# Java常用加解密算法

## 开源实现

谷歌CryptoJS： <https://github.com/brix/crypto-js>

Bitwiseshiftleft-sjcl：<https://github.com/bitwiseshiftleft/sjcl>

Java实现：<https://github.com/lugeek/Encryption/>

## 常用算法介绍

【Java安全】关于Java中常用加密/解密方法的实现：<https://www.cnblogs.com/tancky/p/6409823.html>

JAVA中常用的加密算法总结：

<https://blog.csdn.net/chenbing81/article/details/51914151>

<http://blog.sina.com.cn/s/blog_6259ebd50102v57c.html>

### base64

### 原理：

Base64的码表只有64个字符， 如果要表达64个字符的话，使用6的bit即可完全表示(2的6次方为64)。

因为Base64的编码只有6个bit即可表示，而正常的字符是使用8个bit表示， 8和6的最小公倍数是24，所以4个Base64字符可以表示3个标准的ascll字符；

如果是字符串转换为Base64码， 会先把对应的字符串转换为ascll码表对应的数字， 然后再把数字转换为2进制， 比如a的ascll码味97， 97的二进制是：01100001， 把8个二进制提取成6个，剩下的2个二进制和后面的二进制继续拼接， 最后再把6个二进制码转换为Base64对于的编码，（当转换到最后， 最后的字符不足3个字符咋办， 如果不足三个字符的话，我们直接在最后添加‘＝’） 以下为具体的解析过程案例：

字符串 a b c

ASCII 97 98 99

8bit 01100001 01100010 01100011

6bit 011000 010110 001001 100011

十进制 24 22 9 35

对应编码 Y W J j

#### 应用场景

加密URL

传输二进制数据，如邮件主题及图片等

加密cookie

#### 说明：

Base64是网络上最常见的用于传输8Bit字节码的编码方式之一，Base64就是一种基于64个可打印字符来表示二进制数据的方法。

Base64要求把每三个8Bit的字节转换为四个6Bit的字节（3\*8 = 4\*6 = 24），然后把6Bit再添两位高位0，组成四个8Bit的字节，也就是说，转换后的字符串理论上将要比原来的长1/3。

ase64索引表，字符选用了"A-Z、a-z、0-9、+、/" 64个可打印字符

#### 应用：

Base64编码可用于在HTTP环境下传递较长的标识信息。

Mozilla Thunderbird和Evolution用Base64来保密电子邮件密码。

Base64 也会经常用作一个简单的“加密”来保护某些数据，而真正的加密通常都比较繁琐。

垃圾讯息传播者用Base64来避过反垃圾邮件工具，因为那些工具通常都不会翻译Base64的讯息。

在LDIF档案，Base64用作编码字串。

### AES

### 原理：

参考链接：<https://blog.csdn.net/simple_man_just/article/details/69258923>

AES加密过程涉及到4种操作：字节替代（SubBytes）、行移位（ShiftRows）、列混淆（MixColumns）和轮密钥加（AddRoundKey）。解密过程分别为对应的逆操作。由于每一步操作都是可逆的，按照相反的顺序进行解密即可恢复明文。加解密中每轮的密钥分别由初始密钥扩展得到。算法中16字节的明文、密文和轮密钥都以一个4x4的矩阵表示

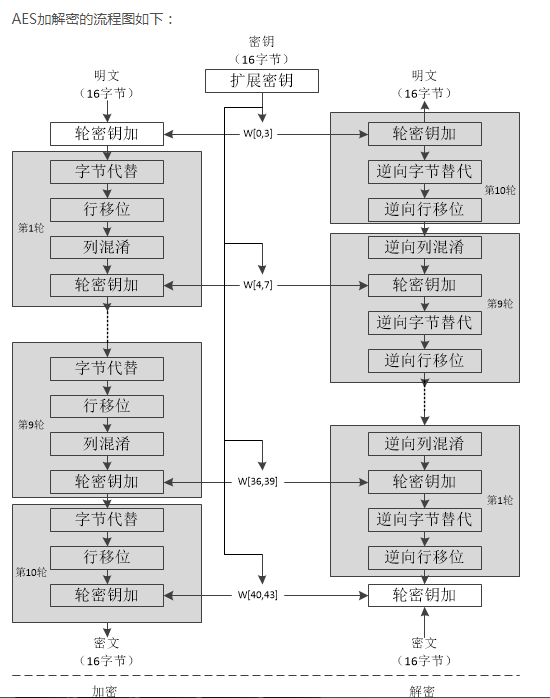
AES 加密过程在一个称为 “体 (state)” 的44字节矩阵上进行，其初值是一个16字节的明文区块 (Rijndael 加密法因支持更大的区块，其矩阵行数可视情况增加)。加密时，各轮 AES 加密循环 (除最后一轮外) 均包含4个步骤。

AddRoundKey ‐ 矩阵中的每一个字节都与该轮密钥(round key) 做 XOR 运算；每个子密钥由密钥生成方案产生。

SubBytes ‐ 通过一个非线性的替换函数，用查找表的方式把每个字节替换成对应的字节。

ShiftRows ‐ 将矩阵中的每个横列进行循环式移位。

MixColumns ‐ 为了充分混合矩阵中各个直行的操作。这个步骤使用线性转换来混合每内联的四个字节。最后一个加密循环中省略 MixColumns 步骤，而以另一个 AddRoundKey 取代。



#### 应用场景

#### 说明：

　 密码算法要求是可逆的，这样解密算法才能正确的恢复明文。拿AES来说，在密钥固定的情况下，明文和密文在整个输入空间是一一对应的。因此算法的各个部件也都是可逆的，再将各个部件的操作顺序设计成可逆的，密文就能正确的解密了。

美国国家标准技术研究所在2001年发布了高级加密标准（AES）。AES是一个对称分组密码算法，旨在取代DES成为广泛使用的标准。

　　根据使用的密码长度，AES最常见的有3种方案，用以适应不同的场景要求，分别是AES-128、AES-192和AES-256。本文主要对AES-128进行介绍，另外两种的思路基本一样，只是轮数会适当增加

#### 应用：

实现参考：

<https://github.com/wwwtyro/cryptico>

<https://github.com/digitalbazaar/forge#aes>

<https://github.com/xuzongbao/encryption>

### RSA

## 1.1 常用加密解密算法

### 相关概念

数据加密算法：

<https://baike.baidu.com/item/%E6%95%B0%E6%8D%AE%E5%8A%A0%E5%AF%86%E7%AE%97%E6%B3%95/3030864>

公钥与私钥：

<http://www.blogjava.net/yxhxj2006/archive/2012/10/15/389547.html>

单向加密 对称加密 非对称加密：

<https://www.cnblogs.com/yx520zhao/p/6658688.html>

数据加密的一些浅谈总结：

<https://blog.csdn.net/maiwc/article/details/51766131>