**QR code generation/reading with ISO18004 – Java**

QR code Encoder

**Introduction:**

The QrEncoder class generates a QR Code image that contains text data, along with two lines of additional text: one at the top and one at the bottom of the image. The code uses the ZXing library to encode the text data and generate the QR Code image. The image is saved to a specified file path with a specified name.

**Libraries Used:**

* **java.awt.Color** - Used to set colors for the QR code image and the text.
* **java.awt.Font** - Used to set font for the text.
* **java.awt.Graphics2D** - Used to create graphics and draw the QR code image and the text.
* **java.awt.image.BufferedImage** - Used to create a BufferedImage to draw the QR code pixels and the text.
* **java.io.File** - Used to create a file to save the generated QR code image.
* **javax.imageio.ImageIO** - Used to write the generated QR code image to a file.
* **com.google.zxing.BarcodeFormat** - Used to specify the format of the barcode to be generated (QR code in this case).
* **com.google.zxing.EncodeHintType** - Used to specify the encoding hints for generating the QR code.
* **com.google.zxing.WriterException** - Used to handle exceptions that occur during QR code generation.
* **com.google.zxing.common.BitMatrix** - Used to represent the QR code as a matrix of bits.
* **com.google.zxing.qrcode.QRCodeWriter** - Used to create and encode the QR code.
* **com.google.zxing.qrcode.decoder.ErrorCorrectionLevel** - Used to specify the error correction level for the QR code.

**Input Parameters:**

The QrEncoder class takes several input parameters:

**width**: the width of the QR Code image, in pixels.

**height**: the height of the QR Code image, in pixels.

**qrCodeData**: the text data to encode in the QR Code.

**topText**: the text to display at the top of the QR Code image.

**bottomText**: the text to display at the bottom of the QR Code image.

**fileType**: the file format to save the QR Code image as, such as "png" or "jpg".

**path**: the file path and name where the QR Code image should be saved.

**Code Details:**

1. The code imports several classes from the ZXing library to encode the text data and generate the QR Code image.
2. The code defines a constant variable "fpath" to retrieve the file path from the system environment variable, and a variable "rename" to specify the file name of the QR Code image.
3. The main() method takes the input parameters, generates the QR Code image, draws the top and bottom text on the image, and saves the image to the specified file path in the specified file format.
4. The getQRCodeParams() method sets the error correction level for the QR Code image to "L" (low), which means that the image can still be scanned even if up to 7% of the code is damaged or obscured.

**Functions and Methods:**

**Main Function:**

The main function generates a QR code image with text at the top and bottom and saves it to a file. It takes no input parameters and returns nothing. The main steps in the function are:

* Initialize the width, height, QR code data, top text, bottom text, and file type.
* Generate the QR code using the **QRCodeWriter** class and the **encode** method.
* Create a **BufferedImage** to draw the QR code pixels and the text.
* Draw the QR code image using the **Graphics2D** class.
* Draw the top and bottom text using the **Graphics2D** class.
* Save the QR code image to a file using the **ImageIO** class.

**The usage of for loop:**

The for loop is used to iterate through each pixel in the QR code image and draw a black square if the corresponding bit in the QR code matrix is set to 1. Specifically, the loop starts at the top-left corner of the image (i=0, j=0) and iterates through every row (i) and column (j) of the image, drawing a square at position (i, j) if the bit at position (i, j) in the QR code matrix is set to 1. This process allows the program to generate the black-and-white QR code image that corresponds to the input data.

**getQRCodeParams Method:**

The **getQRCodeParams** method is a helper method that returns a map of encoding hints for generating the QR code. It takes no input parameters and returns a **Map<EncodeHintType, Object>** object. The method sets the error correction level for the QR code to be generated and returns the map.

**ZXing library**

ZXing is a free and open-source library that is used for creating and reading barcodes, including QR codes. It was developed by Google and released under the Apache License 2.0.

The library provides APIs for encoding and decoding different types of barcodes, including 1D and 2D barcodes, such as UPC-A, EAN-13, Code 39, QR Code, Data Matrix, and more. It can be used in a variety of programming languages, including Java, C++, C#, Python, and Ruby, among others.

In the provided code, the ZXing library is used to encode the input data into a QR code matrix using the **QRCodeWriter** class, which takes the input data, barcode format (in this case, QR\_CODE), and the desired width and height of the barcode image as parameters. The library also provides a **BitMatrix** class to represent the matrix of bits that make up the QR code image.

Additionally, the **EncodeHintType** enum is used to specify additional parameters for encoding the QR code, such as the error correction level, which is set to "L" (Low) in this case. This helps to ensure that the QR code can still be read even if parts of it are damaged or obscured.

**Conclusion:**

The QrEncoder class provides a simple and customizable way to generate QR Code images with additional text. By specifying the input parameters, users can generate QR Codes for various purposes and save them to a specified file path.

QR code Decoder

**Overview**

This Java program reads a QR code image file and decodes the information contained in the QR code using the ZXing library. The decoded information is then printed to the console.

**Libraries Used**

* **com.google.zxing**: The ZXing ("zebra crossing") library is an open-source, multi-format 1D/2D barcode image processing library that can read QR codes among other types of barcodes

**Exceptions**

* **FileNotFoundException**: This exception is thrown if the QR code image file cannot be found.
* **IOException**: This exception is thrown if an input or output error occurs.
* **NotFoundException**: This exception is thrown by the ZXing library if a QR code cannot be found in the image.

The program can quickly and accurately read the information contained in a QR code, saving time and increasing efficiency as well astThe program uses open-source libraries, making it a cost-effective solution for businesses.

Other than that the program can only read QR codes and is not capable of reading other types of barcodes and the program is run using a command-line interface, which may not be user-friendly for non-technical users.