The Java™ Tutorials

Trail: Security Features in Java SE
Lesson: Signing Code and Granting It Permissions
Section: Steps for the Code Signer

Generate Keys

If a code signer does not yet have a suitable private key for signing the code, the key must first be generated, along with a corresponding public key that can be used by the code receiver's runtime system to verify the signature.

Since this lesson assumes that you don't yet have such keys, you are going to create a keystore named `examplestore` and create an entry with a newly generated public/private key pair (with the public key in a certificate).

Type the following command in your command window to create a keystore named `examplestore` and to generate keys:

```
keytool -genkey -alias signFiles -keystore examplestore
```

You will be prompted to enter passwords for the key and keystore.

Subparts of the keytool Command

Let's look at what each of the `keytool` subparts mean.

- The command for generating keys is *-genkey*.
- The *-alias signFiles* subpart indicates the alias to be used in the future to refer to the keystore entry containing the keys that will be generated.
- The *-keystore examplestore* subpart indicates the name (and optionally path) of the keystore you are creating or already using.
- The *storepass* value that you are promted for specifies the keystore password.
- The *keypass* value that you are prompted for specifies a password for the private key about to be generated. You will always need this password in order to access the keystore entry containing that key. The entry doesn't have to have its own password. When you are prompted for the key password, you are given the option of letting it be the same as the keystore password.

Note: For security reasons you should not set your key or keystore passwords on the command line, because they can be intercepted more easily that way.

Distinguished-Name Information

If you use the preceding `keystore` command, you will be prompted for your distinguished-name information. Following are the prompts; the bold indicates what you should type.

```
What is your first and last name?
  [Unknown]:  Susan Jones
What is the name of your organizational unit?
  [Unknown]:  Purchasing
What is the name of your organization?
  [Unknown]:  ExampleCompany
What is the name of your City or Locality?
  [Unknown]:  Cupertino
What is the name of your State or Province?
  [Unknown]:  CA
What is the two-letter country code for this unit?
  [Unknown]:  US
Is <CN=Susan Jones, OU=Purchasing, O=ExampleCompany,
    L=Cupertino, ST=CA, C=US> correct?
  [no]:  y
```

Command Results

The `keytool` command creates the keystore named `examplestore` (if it doesn't already exist) in the same directory in which the command is executed. The command generates a public/private key pair for the entity whose distinguished name has a common name of Susan Jones and the organizational unit of Purchasing.

The command creates a self-signed certificate that includes the public key and the distinguished-name information. (The distinguished name you supply will be used as the "subject" field in the certificate.) This certificate will be valid for 90 days, the default validity period if you don't specify a *-validity* option. The certificate is associated with the private key in a keystore entry referred to by the alias `signFiles`.

Self-signed certificates are useful for developing and testing an application. However, users are warned that the application is signed with an untrusted certificate and asked if they want to run the application. To provide users with more confidence to run your application, use a certificate issued by a recognized certificate authority.

Note: The command could be shorter if option defaults are accepted or you wish to be prompted for various values. Whenever you execute a `keytool` command, defaults are used for unspecified options that have default values, and you are prompted for any required values. For the `genkey` command, options with default values include *alias* (whose default is `mykey`), *validity* (90 days), and *keystore* (the file named `.keystore` in your home directory). Required values include *dname*, *storepass*, and *keypass*.

Previous page: Create a JAR File Containing the Class File
Next page: Sign the JAR File