

Programación de servicios y procesos

Act5.1. Comprobación de
integridad

Francisco José García Cutillas | 2FPGS_DAM



Índice

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

Ejercicio 1


Comprobación de integridad.

Programa una aplicación que reciba como parámetros de entrada un fichero y su HASH SHA-512 y que certifique la integridad del fichero.

Puedes obtener ficheros y HASH de pruebas en la sección de descargas del IDE Eclipse:

<https://www.eclipse.org/downloads/>

Para la realización de esta práctica nos hemos creado un fichero prueba.txt con el siguiente contenido en la carpeta raíz del proyecto.

 prueba.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Documento de prueba para la práctica 5.1

Ahora con la ayuda de <https://www.fileformat.info/tool/hash.htm> que es una calculadora online del hash de un fichero que subamos, obtenemos el siguiente hash SHA-512.

SHA-512	e62b9b432c38710a16326edd4712c4f15185fbf2cd86d3a47afb453c6d8f72e43ce385e7692e2ab4fa0e8211ac8e28de32ad31997c2130036756cd5bd8be8796
---------	--

En nuestro programa hemos creado un método que calcula el hash actual de un fichero que se le pasa por parámetro, y otro método que comprueba que el hash recibido por parámetro coincide con el actual del fichero que también recibe por parámetro.

```
package com.myccompany.psp_act5_1;

import java.io.File;
import java.io.IOException;
import java.nio.file.Files;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

/**
 *
 * @author Fran
 */
public class PSP_Act5_1 {

    private static final String ALGORITMO = "SHA-512";

    public static void main(String[] args) {

        String hash = "e62b9b432c38710a16326edd4712c4f15185fbf2cd86d3a47afb453c6d8f72e43ce385e7692e2ab4fa0e8211ac8e28de32ad31997c2130036756cd5bd8be8796";
        File fichero = new File("prueba.txt");

        System.out.println("**** Comprobación de la integridad de un fichero ****");

        try {

            //Comprobación de la integridad
            if (certificaIntegridad(fichero, hash)) {

                System.out.println("Integridad correcta");

            } else {

                System.out.println("Integridad incorrecta");

            }

            System.out.println("Hash actual del fichero: " + calcularHash(fichero));
            System.out.println("Hash que nos pasan: " + hash);

        } catch (IOException | NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
    }
}
```

Act5.1. Comprobación de integridad

```
39         } catch (IOException ex) {
40             System.out.println("Error: " + ex.getMessage());
41         } catch (NoSuchAlgorithmException ex) {
42             System.out.println("Error: " + ex.getMessage());
43         }
44     }
45
46     //Método para comprobar la integridad del fichero
47     private static boolean certificaIntegridad (File fichero, String hash) throws IOException, NoSuchAlgorithmException{
48         String hashCalculado = calcularHash(fichero);
49         if (hash.equals(hashCalculado)) {
50             return true;
51         } else{
52             return false;
53         }
54     }
55
56     //Método que calcula el Hash de un fichero que se le pasa por parámetro
57     private static String calcularHash(File fichero) throws IOException, NoSuchAlgorithmException{
58         MessageDigest algoritmo = MessageDigest.getInstance("SHA-256");
59         byte[] lecturaFichero = Files.readAllBytes(fichero.toPath());
60         byte[] hashByte = algoritmo.digest(lecturaFichero);
61
62         //Convertimos el hash de tipo array de byte a un string
63         StringBuilder sb = new StringBuilder();
64         for(byte hash : hashByte){
65             sb.append(String.format("%02x", hash));
66         }
67
68         return sb.toString();
69     }
70 }
```

Si comprobamos la integridad con el hash generado desde la calculadora online y desde el generado por el fichero, vemos que coinciden.

```
-----< com.myccompany:PSP_Act5_1 >-----
Building PSP_Act5_1 1.0-SNAPSHOT
[ jar ]
--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSP_Act5_1 ---
*** Comprobación de la integridad de un fichero ***
Integridad correcta
Hash actual del fichero: e62b9b432c38710a16326edd4712c4f15185fbf2cd86d3a47afb453c6d8f72e43ce385e7692e2ab4fa0e8211ac8e28de32ad31997c2130036756cd5bd8be8796
Hash que nos pasan: e62b9b432c38710a16326edd4712c4f15185fbf2cd86d3a47afb453c6d8f72e43ce385e7692e2ab4fa0e8211ac8e28de32ad31997c2130036756cd5bd8be8796

BUILD SUCCESS

Total time: 0.384 s
Finished at: 2024-02-09T08:48:12+01:00
```

Ahora vamos a probar a modificar el fichero. Esto nos modificará su hash, pero nosotros le seguimos introduciendo el mismo que nos había generado la calculadora online.



prueba.txt: Bloc de notas

Archivo Edición Formato Ver Ayuda

Modifico el documento de prueba para la práctica 5.1

Act5.1. Comprobación de integridad

```
--- exec-maven-plugin:3.0.0:exec (default-cli) @ PSP_Act5_1 ---
*** Comprobación de la integridad de un fichero ***
Integridad incorrecta
Hash actual del fichero: 07a0f4ecf4844245ffbf59f9b986088ab0f4ce341a9alc5088e34a3alb570a054cd481eb62bd2d614ff100500e2facaa24ef1e26dc4bf864a35fcf29298e9037
Hash que nos pasan: e62b9b432c38710a16326edd4712c4f15185f9f2cd86d3a47afb453c6d8f72e43ce385e7692e2ab4fa0e821lac8e28de32ad31997c2130036756cd5bd8be8796
-----
BUILD SUCCESS
-----
Total time: 0.379 s
Finished at: 2024-02-08T08:50:34+01:00
-----
```

Podemos observar que al modificar el fichero, se modifica su hash, por lo que la integridad ya no sería correcta con el hash anterior.