# JS和JAVA前后台混合使用加密解密工具

# 1 对称加密

## 1.1 前台JS

目前只编写了AES和DES对称加密的前后台通用，其他的也非常简单。

### 1.1.1 演示Demo路径

jeesite\src\main\webapp\static\crypto-js\3.1.9-1\crypto.html

<script type="text/javascript">  
 // DES-Encrypt  
 var ***des\_ciphertext*** = *encryptByDES*('我有一个消息', 'des@enc@');  
 ***console***.log("DES加密内容："+***des\_ciphertext***);//06tpmY+9V9mPI6AaHXO+IgeLDlDkWUbN  
 // DES-Decrypt  
 var ***des\_plaintext*** = *decryptByDES*(***des\_ciphertext***.toString(), 'des@enc@');  
 ***console***.log("DES解密内容："+***des\_plaintext***);//我有一个消息  
  
 // AES-Encrypt  
 var ***aes\_ciphertext*** = *encryptByAES*('我有一个消息', 'aes@encrypt@key@');  
 ***console***.log("AES加密内容："+***aes\_ciphertext***);//SFNUNGMqvMrMdP9+00Iov6BiefbHpN3e0KTMWo/nHtI=  
 // AES-Decrypt  
 var ***aes\_plaintext*** = *decryptByAES*(***aes\_ciphertext***.toString(), 'aes@encrypt@key@');  
 ***console***.log("AES解密内容："+***aes\_plaintext***);//我有一个消息  
  
 /\*\*  
 \* DES加密  
 \* @param message  
 \* @param key  
 \* @returns {string}  
 \*/  
 function *encryptByDES*(message, key) {  
 var keyHex = CryptoJS.enc.Utf8.parse(key);  
 var encrypted = CryptoJS.DES.encrypt(message, keyHex, {  
 mode: CryptoJS.mode.ECB,  
 padding: CryptoJS.pad.Pkcs7  
 });  
 return encrypted.toString();  
 }  
  
 /\*\*  
 \* DES解密  
 \* @param message  
 \* @param key  
 \* @returns {string}  
 \*/  
 function *decryptByDES*(message, key) {  
 var keyHex = CryptoJS.enc.Utf8.parse(key);  
 var decrypted = CryptoJS.DES.decrypt(message, keyHex, {  
 mode: CryptoJS.mode.ECB,  
 padding: CryptoJS.pad.Pkcs7  
 });  
 return decrypted.toString(CryptoJS.enc.Utf8);  
 }  
  
 /\*\*  
 \* AES加密  
 \* @param message  
 \* @param key  
 \* @returns {string}  
 \*/  
 function *encryptByAES*(message, key) {  
 var keyHex = CryptoJS.enc.Utf8.parse(key);  
 var encrypted = CryptoJS.AES.encrypt(message, keyHex, {  
 mode: CryptoJS.mode.ECB,  
 padding: CryptoJS.pad.Pkcs7  
 });  
 return encrypted.toString();  
 }  
  
 /\*\*  
 \* AES解密  
 \* @param message  
 \* @param key  
 \* @returns {string}  
 \*/  
 function *decryptByAES*(message, key) {  
 var keyHex = CryptoJS.enc.Utf8.parse(key);  
 var decrypted = CryptoJS.AES.decrypt(message, keyHex, {  
 mode: CryptoJS.mode.ECB,  
 padding: CryptoJS.pad.Pkcs7  
 });  
 return decrypted.toString(CryptoJS.enc.Utf8);  
 }  
</script>

### 1.1.2 Console日志

DES加密内容：06tpmY+9V9mPI6AaHXO+IgeLDlDkWUbN

DES解密内容：我有一个消息

AES加密内容：SFNUNGMqvMrMdP9+00Iov6BiefbHpN3e0KTMWo/nHtI=

AES解密内容：我有一个消息

## 1.2 后台JAVA

### 1.2.1 演示Demo类

com.thinkgem.jeesite.common.security.util. CipherUtil

//DES的key长度为8位的字符串，否则会报错  
***log***.info("前后台通用DES加密：{}",encodeDES("des@enc@","我有一个消息"));  
***log***.info("前后台通用DES解密：{}",decodeDES("des@enc@","06tpmY+9V9mPI6AaHXO+IgeLDlDkWUbN"));  
//AES的key长度为16位的字符串，否则会报错  
***log***.info("前后台通用AES加密：{}",encodeAES("aes@encrypt@key@","我有一个消息"));  
***log***.info("前后台通用AES加密：{}",decodeAES("aes@encrypt@key@","SFNUNGMqvMrMdP9+00Iov6BiefbHpN3e0KTMWo/nHtI="));

### 1.2.2 Console日志

2018-06-24 09:55:51,275 INFO [jeesite.common.security.util.CipherUtil] - 前后台通用DES加密：06tpmY+9V9mPI6AaHXO+IgeLDlDkWUbN

2018-06-24 09:55:51,278 INFO [jeesite.common.security.util.CipherUtil] - 前后台通用DES解密：我有一个消息

2018-06-24 09:55:51,278 INFO [jeesite.common.security.util.CipherUtil] - 前后台通用AES加密：SFNUNGMqvMrMdP9+00Iov6BiefbHpN3e0KTMWo/nHtI=

2018-06-24 09:55:51,281 INFO [jeesite.common.security.util.CipherUtil] - 前后台通用AES加密：我有一个消息

# 2 非对称加密

## 2.1 前台JS

目前只编写了RSA非对称加密的前后台通用，其他的也非常简单。

### 2.1.1 演示Demo路径

## 2.2 后台JAVA

### 2.2.1 演示Demo类

附：

前台加密算法demo（DES、AES）：<https://github.com/brix/crypto-js>

前台加密算法demo（RSA）：<https://github.com/travist/jsencrypt>

后台加密算法demo：<https://gitee.com/FlyTank/secure-util>