二维码下载

作者：qinaichen
原文链接：https://ld246.com/article/1498470981752
来源网站：链滴

生成4位随机数

```
public static String generateRandom(){
    String fourRandom = "";
    //产生4位的随机数(不足4位前加零)
    int randomNum = (int)(Math.random()*10000);
    fourRandom = randomNum + "";
    int randLength = fourRandom.length();
    if(randLength<4){
        for(int i=1; i<=4-randLength; i++)
            fourRandom = fourRandom + "0";
    }
    return fourRandom;
}
```

通过浏览器进行下载二维码

```
public void downloadQr(RequestContext requestContext, HttpServletResponse response) th
ows IOException {
    Sign sign = signService.get(requestContext.get("id", String.class));
    String text = "http://www.baidu.com";
    int width = 300;
    int height = 300;
    //二维码的图片格式
    String format = "jpg";
    Hashtable hints = new Hashtable();
    //内容所使用编码
    hints.put(EncodeHintType.CHARACTER_SET, "utf-8");
    BitMatrix bitMatrix = null;
    try {
        bitMatrix = new MultiFormatWriter().encode(text, BarcodeFormat.QR_CODE, width, heig
t, hints);
        //生成二维码
        SimpleDateFormat sf = new SimpleDateFormat("yyyyMMddHHmmss");
        OutputStream out = response.getOutputStream();
        MatrixToImageWriter.writeToStream(bitMatrix, format, out);
        // 创建文件输出流
        response.setContentType("image/jpeg");
        byte buffer[] = new byte[1024];
        // 循环将输入流中的内容读取到缓冲区当中
        int len = 0;
            // 输出缓冲区的内容到浏览器，实现文件下载
        out.write(buffer, 0, len);
        out.close();
    } catch (WriterException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
```

原文链接：二维码下载

```
}
```

直接在下载到本地磁盘

```
public void downloadQr(RequestContext requestContext, HttpServletResponse response) thr
ws IOException {
    String classLoadPath = SignController.class.getClassLoader().getResource("").getPath();
    String imgPath = classLoadPath + ".jpg";
    // QrCodeWriter.write(sign.getId(), 300, 300, imgPath);
    String format = "jpg";// 二维码的图片格式
    Hashtable<EncodeHintType, String> hints = new Hashtable<EncodeHintType, String>();
    hints.put(EncodeHintType.CHARACTER_SET, "utf-8"); // 内容所使用字符集编码
    String text = "http://www.baidu.com";
    try {
        BitMatrix bitMatrix = new MultiFormatWriter().encode(text, BarcodeFormat.QR_CODE, 3
0, 300, hints);
        // 生成二维码
        SimpleDateFormat sf = new SimpleDateFormat("yyyyMMddHHmmss");
        String date = sf.format(new Date());
        String generateRandom = generateRandom();
        File outputFile = new File("C:\\Users\\chun\\Downloads" + File.separator + sign.getNam
() + generateRandom + ".jpg");
        MatrixToImageWriter.writeToFile(bitMatrix, format, outputFile);
        OutputStream out = response.getOutputStream();
        response.reset();
        out.close();
    } catch (WriterException e) {
        // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

MatrixToImageWriter

```
package com.zxy.product.train.web.util;

import com.google.zxing.common.BitMatrix;

import javax.imageio.ImageIO;
import java.io.File;
import java.io.OutputStream;
import java.io.IOException;
import java.awt.image.BufferedImage;


public final class MatrixToImageWriter {

    private static final int BLACK = 0xFF000000;
    private static final int WHITE = 0xFFFFFFFF;
```

```
    private MatrixToImageWriter() {}


    public static BufferedImage toBufferedImage(BitMatrix matrix) {
        int width = matrix.getWidth();
        int height = matrix.getHeight();
        BufferedImage image = new BufferedImage(width, height, BufferedImage.TYPE_INT_RGB)

        for (int x = 0; x < width; x++) {
            for (int y = 0; y < height; y++) {
                image.setRGB(x, y, matrix.get(x, y) ? BLACK : WHITE);
            }
        }
        return image;
    }


    public static void writeToFile(BitMatrix matrix, String format, File file)
            throws IOException {
        BufferedImage image = toBufferedImage(matrix);
        if (!ImageIO.write(image, format, file)) {
            throw new IOException("Could not write an image of format " + format + " to " + file);
        }
    }


    public static void writeToStream(BitMatrix matrix, String format, OutputStream stream)
            throws IOException {
        BufferedImage image = toBufferedImage(matrix);
        if (!ImageIO.write(image, format, stream)) {
            throw new IOException("Could not write an image of format " + format);
        }
    }

}
```

MultiFormatWriter

```
package com.google.zxing;

import com.google.zxing.BarcodeFormat;
import com.google.zxing.EncodeHintType;
import com.google.zxing.Writer;
import com.google.zxing.WriterException;
import com.google.zxing.aztec.AztecWriter;
import com.google.zxing.common.BitMatrix;
import com.google.zxing.datamatrix.DataMatrixWriter;
import com.google.zxing.oned.CodaBarWriter;
import com.google.zxing.oned.Code128Writer;
import com.google.zxing.oned.Code39Writer;
import com.google.zxing.oned.Code93Writer;
import com.google.zxing.oned.EAN13Writer;
import com.google.zxing.oned.EAN8Writer;
```

```
import com.google.zxing.oned.ITFWriter;
import com.google.zxing.oned.UPCAWriter;
import com.google.zxing.oned.UPCEWriter;
import com.google.zxing.pdf417.PDF417Writer;
import com.google.zxing.qrcode.QRCodeWriter;
import java.util.Map;

public final class MultiFormatWriter implements Writer {
    public MultiFormatWriter() {
    }

    public BitMatrix encode(String contents, BarcodeFormat format, int width, int height) throw
 WriterException {
        return this.encode(contents, format, width, height, (Map)null);
    }

    public BitMatrix encode(String contents, BarcodeFormat format, int width, int height, Map
EncodeHintType, ?> hints) throws WriterException {
        Object writer;
        switch(null.$SwitchMap$com$google$zxing$BarcodeFormat[format.ordinal()]) {
        case 1:
            writer = new EAN8Writer();
            break;
        case 2:
            writer = new UPCEWriter();
            break;
        case 3:
            writer = new EAN13Writer();
            break;
        case 4:
            writer = new UPCAWriter();
            break;
        case 5:
            writer = new QRCodeWriter();
            break;
        case 6:
            writer = new Code39Writer();
            break;
        case 7:
            writer = new Code93Writer();
            break;
        case 8:
            writer = new Code128Writer();
            break;
        case 9:
            writer = new ITFWriter();
            break;
        case 10:
            writer = new PDF417Writer();
            break;
        case 11:
            writer = new CodaBarWriter();
            break;
        case 12:
```

```
            writer = new DataMatrixWriter();
            break;
        case 13:
            writer = new AztecWriter();
            break;
        default:
            throw new IllegalArgumentException("No encoder available for format " + format);
        }

        return ((Writer)writer).encode(contents, format, width, height, hints);
    }
}
```