

# IOT Schulung

## Arduino

### 2. Teil

## Tagesablauf

- Vormittag
  - Besprechung mit Teilnehmer
    - Erfahrungen mit Thingworx
    - Eigene Projekte?
    - Hat für die Realisierung eines Projektes etwas gefehlt?
  - Ausgabe neue Hardware
  - Kurzer Ausblick über zukünftige Entwicklungen seitens der Vortragenden
  - Wiederholung des ersten Seminarteiles Teil 1
    - Anschließen der Hardware
    - HTTP Request
    - Neue Bibliothek für Arduino MKR Wifi 1010
    - Arduino IDE Installation für das Inbetriebnehmen der Schulungshardware

## Tagesablauf

- Vormittag
  - Neuer Thingworx Server
    - Aufbau
    - Wichtige Einstellungen für die Inbetriebnahme

## Tagesablauf

- **Nachmittag**
  - Wiederholung des ersten Seminarteiles Teil 2
    - Inbetriebnahme des neuen Arduinos am neuen Server
  - Einbindung eines neuen Sensor (hier der Ultraschallabstandssensor HC SR04)
    - Wie bindet man einen neuen (unbekannten) Sensor ein?
    - Funktionsweise des HC SR04
    - Einbindung des HC SR04 in den Arduino Code
    - Ausgabe im seriellen Monitor
  - Hausübung Drehdecoder KY-040 bis zum 3. Teil
    - Aufgabenstellung

## Besprechung mit Teilnehmer

- Welche Erfahrungen habt ihr gemacht?
- Wurden eigene Projekte realisiert?
- Wurden Diplomarbeiten realisiert?
- Hat für die Realisierung eines Projektes etwas gefehlt?

## Neue Hardware

- Arduino MKR Wifi 1010
- Ultraschall Abstandssensor HC SR04
- Kodierter Drehschalter KY-040
- Steckboard
- Benötige Jumper Kabel zum Verbinden

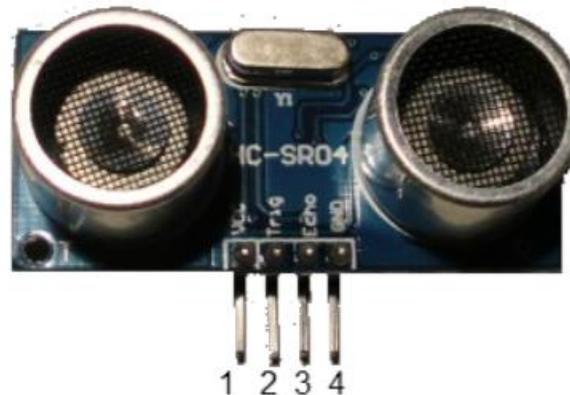
## Neue Hardware

- Arduino MKR Wifi 1010
  - Link: <https://store.arduino.cc/mkr-wifi-1010>
  - ESP32 basierter Mikrocontroller mit Bluetooth und Wifi- Anbindung
  - Günstiger als MKR 1000 Modul
  - Entwicklung einer GUI (Graphical User Interface) auf ESP32 Modul im Gange



## Neue Hardware

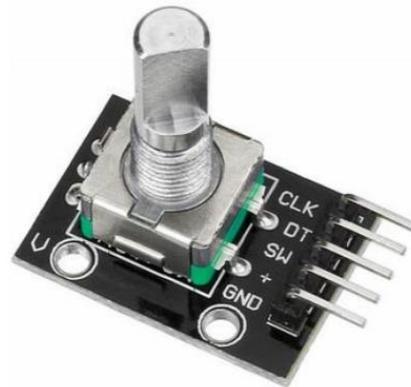
- Ultraschall Abstandssensor HC SR04
  - Datenblatt: [https://www.mikrocontroller.net/attachment/218122/HC-SR04\\_ultraschallmodul\\_beschreibung\\_3.pdf](https://www.mikrocontroller.net/attachment/218122/HC-SR04_ultraschallmodul_beschreibung_3.pdf)
  - Abstand wird basierend auf einer Ultraschall Laufzeitmessung ermittelt



Quelle: [https://www.mikrocontroller.net/attachment/218122/HC-SR04\\_ultraschallmodul\\_beschreibung\\_3.pdf](https://www.mikrocontroller.net/attachment/218122/HC-SR04_ultraschallmodul_beschreibung_3.pdf) (19.10.2019)

## Neue Hardware

- Kodierter Drehschalter KY-040
  - Datenblatt: <https://www.handsontec.com/dataspecs/module/Rotary%20Encoder.pdf>
  - Ermittlung zur Stellung eines rotierenden Körpers inkl. Drehrichtungsangabe



Quelle: <https://www.handsontec.com/dataspecs/module/Rotary%20Encoder.pdf> (19.10.2019)

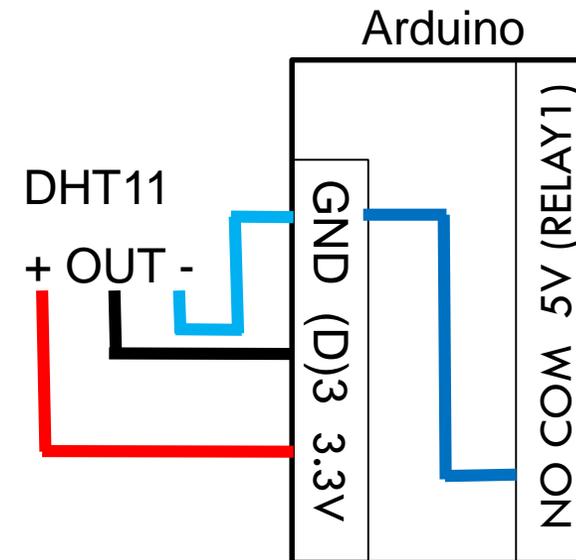
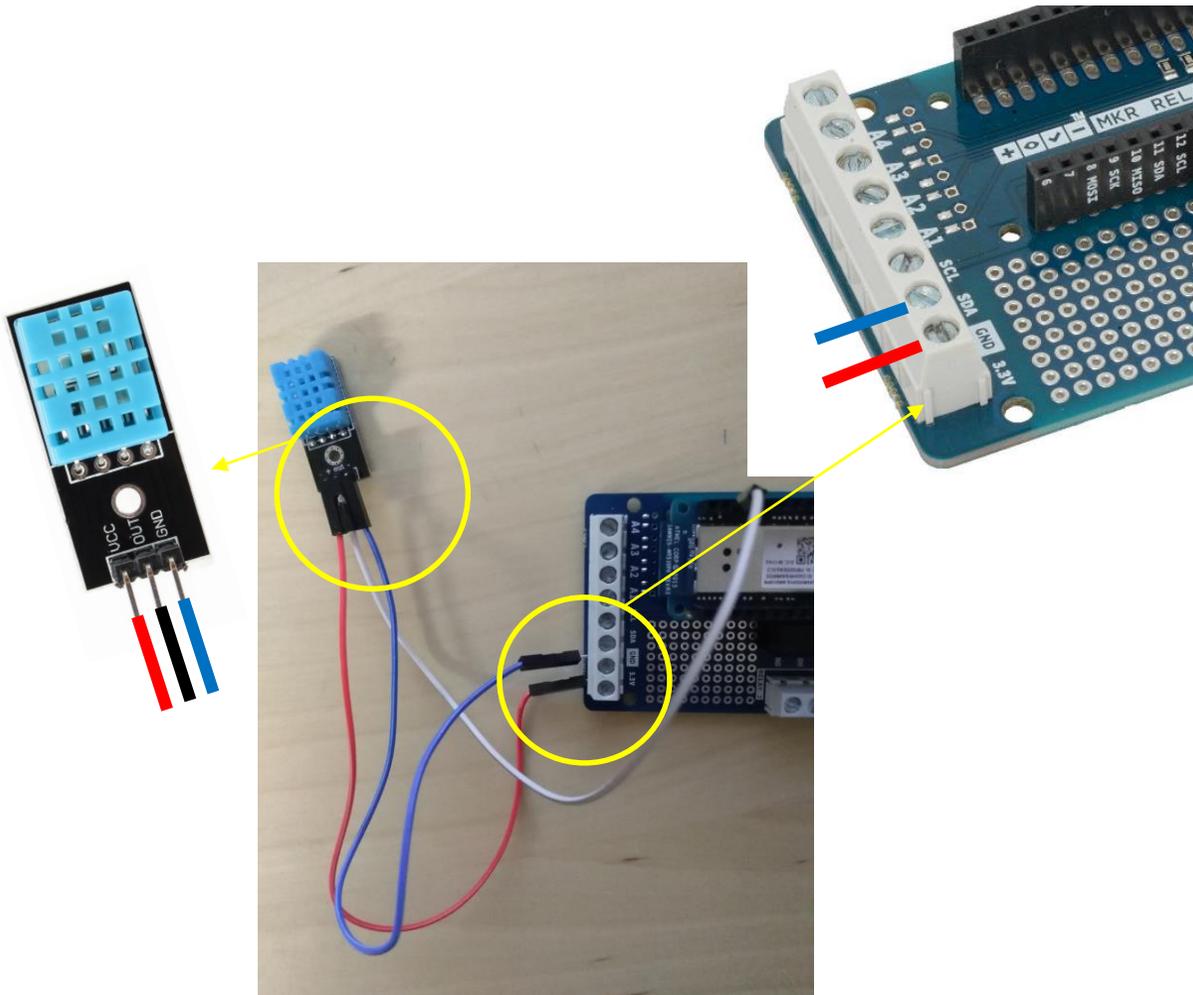
## Zukünftige Entwicklungen

## ESP32 Webserver

- Erstellung einer GUI (Graphical User Interface) für das Handling von Thingworx
- Verbindung mittels Laptop/PC/Handy mit Arduino MKR Wifi 1010
- Einstellungen des Servers, Things, Passwörter, usw. per Drop Down Menü
- Ebenfalls werden oft verwendete Sensoren im Bereich Maschinenbau eingebunden
- Idee dahinter: Schnelles schicken der Daten von oft verwendeten Sensoren im Bereich Maschinenbau ohne Erstellung eines Codes (ohne Programmierkenntnisse) an die Thingworx Cloud.
- IN ENTWICKLUNG!!! von Martin Schubert und Armin Fischer (HTL Wiener Neustadt, Abteilung Automatisierungstechnik) → [https://youtu.be/lytl2\\_7vRSE](https://youtu.be/lytl2_7vRSE)

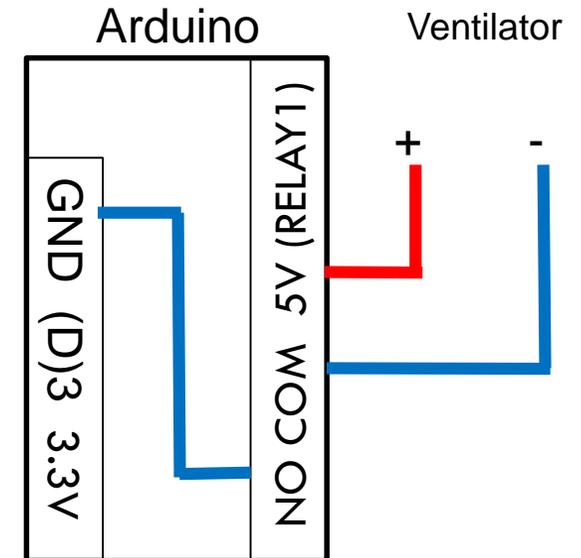
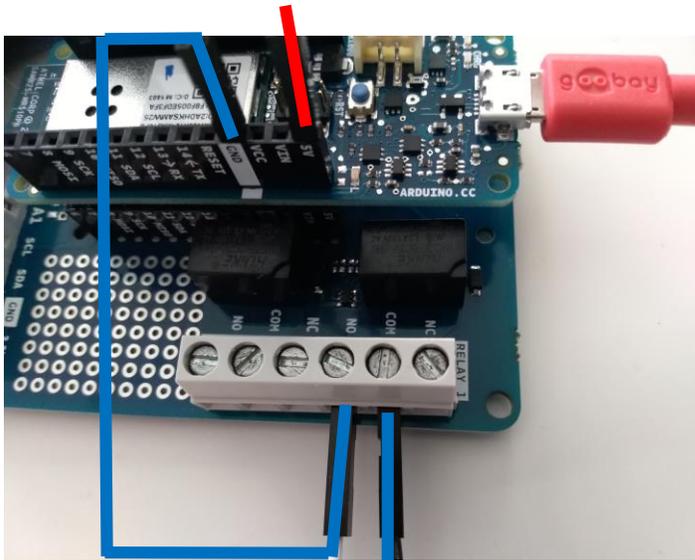
Wiederholung des ersten Seminarteiles

Anschluss DHT11



Wiederholung des ersten Seminarteiles

Anschluss Ventilator



Quelle: Seminarunterlagen IOT Schulung Teil 1, Armin Fischer

Wiederholung des ersten Seminarteiles

Datenübertragung

# WIE WERDEN DATEN IN DIE CLOUD ÜBERTRAGEN?

- Thing 1
  - Property 1
  - Property 2
  - ...

- Thing 2
  - Property 1
  - Property 2
  - ...

CLOUD <https://twx.htl.schule>  
(Server Mödling mit  
Thingworx Composer Software)

- Arduino (Thing)
  - Temperatur (Property)
  - Feuchtigkeit (Property)
  - Ventilator (Property)

Kommunikation  
Arduino und Cloud  
Http Request  
Sicherheitschlüssel  
zur Anmeldung



Arduino MKR1000 (Thing)



DHT11 (Property)



Ventilator (Property)

## Wiederholung des ersten Seminarteiles

## HTTP Request

## HTTP REQUEST

- Protokoll zum Übertragen von Daten (genauso wie im Browser)
- 1 HTTP Request um 1 Sensorwert (Property) vom Arduino (Thing) auf die Cloud zu übertragen
- HTTP Request enthält Sicherheitsschlüssel um eine automatische Authentifizierung/Anmeldung zur Cloud zu erhalten
- Beispiel

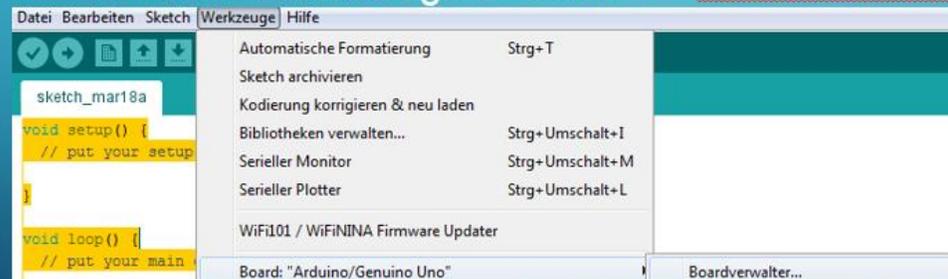
Thingworx Server Adresse	Thing	Property
<a href="https://twx.htl.schule/Thingworx/Things/920417_18FISA_00_Dummy/Properties/Sensorwert_1">https://twx.htl.schule/Thingworx/Things/920417_18FISA_00_Dummy/Properties/Sensorwert_1</a>		
<a href="https://twx.htl.schule/Thingworx/Things/920417_18FISA_00_Dummy/Properties/Sensorwert_1?appKey=de5d03ab-6cac-4e13-a4a9-9b5bfe9f7eda">?appKey=de5d03ab-6cac-4e13-a4a9-9b5bfe9f7eda</a>		
Sicherheitsschlüssel		

Wiederholung des ersten Seminarteiles

Konfigurieren der Arbeitsumgebung

# ARDUINO IDE

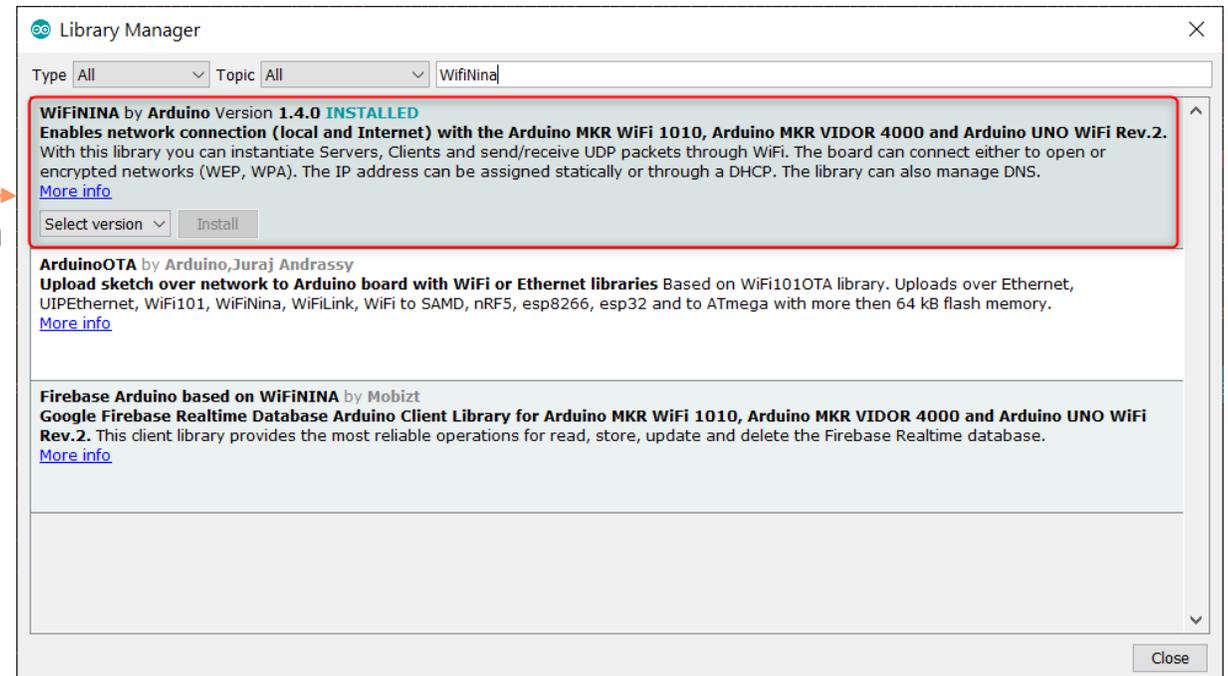
- Software zum Programmieren von Arduino Mikrocontroller
- Download unter <https://www.arduino.cc/en/main/software>
- Um das Arduino MKR1010 Board verwenden zu können muss dies installiert werden: **Werkzeuge > Board > Boardverwalter > Arduino SAMD Boards**



## Wiederholung des ersten Seminarteiles

## Neue Bibliotheken für Arduino MKR 1010

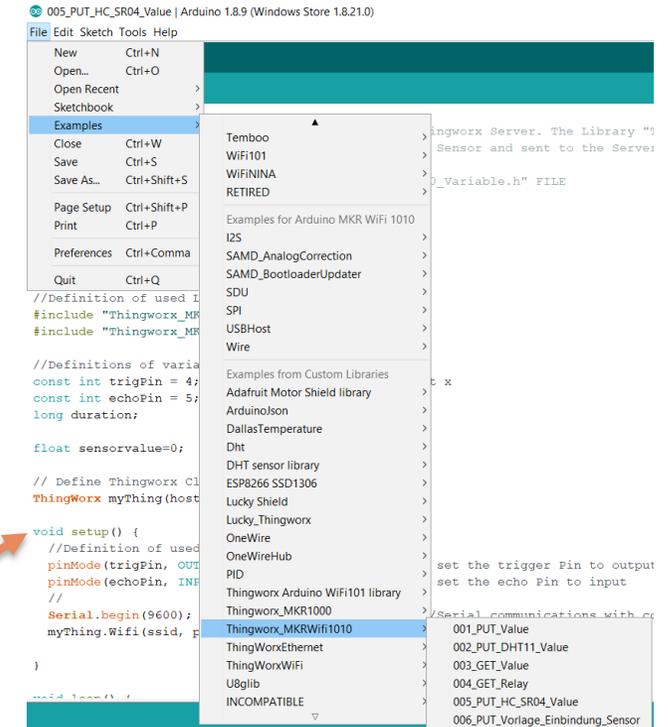
- Der neue Prozessor (jetzt ESP32 vorher SAMD21 Cortex-M0+) bedingt neue Befehle. Darum muss eine neue Library eingebunden werden.
- Libraries = Vorprogrammierte Codes, welche mit Befehlen aufgerufen werden.
- Benötige Libraries für den HTTP Request
  - WifiNINA.h
  - (Werkzeuge > Bibliotheken verwalten > WifiNINA.h)
  - Thingworx MKR1010
  - Dht.h



## Wiederholung des ersten Seminarteiles

- Libraryname: Thingworx\_MKRWifi1010
- Library von Armin Fischer geschrieben
- Muss in folgenden Ordner kopiert werden \Documents\Arduino\libraries
- Mit dieser Library sind folgende Operationen möglich
  - Verbinden mit WLAN
  - Abfragen eines Wertes am Thingworx Server
  - Schicken eines Wertes zum Thingworx Server
- In der Arduino IDE unter Beispiele befinden sich auch vorprogrammierte Beispiele.

## Neue Bibliotheken für Arduino MKR 1010



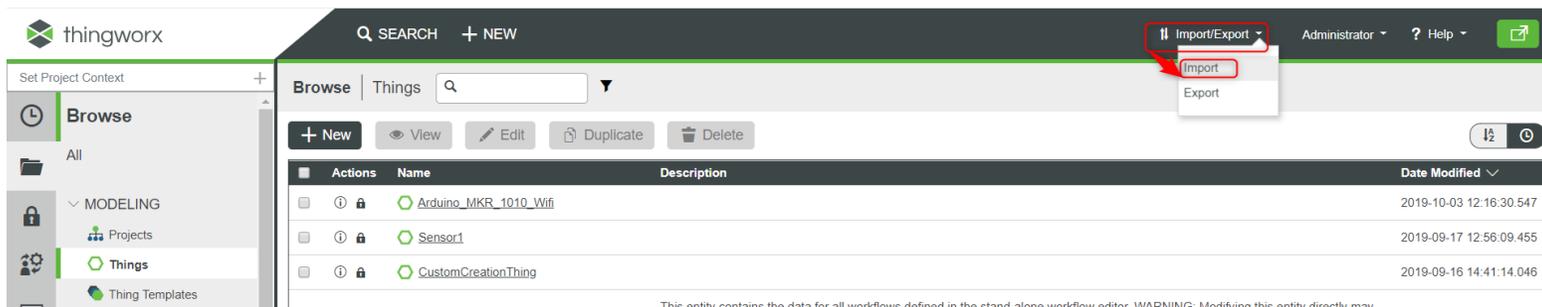
## Neuer Thingworx Server

- Neue Serverarchitektur (weltweit einzigartig) an der HTL Mödling
- Jede Schule hat einen Thingworxadministrator, welcher berechtigt ist, bis zu derzeit 2 Instanzen zu erstellen.
- Eine Instanz besteht aus einem Vuforia Studio Experience und einem Thingworx Composer Server.
- Vorteile zum Vorgänger:
  - Jede Schule arbeitet unabhängig der anderen Schulen
  - Für Laborübungen können Instanzen einfach gelöscht und wieder erstellt werden.

## Neuer Thingworx Server

## Einstellungen

- Es müssen folgende Einstellungen am Server gemacht werden um Daten mit einem Thing auszutauschen
1. Auf den Server eine limitierte Admingruppe erstellen
    - Es wird dazu ein File namens “Things\_CustomCreationThing.xml“ benötigt. ACHTUNG: Dieses File ist gerade in Bearbeitung und wird verbessert.
    - Importieren dieses Files



Neuer Thingworx Server

Importieren des Files

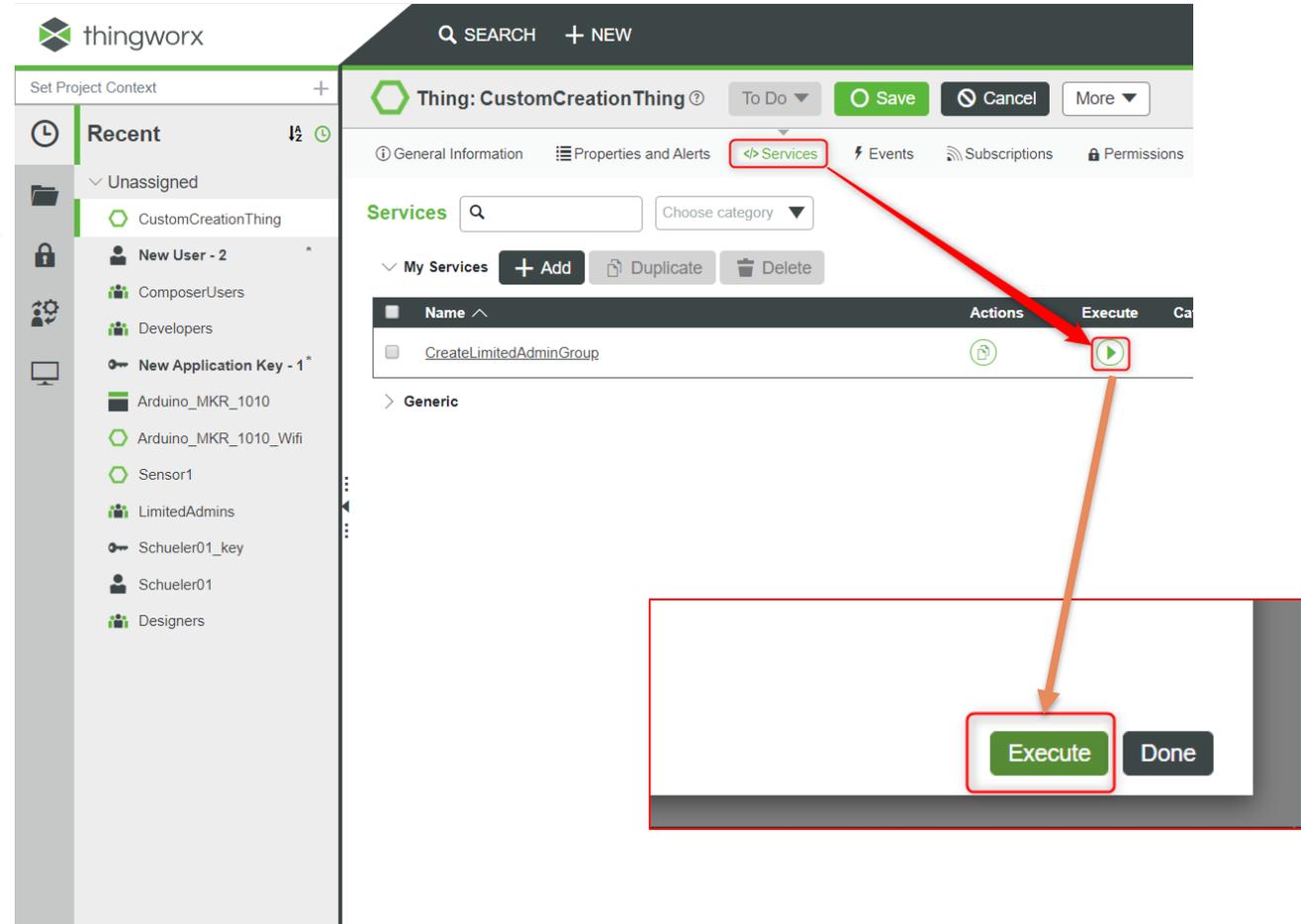
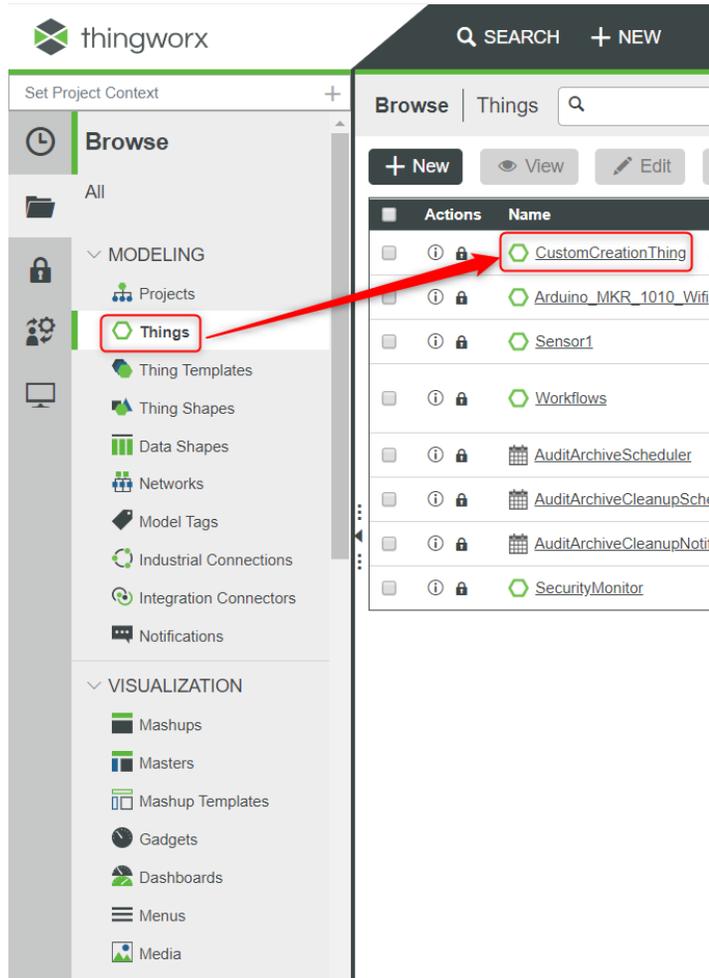
The screenshot shows the Thingworx web interface with the 'Import' dialog box open. The dialog has the following fields and options:

- Import Option:** From File
- Import Type:** Entity
- Use default Persistence Provider
- Include Subsystems
- Import Source:** Single File
- File Name (required):** Things\_CustomCreationThing
- File is required:**

A file explorer window is overlaid on the right, showing the file 'Things\_CustomCreationThing' selected in the 'DATA (D:)' directory. The file name is entered in the 'Dateiname:' field of the explorer. Red arrows point from the 'Browse' button in the import dialog to the file explorer, and from the 'Dateiname:' field to the 'Import' button in the dialog.

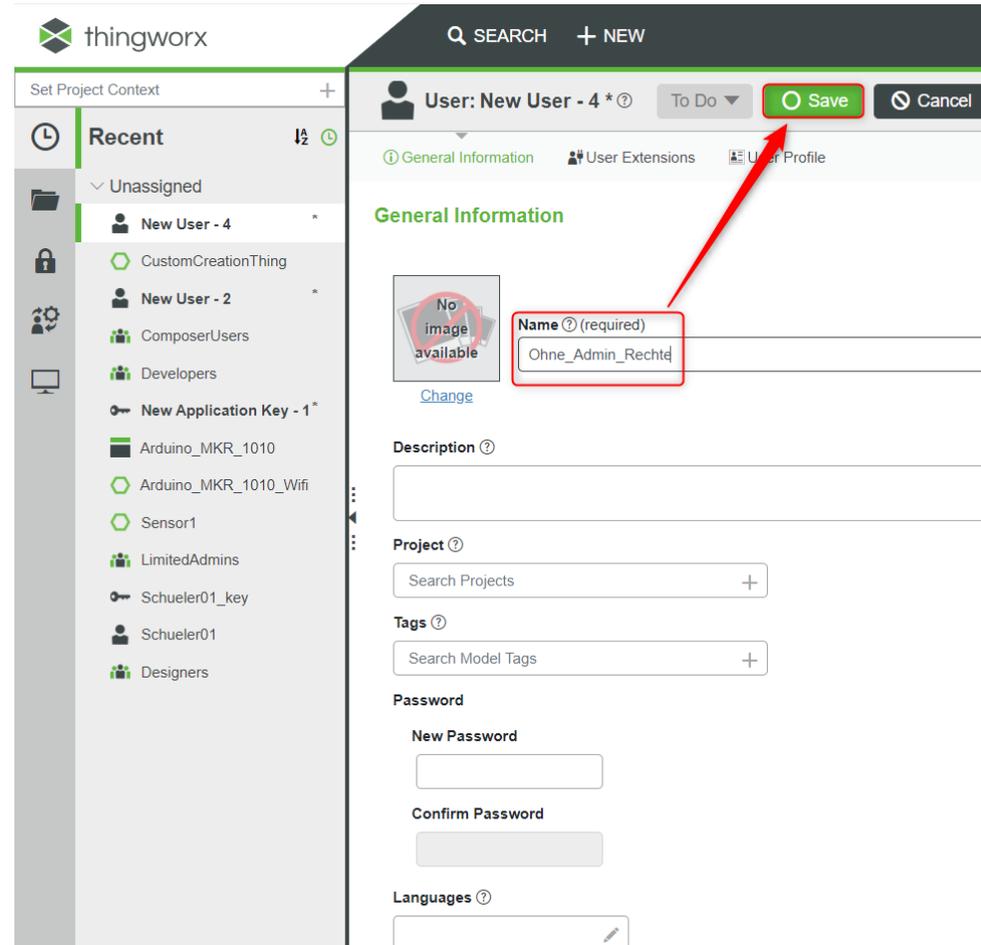
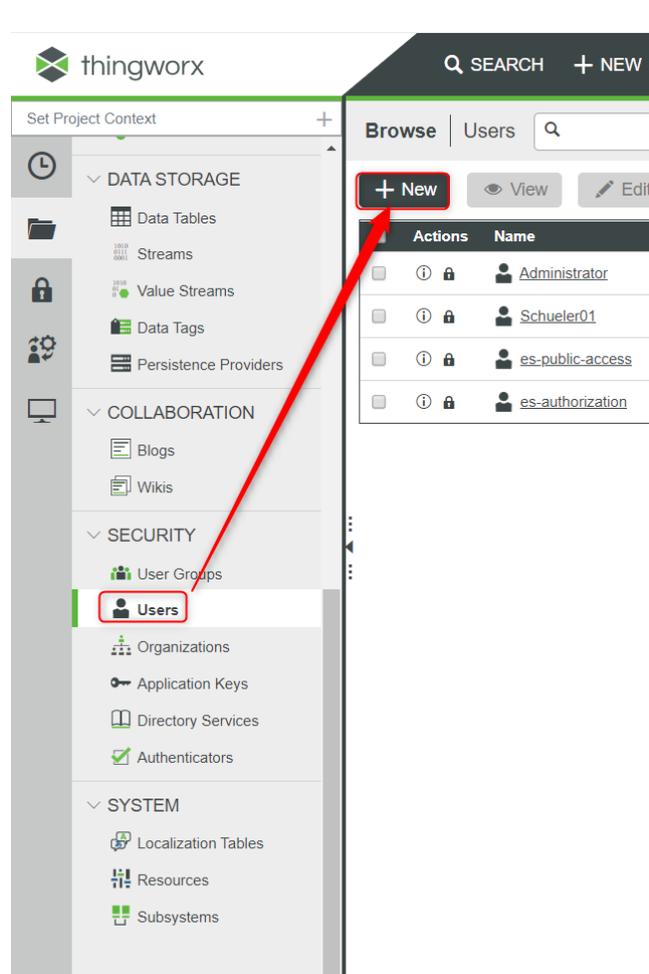
## Neuer Thingworx Server

## Service des Custom Creation Thing ausführen



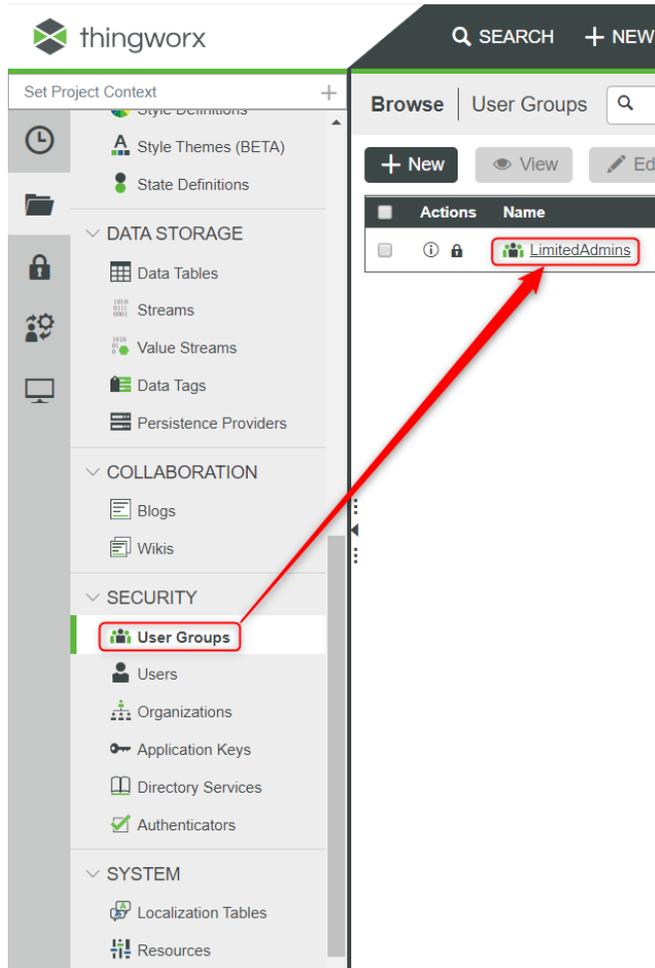
## Neuer Thingworx Server

## Neuen User „ohne Admin Rechte“ erstellen

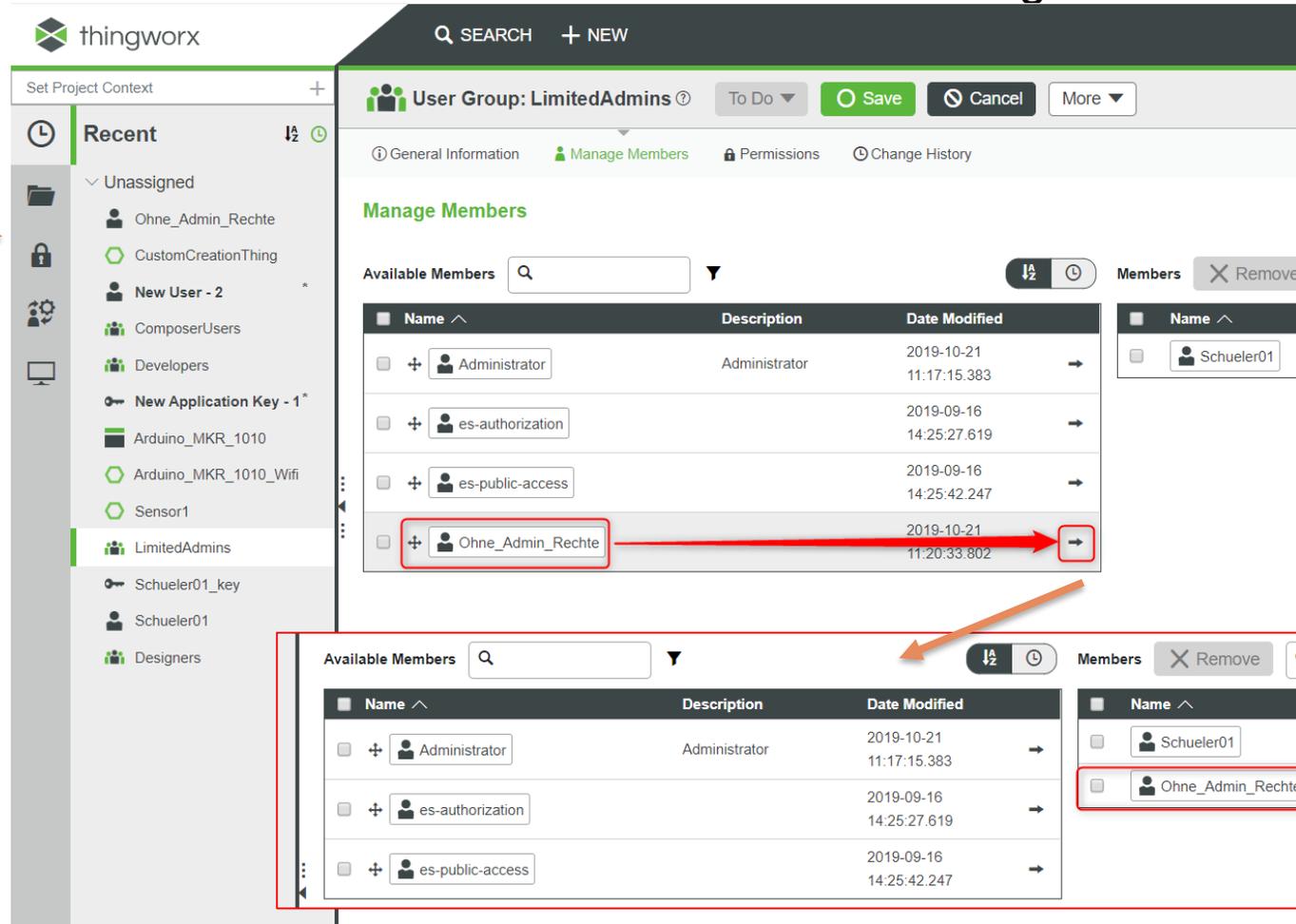


# Neuer Thingworx Server

# Neuen User „ohne Admin Rechte“ zur Gruppe LimitedAdmins hinzufügen



The screenshot shows the Thingworx interface with the 'User Groups' section selected in the left sidebar. The 'LimitedAdmins' group is highlighted in the 'Browse' view. A red arrow points from the 'User Groups' menu item to the 'LimitedAdmins' group, and another red arrow points from the 'LimitedAdmins' group to the 'Recent' list in the next screenshot.



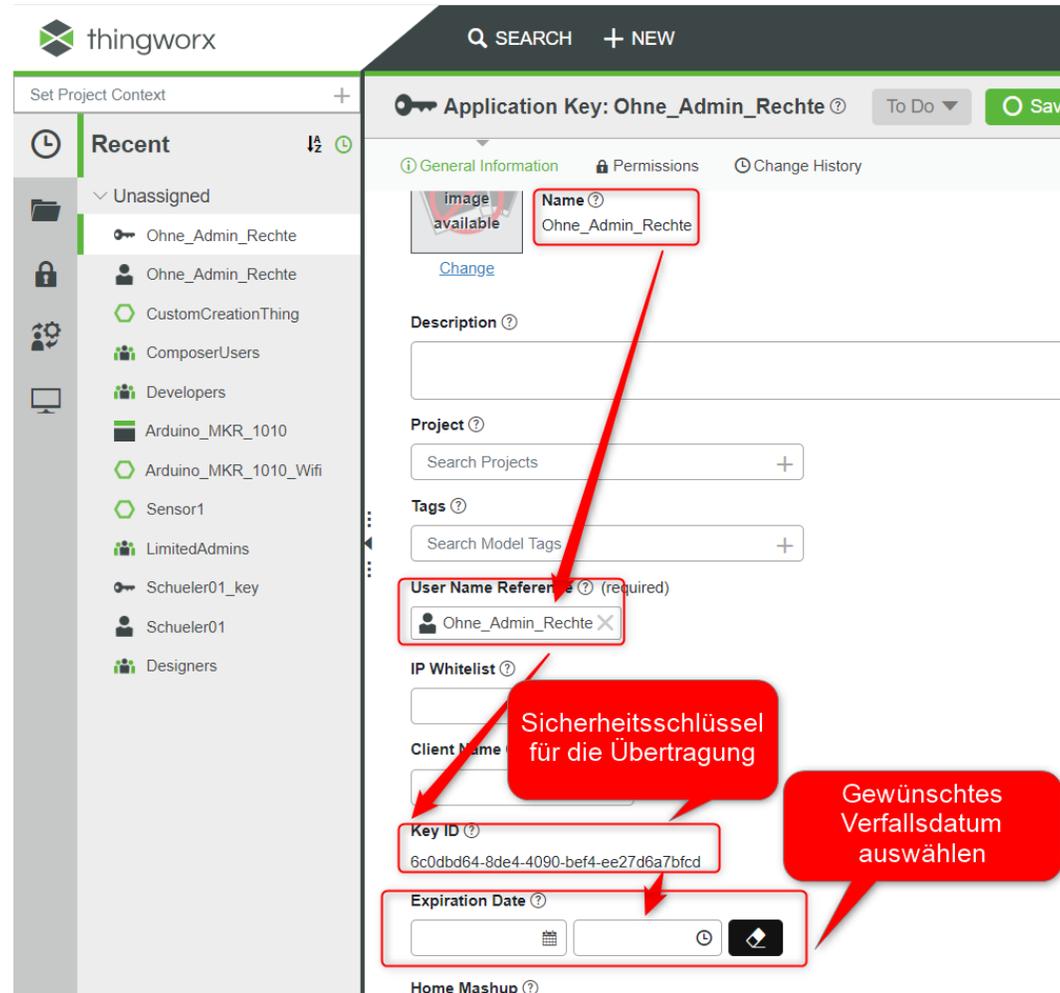
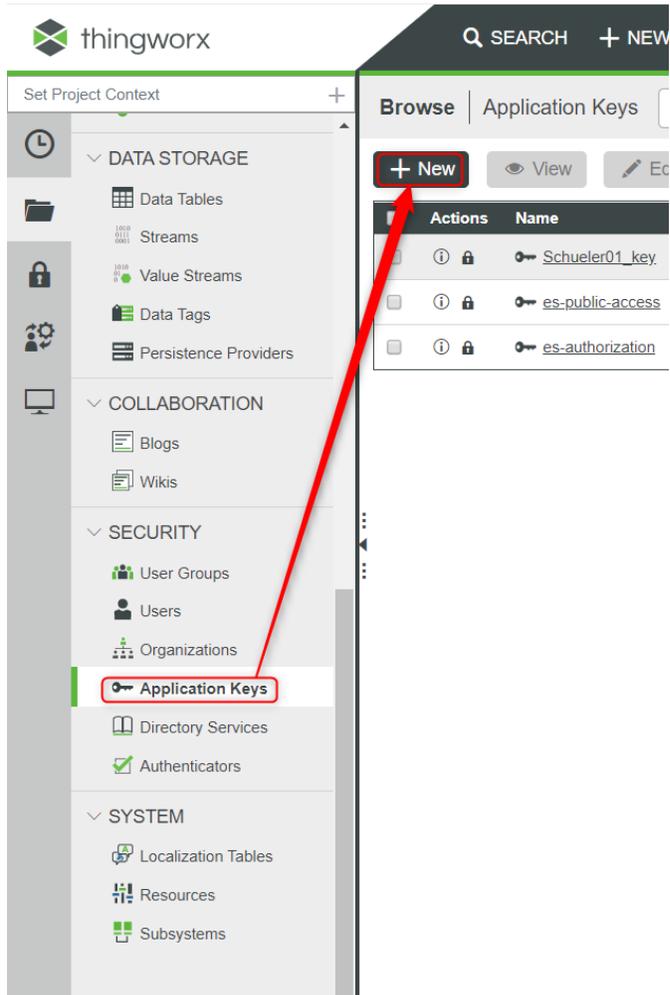
The screenshot shows the 'Manage Members' page for the 'LimitedAdmins' group. The 'Available Members' table lists several users, with 'Ohne\_Admin\_Rechte' highlighted. A red arrow points from the 'Ohne\_Admin\_Rechte' row to the 'Members' table, where it is now listed. An orange arrow points from the 'LimitedAdmins' group in the 'Recent' list to the 'Manage Members' page.

Name	Description	Date Modified
Administrator	Administrator	2019-10-21 11:17:15.383
es-authorization		2019-09-16 14:25:27.619
es-public-access		2019-09-16 14:25:42.247
Ohne_Admin_Rechte		2019-10-21 11:20:33.802

Name	Description
Schueler01	
Ohne_Admin_Rechte	

Neuer Thingworx Server

Sicherheitsschlüssel für den neuen User „ohne Admin Rechte“ erstellen



## Neuer Thingworx Server

## Dem User die erforderlichen Befugnisse für den Zugriff auf ein Thing erteilen

thingworx

SEARCH + NEW

Browse Things

+ New View Edit Duplicate Delete

Actions	Name	
<input type="checkbox"/>	CustomCreationThing	
<input type="checkbox"/>	Arduino_MKR_1010_Wifi	
<input type="checkbox"/>	Sensor1	
<input type="checkbox"/>	Workflows	This entity contains corrupt data
<input type="checkbox"/>	AuditArchiveScheduler	SchedulerThing
<input type="checkbox"/>	AuditArchiveCleanupScheduler	Scheduler for cleanup
<input type="checkbox"/>	AuditArchiveCleanupNotificationScheduler	Notify subscribe
<input type="checkbox"/>	SecurityMonitor	Security Monitor

Eigenes Thing auswählen

thingworx

SEARCH + NEW

Thing: Arduino\_MKR\_1010\_Wifi

General Information Properties and Alerts Services Events Subscriptions **Permissions**

Properties Alerts

My Properties + Add Duplicate Delete Manage Bindings Refresh

Name	Actions	Source	Default Value	Value
# DHT11_Moist	<input type="checkbox"/>			76

SEARCH + NEW

Permissions Entities \* Save Done Cancel

Visibility Run Time Design Time

Arduino\_MKR\_1010\_Wifi

Run Time

All Properties, Services, and Events

Remove Bulk Set Search Users or Groups +

User or Group	Property Read	Property Write	Service Execute	Event Execute	Event Subscribe
Schueler01	<input checked="" type="checkbox"/>				
Ohne_Admin_Rechte	<input checked="" type="checkbox"/>				

Property, Service, or Event Overrides Search Properties, Services or Events +

## Inbetriebnahme alte Hardware

Einstellungen in der  
Thingworx\_MKRWifi1010\_Variable.h Library

```
const unsigned long TPOST = 5000; //Time between requests to TwX server (every 5 secs)

//Wifi - Variables
//char* ssid = " "; //WiFi SSID
//char* password = " "; //WiFi Pass
char* ssid = " "; //WiFi SSID
char* password = " "; //WiFi Pass

//Host Thingworx
char* host = " ".twx.htl.schule"; //TWX Host for
unsigned int port = 443; //TWX host port for https

//Thingworx Variables
//char appKey[] = "c311cc04-b617-47ca-b2cd-584610ee5cbc"; //API Key from TwX
char appKey[] = " ";
char thingName[] = "Arduino_MKR_1010_Wifi";
//->Timing Vars
unsigned long lastConnectionTime = 0; //Last connection ms time between server requests

#endif
```

Wifi Einstellungen

Server Einstellungen

Sicherheitsschlüssel

Thing-Name

## Inbetriebnahme alte Hardware

- Selbstständige Inbetriebnahme der alten Schulungshardware auf neuem Server und mit MKR Wifi 1010 Controller
- Präsentation von letzter Schulung zur Hilfe nehmen!
- Schritte:
  - Richtiges Anschließen der Komponenten (siehe Folien vorher)
  - Arduino IDE konfigurieren (siehe vorher)
  - Server konfigurieren (siehe vorher)
  - Ein Thing und die zugehörigen Properties erstellen (siehe .ppt letzte Schulung)
  - In der Arduino IDE die Examples der Bibliothek durchprobieren

## Einbindung von neuem Sensor

- Um gängige Sensoren am Arduino anzuschließen gibt es 2 Varianten:
  - „Wissenschaftliche“ Methode: Durchlesen des Datenblattes des Sensors und ausprogrammieren des benötigten Algorithmus.
  - „Praktische“ Methode: Google Suche verwenden.
- Da von den Teilnehmern keine Programmierkenntnisse gefordert sind, wird hier auf die praktische Methode eingegangen.
- Hier sollte nochmals erwähnt werden: Für das Realisieren eigener Projekte ist das Erlernen der Grundkenntnisse der Arduino Programmiersprache (C++) unumgänglich!

## Einbindung von neuem Sensor

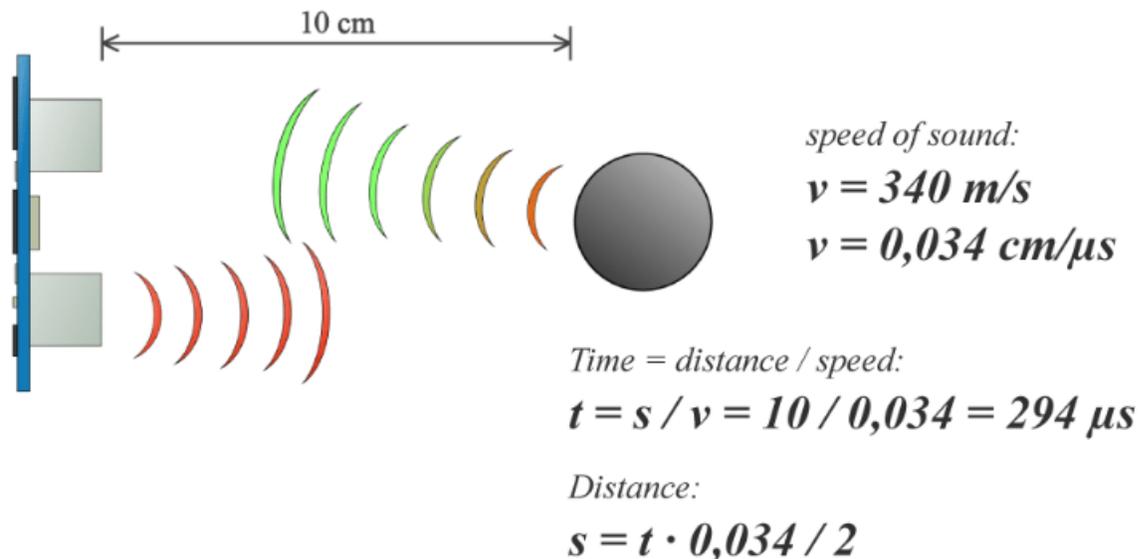
## Praktische Methode

- Arduino ist ein oft verwendeter Mikrocontroller. Viele Millionen Menschen hantieren damit weltweit.
- Man findet daher im Internet viele Bedienungsanleitungen, Algorithmen und Schaltpläne zu oft verwendeten Sensoren.
- Der Abstands-Ultraschallsensor HC SR04 ist ein oft verwendetet Sensor.
- Eine gute Anleitung befindet sich unter:  
<https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>

## Einbindung von neuem Sensor

## Wirkungsweise HC SR04

- Mit Hilfe des Wissens der Schallgeschwindigkeit und dem Messen der Zeit, kann eine Distanz  $s$  mit Hilfe der Formel  $s = v \cdot t$  ermittelt werden.
- Sehen wir uns folgendes Beispiel an:



Quelle: <https://howtomechatronics.com/tutorials/arduino/ultrasonic-sensor-hc-sr04/>

## Einbindung von neuem Sensor

## Verschaltung

- Auf das Steckbrett sollen 5V („+“) und GND („-“) mit Hilfe der Jumper Kabel geleitet werden.
- Der Port VCC des HC SR04 wird auf die 5V des Steckbrettes geleitet.
- Der Port GND des HC SR04 wird auf das GND des Steckbrettes geleitet.
- Der Port Trig des HC SR04 wird auf den digitalen Port D4 des Arduino MKR 1010 geleitet.
- Der Port Echo des HC SR04 wird auf den digitalen Port D5 des Arduino MKR 1010 geleitet.

## Einbindung von neuem Sensor

- Es ist auch folgender Algorithmus angegeben:
- Bei den Code Beispielen der Thingworx\_MKRWifi1010 Library befindet sich 006\_PUT\_Vorlage\_Einbindung\_Sensor
- Diese Vorlage kann für das Anbinden eines Sensors verwendet werden. Mit einem HTTP-Request wird der Wert zum Server geschickt.
- Die hier gezeigten Code Teile müssen nun an den richtigen Platz des Beispielen kopiert werden!

```

10.  const int trigPin = 9;
11.  const int echoPin = 10;
12.
13.  // defines variables
14.  long duration;
15.  int distance;
16.
17.  void setup() {
18.    pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
19.    pinMode(echoPin, INPUT); // Sets the echoPin as an Input
20.    Serial.begin(9600); // Starts the serial communication
21.  }
22.
23.  void loop() {
24.    // Clears the trigPin
25.    digitalWrite(trigPin, LOW);
26.    delayMicroseconds(2);
27.
28.    // Sets the trigPin on HIGH state for 10 micro seconds
29.    digitalWrite(trigPin, HIGH);
30.    delayMicroseconds(10);
31.    digitalWrite(trigPin, LOW);
32.
33.    // Reads the echoPin, returns the sound wave travel time in microseconds
34.    duration = pulseIn(echoPin, HIGH);
35.
36.    // Calculating the distance
37.    distance= duration*0.034/2;
38.
39.    // Prints the distance on the Serial Monitor
40.    Serial.print("Distance: ");
41.    Serial.println(distance);
42.  }

```

## Einbindung von neuem Sensor

### VORLAGE 006\_PUT\_Vorlage\_Einbindung\_Sensor

```
006_PUT_Vorlage_Einbindung_Sensor
//Definition of used Libraries
#include "Thingworx_MKRWifi1010.h"
#include "Thingworx_MKRWifi1010_Variable.h"

//Definitions of variables for the sensor
float sensorvalue=0;

// Define Thingworx Class (1 per Thing)
ThingWorx myThing(host, port, appKey, thingName);

void setup() {
  //Definition of used Pins by the Sensor
  //
  Serial.begin(9600);
  myThing.Wifi(ssid, password);
}

void loop() {
  if (millis() - lastConnectionTime > TPOST) //Send request to server every TPOST seconds
  {
    //Code for the sensor, the value should be stored in the variable "sensorvalue"

    //Send data with PUT Request to Thingworx
    myThing.put("Sensorvalue_Thingworx", sensorvalue); //Send distance to server platform

    lastConnectionTime = millis(); //Refresh last connection time for if
  }
}
```

Variablen für den Code

Einstellungen für den Code

Code für den Sensor

```
10. const int trigPin = 9;
11. const int echoPin = 10;
12.
13. // defines variables
14. long duration;
15. int distance;
16.
17. void setup() {
18.   pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
19.   pinMode(echoPin, INPUT); // Sets the echoPin as an Input
20.   Serial.begin(9600); // Starts the serial communication
21. }
22.
23. void loop() {
24.   // Clears the trigPin
25.   digitalWrite(trigPin, LOW);
26.   delayMicroseconds(2);
27.
28.   // Sets the trigPin on HIGH state for 10 micro seconds
29.   digitalWrite(trigPin, HIGH);
30.   delayMicroseconds(10);
31.   digitalWrite(trigPin, LOW);
32.
33.   // Reads the echoPin, returns the sound wave travel time in microseconds
34.   duration = pulseIn(echoPin, HIGH);
35.
36.   // Calculating the distance
37.   distance= duration*0.034/2;
38.
39.   // Prints the distance on the Serial Monitor
40.   Serial.print("Distance: ");
41.   Serial.println(distance);
42. }
```

## Einbindung von neuem Sensor

- Es entsteht folgender Code:
- Dieser Code ist als Nr. 005 In der Bibliothek vorhanden.
- Der Algorithmus wird hier beschrieben.

005\_PUT\_HC\_SR04\_Value

```
#include "Thingworx_MKRWifi1010.h"
#include "Thingworx_MKRWifi1010_Variable.h"
```

Es werden die benötigten Libraries eingebunden

```
//Definitions of variables for the sensor
const int trigPin = 4; //Trigger Pin on digital port x
const int echoPin = 5; //Echo Pin on digital port y
long duration;
```

Der Trigger Pin befindet sich auf D4 und der Echo Pin auf D5. Es wird eine Variable „Duration“ und „sensorvalue“ erstellt.

```
float sensorvalue=0;
```

```
// Define Thingworx Class (1 per Thing)
ThingWorx myThing(host, port, appKey, thingName);
```

```
void setup() {
  //Definition of used Pins by the Sensor
```

```
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
```

Der Pin 4 ist ein digitaler Output. Der Pin 5 ein digitaler Input

```
  Serial.begin(9600);
  myThing.Wifi(ssid, password);
```

//Serial communications with computer at 9600 bauds for debug purposes  
//Start the Wifi Connection

```
}
```

```
void loop() {
  if (millis() - lastConnectionTime > TPOST) //Send request to server every TPOST seconds
  {
```

```
    //Code for the sensor, the value should be stored in the variable "sensorvalue"
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    // Sets the trigPin on HIGH state for 10 micro seconds
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    // Reads the echoPin, returns the sound wave travel time in microseconds
    duration = pulseIn(echoPin, HIGH);
    sensorvalue= duration*0.034/2;
```

Benötigte Algorithmus für die Verwendung des HC SR04.

```
    //Send data with PUT Request to Thingworx
    myThing.put("HC_SR04",sensorvalue); //Send distance to server platform
```

Der Wert der Variable „sensorvalue“ wird auf die Property „HC\_SR04“ geschickt

```
    lastConnectionTime = millis(); //Refresh last connection time for if
```

```
}
}
```

## Einbindung von neuem Sensor

## Ausgabe des seriellen Monitors

- Da der Arduino keine Anzeige hat, gibt es folgende Möglichkeit um den aktuellen Status auslesen zu können → der serielle Monitor.
- In der Arduino IDE wird rechts oben auf das Lupensymbol geklickt. 
- Wenn der Beispielcode 005 ausgeführt wird, dann ergibt sich folgende Ausgabe:

## Einbindung von neuem Sensor

## Ausgabe des seriellen Monitors

```

COM3
Attempting to connect to SSID: ELVISO 70
SSID: ELVISO 70
IP Address: 192.168.43.229
Signal strength (RSSI):-31 dBm
Connected to: 64cbe.twx.htl.schule:443
PUT /Thingworx/Things/Arduino_MKR_1010_Wifi/Properties/HC_SR04 HTTP/1.1
Host: 64cbe.twx.htl.schule
Content-Type: application/json
Content-Length: 17
Connection: close
x-thingworx-session: false
appKey: 6c0dbd64-8de4-4090-bef4-ee27d6a7bfcd
{"HC_SR04":92.55}

***ANTWORT VOM CLIENT***
HTTP/1.1 200
Server: nginx
Date: Mon, 21 Oct 2019 12:11:01 GMT
Content-Type: text/html;charset=UTF-8
Transfer-Encoding: chunked
Connection: keep-alive
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Content-Security-Policy: frame-ancestors 'self'
X-Frame-Options: SAMEORIGIN
Expires: 0
Cache-Control: no-store, no-cache
Cache-Control: post-check=0, pre-check=0
Pragma: no-cache
0
  
```

Es wird probiert sich zu einem Wifi Netz zu verbinden

Wenn erfolgreich verbunden wurde, werden wichtige Stati und Eigenschaften angezeigt

Es wird ein PUT Request mit den aufgeführten Eigenschaften abgesetzt. Es Wird auf die Property HC\_SR04 der Wert 92.55 geschickt.

Hier die Antwort des Thingworx Servers. Hier sind die 2 markierten Zeilen von Bedeutung.

In der zweiten Zeile wird der HTTP-Status Code angezeigt. Der Wert 200 zeigt eine erfolgreiche Übertragung an. Jeder andere Wert gibt einen Fehler an.

In der letzten Zeile wird das Problem dargestellt. Bei einem Status Code von 200 gibt es keinen Fehler → der Wert 0 wird dargestellt.

## Einbindung von neuem Sensor

## Beispiele Fehler

- Falscher Propertyname

```
***ANTWORT VOM CLIENT:***  
HTTP/1.1 404
```

```
6b  
Unable To Write HC_SR05 on Arduino_MKR_1010_Wifi - Invalid Property &#x3a; HC_SR05 in Arduino_MKR_1010_Wifi  
0
```

- Falscher Server

```
***ANTWORT VOM CLIENT:***  
HTTP/1.1 503 Service Unavailable
```

```
<html><body><b>Http/1.1 Service Unavailable</b></body> </html>*****
```

- Falscher Sicherheitsschlüssel

```
***ANTWORT VOM CLIENT:***  
HTTP/1.1 401
```

- Link für HTTP Status-Tabelle: <https://de.wikipedia.org/wiki/HTTP-Statuscode>

## Hausübung Drehdecoder

- Bis zum 3. Teil der Schulung (März 2020) soll der kodierte Drehschalter KY-040 selbst in Betrieb genommen werden.
- Auf der Thingworxplattform sollen folgende Properties erstellt und angezeigt werden:
  - Clockwise
  - Counterclockwise
    - Wird gegen oder in den Uhrzeigersinn gedreht?
  - Position
    - Anzeige der aktuellen Position (dabei wird in den Uhrzeigersinn ein Schritt als +1 positiv und gegen den Uhrzeigersinn als -1 negativ gewertet).
- Es wird Anfang Februar eine Erinnerungs- und Hilfestellungsmail von mir geschickt!