知乎 首发于 极乐科技

一份简明的 Base64 原理解析

mzlogin

关注他

46 人赞同了该文章

书接上回，在 记一个 Base64 有关的 Bug 一文里，我们说到了 Base64 的编解码器有不同实现，交叉使用它们可能引发的问题等等。

这一回，我们来对 Base64 这一常用编解码技术的原理一探究竟。

1. Base64 是什么

> Base64 是一种基于 64 个可打印字符来表示二进制数据的表示方法。由于 2^6=64，所以每 6 个比特为一个单元，对应某个可打印字符。3 个字节有 24 个比特，对应于 4 个 Base64 单元，即 3 个字节可由 4 个可打印字符来表示。
> —维基百科

它不是一种加解密技术，是一种简单的编解码技术。

Base64 常用于表示、传输、存储二进制数据，也可以用于将一些含有特殊字符的文本内容编码，以便传输。

比如：

1. 在电子邮件的传输中，Base64 可以用来将 binary 的字节序列，比如附件，编码成 ASCII 字节序列；
2. 将一些体积不大的图片 Base64 编码后，直接内嵌到网页源码里；
3. 将要传递给 HTTP 请求的参数做简单的转换，降低肉眼可读性；
   注：用于 URL 的 Base64 非标准 Base64，是一种变种。
4. 网友们在论坛等公开场合习惯将邮箱地址 Base64 后再发出来，防止被爬虫抓取后发送垃圾邮件。

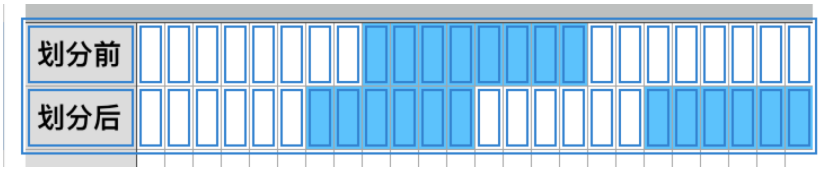
2. Base64 编码原理

标准 Base64 里的 64 个可打印字符是 `A-Za-z0-9+/` ，分别依次对应索引值 0-63。索引表如下：

▲ 赞同 46 ▼ 添加评论 分享 喜欢 收藏 申请转载 ···

数值	字符		数值	字符		数值	字符		数值	字符
0	A		16	Q		32	g		48	w
1	B		17	R		33	h		49	x
2	C		18	S		34	i		50	y
3	D		19	T		35	j		51	z
4	E		20	U		36	k		52	0
5	F		21	V		37	l		53	1
6	G		22	W		38	m		54	2
7	H		23	X		39	n		55	3
8	I		24	Y		40	o		56	4
9	J		25	Z		41	p		57	5
10	K		26	a		42	q		58	6
11	L		27	b		43	r		59	7
12	M		28	c		44	s		60	8
13	N		29	d		45	t		61	9
14	O		30	e		46	u		62	+
15	P		31	f		47	v		63	/

编码时，每 3 个字节一组，共 8bit*3=24bit，划分成 4 组，即每 6bit 代表一个编码后的索引值，划分如下图所示：



这样可能不太直观，举个例子就容易理解了。比如我们对 cat 进行编码：

原文	c								a								t							
ASCII 编码	99								97								116							
二进制位	0	1	1	0	0	0	1	1	0	1	1	0	0	0	0	1	0	1	1	1	0	1	0	0
索引	24								54								5							52
Base64	Y								2								F							0

可以看到 cat 编码后变成了 Y2F0 。

如果待编码内容的字节数不是 3 的整数倍，那需要进行一些额外的处理。

如果最后剩下 1 个字节，那么将补 4 个 0 位，编码成 2 个 Base64 字符，然后补两个 = ：

ASCII 编码	99			
二进制位	0 1 1 0 0 0 1 1	0 0 0 0		
索引	24	48		
Base64	Y	w	=	=

如果最后剩下 2 个字节，那么将补 2 个 0 位，编码成 3 个 Base64 字符，然后补一个 = ：

原文	c	a		
ASCII 编码	99	97		
二进制位	0 1 1 0 0 0 1 1	0 1 1 0 0 0 0 1	0 0	
索引	24	54	4	
Base64	Y	2	E	=

3. 实现一个简易的 Base64 编码器

讲完原理，我们就可以动手实现一个简易的标准 Base64 编码器了，以下是我参考 Java 8 的 java.util.Base64 乱写的一个 Java 版本，仅供参考，主要功能代码如下：

```
public class CustomBase64Encoder {

    /**
     * 索引表
     */
    private static final char[] sBase64 = {
            'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
            'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
            'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X',
            'Y', 'Z', 'a', 'b', 'c', 'd', 'e', 'f',
            'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n',
            'o', 'p', 'q', 'r', 's', 't', 'u', 'v',
            'w', 'x', 'y', 'z', '0', '1', '2', '3',
            '4', '5', '6', '7', '8', '9', '+', '/'
    };

    /**
     * 将 byte[] 进行 Base64 编码并返回字符串
     * @param src 原文
     * @return 编码后的字符串
     */
    public static String encode(byte[] src) {
        if (src == null) {
            return null;
        }

        byte[] dst = new byte[(src.length + 2) / 3 * 4];

        int index = 0;

        // 每次将 3 个字节编码为 4 个字节
        for (int i = 0; i < (src.length / 3 * 3); i += 3) {
            int bits = (src[i] & 0xff) << 16 | (src[i + 1] & 0xff) << 8 | (src[i
            dst[index++] = (byte) sBase64[(bits >>> 18) & 0x3f];
            dst[index++] = (byte) sBase64[(bits >>> 12) & 0x3f];
```

```
        // 处理剩下的 1 个或 2 个字节
        if (src.length % 3 == 1) {
            int bits = (src[src.length - 1] & 0xff) << 4;
            dst[index++] = (byte) sBase64[(bits >>> 6) & 0x3f];
            dst[index++] = (byte) sBase64[bits & 0x3f];
            dst[index++] = '=';
            dst[index] = '=';
        } else if (src.length % 3 == 2) {
            int bits = (src[src.length - 2] & 0xff) << 10 | (src[src.length - 1]
            dst[index++] = (byte) sBase64[(bits >>> 12) & 0x3f];
            dst[index++] = (byte) sBase64[(bits >>> 6) & 0x3f];
            dst[index++] = (byte) sBase64[bits & 0x3f];
            dst[index] = '=';
        }

        return new String(dst);
    }
}
```

这部分源码我也上传到 GitHub 仓库 github.com/mzlogin/spri... 的 base64test 工程里了。

4. 其它知识点

4.1 为什么有的编码结果带回车

在电子邮件中，根据 RFC 822 规定，每 76 个字符需要加上一个回车换行，所以有些编码器实现，比如 sun.misc.BASE64Encoder.encode，是带回车的，还有 java.util.Base64.Encoder.RFC2045，是带回车换行的，每行 76 个字符。

4.2 Base64 的变种

除了标准 Base64 之外，还有一些其它的 Base64 变种。

比如在 URL 的应用场景中，因为标准 Base64 索引表中的 / 和 + 会被 URLEncoder 转义成 %XX 形式，但 % 是 SQL 中的通配符，直接用于数据库操作会有问题。此时可以采用 URL Safe 的编码器，索引表中的 /+ 被换成 -_，比如 java.util.Base64.Encoder.RFC4648_URLSAFE 就是这样的实现。

5. 参考链接

https://zh.wikipedia.org/zh-hans/Base64
zh.wikipedia.org/zh-hans/Base64

base64
www.liaoxuefeng.com/wiki/89769288872…

更多精彩内容，欢迎关注我的公众号「闷骚的程序员」。

发布于 2020-03-08 12:51

编程 base64 Java