链滴

流行的加密技术，MD5、DES、DESede、Blowfish

作者：fnyexx
原文链接：https://ld246.com/article/1355409054037
来源网站：链滴

```
<p> </p>
<pre>package com.xiucaike.common;
import java.security.MessageDigest;
import javax.crypto.Cipher;
import javax.crypto.KeyGenerator;
import javax.crypto.SecretKey;
import org.apache.log4j.Logger;
import org.junit.Test;
import sun.misc.BASE64Encoder;

/**

● @author fnyexx

●

● 加密、类型转换工具

*/

@SuppressWarnings(&quot;restriction&qu
t;)

public class UtilCrpts {

private static final Logger logger = Logger.getLogger(UtilCrpts.class);

/** 定义 加密算法,可用 DES,DESede,Blowfish- 默认DES*/

private String  Algorithm =&quot;DES
quot;;

public String getAlgorithm() {

return Algorithm;

}

public void setAlgorithm(String algorithm) {

Algorithm = algorithm;

}


/** 生成密钥,随机生成
 * @return byte[] 返回生成的密钥
 */
 public String createSecretKey() {

    KeyGenerator keygen = null;

    try {
        keygen = KeyGenerator.getInstance(Algorithm);
    } catch (Exception e) {
        logger.error(&quot;NoSuchAlgorithmException!&quot;);
    }
    SecretKey deskey = keygen.generateKey();

    return byte2hex(deskey.getEncoded());
 }
```

```
/**
 * 将指定的数据根据提供的密钥进行加密
 * @param input 需要加密的数据
 * @param key 密钥
 * @return byte[] 加密后的数据
 * @throws Exception
 */
public byte[] encryptData(String input, String key) {

    SecretKey deskey = new javax.crypto.spec.SecretKeySpec(stringToByte(key.replace(&quot;
&quot;, &quot;&quot;)), Algorithm);
    Cipher c1 = null;
    byte[] cipherByte = null;

    try {
        c1 = Cipher.getInstance(Algorithm);
        c1.init(Cipher.ENCRYPT_MODE, deskey);
        cipherByte = c1.doFinal(input.getBytes());
    } catch (Exception e) {
        logger.error(&quot;error!&quot;);
    }

    return cipherByte;
}

/**
 * 将给定的已加密的数据通过指定的密钥进行解密
 * @param input 待解密的数据
 * @param key 密钥
 * @return byte[] 解密后的数据
 * @throws UnsupportedEncodingException
 * @throws Exception
 */
public String decryptData(String input, String key) {

    SecretKey deskey = new javax.crypto.spec.SecretKeySpec(stringToByte(key.replace(&quot;
&quot;, &quot;&quot;)), Algorithm);
    Cipher c1 = null;
    byte[] clearByte = null;
    String decryptStr = null;

    try {
        c1 = Cipher.getInstance(Algorithm);
        c1.init(Cipher.DECRYPT_MODE, deskey);
        clearByte = c1.doFinal(stringToByte(input.replace(&quot;:&quot;, &quot;&quot;)));
        decryptStr = new String(clearByte,&quot;utf-8&quot;);
    } catch (Exception e) {
        logger.error(&quot;error&quot;);
    }

    return decryptStr;
}
```

```
/**
 *  对密码进行MD5加密(非对称加密，不能解密，适用于密码匹配)
 * @param :       String 原始密码
 * @return:       String 密码密文
 */
public String MD5Encrypt(String pwd){

    MessageDigest md = null;

    try {
        md = MessageDigest.getInstance(&quot;MD5&quot;);
    } catch (Exception e) {
        logger.error(&quot;NoSuchAlgorithmException!&quot;);
    }
    md.update(pwd.getBytes());
    byte[] result = md.digest();
    return byteToString(result);
}

/**
 * 字节码转换成16进制字符串
 * @param byte[]b 输入要转换的字节码
 * @return String 返回转换后的16进制字符串
 */
public String byte2hex(byte[] b) {
    String hs = &quot;&quot;;
    String stmp = &quot;&quot;;
    for (int n = 0; n &lt; b.length; n++) {
        stmp = (java.lang.Integer.toHexString(b[n] &amp; 0XFF));
        if (stmp.length() == 1)
            hs = hs + &quot;0&quot; + stmp;
        else
            hs = hs + stmp;
        if (n &lt; b.length - 1)
            hs = hs + &quot;:&quot;;
    }
    return hs.toUpperCase();
}

/**
 * 16进制字符串转换为byte数组 &quot;1212&quot;={12,12}
 * @param strIn
 *            需要转换的字符串
 * @return 转换后的byte数组
 * @throws IllegalArgumentException
 */
public byte[] stringToByte(String hex)
        throws IllegalArgumentException
{

    if (hex.length() % 2 != 0)
    {
        throw new IllegalArgumentException();
```

```
    }

    char[] arr = hex.toCharArray();
    byte[] b = new byte[hex.length() / 2];

    for (int i = 0, j = 0, l = hex.length(); i &lt; l; i++, j++)
    {
        String swap = &quot;&quot; + arr[i++] + arr[i];
        int byteint = Integer.parseInt(swap, 16) &amp; 0xFF;
        b[j] = new Integer(byteint).byteValue();
    }
    return b;
}

/**
 *  二进制 byte[] 转字符串   {12,12}=&quot;1212&quot;
 * @param :       byte[]
 * @return:       String
 * @throws        AppException
 */
public String byteToString(byte[] b) {
    String hs = &quot;&quot;;
    String stmp = &quot;&quot;;
    for (int n = 0; n &lt; b.length; n++) {
        stmp = (Integer.toHexString(b[n] &amp; 0XFF));
        if (stmp.length() == 1) {
            hs = hs + &quot;0&quot; + stmp;
        } else {
            hs = hs + stmp;
        }
        if (n &lt; (b.length - 1)) {
            hs = hs + &quot;&quot;;
        }
    }
    return hs;
}
/**
 * int类型转换为byte[]类型
 * @param input 待转换的int值
 * @return byte[]
 */
public byte[] intTobyte(int input) {
    byte[] output = new byte[4];
    output[0] = (byte) (input &amp; 0xff);
    output[1] = (byte) ((input &gt;&gt; 8) &amp; 0xff);
    output[2] = (byte) ((input &gt;&gt; 16) &amp; 0xff);
    output[3] = (byte) (input &gt;&gt;&gt; 24);
    return output;
}

/**
 * byte[]类型转换为int类型
 * @param input 待转换的byte值
 * @return int
```

```
 */
public int byteToint(byte[] input) {
    int output = 0;
    if (input.length != 4) {

        logger.error(&quot;error!&quot;);
    }
    output = (input[0] &amp; 0xff) | ((input[1] &lt;&lt; 8) &amp; 0xff00)
            | ((input[2] &lt;&lt; 24) &gt;&gt;&gt; 8) | (input[3] &lt;&lt; 24);
    return output;
}


/**
 * 字节数组使用Base64进行编码
 * @param value 待编码的字节数组
 * @return 使用Base64进行编码的字节数组
 */
public String encdoeStrUsingBase64(byte[] value){

    String encodeTxt = &quot;&quot;;
    if( value!=null &amp;&amp; value.length&gt;0 ){
        encodeTxt= new BASE64Encoder().encode(value);
    }

    return encodeTxt;
}

//DES测试   DESede,Blowfish操作一样
@Test
public void test(){

    String key = createSecretKey();
    String text = &quot;123asdf んぁ” ●☆我是好人/.,&quot;;
    String crpt  = byte2hex(encryptData(text , key));

    System.out.println(&quot;生成KEY：&quot; + key);
    System.out.println(&quot;末加密原文 ： &quot; + text);

    System.out.println(&quot;MD5加密后密文：&quot; + MD5Encrypt(text));
    System.out.println(&quot;DES加密后密文：&quot; + crpt);

    System.out.println(&quot;DES解密后的原文：&quot; + decryptData(crpt, key));

}

}</pre>

<p> </p>
```