链滴

zxing 生成带有 logo 的二维码

作者：huihui
原文链接：https://ld246.com/article/1482310307485
来源网站：链滴

package com.feinno.wbs.web.util;
import java.awt.BasicStroke;
import java.awt.Graphics2D;
import java.awt.Image;
import java.awt.Shape;
import java.awt.geom.RoundRectangle2D;
import java.awt.image.BufferedImage;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.Hashtable;
import java.util.Map;
import javax.imageio.ImageIO;
import javax.servlet.http.HttpServletResponse;
import org.apache.commons.lang.StringUtils;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import com.feinno.xframe.util.LogUtils;
import com.google.zxing.BarcodeFormat;
import com.google.zxing.EncodeHintType;
import com.google.zxing.MultiFormatWriter;
import com.google.zxing.common.BitMatrix;
import com.google.zxing.qrcode.decoder.ErrorCorrectionLevel;
import com.sun.image.codec.jpeg.JPEGCodec;
import com.sun.image.codec.jpeg.JPEGImageEncoder;
/**

二维码工具类

@<a href="https://ld246.com/member/author" aria-name="author" class="tooltipped__
ser" target="_blank">author yuandalong

*/
public class QrUtil {
private static final Logger log = LoggerFactory.getLogger(QrUtil.class);
/**

生成二维码

@param str

@param response

*/
public static void encode(String str,String logoPath,String eclevel,String qrcolor, HttpServ
etResponse response,int qrWidth,int qrHeight,int pix)

{
try{
Map hints = new Hashtable();
hints.put(EncodeHintType.CHARACTER_SET, "utf-8");//使用小写的编码，大写会出现]Q2\00
026 开头内容
//ErrorCorrectionLevel.H 容错率：容错率越高,二维码的有效像素点就越多.
if (StringUtils.equals(eclevel, "M")) {
hints.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.M);
}else if (StringUtils.equals(eclevel, "L")) {
hints.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.L);
}else if (StringUtils.equals(eclevel, "Q")) {
hints.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.Q);
}else if (StringUtils.equals(eclevel, "H")) {
hints.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.H);
}else{
hints.put(EncodeHintType.ERROR_CORRECTION, ErrorCorrectionLevel.M);
}
hints.put(EncodeHintType.MARGIN, 0);//margin 边框设置
BitMatrix martrix = new MultiFormatWriter().encode(str,
BarcodeFormat.QR_CODE, qrWidth, qrHeight, hints);
//二维码
int bgColor = 0xFF000000;
if (!StringUtils.isBlank(qrcolor)) {
bgColor = Integer.parseInt(qrcolor.substring(4), 16);// 转换成 int
}
BufferedImage bufferImage = new BufferedImage(martrix.getWidth(), martrix.getHeight(),
BufferedImage.TYPE_INT_RGB);
for (int x = 0; x < martrix.getWidth(); x++) {
for (int y = 0; y < martrix.getHeight(); y++) {
bufferImage.setRGB(x, y, martrix.get(x, y) ? bgColor : 0xFFFFFFFF);//填充，可设置颜色 颜
的取值为后 6 位
}
}
if (!StringUtils.isEmpty(logoPath)) {
File file = new File(logoPath);
if (file.exists()) {
int width = (int) (qrWidth / pix);
int height = (int)(qrHeight / pix);
Image thumb = generatThumbnails(file, null, width, height, true);
if (thumb != null) {
//插入 logo
Graphics2D graph = bufferImage.createGraphics();
int w = thumb.getWidth(null);
int h = thumb.getHeight(null);
int x = (qrWidth - thumb.getWidth(null)) / 2; //设置 logo 的插入位置
int y = (qrHeight - thumb.getHeight(null)) / 2;
graph.drawImage(thumb, x, y, w, h, null);
Shape shape = new RoundRectangle2D.Float(x, y, w, h, 16, 16); // 后面两个参数是设置周
圆角，数值越大圆角越大
graph.setStroke(new BasicStroke(3f));
graph.draw(shape);
graph.dispose();
}
}

}
ImageIO.write(bufferImage, "jpg", response.getOutputStream());
}catch(Exception e){
log.error(LogUtils.getExceptionTrace(e));
}
}
/**

生成 logo 缩略图

@param file 输入的文件流

@param outputPath 输出路径

@param width 缩略图宽

@param height 缩略图高

@param proportion 是否等比例缩放

*/
private static Image generatThumbnails(File file, String outputPath, int width, int height,
oolean proportion)
{
log.info("缩略图宽：{}, 高：{}", new Object[]{width, height});
try {
BufferedImage img = ImageIO.read(file);
if (img.getWidth(null) == -1) {
log.info("图片无法读取！");
return null;
}
if (width <=0 || height <= 0) {
log.info("新生成的缩略图宽高不得小于 0！");
return null;
}
int newWidth;
int newHeight;
if (proportion) {
//等比例压缩
double rate1 = ((double)img.getWidth(null)) / (double)width + 0.1;
double rate2 = ((double)img.getHeight(null)) / (double)height + 0.1;
log.info("缩放比例 1：{}， 缩放比例 2：{}, 原生宽度：{}， 原生高度：{}", new Object[]{rate1,
ate2, img.getWidth(null), img.getHeight(null)});
//按照缩放比率大的进行缩放
double rate = rate1 > rate2 ? rate1 : rate2;
newWidth = (int) (((double) img.getWidth(null)) / rate);
newHeight = (int) (((double) img.getHeight(null)) / rate);

}else{
newWidth = width; // 输出的图片宽度
newHeight = height;
}
log.info("缩略图新的宽度：{}, 新的高度：{}", new Object[]{newWidth,newHeight});
BufferedImage tag = new BufferedImage(newWidth, newHeight, BufferedImage.TYPE_IN
_RGB);
/**

Image.SCALE_SMOOTH 的缩略算法 生成缩略图片的平滑度的

优先级比速度高 生成的图片质量比较好 但速度慢

*/
Image thumb = img.getScaledInstance(newWidth, newHeight, Image.SCALE_SMOOTH);<
p>
tag.getGraphics().drawImage(thumb, 0, 0, null);
if (!StringUtils.isEmpty(outputPath)) {
FileOutputStream out = new FileOutputStream(outputPath);
// JPEGImageEncoder 可适用于其他图片类型的转换
JPEGImageEncoder encoder = JPEGCodec.createJPEGEncoder(out);
encoder.encode(tag);
out.close();
}
return thumb;
} catch (IOException e) {
log.error(LogUtils.getExceptionTrace(e));
return null;
}
}
}