

[首页](#) [资讯](#) [精华](#) [论坛](#) [问答](#) [博客](#) [专栏](#) [群组](#) [更多 ▼](#)

[您还未登录！](#) [登录](#) [注册](#)

[suflow](#)

- [博客](#)
- [微博](#)
- [相册](#)
- [收藏](#)
- [留言](#)
- [关于我](#)

[二维码 编码原理简介](#)

博客分类:

- [Java](#)
- [图形图像](#)
- [移动开发](#)

[二维码算法编码](#)

一、什么是二维码:

二维码 (2-dimensional bar code) , 是用某种特定的几何图形按一定规律在平面 (二维方向上) 分布的黑白相间的图形记录数据符号信息的。

在许多种类的二维条码中, 常用的码制有: Data Matrix, Maxi Code, Aztec, QR Code, Vericode, PDF417, Ultracode, Code 49, Code 16K等。

1. 堆叠式/行排式二维条码, 如, Code 16K、Code 49、PDF417 (如下图) 等



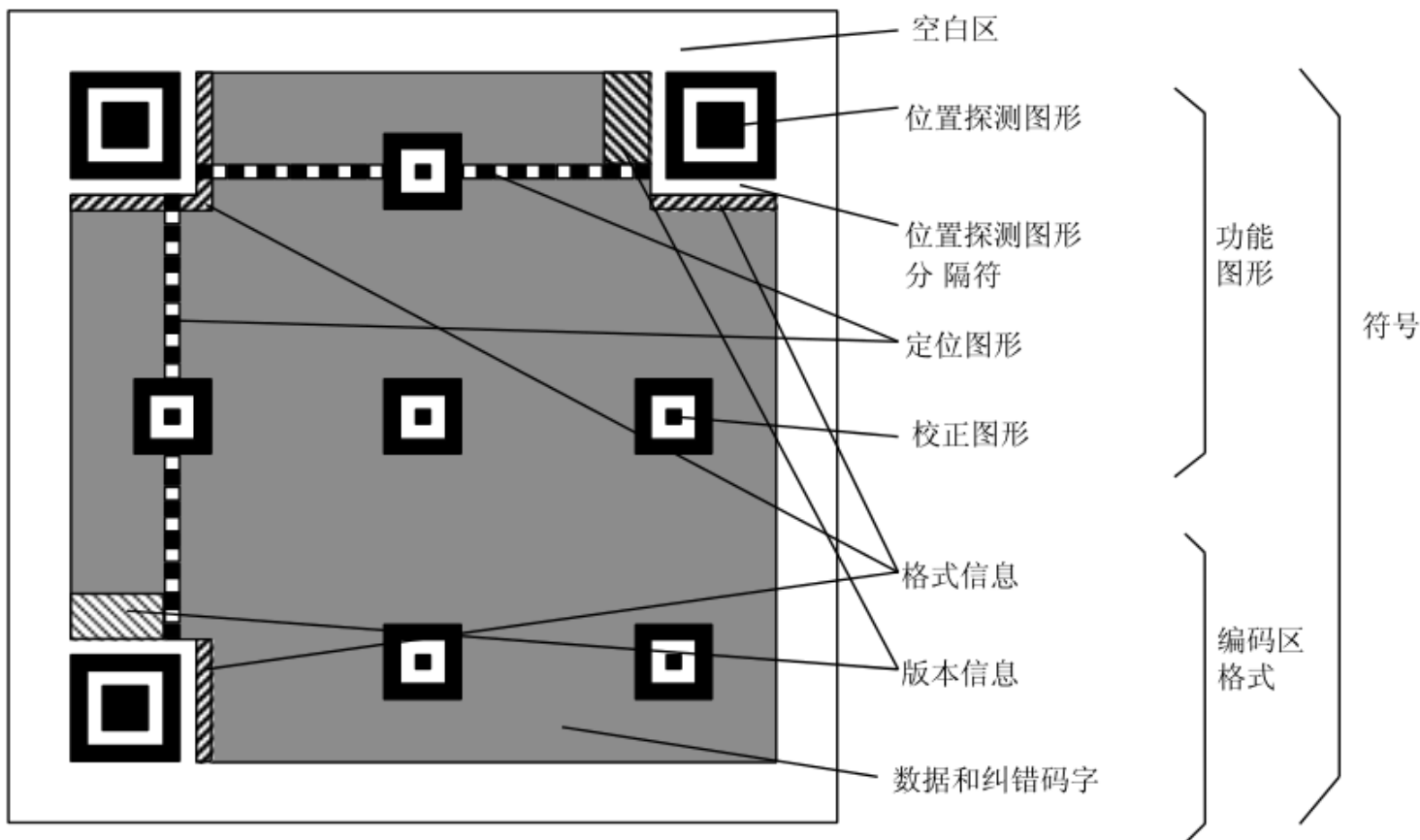
2. 矩阵式二维码, 最流行莫过于QR CODE

二维码的名称是相对与一维码来说的, 比如以前的条形码就是一个“一维码”, 它的优点有: 二维码存储的数据量更大; 可以包含数字、字符, 及中文文本等混合内容; 有一定的容错性 (在部分损坏以后可以正常读取); 空间利用率高等。

二、QR CODE 介绍

QR(Quick-Response) code是被广泛使用的一种二维码, 解码速度快。

它可以存储多用类型



如上图时一个qrcode的基本结构，其中：

位置探测图形、位置探测图形分隔符、定位图形：用于对二维码的定位，对每个QR码来说，位置都是固定存在的，只是大小规格会有所差异；

校正图形：规格确定，校正图形的数量和位置也就确定了；

格式信息：表示改二维码的纠错级别，分为L、M、Q、H；

版本信息：即二维码的规格，QR码符号共有40种规格的矩阵（一般为黑白色），从21x21（版本1），到177x177（版本40），每一版本符号比前一版本 每边增加4个模块。

数据和纠错码字：实际保存的二维码信息，和纠错码字（用于修正二维码损坏带来的错误）。

简要的编码过程：

1. 数据分析：确定编码的字符类型，按相应的字符集转换成符号字符；选择纠错等级，在规格一定的条件下，纠错等级越高其真实数据的容量越小。

2. 数据编码：将数据字符转换为位流，每8位一个码字，整体构成一个数据的码字序列。其实知道这个数据码字序列就知道了二维码的数据内容。

QR码资料容量	
数字	最多7,089字符
字母	最多4,296字符
二进制数 (8 bit)	最多2,953 字节
日文汉字 / 片假名	最多1,817字符 (采用Shift JIS)
中文汉字	最多984字符 (采用UTF-8)
中文汉字	最多1,800字符 (采用BIG5)

模式	指示符
ECI	0111
数字	0001
字母数字	0010
8 位字节	0100
日本汉字	1000
中国汉字	1101
结构链接	0011
FNC1	0101 (第一位置) 1001 (第二位置)
终止符 (信息结尾)	0000

数据可以按照一种模式进行编码，以便进行更高效的解码，例如：对数据：01234567编码（版本1-H），

1) 分组：012 345 67

2) 转成二进制：012→0000001100

345→0101011001

67 →1000011

3) 转成序列：0000001100 0101011001 1000011

4) 字符数 转成二进制：8→0000001000

5) 加入模式指示符（上图数字）0001：0001 0000001000 0000001100 0101011001 1000011

对于字母、中文、日文等只是分组的方式、模式等内容有所区别。基本方法是一致的

3. 纠错编码：按需要将上面的码字序列分块，并根据纠错等级和分块的码字，产生纠错码字，并把纠错码字加入到数据码字序列后面，成为一个新的序列。

错误修正容量	
L水平	7%的字码可被修正
M水平	15%的字码可被修正
Q水平	25%的字码可被修正
H水平	30%的字码可被修正

在二维码规格和纠错等级确定的情况下，其实它所能容纳的码字总数和纠错码字数也就确定了，比如：版本10，纠错等级时H时，总共能容纳346个码字，其中224个纠错码字。

就是说二维码区域中大约1/3的码字是冗余的。对于这224个纠错码字，它能够纠正112个替代错误（如黑白颠倒）或者224个据读错误（无法读到或者无法译码），

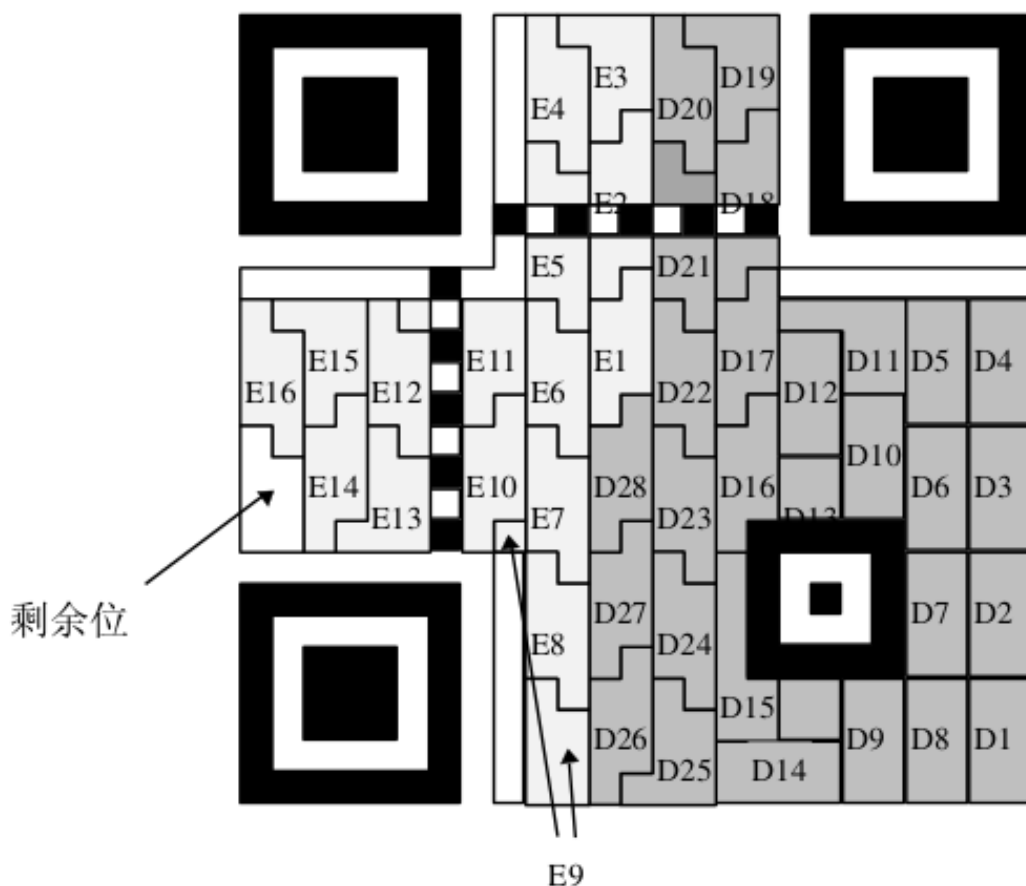
这样纠错容量为： $112/346=32.4\%$

4. 构造最终数据信息：在规格确定的条件下，将上面产生的序列按次序放如分块中

按规定把数据分块，然后对每一块进行计算，得出相应的纠错码字区块，把纠错码字区块按顺序构成一个序列，添加到原先的数据码字序列后面。

如：D1, D12, D23, D35, D2, D13, D24, D36, ... D11, D22, D33, D45, D34, D46, E1, E23, E45, E67, E2, E24, E46, E68, ...

构造矩阵：将探测图形、分隔符、定位图形、校正图形和码字模块放入矩阵中。



把上面的完整序列填充到相应规格的二维码矩阵的区域中

6. 掩模：将掩模图形用于符号的编码区域，使得二维码图形中的深色和浅色（黑色和白色）区域能够比率最优的分布。

一个算法，不研究了，有兴趣的同学可以继续。

7. 格式和版本信息：生成格式和版本信息放入相应区域内。

版本7-40都包含了版本信息，没有版本信息的全为0。二维码上两个位置包含了版本信息，它们是冗余的。

版本信息共18位，6X3的矩阵，其中6位时数据为，如版本号8，数据位的信息为 001000，后面的12位是纠错位。

至此，二维码的编码流程基本完成了，下面就来实践一下吧，当然不用自己再去编写上面的算法了，使用三方包zxing 就可以了

编码：

```
public static void encode(String content, String format, String filePath) {
try {
    Hashtable hints = new Hashtable();//设置编码类型
    hints.put(EncodeHintType.CHARACTER_SET, DEFAULT_ENCODING);
    //编码
    BitMatrix bitMatrix = new QRCodeWriter().encode(content,
BarcodeFormat.QR_CODE, DEFAULT_IMAGE_WIDTH,
DEFAULT_IMAGE_HEIGHT, hints);
    //输出到文件，也可以输出到流
    File file = new File(filePath);
    MatrixToImageWriter.writeToFile(bitMatrix, format, file);

} catch (IOException e) {
    e.printStackTrace();
} catch (WriterException e1) {
    e1.printStackTrace();
}
}
```

解码： BufferedImage image = ImageIO.read(file);//读取文件
LuminanceSource source = new BufferedImageLuminanceSource(image);
BinaryBitmap bitmap = new BinaryBitmap(new HybridBinarizer(
source));

//解码

```
Result result = new MultiFormatReader().decode(bitmap);
String resultStr = result.getText();
System.out.println(resultStr);
```

Done！轮到你了！

参考内容及资料：



<http://zh.wikipedia.org/wiki/QR%E7%A2%BC>

<http://code.google.com/p/zxing/>

<http://www.google.com/>

qrcoe编码码标准

- [查看图片附件](#)

分享到:  

[Ubuntu设置中文编码和文本的编码转换](#)

- 2011-06-22 11:13
- 浏览 43756
- [评论\(4\)](#)
- 分类:[编程语言](#)
- [相关推荐](#)

评论

4 楼 [mfdefs](#) 2014-11-17

nice ! !

3 楼 [浙沥枫](#) 2012-12-05

不错哟.....

2 楼 [zhouwei849712382](#) 2012-11-27



1 楼 [hillshills](#) 2012-05-02

了解了! 😊


发表评论



[您还没有登录,请您登录后再发表评论](#)



suflow

- 浏览: 79658 次
- 性别: 
- 来自: 杭州