## 一维码与二维码



一维码是由粗细不同、黑白相间的条或空及相应的字符组成的标记即传统条码

二维码是用某种特定的几何图案按一定规律在平面上的条、空间的图形记录数据符号信息

## 二维码分类

### 线性堆叠式二维码

编码原理：建立在一维条码的基础上,按需要堆积成两行或多行



### 矩阵式二维码

编码原理：在一个矩形空间通过黑白像素在矩阵中的不同分布进行编码

在矩阵的相应元素位置上，用点的出现表示二进制1，点的不出现表示二进制0

 二维码可容纳：1850个大写字母/2710个数字/1108个字节/500多个汉字

## 二维码国际标准

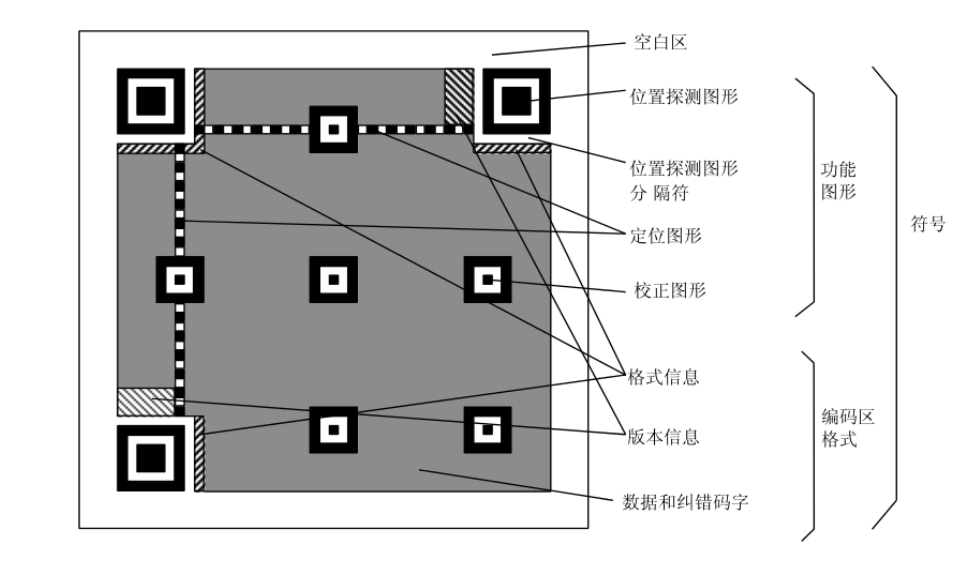
PDF417：不支持中文

DM：专利未公开，需要支付专利费用

QR Code:专利公开，支持中文

QR Code与其他二维码相比，具有识读速度快、数据密度大、占用空间小的优势

## 二维码结构



## php生成二维码

### phpQRcode：

官方文档：

<http://phpqrcode.sourceforge.net/>

#### 生成普通二维码

*/\*\*  
 \* 生成二维码  
 \** ***@author*** *xichao<952002423****@qq****.com>  
 \* 错误级别  
 \* define('QR\_ECLEVEL\_L', 0); 7%  
 \* define('QR\_ECLEVEL\_M', 1); 15%  
 \* define('QR\_ECLEVEL\_H', 3); 25%  
 \* define('QR\_ECLEVEL\_Q', 2); 30%  
 \*/***function** makeQrcode($url){  
*// import('phpqrcode.phpqrcode',VENDOR\_PATH,'.php');* Vendor(**'phpqrcode.phpqrcode'**);  
 *//二维码内容* $value = $url;  
 */\*\*  
 \* 默认为否，不生成文件，只将二维码图片返回  
 \* 否则需要给出存放生成二维码图片的路径  
 \*/* $outfile=**"quicknew.png"**;  
 *//$outfile='qrcode/'.time().'png';  
 //容错级别  
 //不同的参数表示二维码可被覆盖的区域百分比* $errorLevel = **'L'**;  
 *//生成图片的大小 默认为4* $size = 5;  
 *//生成二维码的空白区域大小* $margin=2;  
 *//保存二维码图片并显示出来，$outfile必须传递图片路径* $saveandprint=**true**;  
 *//生成二维码图片* \QRcode::*png*($value,$outfile , $errorLevel, $size, $margin,$saveandprint);  
 *//获取图片类型* $type = *getimagesize*($outfile)[**'mime'**];  
 *//获取图片二进制流* $imgData = *file\_get\_contents*($outfile);  
 *//图片二进制流转base64* $base64String = **'data:'** . $type . **';base64,'** . *base64\_encode*($imgData);  
 **return** $base64String;  
*// $QR = $filename; //已经生成的原始二维码图片文件  
// $QR = imagecreatefromstring(file\_get\_contents($QR));  
// //输出图片  
// imagepng($QR, 'qrcode.png');  
// imagedestroy($QR);*}

#### 生成带logo二维码

*/\*\*  
 \* 生成带logo的二维码  
 \** ***@param*** *$url  
 \** ***@author*** *xichao<952002423****@qq****.com>  
 \*/***function** makePicQrcode($url){  
 Vendor(**'phpqrcode.phpqrcode'**);  
 *//二维码内容* $value = $url;  
 */\*\*  
 \* 默认为否，不生成文件，只将二维码图片返回  
 \* 否则需要给出存放生成二维码图片的路径  
 \*/* $outfile=**"quicknew.png"**;  
 *//$outfile='qrcode/'.time().'png';  
 //容错级别  
 //不同的参数表示二维码可被覆盖的区域百分比* $errorLevel = **'L'**;  
 *//生成图片的大小 默认为4* $size = 12;  
 *//生成二维码的空白区域大小* $margin=2;  
 *//保存二维码图片并显示出来，$outfile必须传递图片路径* $saveandprint=**true**;  
 *//生成二维码图片* \QRcode::*png*($value,$outfile , $errorLevel, $size, $margin,$saveandprint);  
 $logo=**"logo.png"**;  
 $QR=$outfile;  
 */\*\*  
 \* 将logo嵌套在二维码中  
 \* 参数一：目标图象连接资源  
 \* 参数二：源图象连接资源。  
 \* 参数三：目标 X 坐标点  
 \* 参数四： 目标 Y 坐标点 logo图片嵌入后坐标  
 \* 参数五： 源的 X 坐标点 logo图片嵌入前坐标 相对移动原图X坐标  
 \* 参数六：源的 Y 坐标点  
 \* 参数七：目标宽度  
 \* 参数八：目标高度 logo图片嵌入后高度  
 \* 参数九：源图像宽度 logo图片嵌入前高度  
 \* 参数十：源图像高度  
 \*/* **if** ($logo !== **FALSE**) {  
 $QR = *imagecreatefromstring*(*file\_get\_contents*($QR));  
 $logo = *imagecreatefromstring*(*file\_get\_contents*($logo));  
 $QR\_width = *imagesx*($QR);*//二维码图片宽度* $QR\_height = *imagesy*($QR);*//二维码图片高度* $logo\_width = *imagesx*($logo);*//logo图片宽度* $logo\_height = *imagesy*($logo);*//logo图片高度* $logo\_qr\_width = $QR\_width\*0.4; *//logo图片在二维码图片中宽度大小* $scale = $logo\_width/$logo\_qr\_width;  
 $logo\_qr\_height = $logo\_height/$scale; *//logo图片在二维码图片中高度大小  
 // 目标 X 坐标点* $from\_width = ($QR\_width - $logo\_qr\_width)/2;  
 *//重新组合图片并调整大小  
 imagecopyresampled*($QR, $logo, $from\_width, $from\_width, 0, 0, $logo\_qr\_width, $logo\_qr\_height, $logo\_width, $logo\_height);  
 *imagepng*($QR,**'SWE.png'**);  
 }  
}

## Php解码

### [php-qrcode-detector-decoder](https://github.com/khanamiryan/php-qrcode-detector-decoder)

zing移植到php

文档：<https://github.com/zxing/zxing>

<https://github.com/khanamiryan/php-qrcode-detector-decoder>

安装：

composer require khanamiryan/qrcode-detector-decoder

使用

//解密不带logo二维码

$image = **"hello\_world.png"**;//二维码图片  
$qrcode=**new** QrReader($image);  
**return** $qrcode->text();