**.net c#**

using System.Security.Cryptography;

/// <summary>

/// ==========URL DES加密,不支持大小等于号（英文状态下的）=============

/// </summary>

/// <param name="str">要加密的字符串</param>

/// <param name="IV">向量8位</param>

/// <param name="Key">密钥8位</param>

/// <returns></returns>

public static String URLEncrypt(String str, string IV, string Key)

{

byte[] bKey = Encoding.UTF8.GetBytes(Key);

byte[] bIV = Encoding.UTF8.GetBytes(IV);

byte[] bStr = Encoding.UTF8.GetBytes(str);

try

{

DESCryptoServiceProvider desc = new DESCryptoServiceProvider();

MemoryStream mStream = new MemoryStream();

CryptoStream cStream = new CryptoStream(mStream, desc.CreateEncryptor(bKey, bIV), CryptoStreamMode.Write);

cStream.Write(bStr, 0, bStr.Length);

cStream.FlushFinalBlock();

return Convert.ToBase64String(mStream.ToArray()).Replace('+', '\_').Replace('/', '@');

}

catch (Exception e)

{

return "加密失败！" + e.Message;

}

}

/// <summary>

/// ===================== URL DES解密 ========================

/// </summary>

/// <param name="DecryptStr">要解密的字符串</param>

/// <param name="IV">向量8位</param>

/// <param name="Key">密钥8位</param>

/// <returns></returns>

public static String URLDecrypt(String DecryptStr, string IV, string Key)

{

DecryptStr = DecryptStr.Replace('\_', '+').Replace('@', '/');

try

{

byte[] bKey = Encoding.UTF8.GetBytes(Key);

byte[] bIV = Encoding.UTF8.GetBytes(IV);

byte[] bStr = Convert.FromBase64String(DecryptStr);

DESCryptoServiceProvider desc = new DESCryptoServiceProvider();

MemoryStream mStream = new MemoryStream();

CryptoStream cStream = new CryptoStream(mStream, desc.CreateDecryptor(bKey, bIV), CryptoStreamMode.Write);

cStream.Write(bStr, 0, bStr.Length);

cStream.FlushFinalBlock();

return Encoding.UTF8.GetString(mStream.ToArray());

}

catch (Exception e)

{

return "解密失败！" + e.Message;

}

}

**Java**

package com**.**htaiyun**.**utils**;**

**import** java**.**security**.**Key**;**

**import** java**.**security**.**spec**.**AlgorithmParameterSpec**;**

**import** javax**.**crypto**.**Cipher**;**

**import** javax**.**crypto**.**SecretKeyFactory**;**

**import** javax**.**crypto**.**spec**.**DESKeySpec**;**

**import** javax**.**crypto**.**spec**.**IvParameterSpec**;**

**import** org**.**apache**.**commons**.**codec**.**binary**.**Base64**;**

public class DESEncrypt **{**

private static final byte**[]** DESkey **=** "12345678"**.**getBytes**();**// 设置密钥，略去

private static final byte**[]** DESIV **=** "87654321"**.**getBytes**()** **;**// 设置向量，略去

//加密算法的参数接口，IvParameterSpec是它的一个实现

static AlgorithmParameterSpec iv **=** **null;**

private static Key key **=** **null;**

public DESEncrypt**()** **throws** Exception **{**

**this(**DESkey**,**DESIV**);**

**}**

public DESEncrypt**(**String DESkey**,**String DESIV**)** **throws** Exception **{**

**this(**DESkey**.**getBytes**(),**DESIV**.**getBytes**());**

**}**

private DESEncrypt**(**byte**[]** DESkey**,**byte**[]** DESIV**)** **throws** Exception **{**

// 设置密钥参数

DESKeySpec keySpec **=** **new** DESKeySpec**(**DESkey**);**

// 设置向量

iv **=** **new** IvParameterSpec**(**DESIV**);**

// 获得密钥工厂

SecretKeyFactory keyFactory **=** SecretKeyFactory**.**getInstance**(**"DES"**);**

key **=** keyFactory**.**generateSecret**(**keySpec**);**// 得到密钥对象

**}**

/\*\*

\* @param data

\* @return

\* @throws 加密

\*/

public String encode**(**String data**)** **throws** Exception **{**

// 得到加密对象Cipher

Cipher enCipher **=** Cipher**.**getInstance**(**"DES/CBC/PKCS5Padding"**);**

// 设置工作模式为加密模式，给出密钥和向量

enCipher**.**init**(**Cipher**.**ENCRYPT\_MODE**,** key**,** iv**);**

byte**[]** pasByte **=** enCipher**.**doFinal**(**data**.**getBytes**(**"utf-8"**));**

**return** Base64**.**encodeBase64String**(**pasByte**).**replaceAll**(**"\\+"**,** "\_"**).**replaceAll**(**"\\/"**,** "@"**);**

**}**

/\*\*

\* @param data

\* @return

\* @throws 解密

\*/

public String decode**(**String data**)** **throws** Exception **{**

Cipher deCipher **=** Cipher**.**getInstance**(**"DES/CBC/PKCS5Padding"**);**

deCipher**.**init**(**Cipher**.**DECRYPT\_MODE**,** key**,** iv**);**

byte**[]** pasByte **=** deCipher**.**doFinal**(**Base64**.**decodeBase64**(**data**.**replaceAll**(**"\_"**,** "+"**).**replaceAll**(**"@"**,** "/"**)));**

**return** **new** String**(**pasByte**,** "UTF-8"**);**

**}**

public static void main**(**String**[]** args**)** **throws** Exception **{**

DESEncrypt tools **=** **new** DESEncrypt**();**

System**.**out**.**println**(**"加密:" **+** tools**.**encode**(**"http://192.168.1.2/a.doc"**));**

System**.**out**.**println**(**"解密:" **+** tools**.**decode**(**tools**.**encode**(**"http://192.168.1.2/a.doc "**)));**

**}**

**}**

注意：如果加密结果中出现了\n换行符，则必须替换掉。原因是：根据RFC822规定，BASE64Encoder编码每76个字符，还需要加上一个回车换行。部分Base64编码的Java库还按照这个标准实行。

解决方案

1、换用Apache的 commons-codec.jar， Base64.encodeBase64String(byte[]）得到的编码字符串是不带换行符的。

或者：

2、用字符串对象的replaceAll方法替换掉\r和\n，代码如下：replaceAll("[\\s\*\t\n\r]", "");

如果加密结果中出现了中划线“-”，表示使用转换base64字符串函数用错了。不能使用BASE64.encodeBase64URLString，要使用BASE64.encodeBase64String。

**PHP**

**一、PHP 5.5.0以前的加密，采用的是mcrypt\_cbc加密函数：**

<?php

**class** DES

{

**var** $key;

**var** $iv; //偏移量

**function** DES( $key, $iv=0 ) {

//key长度8例如:1234abcd

$this->**key** = $key;

**if**( $iv == 0 ) {

$this->iv = $key; //默认以$key 作为 iv

} **else** {

$this->iv = $iv; //mcrypt\_create\_iv ( mcrypt\_get\_block\_size (MCRYPT\_DES, MCRYPT\_MODE\_CBC), MCRYPT\_DEV\_RANDOM );

}

}

**function** encrypt($str) {

//加密，返回base64编码字符串，之后要替换"+"为 "\_"， "/"为 "@"

$size = **mcrypt\_get\_block\_size** ( **MCRYPT\_DES**, **MCRYPT\_MODE\_CBC** );

$str = $this->pkcs5Pad ( $str, $size );

$sTemp = **base64\_encode** ( ( **mcrypt\_cbc**(**MCRYPT\_DES**, $this->**key**, $str, **MCRYPT\_ENCRYPT**, $this->iv ) ) );

$sTemp = **str\_replace**('+','\_',$sTemp);

$sTemp = **str\_replace**('/','@',$sTemp);

**return** $sTemp;

}

**function** pkcs5Pad($text, $blocksize) {

$pad = $blocksize - (**strlen** ( $text ) % $blocksize);

**return** $text . **str\_repeat** ( **chr** ( $pad ), $pad );

}

}

$sUrl = 'http://www.a.com/test/test.php?id=1';

$ODes = **new** DES('11111111','22222222');

**echo** $ODes->encrypt($sUrl);

?>

**一、PHP 5.5.0以后的加密，采用的是mcrypt\_encrypt加密函数：**

<?php

**class** DES

{

**var** $key; //密钥

**var** $iv; //向量

**function** DES( $key, $iv=0 ) {

//key长度8例如:1234abcd

$this->**key** = $key;

**if**( $iv == 0 ) {

$this->iv = $key; //默认以$key 作为 iv

} **else** {

$this->iv = $iv; //mcrypt\_create\_iv ( mcrypt\_get\_block\_size (MCRYPT\_DES, MCRYPT\_MODE\_CBC), MCRYPT\_DEV\_RANDOM );

}

}

**function** encrypt($str) {

//加密，返回base64编码字符串，之后要替换"+"为 "\_"， "/"为 "@"

$size = **mcrypt\_get\_block\_size** ( **MCRYPT\_DES**, **MCRYPT\_MODE\_CBC** );

$str = $this->pkcs5Pad ( $str, $size );

$sTemp = **base64\_encode** ( ( **mcrypt\_encrypt**(**MCRYPT\_DES**, $this->**key**, $str, **MCRYPT\_MODE\_CBC**, $this->iv ) ) );

$sTemp = **str\_replace**('+','\_',$sTemp);

$sTemp = **str\_replace**('/','@',$sTemp);

**return** $sTemp;

}

**function** pkcs5Pad($text, $blocksize) {

$pad = $blocksize - (**strlen** ( $text ) % $blocksize);

**return** $text . **str\_repeat** ( **chr** ( $pad ), $pad );

}

}

$sUrl = 'http://www.a.com/test/test.php?id=1';

$ODes = **new** DES('1234abcd','1234abcd');

**echo** $ODes->encrypt($sUrl);

?>

**注意：向量必填**