**Cyber Physical Systems**

Wintersemester 2020/21

Hochschule München University of Applied Sciences

# Lektion 08: OPC UA Server einrichten und mit Android APP testen

von: Max Holzmann

In der Rallye Anwendung wird der OPC UA Server auf dem RallyMaster dafür genutzt, um für jegliche Art von OPC Client relevante Werte aus dem RallyMaster zu veröffentlichen. Konkret geht es hierbei, um die Anzahl an teilnehmenden RallyESPs, den Namen des aktuell RallyESPs sowie dessen Anzahl an Durchfahrten.

## Ziele

- Aufsetzen eines einfachen OPC UA Server auf dem Rasberry Pi mit Python
- Auslesen der Nodes Des OPC Server mit einer beliebigen OPC UA App

## Vorraussetzungen und erforderliches Equipment

- Die Tutorial Lektionen ...
  01: Grundlegende Einstellungen und Installationen link
  03: Konfigurieren des WLAN Chips des Raspberry pi als Acesspoint link
  04: Node Red installieren link
  ... müssen erledigt sein
- Hardware
  - 1x Raspberry Pi 3 B
  - 1x Android Smartphone
- Verwendete Software und verwendete Tools in diesem Tutorial
  - OPC UA Cleint for Android link zu Google Play

## Lösungsschritte

1. Installation der Python Bibliothek für OPC UA link to Git Repository

   ```
   pip3 install opcua
   ```

2. Anpassen des *server-minimal.py* Beispiels der Bibliothek für den Rallye OPC UA Server. Zum Datenaustausch von Node Red zum OPC UA Server wird eine json Datei verwendet. In dieser Datei schreibt Node Red jede Sekunde die aktuellen Daten und der OPC Server liest diese anschließend wieder aus.

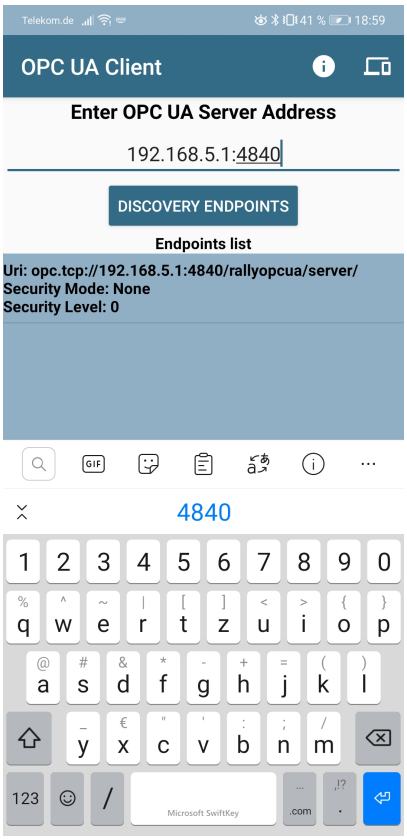
```

```

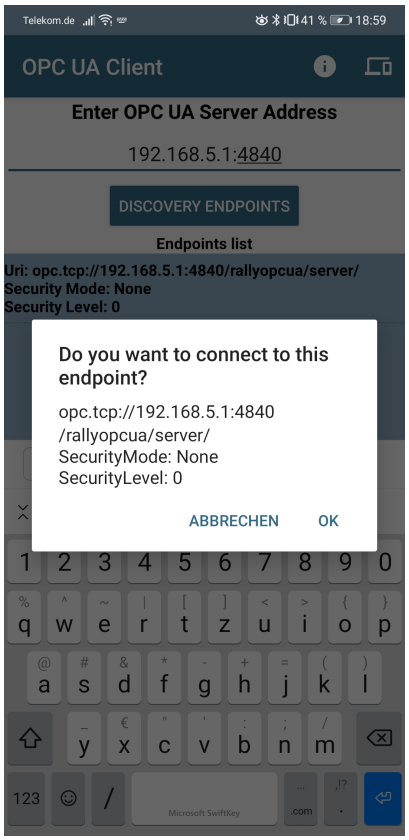
```
import sys
sys.path.insert(0, "..")
import time
import json

from opcua import ua, Server


if __name__ == "__main__":

    # setup server
    server = Server()
    server.set_endpoint("opc.tcp://192.168.5.1:4840/rallyopcua/server/")

    # setup namespace
    uri = "http://rally.bayern"
    idx = server.register_namespace(uri)
    print("node index: {}".format(idx))

    # get Objects node
    objects = server.get_objects_node()

    # adddress space with the data nodes
    rallyobj = objects.add_object(idx, "RallyApp")
    numberofcars = rallyobj.add_variable(idx, "NumberOfCars", 0)
    nameofcurrentcar = rallyobj.add_variable(idx, "NameCurrentCar", "test")
    measuremntsofcurrentcar = rallyobj.add_variable(idx, "MeasurenemtsCurrentCar",
0)

    # starting!
    server.start()

    try:
        while True:
            time.sleep(1)
            try:
                #get the Data from node Red via a json file
                jsonfile = open("/home/pi/rally/opcValues.json", "r")
                jsondata = jsonfile.read()
                #read json conntent
                jsondata = json.loads(jsondata)
                jsonfile.close()
                #set datanodes of OPC UA Server
                numberofcars.set_value(jsondata["numberOfCars"])
                nameofcurrentcar.set_value(jsondata["currentCarName"])

measuremntsofcurrentcar.set_value(jsondata["currentCarMeasurements"])
            except:
                pass
    finally:
        #close connection, remove subcsriptions, etc
        server.stop()
```

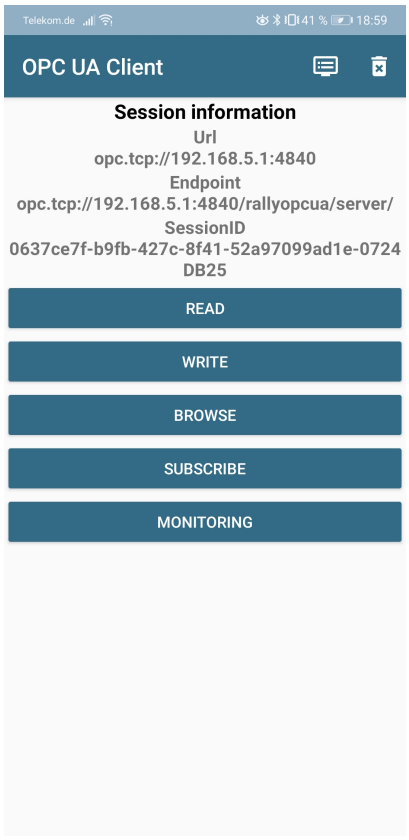
3. Installieren der Android App *OPC UA Client* aus dem Play Store.

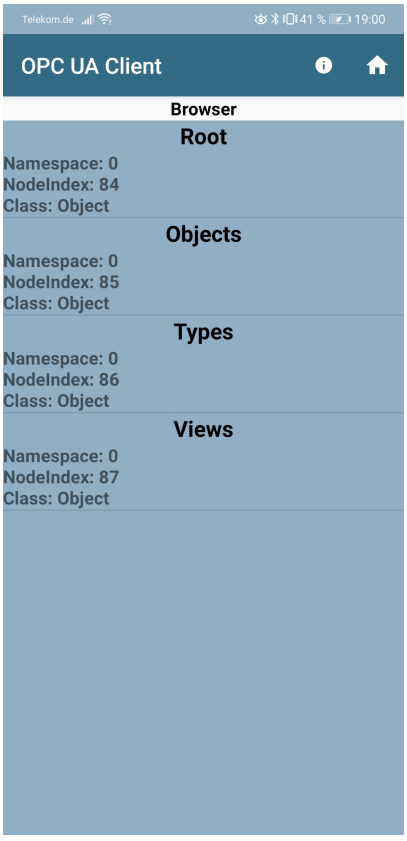
4. Verbinden des Smartphones mit dem WLAN Acesspoint des Rallye Masters

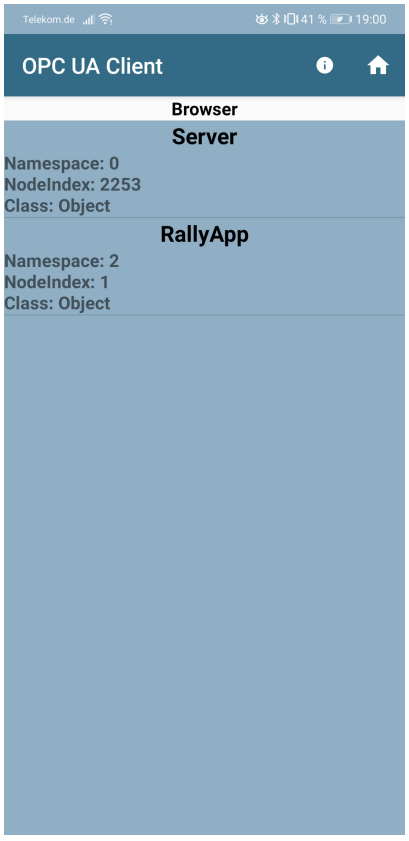
5. Der Bilderschritt-Anleitung folgen, um die Werte des Rallye OPC UA Servers lesen zu können.

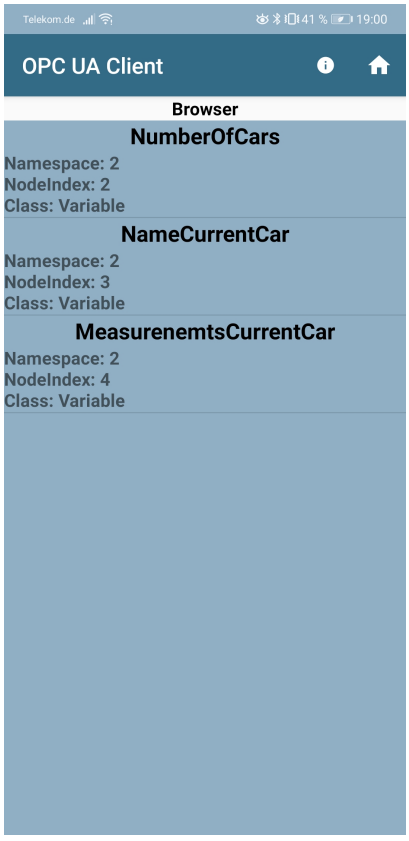


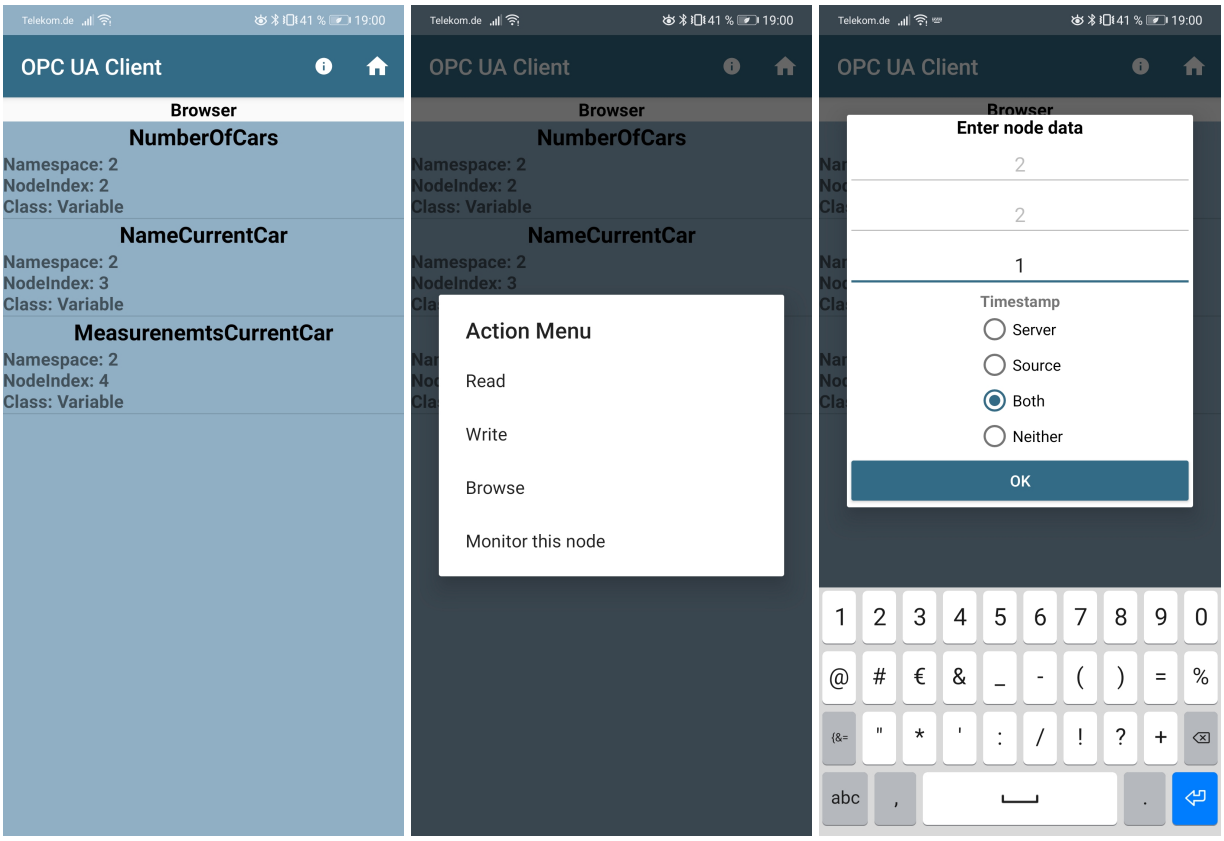



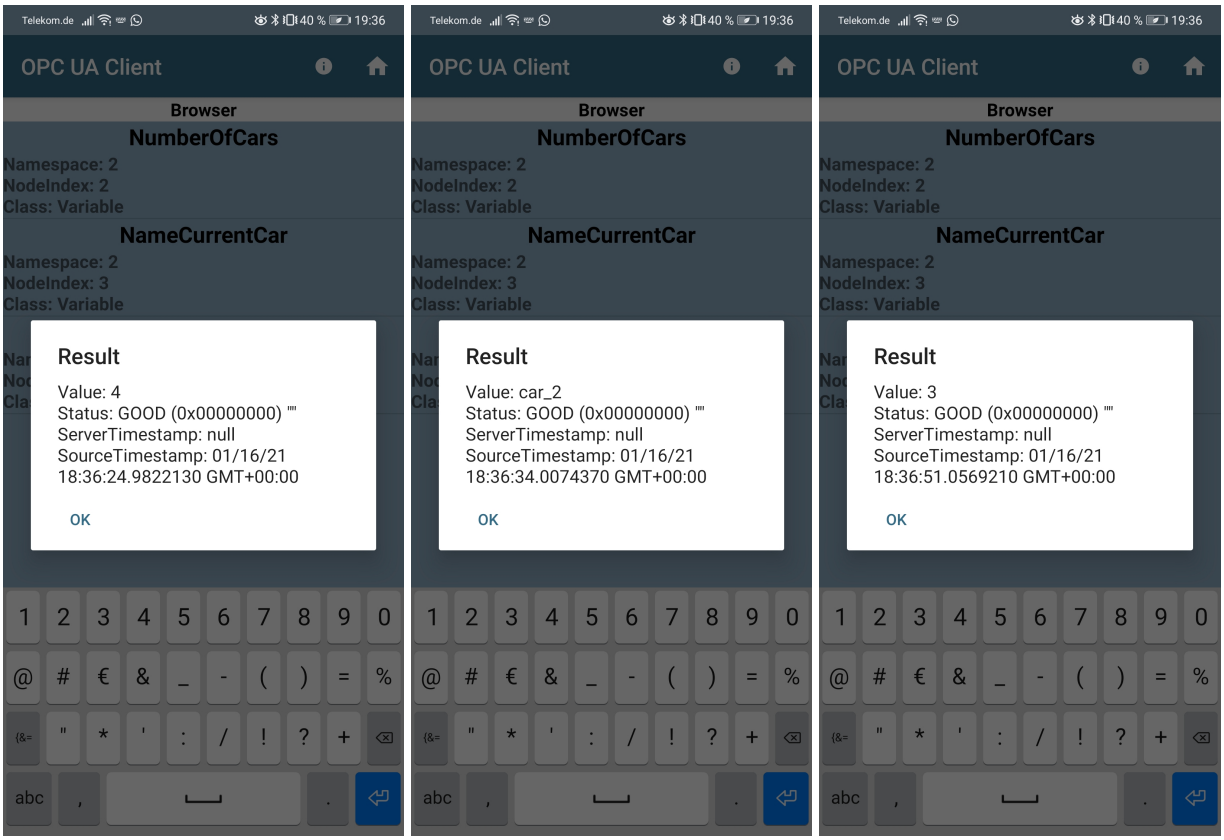



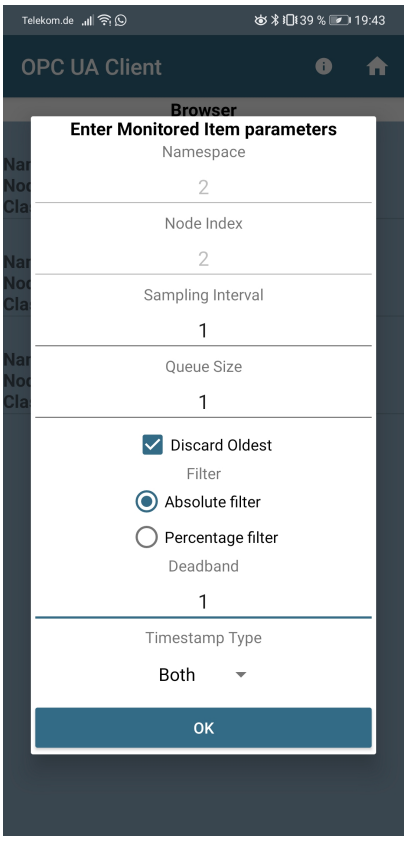




Telekom.de 41 % 19:00
OPC UA Client
Browser
NumberOfCars
Namespace: 2
NodeIndex: 2
Class: Variable
NameCurrentCar
Namespace: 2
NodeIndex: 3
Class: Variable
MeasurenemtsCurrentCar
Namespace: 2
NodeIndex: 4
Class: Variable
Action Menu
Read
Write
Browse
Monitor this node
Enter node data
2
2
1
Timestamp
Server
Source
Both
Neither
OK


Telekom.de 40 % 19:36
OPC UA Client
Browser
NumberOfCars
Namespace: 2
NodeIndex: 2
Class: Variable
NameCurrentCar
Namespace: 2
NodeIndex: 3
Class: Variable
Result
Value: 4
Status: GOOD (0x00000000) ""
ServerTimestamp: null
SourceTimestamp: 01/16/21 18:36:24.9822130 GMT+00:00
OK
Result
Value: car_2
Status: GOOD (0x00000000) ""
ServerTimestamp: null
SourceTimestamp: 01/16/21 18:36:34.0074370 GMT+00:00
OK
Result
Value: 3
Status: GOOD (0x00000000) ""
ServerTimestamp: null
SourceTimestamp: 01/16/21 18:36:51.0569210 GMT+00:00
OK




# Quellen

[1] FreeOpcUa

[2] OPC Foundation