Código Fuente

```java
package factory.method;
/**
 * @author Victor Lavalle
 */
public interface AlgoritmoEncriptamiento {

    public abstract void configurar();

    public abstract String encriptar(String texto);

    public abstract String desencriptar(String textoEncriptado);
}
```

```java
package factory.method;
/**
 * @author Victor Lavalle
 */
public class FactoryAlgoEncripKHash implements FactoryAlgoEncriptamiento {

    @Override
    public AlgoritmoEncriptamiento create() {
        AlgoritmoEncriptamiento algo = new KHash();
        algo.configurar();
        return algo;
  }

}
```

```java
package factory.method;
/**
 * @author Victor Lavalle
 */
public class FactoryAlgoEncripGHash implements FactoryAlgoEncriptamiento {

    @Override
    public AlgoritmoEncriptamiento create() {
        AlgoritmoEncriptamiento algo = new GHash();
        algo.configurar();
        return algo;
    }

}
```

```java
package factory.method;
/**
 * @author Victor Lavalle
 */
public interface FactoryAlgoEncriptamiento {

    public AlgoritmoEncriptamiento create();

}
```

```java
package factory.method;

import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.SecretKeySpec;
import java.security.spec.KeySpec;
import java.util.Base64;

public class GHash implements AlgoritmoEncriptamiento {
    private static String secretKey ;
    private static String salt ;
    private byte[] gHashInitVector = { 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 };
    private IvParameterSpec gHashInitVectorHash;

    @Override
    public void configurar() {
        secretKey = "GHash!!!!";
         salt = "GHash!!!!";
         IvParameterSpec gHashInitVectorHash = new IvParameterSpec(gHashInitVector);
    }

    @Override
    public String encriptar(String str) {
        System.out.println("Encrypting String with GHash");
        try {
            SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
            KeySpec spec = new PBEKeySpec(secretKey.toCharArray(), salt.getBytes(), 65536, 256);
            SecretKey tmp = factory.generateSecret(spec);
            SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");
            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE, secretKey, gHashInitVectorHash);
            return Base64.getEncoder().encodeToString(cipher.doFinal(str.getBytes("UTF-8")));
        }
        catch (Exception e) {
            System.out.println("Error while encrypting with GHash: " + e.toString());
  }
        return null;
    }

     @Override
     public String desencriptar(String str) {
         System.out.println("Decrypting String with GHash");
         try {
             SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
             KeySpec spec = new PBEKeySpec(secretKey.toCharArray(), salt.getBytes(), 65536, 256);
             SecretKey tmp = factory.generateSecret(spec);
             SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");
             Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
             cipher.init(Cipher.DECRYPT_MODE, secretKey, gHashInitVectorHash);
             return new String(cipher.doFinal(Base64.getDecoder().decode(str)));}
```

```
        catch (Exception e) {
  System.out.println("Error while decrypting: " + e.toString());
                    }
            return null;
  }
}
```

```
package factory.method;

import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.SecretKeyFactory;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.PBEKeySpec;
import javax.crypto.spec.SecretKeySpec;
import java.security.spec.KeySpec;
import java.util.Base64;

public class KHash implements AlgoritmoEncriptamiento {
    private static String secretKey;
    private static String salt ;
    private byte[] kHashInitVector = {0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0};
    private IvParameterSpec kHashInitVectorHash;

    @Override
    public void configurar() {
         secretKey = "GHash!!!!";
         salt = "GHash!!!!";
        IvParameterSpec kHashInitVectorHash = new
IvParameterSpec(kHashInitVector);
    }

    @Override
    public String encriptar(String str) {
        System.out.println("Encrypting String with KHash");
        try {
            SecretKeyFactory factory =
SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
            KeySpec spec = new PBEKeySpec(secretKey.toCharArray(),
salt.getBytes(), 65536, 256);
            SecretKey tmp = factory.generateSecret(spec);
            SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");

            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
            cipher.init(Cipher.ENCRYPT_MODE, secretKey, kHashInitVectorHash);
            return
Base64.getEncoder().encodeToString(cipher.doFinal(str.getBytes("UTF-8")));
        } catch (Exception e) {
            System.out.println("Error while encrypting with KHash: " +
e.toString());
        }
        return null;
    }

    @Override
    public String desencriptar(String str) {
        System.out.println("Decrypting String with KHash");
        try {
```

```
            SecretKeyFactory factory = SecretKeyFactory.getInstance("PBKDF2WithHmacSHA256");
            KeySpec spec = new PBEKeySpec(secretKey.toCharArray(), salt.getBytes(), 65536, 256);
            SecretKey tmp = factory.generateSecret(spec);
            SecretKeySpec secretKey = new SecretKeySpec(tmp.getEncoded(), "AES");

            Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5PADDING");
            cipher.init(Cipher.DECRYPT_MODE, secretKey, kHashInitVectorHash);
            return new String(cipher.doFinal(Base64.getDecoder().decode(str)));
        } catch (Exception e) {
            System.out.println("Error while decrypting: " + e.toString());
        }
        return null;
    }

}
```

```
package factory.method;
/**
 * @author Victor Lavalle
*/
public class Cliente {

    public Cliente(){ }

    public String encriptar(String texto, FactoryAlgoEncriptamiento factory) {
        AlgoritmoEncriptamiento encriptador = factory.create();
        return encriptador.encriptar(texto);
    }

    public String desencriptar(String texto, FactoryAlgoEncriptamiento factory) {
        AlgoritmoEncriptamiento desencriptador = factory.create();
        return desencriptador.desencriptar(texto);
    }

  public static void main(String args[]) {
      String cadenaEncriptada;
      cadenaEncriptada = new Cliente().encriptar("Hola",new FactoryAlgoEncripGHash());
      System.out.println(cadenaEncriptada+"\n");

      String cadenaDesencriptada;
      cadenaDesencriptada=new Cliente().desencriptar("0xQM9GGhFwHK4h1TFmFbBw==",new FactoryAlgoEncripGHash());
      System.out.println(cadenaDesencriptada+"\n");

      System.out.println("-------------------------------");

      cadenaEncriptada = new Cliente().encriptar("Hola", new FactoryAlgoEncripKHash());
      System.out.println(cadenaEncriptada+"\n");

       cadenaDesencriptada=new Cliente().desencriptar("jLkPaQStligov95QZ487Zg==",new FactoryAlgoEncripKHash());
      System.out.println(cadenaDesencriptada+"\n");
        }
}
```

Diagrama de Clases

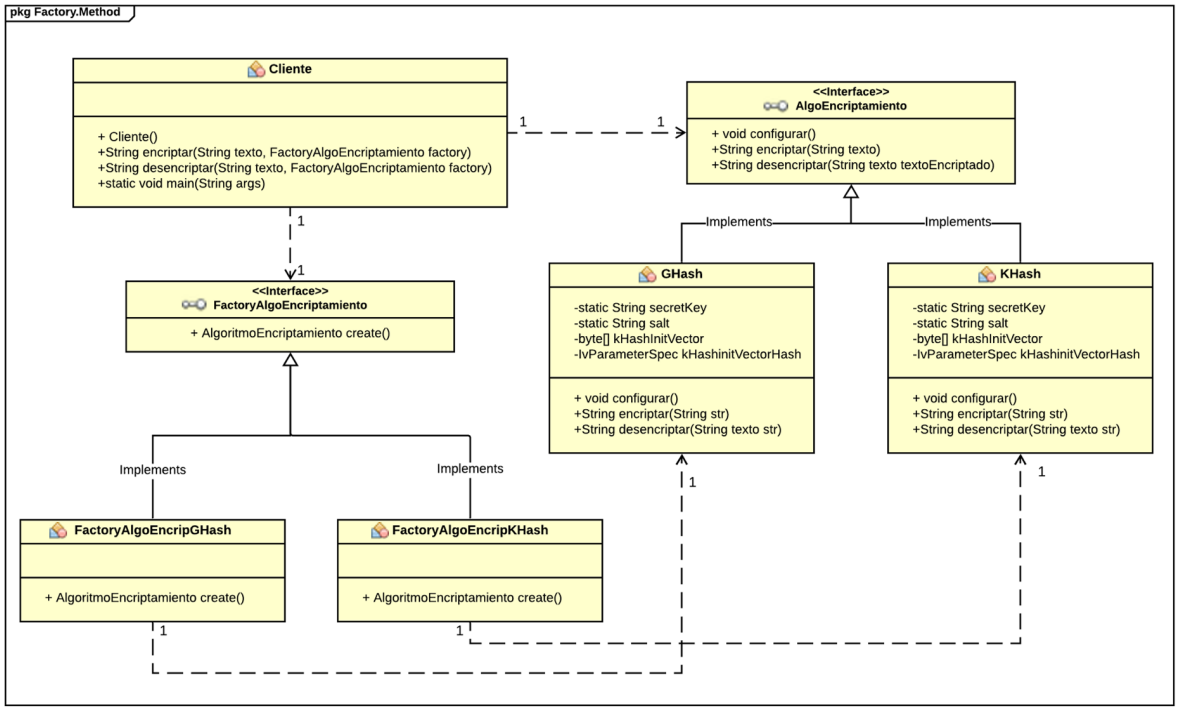
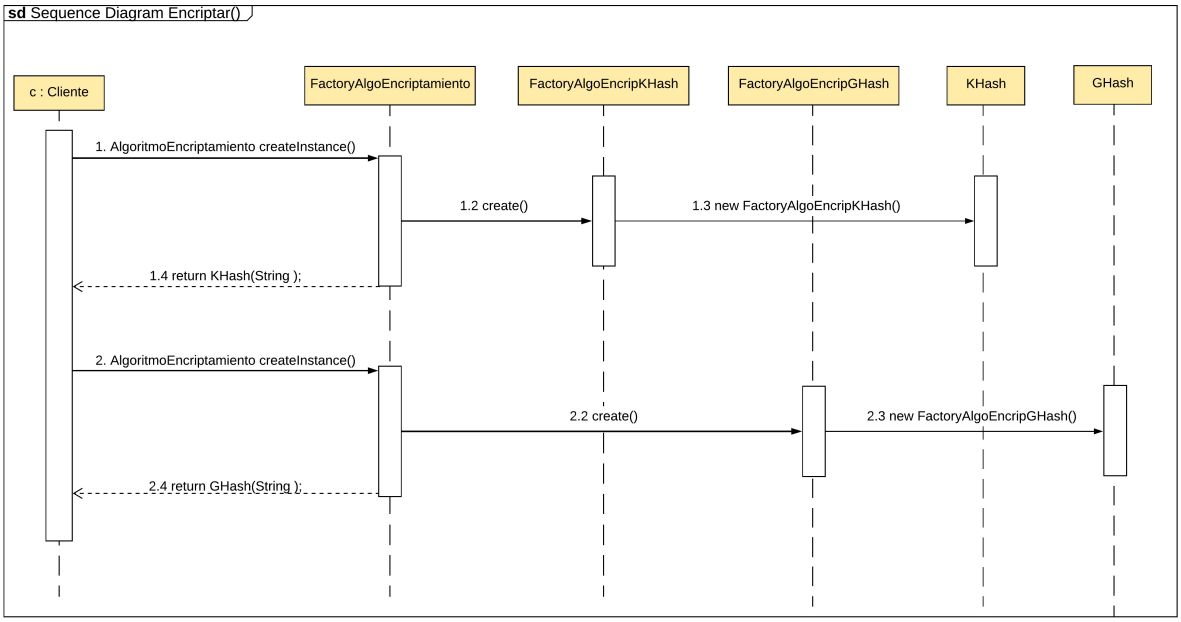


Diagrama de Secuencia



Ejecución del Programa