QR-CODES ERSTELLEN

GRUBER DANIELA
TUCEK JOSEF THOMAS

Start

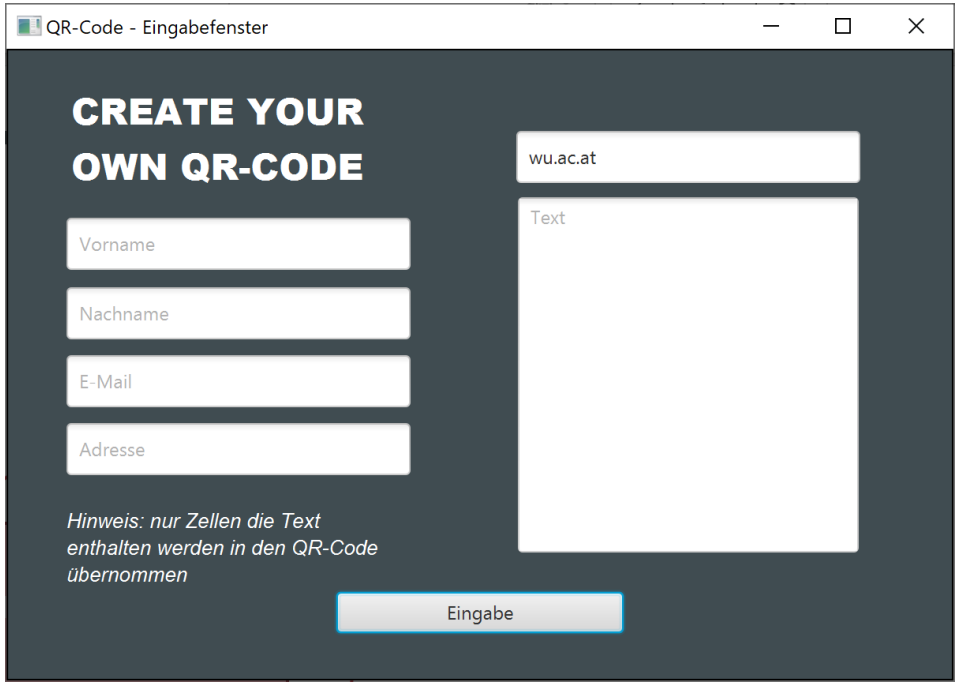
HAUPTPROGRAMM
QRCode.rxj

inputQR.fxml

JAVA BIB -1.
core-3.4.1.jar

JAVA BIB – II.
javase-3.4.1.jar

CONTROLLER
QRCodeController.rxj



Programm für
BILDAUSGABE
bild.rxj

showQR.fxml

Ende

A

Jar-Dateien

Da keine JAR-Bibliotheken verfügbar waren, musste mithilfe der POM-Datei für Apache Maven eine Jar Datei erstellt werden

B

Controller

Im Controller.rxj werden die Infos aus dem GUI geladen und QR Code generiert

C

Hauptprogramm

Dieses Programm wird ausgeführt und liefert die Eingabefelder (GUI)

D

Bildausgabe

Dieses Programm wird durch den Controller aufgerufen und ausgeführt, es liefert den fertigen QR Code in eigenem Fenster

KOMPONENTEN DES PROGRAMMS

JAR-DATEIEN ERSTELLEN

- Für ZXing stehen keine JAR-Dateien zum Download zur Verfügung
- Können mit Apache Maven erzeugt werden, weil POM-Dateien vorhanden sind

ABLAUF:

1. Download von Zxing als ZIP-Datei von Github
2. Installation von Maven

Erzeugung der JAR-Dateien mit Maven:

```
cd zxing-3.4.1/core
mvn install
```

→ JAVA BIB -I. core-3.4.1.jar

```
cd zxing-3.4.1/javase
 mvn install
```

→ JAVA BIB -II. javase-3.4.1.jar

Ausführen mit Java:

```
javac -classpath * Barcodes.java
java -cp "core-3.4.1.jar;javase-3.4.1.jar;" Barcodes
```

Beispiel für Path-Variable: - User-Verzeichnis ist anzupassen

```
set CLASSPATH=%CLASSPATH%;C:\Users\laptop2pc\Downloads\core-3.4.1.jar;C:\Users\laptop2pc\Downloads\javase-3.4.1.jar;
```

CONTROLLER

1. Button-Click & Rückgabewerte der Eingabefelder

* id-Labels sind die Eingabemöglichkeiten im GUI

2. Array zur Verarbeitung der Infos

* es wird evaluiert, in welchen Feldern Werte eingetragen wurden und diese anschließend ausgegeben

```
3   -- CONTROLLER
4
5   ::routine buttonClicked public
6   use arg primaryStage
7     slotDir=arg(arg())
8
9   -- get the labels
10
11  /* @get(idVorname) */
12  /* @get(idName) */
13  /* @get(idAdresse) */
14  /* @get(idEmail) */
15  /* @get(idWebseite) */
16  /* @get(idText) */
17  /* @get(idBild) */
18
19  textarray=.bsf~bsf.createJavaArray("java.lang.String",6)
20
21  do i=1 to 6
22      textarray[i]=''  -- textarray mit leeren String initialisieren
23  end
24  if (idVorname~getText<>'') then textarray[1]= "Vorname:" idVorname~getText ||';'
25  if (idName~getText<>'') then textarray[2]= "Name:" idName~getText ||';'
26  if (idAdresse~getText<>'') then textarray[3]= "Adresse:" idAdresse~getText ||';'
27  if (idEmail~getText<>'') then textarray[4]= "E-Mail: " idEmail~getText ||';'
28  if (idWebseite~getText<>'') then textarray[5]= "https://" || idWebseite~getText ||' '
29  if (idText~getText<>'') then textarray[6]= idText~getText
30
31  text=''
32  do i=1 to 6
33      text=text||textarray[i]
34  end
35
36  say text
37
```

nur ausgefüllte Textfelder werden im QR-Code hinzugefügt

CONTROLLER

REXX:

```
37
38  --========================== QR CODE ================================================
39  BarcodeFormat=BSF.import("com.google.zxing.BarcodeFormat")
40  bitMatrix=BSF.import("com.google.zxing.qrcode.QRCodeWriter")
41
42  width=.bsf~new("java.lang.Integer",300)
43  height=.bsf~new("java.lang.Integer",400)
44  bitMatrix=.bsf~new("com.google.zxing.common.BitMatrix")
45  QRWriter=.bsf~new("com.google.zxing.qrcode.QRCodeWriter")
46  bitMatrix=QRWriter~encode(text,BarcodeFormat~QR_CODE,width,height)
47
```

=

JAVA:

```
68      BitMatrix bitMatrix = new QRCodeWriter().encode(text, BarcodeFormat.QR_CODE, width,
        height);
69      //BitMatrix bitMatrix = new QRCodeWriter().encode(text, BarcodeFormat.DATA_MATRIX,
        width, height);
```

CONTROLLER

JPEG-Bild (QR Code) wird kreiert

Automatischer Aufruf des Programms zum Anzeigen des Bildes

```
47
48  ------------------------- write image ------------------------------
    --------------
49
50  ImageWriter=BSF.import("com.google.zxing.client.j2se.MatrixToImageWrit
    er")
51  file1=.bsf~new("java.io.File","qrcode_test.jpg")
52  filestream1=.bsf~new("java.io.FileOutputStream",file1)
53  ImageWriter~writeToStream(bitMatrix,'jpeg',filestream1)
54
55  --======================== show picture
    ============================================
56
57
58  parse source system invocation filename.
59  --say system
60  select
61  when system = 'WindowsNT' then
62     bild.rxj
63  when system = 'DARWIN' | system = 'LINUX' then
64     rexxj.sh bild.rxj
65     otherwise
66     bild.rxj
67  end
68  --bild.rxj
69  --
    ========================================================================
    ============
70
71
```

HAUPTPROGRAMM

```
20  ::requires "BSF.CLS"    -- get Java support
21
22  ::class RexxApplication -- implements the abstract class "javafx.application.Application"
23
24  ::method start          -- Rexx method "start" implements the abstract method
25    use arg primaryStage  -- fetch the primary stage (window)
26    primaryStage~setTitle("QR-Code - Eingabefenster")
27
28    fxmlUrl=.bsf~new("java.net.URL", "file:inputQR.fxml")
29    rootNode=bsf.loadClass("javafx.fxml.FXMLLoader")~load(fxmlUrl)
30
31    scene=.bsf~new("javafx.scene.Scene", rootNode)
32    primaryStage~setScene(scene)  -- set the stage
33    primaryStage~show             -- show the stage
34
```

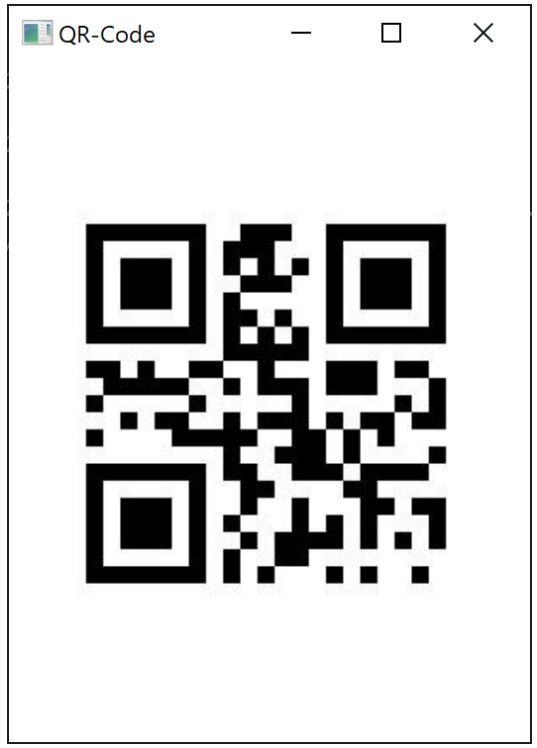
Ausführung von Rexx Script QRCodeController.rxj

* alle Eingaben in den erstellten Feldern werden im Controller aufgerufen

FXML

```
1   <?xml version="1.0" encoding="UTF-8"?>
2
3   <?import javafx.scene.control.Button?>
4   <?import javafx.scene.control.Label?>
5   <?import javafx.scene.control.TextArea?>
6   <?import javafx.scene.control.TextField?>
7   <?import javafx.scene.layout.AnchorPane?>
8   <?import javafx.scene.text.Font?>
9   <?language rexx?>
10
11  <AnchorPane maxHeight="-Infinity" maxWidth="-Infinity" minHeight="-
    Infinity" minWidth="-Infinity" prefHeight="400.0" prefWidth="600.0"
    style="-fx-border-color: #000000; -fx-background-color: #404c51;"
    xmlns="http://javafx.com/javafx/17" xmlns:fx="http://javafx.com/fxml/1">
12
13      <fx:script source="QRCodeController.rxj" />
14
15      <children>
16
```

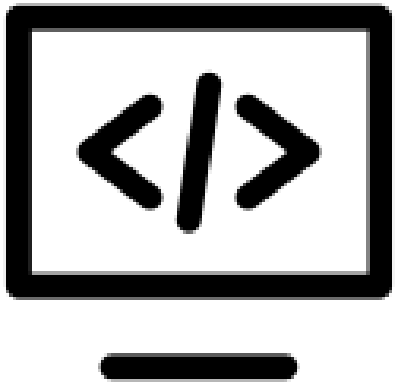
PROGRAMM FÜR BILDAUSGABE




```
::requires "BSF.CLS"    -- get Java support

::class RexxApplication -- implements the abstract class
"javafx.application.Application"

::method start          -- Rexx method "start" implements the abstract
method
  use arg primaryStage  -- fetch the primary stage (window)
  primaryStage~setTitle("QR-Code")

  fxmlUrl=.bsf~new("java.net.URL", "file:showQR.fxml")
  rootNode=bsf.loadClass("javafx.fxml.FXMLLoader")~load(fxmlUrl)

  scene=.bsf~new("javafx.scene.Scene", rootNode)    -- create a scene
  for our document
  primaryStage~setScene(scene)  -- set the stage to our scene
  primaryStage~show             -- show the stage (and thereby our
  scene)
```

FXML

```
<?xml version="1.0" encoding="UTF-8"?>

<?import javafx.scene.image.Image?>
<?import javafx.scene.image.ImageView?>
<?import javafx.scene.layout.AnchorPane?>
<?language rexx?>
```



VIELEN DANK!

WIR BEDANKEN UNS FÜR EURE AUFMERKSAMKEIT UND WÜNSCHEN VIEL SPASS BEIM AUSPROBIEREN!