## java后台生成二维码

1. 导包



1. 写入生成二维码的函数

|  |
| --- |
| **public** **static** BufferedImage encodeImage(String contents){  BufferedImage image = **null**;  **try** {  Qrcode qrcode = **new** Qrcode();  /\*\*\*表示的字符串长度： 容错率(ECC) 显示编码模式(EncodeMode)及版本(Version)有关\*\*\*/  /\*二维码的纠错级别(排错率)，共有四级：可选L(7%)、M(15%)、Q(25%)、H(30%)(最高H)。  纠错信息同样存储在二维码中，纠错级别越高，纠错信息占用的空间越多，那么能存储的有用信息就越少,对二维码清晰度的要求越小 \*/  qrcode.setQrcodeErrorCorrect('M');  //编码模式：Numeric 数字, Alphanumeric 英文字母,Binary 二进制,Kanji 汉字(第一个大写字母表示)  qrcode.setQrcodeEncodeMode('B');  /\*  二维码的版本号：也象征着二维码的信息容量；二维码可以看成一个黑白方格矩阵，版本不同，矩阵长宽方向方格的总数量分别不同。  1-40总共40个版本，版本1为21\*21矩阵，版本每增1，二维码的两个边长都增4；  版本2 为25x25模块，最高版本为是40，是177\*177的矩阵；  \*/  qrcode.setQrcodeVersion(7);  //获取内容的字节数组，设置编码格式  **byte** [] contentBytes = contents.getBytes("UTF-8");  //图片尺寸,会根据version的变大，而变大，自己需要计算  **int** imgSize = 139;  image = **new** BufferedImage(imgSize, imgSize, BufferedImage.*TYPE\_INT\_RGB*);  //获取画笔  Graphics2D gs = image.createGraphics();  //设置背景色 白色  gs.setBackground(Color.*WHITE*);  gs.clearRect(0, 0, imgSize, imgSize);  //设定图像颜色 黑色  gs.setColor(Color.*BLACK*);  // 设置偏移量，不设置可能导致二维码生产错误(解析失败出错)  **int** pixoff = 2;  //二维码输出  **if**(contentBytes.length > 0 && contentBytes.length < 150){  **boolean** [][] code = qrcode.calQrcode(contentBytes);  **int** codeLen = code.length;  **for**(**int** i = 0; i < codeLen; i++ ){  **for**(**int** j = 0; j < codeLen; j++){  **if**(code[j][i]){  gs.fillRect(j \* 3 + pixoff, i \* 3 + pixoff, 3, 3);  }  }  }  }**else**{  System.*out*.println("This is content bytes length not in [0,150].");  }  gs.dispose();  image.flush();  } **catch** (Exception e) {  System.*out*.println("生成二维码失败"+e.getMessage());  }  **return** image;  } |

1. 测试函数

|  |
| --- |
| **public** **static** **void** writeToFile(String contents,String format,File file){  BufferedImage image = *encodeImage*(contents);  **try** {  ImageIO.*write*(image, format, file);  } **catch** (IOException e) {  System.*out*.println("二维码写入文件失败"+e.getMessage());  }  }  //contents是二维码的内容 format是图片类型 file是写入文件路径 |

1. 测试

|  |
| --- |
| **public** **static** **void** main(String []args){  createrweima s=**new** createrweima();  s.testWriteToFile();  } |

1. 全部代码

|  |
| --- |
| package gongju;  import java.awt.Color;  import java.awt.Graphics2D;  import java.awt.image.BufferedImage;  import java.io.File;  import java.io.IOException;  import java.io.OutputStream;  import javax.imageio.ImageIO;  import com.swetake.util.Qrcode;  public class createrweima {  public static BufferedImage encodeImage(String contents){  BufferedImage image = null;  try {  Qrcode qrcode = new Qrcode();  /\*\*\*表示的字符串长度： 容错率(ECC) 显示编码模式(EncodeMode)及版本(Version)有关\*\*\*/  /\*二维码的纠错级别(排错率)，共有四级：可选L(7%)、M(15%)、Q(25%)、H(30%)(最高H)。  纠错信息同样存储在二维码中，纠错级别越高，纠错信息占用的空间越多，那么能存储的有用信息就越少,对二维码清晰度的要求越小 \*/  qrcode.setQrcodeErrorCorrect('M');  //编码模式：Numeric 数字, Alphanumeric 英文字母,Binary 二进制,Kanji 汉字(第一个大写字母表示)  qrcode.setQrcodeEncodeMode('B');  /\*  二维码的版本号：也象征着二维码的信息容量；二维码可以看成一个黑白方格矩阵，版本不同，矩阵长宽方向方格的总数量分别不同。  1-40总共40个版本，版本1为21\*21矩阵，版本每增1，二维码的两个边长都增4；  版本2 为25x25模块，最高版本为是40，是177\*177的矩阵；  \*/  qrcode.setQrcodeVersion(7);  //获取内容的字节数组，设置编码格式  byte [] contentBytes = contents.getBytes("UTF-8");  //图片尺寸,会根据version的变大，而变大，自己需要计算  int imgSize = 139;  image = new BufferedImage(imgSize, imgSize, BufferedImage.TYPE\_INT\_RGB);  //获取画笔  Graphics2D gs = image.createGraphics();  //设置背景色 白色  gs.setBackground(Color.WHITE);  gs.clearRect(0, 0, imgSize, imgSize);  //设定图像颜色 黑色  gs.setColor(Color.BLACK);  // 设置偏移量，不设置可能导致二维码生产错误(解析失败出错)  int pixoff = 2;  //二维码输出  if(contentBytes.length > 0 && contentBytes.length < 150){  boolean [][] code = qrcode.calQrcode(contentBytes);  int codeLen = code.length;  for(int i = 0; i < codeLen; i++ ){  for(int j = 0; j < codeLen; j++){  if(code[j][i]){  gs.fillRect(j \* 3 + pixoff, i \* 3 + pixoff, 3, 3);  }  }  }  }else{  System.out.println("This is content bytes length not in [0,150].");  }  gs.dispose();  image.flush();  } catch (Exception e) {  System.out.println("生成二维码失败"+e.getMessage());  }  return image;  }  public static void writeToFile(String contents,String format,File file){  BufferedImage image = encodeImage(contents);  try {  ImageIO.write(image, format, file);  } catch (IOException e) {  System.out.println("二维码写入文件失败"+e.getMessage());  }  }  public static void writeToStream(String contents,String format,OutputStream stream){  BufferedImage image = encodeImage(contents);  try {  ImageIO.write(image, format, stream);  } catch (IOException e) {  System.out.println("二维码写入流失败"+e.getMessage());  }  }  public void testWriteToFile() {  String contents = "http://blog.csdn.net/typa01\_kk";  String format = "png"; //格式  // File logoImg = new File("D:"+File.separator+"logo.jpg");  File img = new File("D://dd.png");  //生成二维码  createrweima.writeToFile(contents, format, img);  //添加logo图片  // File img1 = new File("D:"+File.separator+"csdnlogo.jpg");  // EncodeImgZingLogo.writeToFile(img, logoImg, format, img1);    //解析二维码  // String content = DecodeImgQrcode.decodeImg(img);  // System.out.println("1:"+content);  //String content1 = DecodeImgQrcode.decodeImg(img1);  //System.out.println("2:"+content1);  }  public static void main(String []args){  createrweima s=new createrweima();  s.testWriteToFile();    }  } |

https://blog.csdn.net/typa01\_kk/article/details/46676391