



Featured Books

Java Coding Guideli
Recommendations f
Reliable and Secure
Programs (SEI Series
Software Engineerin

Head First Design Pa

Java Concurrency ir

Java Performance

Java Puzzlers: Traps
and Corner Cases

Head First Object-Or

Analysis and Desigr

**Clean Code: A Hand
Agile Software Craft**

Algorithms Unlocke

**Data Structures and
Algorithm Analysis i
(3rd Edition)**

**Refactoring: Improvi
Design of Existing C**

**The Pragmatic Progr
From Journeyman to**

**Code Complete: A P
Handbook of Softwa
Construction, Secor**

**Cracking the Coding
Interview: 150 Progr
Questions and Solut**

**The Clean Coder: A
Conduct for Profess
Programmers (Robe
Martin Series)**

File Encryption and Decryption Simple Example

Last Updated on 19 June 2014 | Print Email

Java Performance **Clean Code** **The Clean Coder** **The Pragmatic Programmer**

Encryption and decryption are fundamental requirements of every secure-aware application, therefore the Java platform provides strong support for encryption and decryption through its **Java Cryptographic Extension (JCE)** framework which implements the standard cryptographic algorithms such as AES, DES, DESede and RSA. This tutorial shows you how to basically encrypt and decrypt files using the *Advanced Encryption Standard (AES)* algorithm. AES is a symmetric-key algorithm that uses the same key for both encryption and decryption of data.

1. Basic Steps

Here are the general steps to encrypt/decrypt a file in Java:

- Create a `Key` from a given byte array for a given algorithm.
- Get an instance of `Cipher` class for a given algorithm transformation. See document of the Cipher class for more information regarding supported algorithms and transformations.
- Initialize the `Cipher` with an appropriate mode (encrypt or decrypt) and the given `Key`.
- Invoke `doFinal(input_bytes)` method of the `Cipher` class to perform encryption or decryption on the `input_bytes`, which returns an encrypted or decrypted byte array.
- Read an input file to a byte array and write the encrypted/decrypted byte array to an output file accordingly.

Now, let's see some real examples.

2. The CryptoUtils class

Here's a utility class that provides two utility methods, one for encrypt a file and another for decrypt a file:

```java
package net.codejava.crypto;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
import java.security.InvalidKeyException;
import java.security.Key;
import java.security.NoSuchAlgorithmException;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import javax.crypto.spec.SecretKeySpec;

/**
 * A utility class that encrypts or decrypts a file.
 * @author www.codejava.net
 *
 */
public class CryptoUtils {
    private static final String ALGORITHM = "AES";
    private static final String TRANSFORMATION = "AES";

    public static void encrypt(String key, File inputFile, File o
            throws CryptoException {
        doCrypto(Cipher.ENCRYPT_MODE, key, inputFile, outputFile)
    }

    public static void decrypt(String key, File inputFile, File o
            throws CryptoException {
        doCrypto(Cipher.DECRYPT_MODE, key, inputFile, outputFile)
    }

    private static void doCrypto(int cipherMode, String key, File
            File outputFile) throws CryptoException {
        try {
            Key secretKey = new SecretKeySpec(key.getBytes(), ALG
            Cipher cipher = Cipher.getInstance(TRANSFORMATION);
            cipher.init(cipherMode, secretKey);

            FileInputStream inputStream = new FileInputStream(inp
            byte[] inputBytes = new byte[(int) inputFile.length()
            inputStream.read(inputBytes);

            byte[] outputBytes = cipher.doFinal(inputBytes);

            FileOutputStream outputStream = new FileOutputStream(
            outputStream.write(outputBytes);

            inputStream.close();
            outputStream.close();

        } catch (NoSuchPaddingException | NoSuchAlgorithmExceptio
                | InvalidKeyException | BadPaddingException
                | IllegalBlockSizeException | IOException ex) {
            throw new CryptoException("Error encrypting/decryptin
        }
    }
}
```

Both the methods **`encrypt()`** and **`decrypt()`** accepts a key, an input file and an output file as parameters, and throws a **`CryptoException`** which is a custom exception written as below:

```
package net.codejava.crypto;

public class CryptoException extends Exception {

    public CryptoException() {
    }

    public CryptoException(String message, Throwable throwable) {
        super(message, throwable);
    }
}
```

This custom exception eliminates the messy throws clause, thus make the caller invoking those methods without catching a lengthy list of original exceptions.

3. The CryptoUtilsTest class

The following code is written for a test class that tests the `CryptoUtils` class above:

```
package net.codejava.crypto;

import java.io.File;

/**
 * A tester for the CryptoUtils class.
 * @author www.codejava.net
 *
 */
public class CryptoUtilsTest {
    public static void main(String[] args) {
        String key = "Mary has one cat1";
        File inputFile = new File("document.txt");
        File encryptedFile = new File("document.encrypted");
        File decryptedFile = new File("document.decrypted");

        try {
            CryptoUtils.encrypt(key, inputFile, encryptedFile);
            CryptoUtils.decrypt(key, encryptedFile, decryptedFile
        } catch (CryptoException ex) {
            System.out.println(ex.getMessage());
            ex.printStackTrace();
        }
    }
}
```

This test program simply encrypts a text file, and then decrypts the encrypted file. Note that the key used for encryption and decryption here is a string “Mary has one cat”;

4. Note about key size

The AES algorithm requires that the key size must be 16 bytes (or 128 bit). So

if you provide a key whose size is not equal to 16 bytes, a `java.security.InvalidKeyException` will be thrown. In case your key is longer, you should consider using a padding mechanism that transforms the key into a form in which its size is multiples of 16 bytes. See the Cipher class Javadoc for more details.

References

- Java SE Security
- Advanced Encryption Standard (AES)
- Cipher class Javadoc

Attachments:

SimpleFileEncryptionDecryptionDemo.zip [Java source code] 10 kB

Add comment

Name E-mail

comment

1000 symbols left

Notify me of follow-up comments

Type the text

Privacy & Terms

Send

Comments

1 2

#20 **Nam** 2015-11-04 19:55 **0**

Quoting Anonymos:

> if a input file is .dll file ? How to make a .dll file for this

The input can be any file.
If you want to make a .dll for this code, use JNI. google for Java JNI.

Quote

#19 **Anonymos** 2015-10-31 08:33 **0**

if a input file is .dll file ? How to make a .dll file for this

Quote

#18 **thava** 2015-09-28 07:03 **0**

hi da thava

Quote

#17 **Wayne** 2015-09-17 13:47 **0**

when I compile the first class in Linux, I get the following errors. Am I missing something?

CryptoUtils.java:29: error: cannot find symbol
throws CryptoException {
^
symbol: class CryptoException
location: class CryptoUtils
CryptoUtils.java:34: error: cannot find symbol
throws CryptoException {
^
symbol: class CryptoException
location: class CryptoUtils
CryptoUtils.java:39: error: cannot find symbol
File outputFile) throws CryptoException {
^
symbol: class CryptoException
location: class CryptoUtils
CryptoUtils.java:60: error: cannot find symbol
throw new CryptoException("Error encrypting/decrypting file", ex);
^
symbol: class CryptoException
location: class CryptoUtils

Quote

#16 **chandrasekar** 2015-09-14 02:38 **-1**

pls i need java source code

Quote

#15 **Shadab Shamsi** 2015-09-11 06:24 **-2**

Thanks a lot.The code is really well written.

Quote

#14 **Carlos** 2015-07-27 15:09 **0**

How can I change the key?

Quote

#13 **Carlos** 2015-07-27 15:08 **-2**

The first time I use a key, it works. When I change the key after the first time, the program does not work (not matter what I do) until I change the key back to the first one.

Quote

#12 **swanand hegde** 2015-07-21 04:46 **-2**

Where is the key in the source code

Quote

#11 **swanand hegde** 2015-07-21 04:32 **0**

where is the key in the attachment

Quote

1 2

Refresh comments list

RSS feed for comments to this post