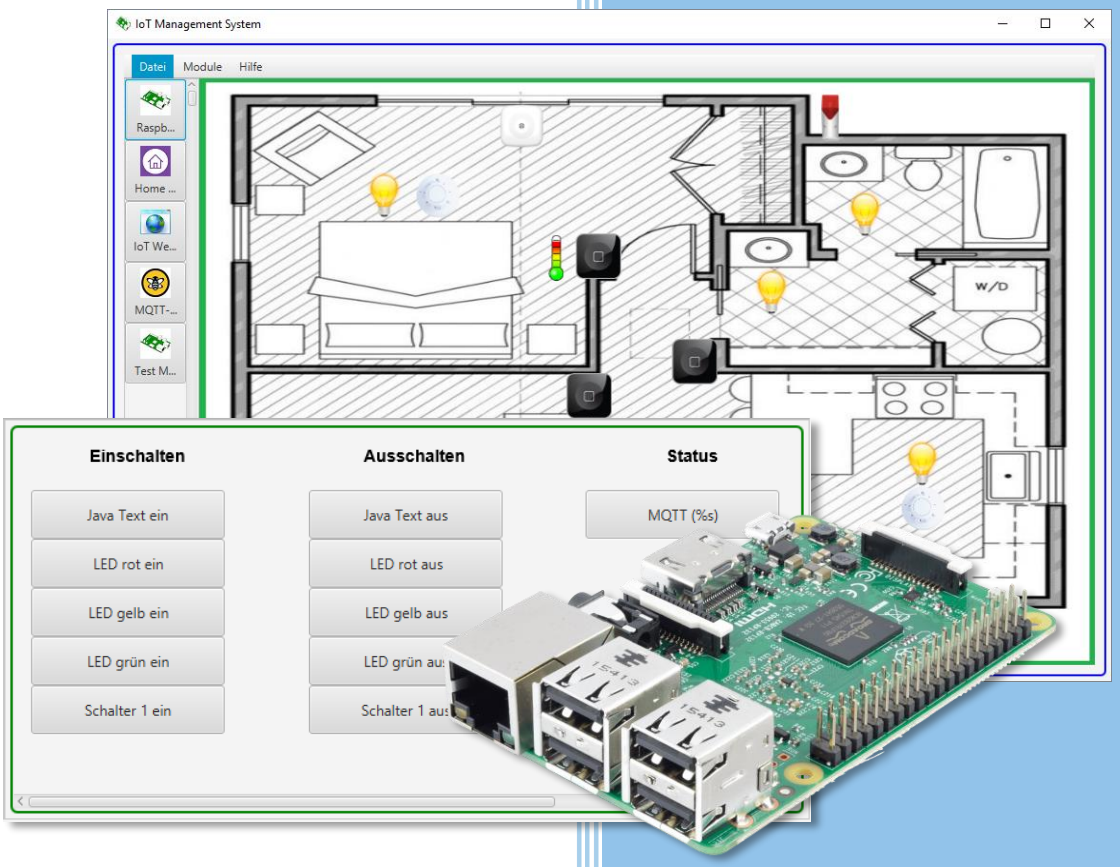
Internet der Dinge (IoT)



Adrian Meier

Privat

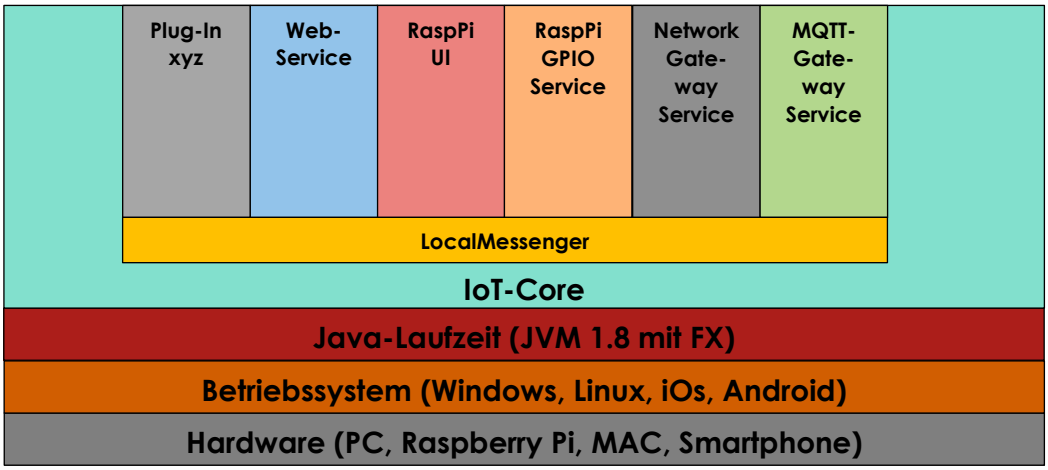
29.3.2017

Inhaltsverzeichnis

Applikationsbeschreibung

Systemaufbau

Applikation-Schichtenmodel

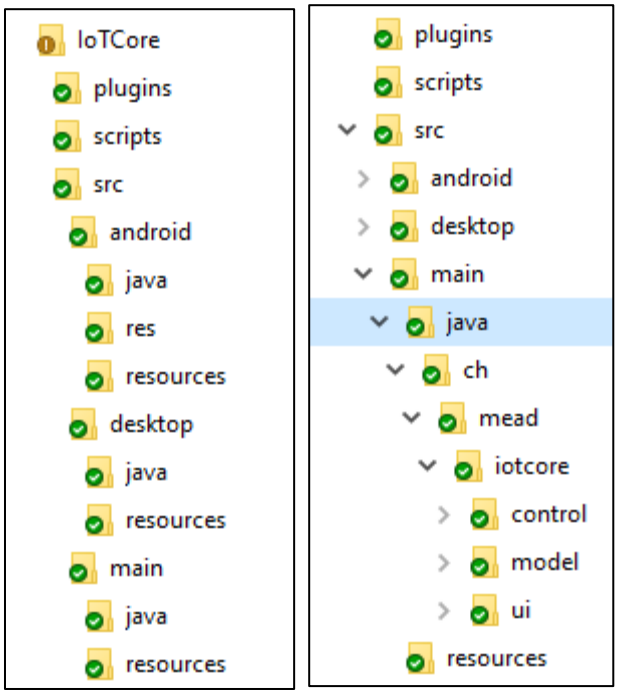


Wie aus der Grafik zu sehen ist, geht der Kommunikationsweg immer über den Local-Messenger. Das heisst, die einzelnen Moduln (Plug-Ins) stehen nie in direktem Kontakt zueinander. Somit ist die Austausch- und Abschaltbarkeit von Modulen gewährleistet.

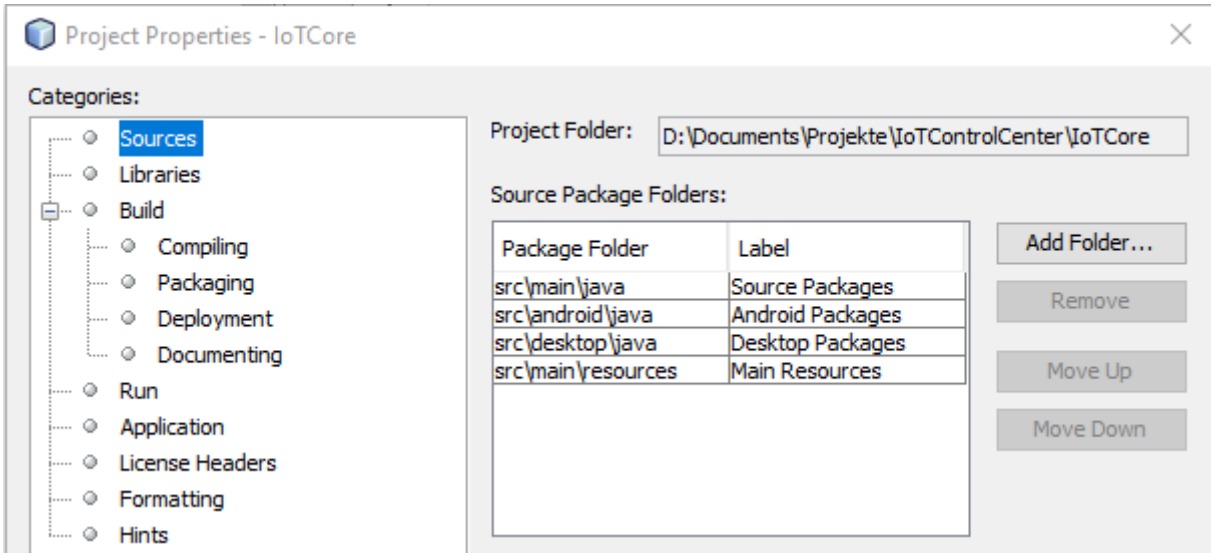
Projektstruktur

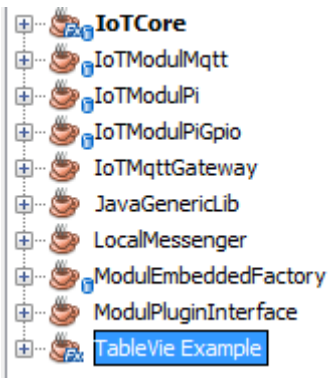
Im Verzeichnissystem

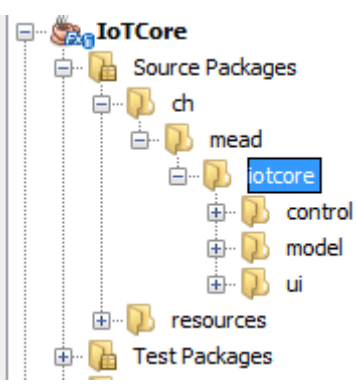
Wenn das Projekt für Multiplattform wie Windows, Mac oder Android vorgesehen ist, wird eine zum Netbeans abweichende Struktur verlangt:

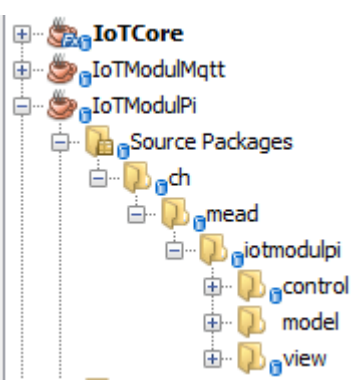


Innerhalb Netbeans



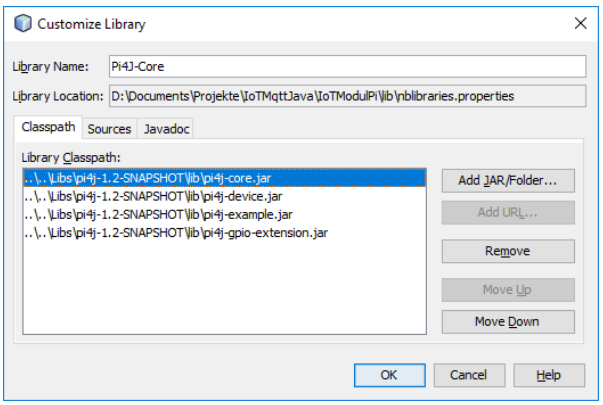


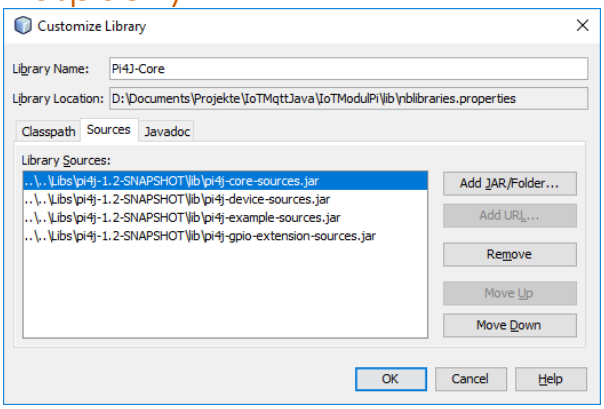


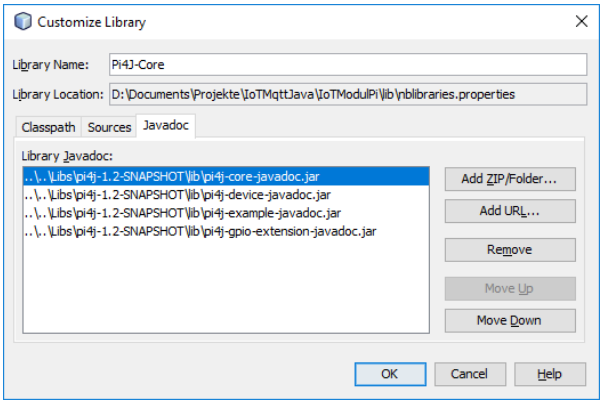


Projektvorbereitung für Bibliotheken

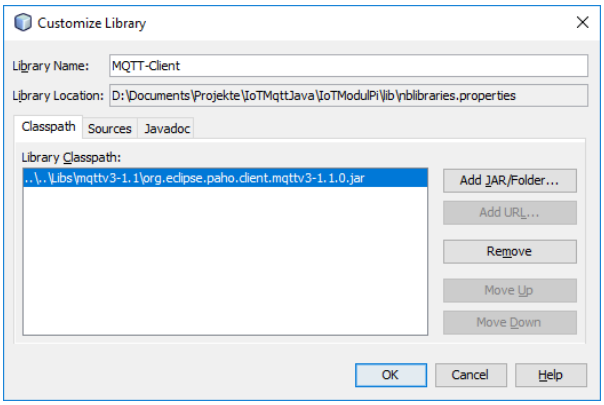
Einbinden von Java für GPIO von Raspberry Pi

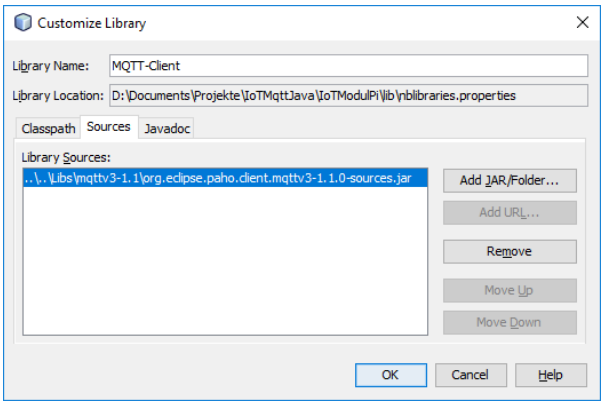




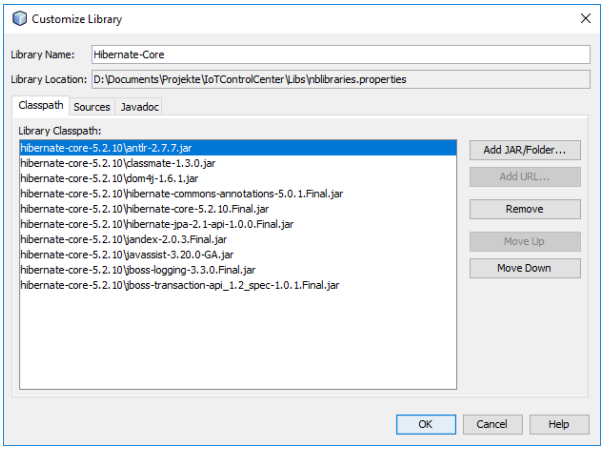


Einbinden von MQTT – Kommunikation



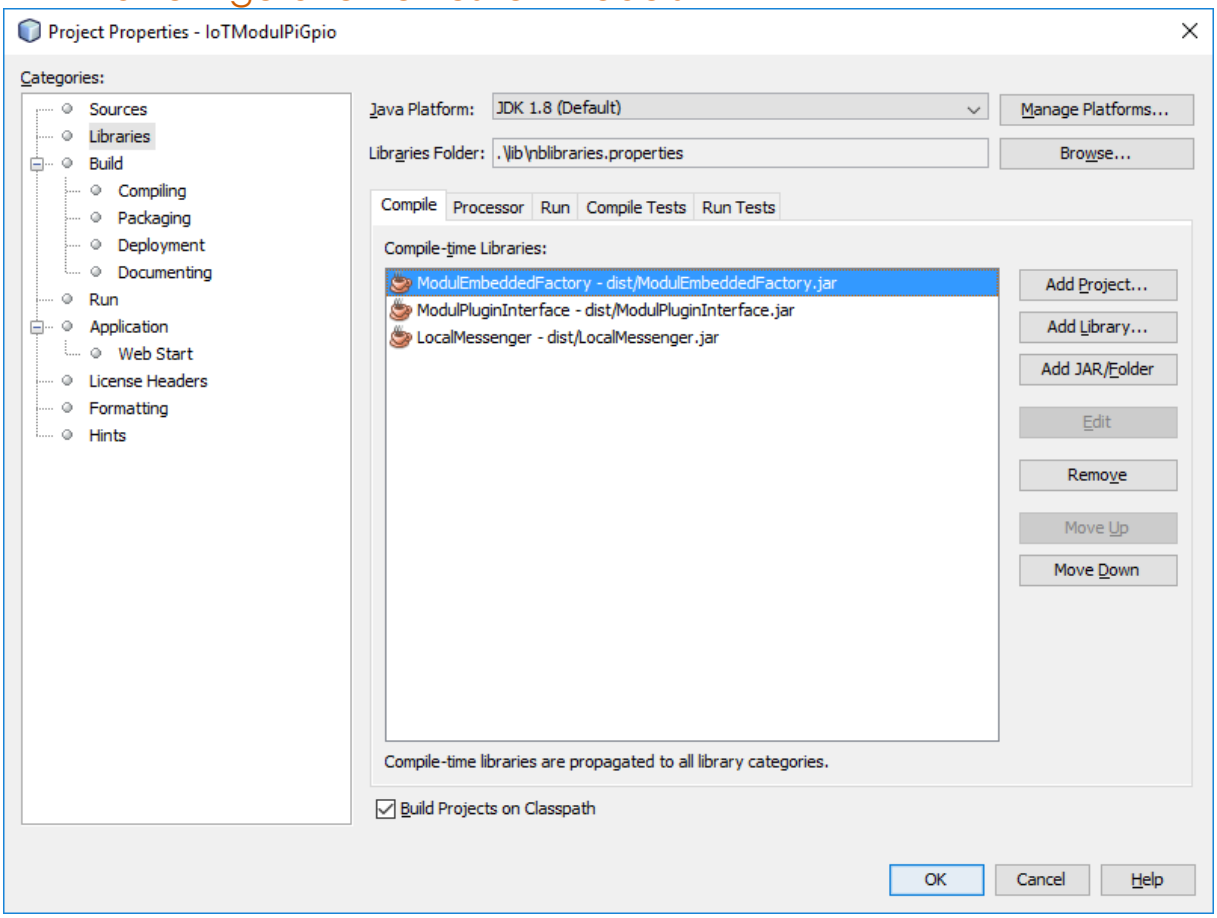


Einbinden von Hibernate für Datenbankanbindung



Projektkonfiguration

Minimalkonfiguration eines IoT-Moduls



Ant-Script Ergänzung

Um die erzeugte Plug-In dem IoT-Core zur Verfügung zu stellen, muss im Modulprojektverzeichnis am Ende der Datei build.xml folgende Einträge hinzugefügt werden:

```
<!-- Modul ins Plugin-Verzeichnis von IoT-Core publizieren  -->
<target name = "-post-jar" >
   <echo message = "Deploy Modul to ../IoTCore/plugin for IoT-Core Plug-Ins" />

   <copy file = "${dist.jar}" todir = "../IoTCore/plugin" />
</target>
```

Plug-In-Implementation

Minimalimplementierung der Plug-In-Schnittstelle

```
1   package ch.mead.iotmodulpigpio.control;
2
3   import ch.mead.modulembeddedfactory.control.EmbModuls;
4   import ch.mead.localmessenger.MsgDispatcher;
5   import ch.mead.modulplugininterface.PluginHandler;
6
7   public class PiGpioPlugin implements PluginHandler {
8       private PluginHandler manager;
9       PiGpioControl ctrl;
10
11      @Override
12      public boolean start(){
13        System.out.println("Starte: "+this.getClass().getName());
14        if (ctrl != null)
15          ctrl.run();
16        return true;
17      }
18      @Override
19      public boolean stop(){
20        System.out.println("Stoppe: "+this.getClass().getName());
21        if (ctrl != null)
22          ctrl.destroy();
23      return true;
24      }
25
26      @Override
27      public void setPluginManager(PluginHandler manager){
28        this.manager = manager;
29      }
30
31      @Override
32      public PluginHandler getPluginManager() {
33          return manager;
34      }
35
36      @Override
37      public EmbModuls initializeModul(MsgDispatcher observer) {
38          ctrl =  new PiGpioControl(observer);
39          return ctrl;
40      }
41
42      @Override
43      public String getDescription() {
44          return "RaspPi GPIO Control";
45      }
46  }
```

Minimalimplementierung der Applikation

Um die Kommunikation zu den Modulen zu gewährleisten, wird der Konstruktor mit dem Oberver eingefügt.

```
1   package ch.mead.iotmodulpigpio.control;
2
3   import ch.mead.localmessenger.MsgDispatcher;
4   import ch.mead.localmessenger.Observer;
5   import ch.mead.localmessenger.impl.ObserveData;
6   import ch.mead.modulembeddedfactory.control.EmbModuls;
7   import java.util.List;
8   import javafx.scene.Node;
9   import javafx.scene.control.MenuItem;
10
11  public class PiGpioControl implements EmbModuls, Observer {
12      MsgDispatcher observer;
13
14      public PiGpioControl(MsgDispatcher observer) {
15          this.observer = observer;
16          this.observer.addSubscription(this); // am Observer anmelden
17      }
18
19      @Override
20      public Node createMainPane() {...3 lines }
23
24      @Override
25      public Node createNavigationPane() {...3 lines }
28
29      @Override
30      public Node createStatusPane() {...3 lines }
33
34      @Override
35      public List<MenuItem> createSubMenu() {...3 lines }
38
39      @Override
40      public void run() {...3 lines }
43
44      @Override
45      public void destroy() {...3 lines }
48
49      @Override
50      public void notification(ObserveData data) {...3 lines }
53  }
```

Design

Modulkonfiguration