**Generate and Read QR Code**

**Objective:**

This document contains the exploration of 2 popular API’s to generate the QR code from the shortened URL. This document will contain the code to generate the QR code along with the pros and cons of each API.

**QR Code:**

A **QRCode** is an arrangement of black and white squares and can be read with various QRCode Scanners and is convenient today because every smartphone has a QRcode scanner app. Following is the sample image.



**API Introduction:**

1. **ZXing (“Zebra Crossing”)** is the popular API for QR code processing in Java. Its library has multiple components. ZXing ("zebra crossing") is an open-source, multi-format 1D/2D barcode image processing library implemented in Java
2. **QRGen:** a simple QRCode generation api for java built on top ZXING

**XZING Implementation :**

Following are the code snippet to Generate the QR Code using XZING :

**Step 1: Maven Dependency**

<dependency>  
 <groupId>com.google.zxing</groupId>  
 <artifactId>core</artifactId>  
 <version>3.3.0</version>  
</dependency>

**Step2 :** function to generate the QR Code. This will take the data, path, charset, widht and height to generate the QR code image which can be written to stream. The image can be writtern in different format such as png/jpg etc.

public static void createQR(  
 String data, String path,  
 String charset,  
 int height, int width)  
 throws WriterException, IOException {  
  
 Map<EncodeHintType,  
 ErrorCorrectionLevel>  
 hintMap  
 = new HashMap<EncodeHintType,  
 ErrorCorrectionLevel>();  
  
 hintMap.put(  
 EncodeHintType.*ERROR\_CORRECTION*,  
 ErrorCorrectionLevel.*L*);  
  
 BitMatrix matrix  
 = new MultiFormatWriter().encode(  
 new String(  
 data.getBytes(charset),  
 charset),  
 BarcodeFormat.*QR\_CODE*, width, height);  
  
 MatrixToImageWriter.*writeToStream*(matrix,"png",new FileOutputStream(new File(path)));  
}

**Step 3 :** Read the QR code from the. Following code read qr code from the file stream and return the data decoded out of QR code.

public static String readQR(  
 String path,  
 String charset)  
 throws FileNotFoundException,  
 IOException,  
 NotFoundException  
{  
  
 Map<EncodeHintType,  
 ErrorCorrectionLevel>  
 hintMap  
 = new HashMap<EncodeHintType,  
 ErrorCorrectionLevel>();  
  
 hintMap.put(  
 EncodeHintType.ERROR\_CORRECTION,  
 ErrorCorrectionLevel.L);  
  
 BinaryBitmap binaryBitmap  
 = new BinaryBitmap(  
 new HybridBinarizer(  
 new BufferedImageLuminanceSource(  
 ImageIO.read(  
 new FileInputStream(path)))));  
  
 Result result  
 = new MultiFormatReader()  
 .decode(binaryBitmap);  
  
 return result.getText();  
}

**QRGen Implementation:**

Note : As of 2.1.0 QRGen is available from [jitpack.io](https://jitpack.io/#kenglxn/QRGen). QRGen is no longer deployed to maven central (ref: #61). Older releases are available from [Maven Central Repository](http://search.maven.org/#browse%7C-852965118).

**Maven Dependency:**

<dependency>  
 <groupId>com.github.kenglxn.QRGen</groupId>  
 <artifactId>javase</artifactId>  
 <version>2.6.0</version>  
</dependency>

**Repository:**

<repositories>  
 <repository>  
 <id>jitpack.io</id>  
 <url>https://jitpack.io</url>  
 </repository>  
</repositories>

**Code Implementation:**

**QR Code Writer :**

Since QRGen is build on the top of XZING, it provides simple and easy API to generate the QR Code which can be written to stream.

QRCode.*from*("http://amex.ShortURLQRCodeGen").to(ImageType.*PNG*)  
 .withSize(200, 200).  
 writeTo(new FileOutputStream(new File(path)));

**QR Code Reader :**

As per my knowledge QRGen API doesn’t have API to read the QR code.

**PROS and CONS**

**ZXING :**

1. Provides the vast support to different formats of QR code.
2. Matured API with respect to QR code
3. Supports both read and write

**QRGen:**

1. Simple and easy to use API
2. Built on the top of XZING.
3. Supports only Generate QR code.

Attached Sample Code to generate the QR Code and read it back using the XZING and QRGen.