**Raspberry Pi Project :**

**„HomeControl“**

* **Starten des Programms**
  + **Parameter**
* **Konfiguration**
  + **Hauptconfig**
  + **Nameconfig**
  + **RSSFeeds**
* **Layout**
* **Server und Netzwerk**
  + **Allgemein**
  + **Einloggen**
  + **AuthActions**
* **Möglichkeiten der Software**

**Aufruf des Programms**

**Das Programm kann mit dem Standard Java Aufruf Befehl gestartet werden:**

**sudo java <VM Parameter> -jar HomeControl.jar <arg>**

**sudo** ist **notwendig**, um von Java aus auf das **PiFace** zuzugreifen.

Für VM (Virtual [Maschine](http://www.dict.cc/deutsch-englisch/Maschine.html)) Parameter kann z.B.:

* Xms256m
* Xmx256m

angegeben werden. Dies würde der Java VM – also auch dem Programm – mehr Arbeitsspeicher geben.

Für das **erste** **Argument** ist ein „**Master** **Passwort**“ vorgesehen.

Dieses kann verwendet werden um sich in der GUI **einzuloggen** oder sich über die **Netzwerk** **Funktion** mit dem **Nutzernamen** „root“ **einzuloggen**.

Wenn man **nicht** im Ordner /**Desktop** ist, muss der komplette **Pfad** angegeben werden.

So kann man das **Programm** wie folgt **starten**:

**sudo java –Xms256m –Xmx256m –jar HomeControl.jar swordfish**

Konfiguration

Es gibt **3** config Dateien um das **Programm** **anzupassen**. Diese befinden sich in der **gleichen** **Directory** wie das Programm:

* config.properties
* NameConfig.properties
* RSSFeeds.txt

Im Folgenden ist ein Beispiel für jede der config Dateien gegeben und eine Erklärung zu jeder Option.

config.properties

* **Version=3**
  + Die Version gibt die Version der config Datei an. Nicht verändern!
* **Rss\_enabled=true**
  + Wenn man Rss\_enabled auf false setzt, wird kein RSS Feed mehr geladen
* **Weather\_City=Schweinfurt**
  + Hier Kann man einstellen für welche Stadt man das Wetter angezeigt bekommen möchte
* **Mpc\_enabled=true**
  + Durch das setzen auf false wird die Funktion des Musikplayers ausgeschalten.
* **Rss\_refresh\_delay\_ms=850000**
  + Rss\_refresh\_delay\_ms gibt an nach wie vielen Millisekunden der RSS Feed aktualisiert werden soll
* **Mpc\_ServerIp=192.168.11.205**
  + Mit dieser Option kann die Server IP, an die die Mpc Befehle gesendet werden, geändert werden.
* **Is\_Testbuild=false**
  + Testbuild sollte auf true gesetzt werden, wenn kein PiFace auf dem Pi steckt.
* **Mpc\_refresh\_delay\_ms=2000**
  + Hier kann man verändern, wie viele Sekunden der Musik Titel vom Mpc Server ausgelesen werden soll
* **Weather\_refresh\_delay=600000**
  + Auch hier kann man den Delay zum erneuern des Wetters ändern
* **Start\_with\_login\_screen=false**
  + Es gibt auch die Möglichkeit einen einlog Bildschirm zu verwenden.
* **Internal\_server\_port=9977**
  + Auch der Port des Eigenen Servers ist konfigurierbar. Falls man von außerhalb Netzwerk befehle zum Pi schicken will, muss dieser Port freigeben werden über TCP & UDP.
* **Dev\_prompt\_enabled=true**
  + Die Dev\_promot ist die schwarze Konsole in der Mitte der GUI. Hier kann man sich mit dem Master Passwort einloggen und befehle direkt ausführen.
* **Dev\_console\_enabled=true**
  + Die Dev\_console ist der weiße Kasten mit der Lianen Schrift. In dieser werden alle Meldungen ausgegeben.
* **Pi\_Build=false**
  + Pi build gibt wiederrum an, ob LED’s angesteuert werden sollen. Dieser Befehl bezieht sich nur auf Netzwerk befehle. Es sollte nur auf true gesetzt sein, wenn das PiFace angeschlossen ist.

NameConfig.properties

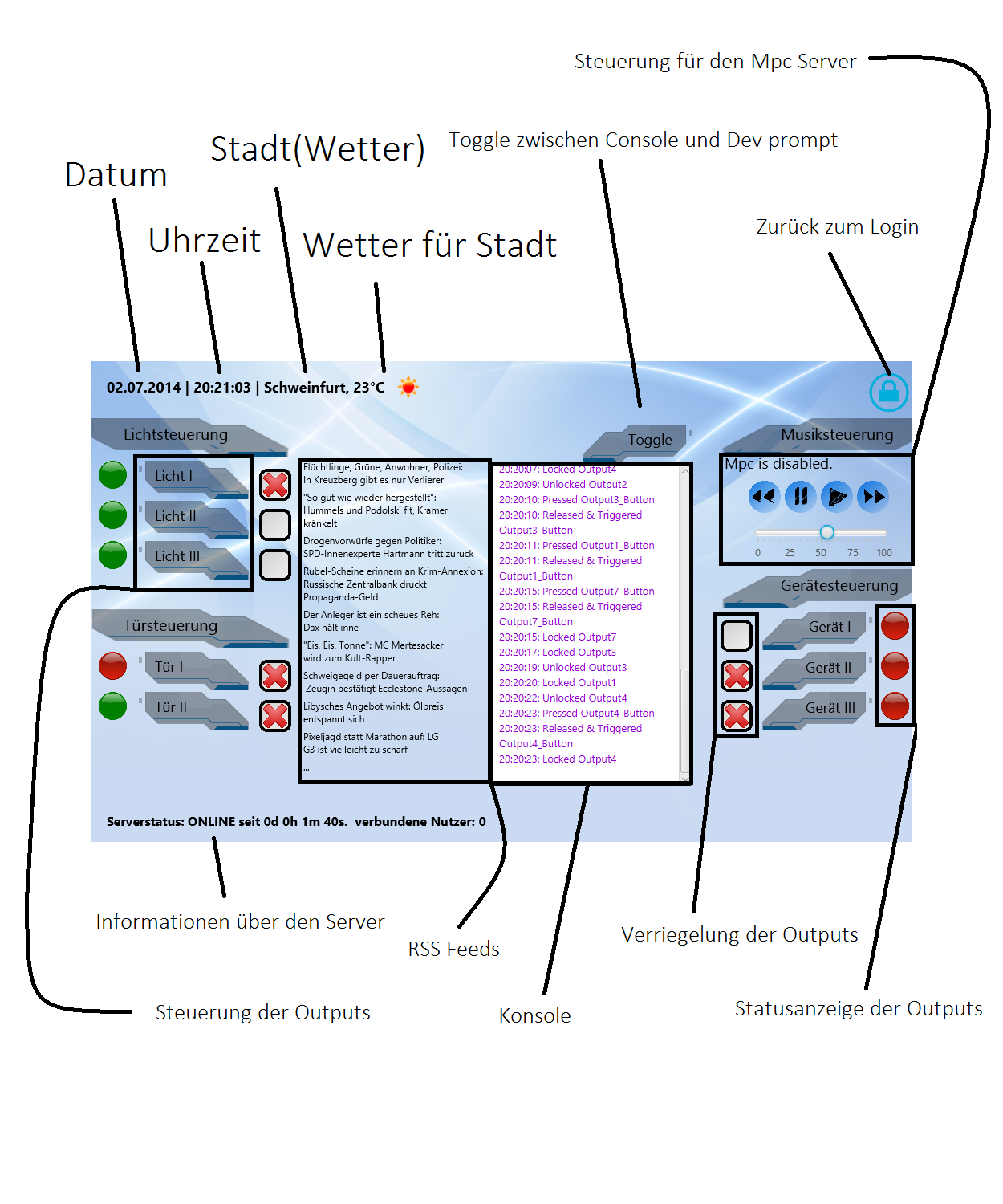
* **Version=1**
* **Head0\_Name=Lichtsteuerung**
* **Output0\_Name=Licht I**
* **Output1\_Name=Licht II**
* **Output2\_Name=Licht III**
* **Head1\_Name=T\u00FCrsteuerung**
* **Output3\_Name=T\u00FCr I**
* **Output4\_Name=T\u00FCr II**
* **Head2\_Name=Ger\u00E4testeuerung**
* **Output5\_Name=Ger\u00E4t I**
* **Output6\_Name=Ger\u00E4t II**
* **Output7\_Name=Ger\u00E4t III**

Diese config sollte selbsterklärend sein. Das \u00FC steht für „ü“. \u00E4 für ein „ä“.

RSSFeeds.txt

* http://www.n-tv.de/rss
* http://www.reddit.com/.rss

In dieser **Datei** kann man **Feeds** **hinzufügen**. Pro Zeile kann man einen Feed hinzufügen. Dabei darf die **maximale** **Anzahl** von **10** nicht überschritten werden

Layout

Server und Netzwerkfunktion

Das **Programm** beinhaltet einen **Java Socket Server**, der **von jedem Gerät angesteuert** werden kann.

Um von **außerhalb** auf den **Server** **zuzugreifen** müssen **Ports** **freigegeben** werden.

Der **Server** ist fähig **befehle** von **Clients** **entgegenzunehmen** und diese ggf. zu **beantworten**.

Funktionsweise und API

Um dem **Server** **Befehle** zu geben, die etwas **bewirken**, muss man sich **einloggen**.

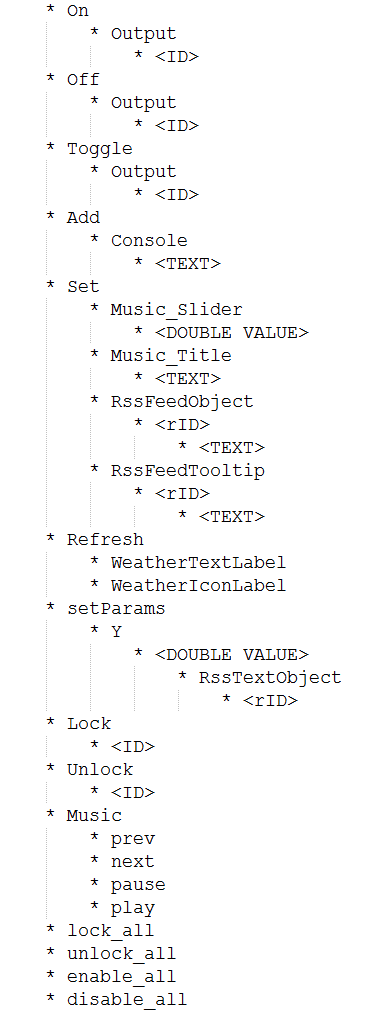
Hierzu muss der **Client** einen **String** an den **Server** **schicken**.

Die **Standartformatierung** für den **String**, welcher zum **Einloggen** benutzt werden ist:

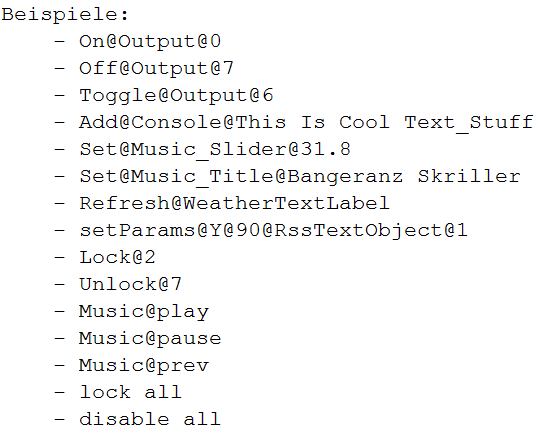
/login <Username> <Passwort>

Wenn Username und Passwort in der Datenbank vorhanden sind, schickt der Server einen sog. Private Key zurück. Dieser besteht aus 32 Zufälligen Zeichen und wird dazu genutzt um sog. AuthAction durchzuführen. Diese funktionieren wie folgt:

/AuthAction <Private Key> <arg1>@<arg2>@<arg3>@<arg4>…

Zurzeit kann man die Folgenden Befehle mit einer AuthAction ausführen

Die ID kann eine beliebige Zahl zwischen 0 und 7 sein. Die rID kann 0-9 sein.



Möglichkeiten der Software

Mit der **Software** und der **Hardware** ist es **Theoretisch** möglich ein **komplettes** **Haus** **elektronisch** über PI’s zu **Steuern**. Dafür bräuchte man **einen** **Master** PI, der sich um alle **befehle** **kümmert** und diese wie ein **Rooter** an die richtigen PI’s weiter **schickt**.

Des Weiteren braucht man für **je 7 Möglichkeiten** zum **Ansteuern** einen neuen **PI**, welcher als **Client** mit dem **Server** **verbunden** werden muss. Dieser hat die **gleiche** **Nutzeroberfläche**, aber **anstatt** **selbst** etwas direkt zu **machen**, gibt er den **Befehl** zuerst an den **Master** **PI** **weiter**, welcher sich um alles weitere kümmert.

Dieser gibt dann dem **richtigen** **PI** den **Befehl** etwas zu tun. Dieses **Konzept** ist mit der **Software**, die auf dem **HomeControl** **Pi installiert** ist **bereits** mit **wenigen Zeilen Code mehr umsetzbar**.

Zuger die **Handy** **App** wäre damit **kompatibel**. So kann man auch **unterwegs** nach dem **Rechten** sehen. Diese könnte dann z.B. auch einen **Video** **Stream**, der von einer **Webcam** auf den **PI** geleitet wird zum **Handy** leiten und das diesen wiederrum anzeigen.

Einige Möglichkeiten sein Haus zu digitalisieren wären:

* Relais und Elektrische Schlösser an Türen, um diese auf und zu zumachen.
* Relais an Lichtern im Haus, um diese an oder aus zu schalten
* Geräte wie z.B eine Kaffeemaschine anschalten
* Sensoren an den Türen um zu sehen wann sie geöffnet wurden ( bei Bedarf auch per email Nachricht an Handy wenn man außer Haus ist )
* Sensoren an den Fenstern um zu sehen, welche Fenster offen sind.
* Motoren an den Fenster sind theoretisch auch möglich um diese automatisch zu bedienen
* Webcam an der Tür als Fernsprechanlage
* Steuerung und Sensoren an Rollläden

Wie man sehen kann, sind die Möglichkeiten schier unbegrenzt.

So kann man sein komplettes Haus für wenig Geld in ein digitales Wunder verwandeln