Charakteristik and Services Tables

Mittwoch, 22. März 2023

12:32

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Actions** | **Admin**  **/User**  **Code** | **Receiving/Sending JSON Code to ESP** | **Android Method**  **BleutoothLeService.** | **Service Constant** | **Characteristic Constant** | **Description** | **Test** |
| Initial User | User | {"cmd": "init\_user", "name": "test-user"} | initUser(String name) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_USER | register as a new user and receive a user ID | ok |
| Make Recipe | User | {"cmd": "make\_recipe", "user": 8858, "recipe": "radler"} | makeRecipe(float user, String recipe) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_USER | mixes the recipe | ok |
| Reset | User | {"cmd": "reset", "user": 9650} | reset(float user) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_USER | Reset the machine so that it can make a new cocktail |  |
| Abort | User | {"cmd": "abort", "user": 483} | Abort(float user) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_USER | Cancels the current recipe |  |
| Define Pump | Admin | {"cmd": "define\_pump", "user": 0, "volume": 1000, "slot": 1} | adminDefinePump(float volume, int slot) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_ADMIN | add pump |  |
| Add liquid | User/Admin | {"cmd": "add\_liquid", "user": \*/0, "liquid": "water", "volume": 30} | addLiquid(float user, String liquid, float volume) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_User | Adds liquid to the cocktail | ok |
| Run Pump | Admin | {"cmd": "run\_pump", "user": 0, "slot": 1, "time": 1000} | adminRunPump(int slot, int time) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_User | Runs the pump for a certain time |  |
| Define Recipe | User/Admin | {"cmd": "define\_recipe", "user": \*/0, "name": "radler", "liquids": [["beer", 250], ["lemonade", 250]]} | defineRecipe(float user, String name, ArrayList<Pair<String, Float>> liquids) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_USER | defines a new recipe or changes an existing recipe | ok |
| Edit Recipe | User/Admin | {"cmd": "edit\_recipe", "user": \*/0, "name": "radler", "liquids": [["beer", 250], ["lemonade", 250]]} | editRecipe(float user,String name, ArrayList<Pair<String, Float>> liquids) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_USER | edit an existing recipe | ok |
| Calibrate Pump | Admin | {"cmd": "calibrate\_pump", "user": 0, "slot": 1, "time1": 10000, "time2": 20000, "volume1": 15.0, "volume2": 20.0} | adminCalibratePump(int slot,int time1, int time2, float volume1, float volume2) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_ADMIN | For calibration, two measured values must be available for which the pump has been running for a different time. This is then used to calculate the flow rate and the pump rate. The times are given in milliseconds and the liquids in millilitres. |  |
| restart | Admin | {"cmd": "restart", "user": 0}  {"cmd": "restart", "user": 0, "factory\_reset": true} | adminRestart(Boolean restFactory) | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_ADMIN | restarts the machine. If factory\_reset is set to true, all settings will also be deleted. | at this point factroy\_recet and without it behave same |
| clean | Admin | {"cmd": "clean", "user": 0} | adminClean() | BluetoothLeService.SERVICE\_READ\_WRITE | BluetoothLeService.CHARACTERISTIC\_MESSAGE\_ADMIN | Clean the machine | ok |
| Status Liquids | Admin | {"pumps":{"1":{"liquid":a","volume":100.0},"2":{"liquid":b","volume":100.0},"3":{"liquid":c","volume":100.0}},"liquids":{"c":100.0,"a":100.0,"b":100.0}} | getCharacteristicValue(BluetoothLeService.SERVICE\_STATUS\_STATE, BluetoothLeService.CHARACTERISTIC\_STATUS\_LIQUIDS); | BluetoothLeService.SERVICE\_STATUS\_STATE | BluetoothLeService.CHARACTERISTIC\_STATUS\_LIQUIDS | Read Pumps Volume | ok |
| Status State | Admin | "ready" | getCharacteristicValue(BluetoothLeService.SERVICE\_STATUS\_STATE,BluetoothLeService.CHARACTERISTIC\_STATUS\_STATE); | BluetoothLeService.SERVICE\_STATUS\_STATE | BluetoothLeService.CHARACTERISTIC\_STATUS\_STATE | Read current State | ok |
| Status Cocktail | Admin | [["a",10.0],["b",10.0]] | getCharacteristicValue(BluetoothLeService.SERVICE\_STATUS\_STATE, BluetoothLeService.CHARACTERISTIC\_STATUS\_COCKTAIL); | BluetoothLeService.SERVICE\_STATUS\_STATE | BluetoothLeService.CHARACTERISTIC\_STATUS\_COCKTAIL | Read Current Cocktail | ok |
| Status Recipes | Admin | {"abc":[["c",5.0],["b",5.0],["a",5.0]],"ab":[["b",5.0],["a",5.0]]} | getCharacteristicValue(BluetoothLeService.SERVICE\_STATUS\_STATE, BluetoothLeService.CHARACTERISTIC\_STATUS\_RECIPES); | BluetoothLeService.SERVICE\_STATUS\_STATE | BluetoothLeService.CHARACTERISTIC\_STATUS\_RECIPES | Read saved Recipes from Device | ok |

**Wirte Characteristic/Send Message to ESP:** to use the Methods it should be make an Instance of **BluetoothLeService** Class and for every Listener it should calls proper Method in **BluetoothLeService** ofcourse in try catch exception with **JSONException**. To receive Response from ESP call **getCharacterisitcValue(String Service, String Characterisitc)**

**Read Charactersitic/Receive Message from ESP:** To read Characteristic from ESP call **getCharacterisitcValue(String Service, String Characterisitc)**