

Programación de servicios y procesos

Act5.2. Almacenamiento seguro

Francisco José García Cutillas | 2FPGS_DAM



Índice

Ejercicio 1	3
-------------------	---

Ejercicio 1

Almacenamiento seguro

Programar una aplicación Java que reciba como parámetros de entrada un fichero y una contraseña, cifre el contenido utilizando AES, almacene el resultado con un nombre de fichero distinto y borre el fichero original.

Programar también la aplicación que realice la transformación inversa.

En el raíz de nuestro proyecto vamos a crear el fichero “ficheroDescifrado.txt”, que será el que vamos a cifrar.



ficheroDescifrado.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Este es un mensaje cifrado, no se lo cuentes a nadie.

Lo primero que vamos a hacer es ejecutar el método “cifrar”, introduciéndole por parámetros la ruta del fichero original, la del fichero a generar cifrado y una key que vamos a generar con el método “obtenerClave” que recibe una contraseña de tipo String.

```

1  package com.myccompany.psp_act5_2;
2
3  import java.io.File;
4  import java.io.FileInputStream;
5  import java.io.FileOutputStream;
6  import java.security.Key;
7  import javax.crypto.Cipher;
8  import javax.crypto.CipherInputStream;
9  import javax.crypto.CipherOutputStream;
10 import javax.crypto.spec.SecretKeySpec;
11
12 /**
13  *
14  * @author Fran
15  */
16 public class PSP_Act5_2 {
17
18     public static void main(String[] args) {
19
20         try {
21
22             Key claveCifrado = obtenerClave(password: "esto es una contraseña", longitud: 16);
23             cifrar(ficheroSinCifrar: "ficheroDescifrado.txt", nombreFicheroCifrar: "ficheroCifrado.txt", key: claveCifrado);
24
25             //descifrar("ficheroCifrado.txt", "ficheroDescifrado.txt", claveCifrado);
26
27         } catch (Exception ex) {
28
29             System.out.println("Error: " + ex.getMessage());
30
31         }
32     }
33
34
35     //Método para obtener la clave de tipo Key a partir de una contraseña de tipo String
36     private static Key obtenerClave(String password, int longitud) {
37
38         Key clave = new SecretKeySpec(key.password.getBytes(), offset: 0, len: longitud, algorithm: "AES");
39
40         return clave;
41     }
42
43 }

```

```

44 //Método para cifrar un fichero. Recibe la ruta del fichero a cifrar, la del nuevo fichero cifrado y una key
45 private static void cifrar(String ficheroSinCifrar, String nombreFicheroCifrar, Key key) throws Exception{
46
47     Cipher cipher = Cipher.getInstance("AES");
48     cipher.init(opmode:Cipher.ENCRYPT_MODE, key);
49
50     FileInputStream fis = new FileInputStream(name:ficheroSinCifrar);
51     FileOutputStream fos = new FileOutputStream(name:nombreFicheroCifrar);
52     CipherOutputStream cos = new CipherOutputStream(o:fos, c:cipher);
53
54     //Leemos el fichero y lo ciframos
55     byte[] buffer = new byte[1024];
56
57     int bytesLeídos = fis.read(b:buffer);
58
59     while(bytesLeídos != -1){
60
61         cos.write(b:buffer, off:0, len:bytesLeídos);
62         bytesLeídos = fis.read(b:buffer);
63
64     }
65
66     cos.close();
67     fos.close();
68     fis.close();
69
70     //Borramos el fichero original
71     File fichero = new File(pathname:ficheroSinCifrar);
72     fichero.delete();
73
74     System.out.println(s:"Fichero cifrado.");
75
76 }
77
78 //Método para descifrar un fichero. Recibe la ruta del fichero cifrado, la del nuevo sin cifrar y una key
79 private static void descifrar(String ficheroCifrado, String nombreFicheroSinCifrar, Key key) throws Exception{
80
81     Cipher cipher = Cipher.getInstance("AES");
82     cipher.init(opmode:Cipher.DECRYPT_MODE, key);
83
84     FileInputStream fis = new FileInputStream(name:ficheroCifrado);
85     FileOutputStream fos = new FileOutputStream(name:nombreFicheroSinCifrar);
86     CipherInputStream cis = new CipherInputStream(i:fis, c:cipher);
87
88     //Leemos el fichero y lo desciframos
89     byte[] buffer = new byte[1024];
90
91     int bytesLeídos = cis.read(b:buffer);
92
93     while(bytesLeídos != -1){
94
95         fos.write(b:buffer, off:0, len:bytesLeídos);
96         bytesLeídos = cis.read(b:buffer);
97
98     }
99
100     cis.close();
101     fos.close();
102     fis.close();
103
104     //Borramos el fichero original
105     File fichero = new File(pathname:ficheroCifrado);
106     fichero.delete();
107
108     System.out.println(s:"Fichero descifrado.");
109
110 }
111

```

```

--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSP_Act5_2 ---
Fichero cifrado.

-----

BUILD SUCCESS


-----

Total time:  0.424 s
Finished at: 2024-02-08T18:11:51+01:00

-----

```

Este será ahora nuestro fichero cifrado.

 ficheroCifrado.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

|-0ÓE|pddâ1&@Dİ€ÈµÄÖ|”š”g!Žeh`upĩº?15µŸĚñÄ5Ø’ää)lw|.0Y¢HR Äš%;

Vamos ahora a volverlo a descifrar.

```
public static void main(String[] args) {
    try {
        Key claveCifrado = obtenerClave(password: "esto es una contraseña", longitud: 16);
        //cifrar("ficheroDescifrado.txt", "ficheroCifrado.txt", claveCifrado);

        descifrar( ficheroCifrado: "ficheroCifrado.txt", nombreFicheroSinCifrar: "ficheroDescifrado.txt", key: claveCifrado);
    } catch (Exception ex) {
        System.out.println("Error: " + ex.getMessage());
    }
}
```

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSP_Act5_2 ---
```


Fichero descifrado

BUILD SUCCESS

Total time: 0.426 s

Finished at: 2024-02-08T18:14:07+01:00

Volvemos a tener el fichero descifrado.

 ficheroDescifrado.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Este es un mensaje cifrado, no se lo cuentes a nadie.