Java Keytool is a digital certificate management platform where the software developer keeps all certificates and private keys in a single place. The Java Keytool Keystore is the perfect solution to maintain the flow of trust and validation of all required certificates.

Okay, let's discuss the various Java Keytool Keystore commands that assist you in generating a certificate signing request and importing a private key and certificate.

The command to generate a Java keystore and keypair:

```
keytool -genkey -alias mydomain -keyalg RSA -keystore keystore.jks -keysize 2048
```

The command to generate a certificate signing request (CSR) for an existing Java keystore:

```
keytool -certreq -alias mydomain -keystore keystore.jks -file mydomain.csr
```

The command for importing a root or intermediate certificate to an existing Java keystore:

```
keytool -import -trustcacerts -alias root -file Thawte.crt -keystore keystore.jks
```

The command for importing a signed primary certificate to an existing Java keystore:

```
keytool -import -trustcacerts -alias mydomain -file mydomain.crt -keystore keystore.jks
```

The command to generate a keystore and a self-signed certificate:

```
keytool -genkey -keyalg RSA -alias selfsigned -keystore keystore.jks -storepass password -validity 360 -k
eysize 2048
```

The list of above commands will assist in generating a keypair and certificate signing request for a certificate.

We also gathered the list of Java Keystore commands to validate the generation process for certificates and CSRs.

The command for checking a standalone certificate:

```
keytool -printcert -v -file mydomain.crt
```

The command for checking which certificates are in a Java keystore:

```
keytool -list -v -keystore keystore.jks
```

The command for checking a particular keystore entry using an alias:

```
keytool -list -v -keystore keystore.jks -alias mydomain
```

Additionally, there are few crucial processes where you need Java Keytool commands. Let's have those commands for further validation.

The command for deleting a certificate from a Java Keytool keystore:

```
keytool -delete -alias mydomain -keystore keystore.jks
```

The command for changing a Java keystore password:

```
keytool -storepasswd -new new_storepass -keystore keystore.jks
```

The command for exporting a certificate from a keystore:

```
keytool -export -alias mydomain -file mydomain.crt -keystore keystore.jks
```

The command to view a list of trusted CA certs:

```
keytool -list -v -keystore $JAVA_HOME/jre/lib/security/cacerts
```

The command for importing new CAs into your trusted certs:

```
keytool -import -trustcacerts -file /path/to/ca/ca.pem -alias CA_ALIAS -keystore $JAVA_HOME/jre/lib/security/cacerts
```

We are sure that this list of commands will save developers' time while implementing a certificate for an existing application or a website.