|  |  |
| --- | --- |
| **Programmierschnittstellen und Softwarequalität**  **Prof. Dr. Ursula Oesing** |  |
| **Projekttermine 1 a und 1b:**  **Messdaten des EMU-Messgeräts**  **unter Verwendung von Threads**  **erfassen und in einer DB speichern** |

**1. Messdaten des EMU-Messgeräts erfassen und speichern**

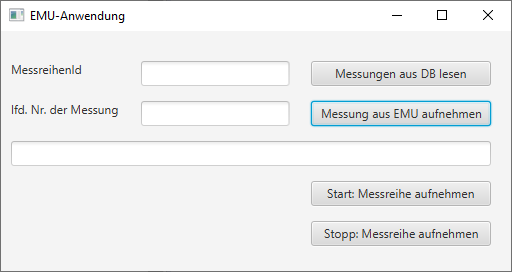
**1.1 Vorstellung der Aufgabe des gesamten Projekts**

In dem gesamten Projekt werden Sie eine Anwendung entwickeln, die Messdaten des EMU-Messgeräts erfassen, speichern, darstellen und auswerten kann. Diese Anwendung wird auch getestet mittels automatisierter Softwaretests. **Sie benötigen zum nächsten Projekttermin das Ergebnis des vorherigen Termins!** Eine Basisanwendung, welche bereits eine Auswahl von Datenbankzugriffen enthält, steht Ihnen auf dem Serverlaufwerk zur Verfügung. Es liegen Beispielprogramme zu Threads ebenfalls auf dem Serverlaufwerk.

**1.2 Aufgaben dieses Projekttermins**

**1.2.1 Basisanwendung übernehmen und ergänzen**

1. Vorhanden ist eine Basisanwendung, welche das MVC-Pattern realisiert. Übernehmen Sie diese. Übernehmen Sie die Bibliotheken für JavaFX. Jetzt müsste die Anwendung laufen. Sie ist allerdings noch nicht funktionstüchtig.

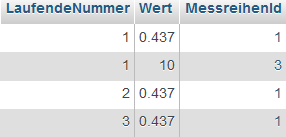


1. Erstellen Sie eine Datenbank mittels MySql, welche die folgenden beiden Tabellen enthält. Erweitern Sie Ihre Basisanwendung, so dass diese auf Ihre Datenbank zugreifen kann.

**Tabelle Messreihe**



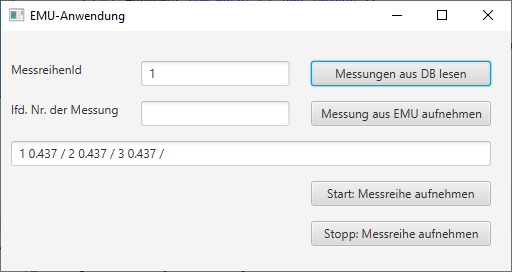
**Tabelle Messung**



Das Attribut *MessreihenId* der Tabelle *Messreihe* ist der Primärschlüssel der Tabelle *Messreihe*. Das Attribut *MessreihenId* der Tabelle *Messung* ist ein Fremdschlüssel. Der Primärschlüssel der Tabelle *Messung* besteht aus der *MessreihenId* und dem Attribut *LaufendeNummer*. Verwenden Sie für die Attribute *MessreihenId* oder *LaufendeNummer* kein Autoincrement.

**Sie benötigen die Tabellen für weitere Projekttermine. Daher über-nehmen Sie diese bitte in der vorgegebenen Form!**

Bei der Eingabe der MessreihenId *1* und Klick auf *Messungen aus DB lesen* werden jetzt die drei Messungen der Messreihe 1 aus der Datenbank gelesen und angezeigt. Die Verbindungsdaten zur Datenbank müssen allerdings vorher angepasst werden!



**1.2.2 Messungen mit dem EMU-Messgerät erfassen und speichern**

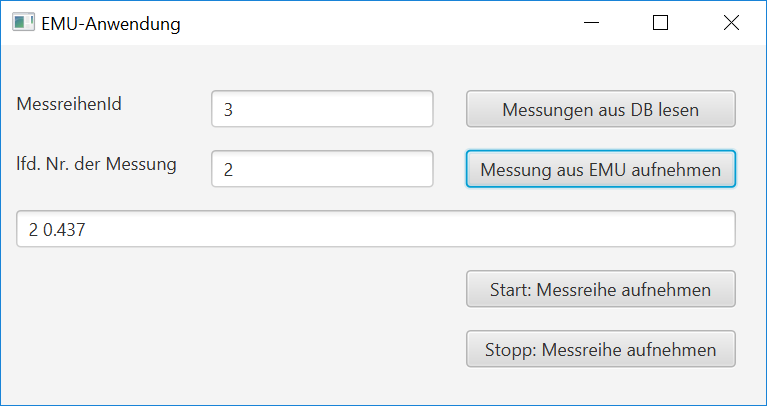
Gegeben seien Verbraucher. Weiterhin haben Sie bereits die Anwendung aus  
der Übung 2 erstellt. Sie benötigen die Enumeration MESSWERT.java und die Klasse EmuCheckConnection.java aus der Übung 2. Übernehmen Sie diese und legen Sie sie in ein Unterpackage *emu* im package *business* der Anwendung aus dem Aufgabenteil 1.2.1. Binden Sie weiterhin die Bibliotheken für den Zugriff auf ein EMU-Messgerät aus der Übung 2 ein.

Erweitern Sie die Anwendung aus dem Aufgabenteil 1.2.1 um die folgende Funktionalität.

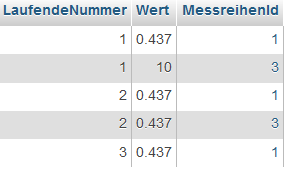
Beim Klick auf den Button *Messung aus EMU aufnehmen* führen Sie mit Hilfe des EMU-Messgeräts eine Messung der Leistung oder der Arbeit des Verbrauchers durch. Speichern Sie die Messung in der Datenbank ab. Sie können diese einer Messreihe zuordnen, die Sie über phpMyAdmin in die Datenbanktabelle eingeben.

**Beispiel**

Es wurde bereits über phpMyAdmin eine Messreihe mit MessreihenId 3 angelegt, siehe oben. Dann wird eine Messung zu der Messreihe mit der vorgegebenen MessreihenId durchgeführt und angezeigt.



Die Messung wird in der Datenbank gespeichert, siehe 4. Zeile.

****

**Hinweis zu 1.2.2**

Bisher werden die Messergebnisse in der Konsole ausgegeben. Sie brauchen Sie aber als double-Wert im Programm.

In der Klasse *EmuCheckConnection* legen Sie die folgenden beiden Attribute und die folgende Methode an.

private String ergebnis;

private boolean ergSchreiben = false;

public double gibErgebnisAus(){

if(ergebnis.contains("\*")){

int a = ergebnis.indexOf("(");

int m = ergebnis.indexOf("\*");

ergebnis = erg.substring(a+1,m);

}

return Double.parseDouble(ergebnis);

}

In der Methode *sendRequest* setzen Sie nach dem Senden Ihres Requests  
*ergSchreiben* auf true. Ab jetzt soll alles, was gelesen wird, nicht nur in der Konsole ausgegeben werden, sondern in *ergebnis* geschrieben werden. Letzteres implementieren Sie in der Methode *run.*

public void run() {

int intZahl;

byte[] byteArray = new byte[1];

while(connected){

try {

intZahl = device.getQueueStatus();

if (intZahl != 0){

intZahl = device.read(byteArray);

System.out.print((char)byteArray[0]);

if(ergSchreiben){

ergebnis = ergebnis + (char)byteArray[0];

}

}

}

catch (FTDIException e) {

e.printStackTrace();

}

try {

sleep(5);

}

catch (InterruptedException e) {

e.printStackTrace();

}

}

}

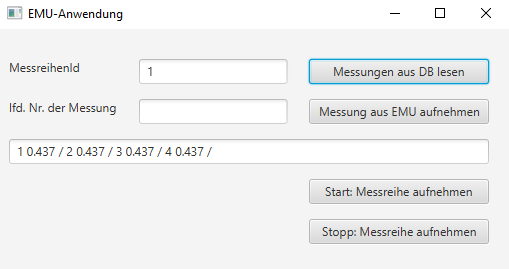
**1.2.3 Mehrere Messungen mit dem EMU-Messgerät erfassen und speichern**

Als Vorbereitung löschen Sie in der Datenbank alle Messungen zu einer der vorhandenen Messreihen.

Legen Sie ein Thread an. Mit Hilfe dieses Threads soll es möglich sein, in  
 regelmäßigen zeitlichen Abständen Messungen zu einer Messreihe vorzuneh-men. Die Messungen sollen in der Datenbank gespeichert werden.

Sie geben als MessreihenID 1 ein. (Das ist die MessreihenID, deren  
Messungen Sie gelöscht haben.) Sie klicken auf den Button *Start: Messreihe aufnehmen.* Dann warten Sie solange, bis mindestens zwei Messungen durchgeführt wurden. (Letzteres können Sie in der Konsole verfolgen.) Dann klicken Sie auf den Button *Stopp: Messreihe aufnehmen* und dann auf *Messungen aus DB lesen.*

Die aufgenommenen Messungen werden angezeigt.



Sie sind auch in der Datenbank gespeichert worden. Sie können die Klasse *Timer* des Beispielprojekts benutzen.