RNHD Documentation

Project:	RNHD Server (v2)
Date:	2018-02-08
RNHD-Version:	1.13.0
Authors:	Danny Schumann

Table of contents

1 Overview

The RNHD Server is an application for ZIGPOS Real Time Localization Systems (RTLS) which runs on the ZIGPOS Gateway. It provides an REST and Websocket API to external applications as well as a web based user interface which uses this API.

2 Compatibility notes

2.1 Changes from 1.10.3 to 1.12.8

- Default baud rate to coordinator changed to 115200
- Added distance plausibility filter to positioning settings
- Added experimental simulation mode
- Added Bespoon integration module
- Added custom items resource to bind customer items to mobile devices
- Added system diagnosis feature

2.2 RNHDv2 compared to RNHDv1

The RNHD Server got a new internal architecture RNHDv2 (since v1.7.15) which will replace the old RNHDv1 architecture. The system can still run in RNHDv1 mode for compatibility reasons. It is planned to abandon RNHDv1 in RNHD Server v2.0.

Most APIs are compatible between RNHDv1 and RNHDv2, but not all. This document describes the latter if not explicitly marked as RNHDv1.

3 Settings

The settings define the behavior of the application. They are stored in the settings.ini file in the setup folder of the RNHD Server application and can be changed over the REST-API or the graphical Web-Client. A subset of the settings is hidden for non-expert users in the graphical user interface.

3.1 BESPOON

The BeSpoon module connects to a BeSpoon Localization server.

Parameter	Type	Values	Description
enabled	boolean	false	enable/disable module
serverIp	String	localhost (when on the same system)	IP or hostname of the BeSpoon localization server
tcpPort	int	8088	TCP port of the BeSpoon server
pollRateMS	int	1000	Poll time in Milliseconds

3.2 CONNECTION

The Connection category contains the settings for the serial connection to the network coordinator.

Parameter	Type	Values	Description
baudrate	int	115200	baudrate to coordinator

		(default)	
serial	string	COM-Port	Name of serial port (e.g. COM8 or /dev/ttyUSB0) or "auto" for automatic detection of the right serial port

3.3 DATA

The Data category contains settings for the connected database.

Parameter	**Type**	**Values**	**Description**
database	string	position_db (default)	name of the MySQL database
user	string		database user
password	string		password of database user
url	string	mysql://localhost (default)	url to database
version	int		The version of the database format. The version will be incremented automatically after a database upgrade and should not be changed manually.
storeDistance	boolean	false	store distances in database (reboot needed to take effect)
storePosition	boolean	true	store current positions in database (reboot needed to take effect) It is strongly recommended to keep this parameter true.
storePositionHistory	boolean	false	store historic positions in database (reboot needed to take effect)
storePmu	boolean	false	store PMU values in database (reboot needed to take effect)
storeRssi	boolean	false	store rssi values in database (reboot needed to take effect)
storeImu	boolean	false	store inertial data in database (reboot needed to take effect)

3.4 FUSION

The Fusion category is used for experimental data fusion functionality.

3.5 GEO

The Geo category is used for the conversion of relative (x, y, z) coordinates to global (long, lat, alt) coordinates. Therefore it is necessary to define the same two points in both coordinate systems.

Parameter	**Type**	**Values**	**Description**
enabled	boolean		switch conversion on or off
invertX	boolean	false (default)	invert X-Axis before conversion

invertY	boolean	true (default)	invert Y-Axis before conversion
x1,y1,x2,y2	double		relative coordinates of the points P1=(x1,y1) and P2=(x2,y2)
lat1,lng1,lat2,lng2	double		geo coordinates of the same points P1 and P2

3.6 OBSTACLE_AVOIDANCE

Obstacle avoidance maps define areas of obstacles which should be avoided by mobile devices. A path between the last and the current mobile position is calculated around the defined obstacles. Obstacles are defined as a grid of black pixels in the obstacle avoidance map. This map is defined as usual background map with a low resolution (recommended are 0.2m/pixel).

Parameter	Type	Values	Description
enabled	boolean	false (default)	enable or disable obstacle avoidance
completePath	boolean	false (default)	performance consuming debugging feature, should always be false
logPathEnabled	boolean	false (default)	log path into file
maxSpeedInMeterPerSecond	double	2.0	The maximum speed of a tag. This should be a realistic value to reduce the performance used for path finding.
obstacleMapId	int		The id of the background map which should be used as obstacle avoidance map.

3.7 POSITIONING

The positioning category is used to control the position calculation behavior.

Parameter	Type	Values	Description
distanceHistoryTimeMS	int	0 (default)	This parameter defines the time in Milliseconds of valid historic distances which will be added to the actual set of measured distances.
distancePlausibilityFilterEnabled	boolean	true (default)	Filters out distances which are obviously to long compared to the calculated position
distancePlausibilityThreshold	int	3	If the distance is X times to long the position will be recalculated without this distance.
fixedZ	double		Z value of mobiles in meter
fixedZModeEnabled	boolean	false (default)	If all mobiles are placed on a

			fixed height (fixedZ) and all anchors are placed on a different height, one can use this mode to increase positioning accuracy by shorten the measured distances.
maxDistanceLength	double	0 = disabled	filter to long distances
minDistanceAccuracy	double	0.0 … <1.0	filter inaccurate distances
minDistances	int	>0	minimum distances needed for position calculation
minPositionMoveThreshold	double	0	RNHDv1 only New positions must be more far away than the threshold to be updated.
permanentPositioningEnabled	boolean	false	RNHDv1 only deprecated
positionAlgorithm	int	0 = disabled 8 = GridBased (experimental) 10 = MultiDistSMDS	The Number of the used positioning algorithm.
useInfrastructure	boolean	true	Experimental, should always be true

3.7.1 POSITIONING/GridBasedPositioning

This positioning algorithm is experimental.

Parameter	**Type**	**Values**	**Description**
accuracyThresholdValue			
anchorNumberWeight			
gridAreaX,gridAreaY,gridAreaZ			
gridMaxX,gridMaxY,maxZ			
gridMinX,gridMinY,minZ			
numberOfConfidenceCandidates			
xAccuracy,yAccuracy,zAccuracy			

3.7.2 POSITIONING/MultiDistSMDS

Parameter	**Type**	**Values**	**Description**
dimension	int	2 = 2D (x,y) 3 = 3D (x,y,z) (default)	dimension (need to reboot to take effect)

3.8 PROTOCOL

The Protocol category contains the settings for the protocol which is used between the network coordinator and the gateway application.

Parameter	Type	Values	Description
protocolVersion	int		The current protocol version to communicate with the coordinator. This is related to the firmware version.
zpFrameEnabled	boolean	true	Needs to be true for coordinators with firmware version 119 or higher, otherwise false

3.9 RANGING

The Ranging category contains the settings for the ranging process between initiator and reflector devices. (Initiator = Anchor, Reflector = Mobile)

Parameter	Type	Values	Description
antennaInitiator	int	0 = ANT2 (default) 1 = ANT1/ANT3	Defines the antenna, which should be used for ranging.
antennaReflector	int	0 = ANT2 (default) 1 = ANT1/ANT3	Defines the antenna, which should be used for ranging.
distCorrectFactor	double	1.0 (default)	RNHDv1 only Distance multiplicator.
distCorrectOffsetP	double	0.0 (default)	RNHDv1 only Distance offset is added after multiplaction.
distCorrectFactorPmuRbl	double	1.0 (default)	Distance multiplicator for PMU ranging.
distCorrectOffsetPmuRbl	double	0.0 (default)	Distance offset for PMU ranging is added after multiplaction.
distCorrectFactorUwbTof	double	1.0 (default)	Distance multiplicator for UWB ranging.
distCorrectOffsetUwbTof	double	0.0 (default)	Distance offset for UWB ranging is added after multiplaction.
distanceAlgorithm	int	0 = disabled 1..8	RNHDv1 only Distance algorithms used to calculate distances out of PMU values.
distanceAlgorithmConsistent	int	0 = disabled 1..3	RNHDv1 only Distances algorithms used for equidistant frequency sets.
distanceNotificationEnabled	boolean	false (default)	RNHDv1 only Send WebSocket notifications on RANGING_DISTANCES topic.
diversityInitiatorEnabled	boolean		Use 2 antennas for ranging.
diversityReflectorEnabled	boolean		Use 2 antennas for ranging.
frequencySet	int	0 = equidistant (default) 1 = Golomb 0 2 = Golomb 1 3 = Golomb 2	RNHDv1 only Defines the frequency steps that are used for ranging: Equidistant see numOfFrequencies Golomb ruler 1

			{1, 3, 7, 25, 30, 41, 44, 56, 69, 76, 77, 86}
			Golomb ruler 2
			{1,12,15,16,25,46,62,85,104,121,126,133,153,159,161}
			Golomb ruler 3
			{1,16,27,36,39,55,60,82,111,113,143,147,153,160,161}

nativeEigenValueLibrary	boolean	false (default)	RNHDv1 only Use native library or java implementation for Eigen-Value calculation.
numOfFrequencies	int	10 … 120 (multiples of 10)	Numbers of frequencies used when equidistant frequencySet is selected.
rangeType	int	0 = extended 1 = PMU (default)	RNHDv1 only 0 means RSSI or UWB ranging dependent on the rangeTypeExtended parameter
rangeTypeExtended	int	0 = RSSI 1 = UWB	RNHDv1 only This setting depends on the device.
rangingDurationUwb	double	17.5 (default)	Ranging duration of UWB ranging in Milliseconds. This depends on the UWB settings of the device and should not be changed by non experts.
startFrequency	int	min: 2324 by regulation 2403 … 2483 max: 2525	Defines the start frequency in Mhz for the ranging. Start and end frequency must be in the defined interval.
stepSize	int	0 = 0.5 Mhz 1 = 1 Mhz	Defines the frequency stepping granularity in Mhz.
logDistance, logPmu, logRssi, logImu, logEhv	boolean	false	Log ranging/sensor results in file

3.10 REST

Parameter	**Type**	**Values**	**Description**
enabled	boolean		enable REST interface
url	string	http://0.0.0.0:8083/rest	Base URL of the REST interface. Set 0.0.0.0 as IP to allow access from all clients in the network.
websocketEnabled	boolean	true (default)	enable WebSocket interface

3.11 OAUTH

The authorization module is described in detail in a separated document [1].

Parameter	Type	Values	Description
enabled	boolean	true	enable / disable OAuth2
authDir	string	/home/zigpos/auth	folder to share data in case multiple applications using the same authorization server (don't touch this parameter)

3.12 RNHD

The RNHD category contains the core settings of the system.

Parameter	Type	Values	Description
anchorMaxNumber	int	0-12 6 = default	Maximum number of anchors to range with (in RNHDv1 maximum is 6)
anchorSelectAlgLevelFactor	double	1 = default	Multiplies the distance to anchors for anchor select algorithms. Set this bigger than 1 to select more anchors in the same height/level. (RNHDv1: Is only used when anchorSelectAlgorithm=3)
anchorSelectAlgorithm	int	0 = predefined 1 = closest *2 = RSSI based (deprecated)* 3 = closest using level factor	RNHDv1 only The algorithm defines which anchors should be used for ranging. The predefined anchors are defined in RNHD/STATIC_ANCHORS/ where each mobile has multiple parameters like this mobile_XX where XX is the short address of the mobile and the value is the short address of the anchor.
beaconEnabled	boolean	usually true	Network beacons may be switched off, if no IEEE802.15.4 communication is needed.
beaconMode	int	0 (default)	RNHDv1 only 0: standard beacon mode 1: experimental beacon mode, which will replace mode 0 in future.
contactNotConnectedIntervalMS	int	time in Milliseconds	RNHDv1 only The interval of requesting a network request of devices which are marked as not

			connected. 10000 for 10s is a usual value.
keepHistoryTimeS	int	time in Seconds	RNHDv1 only Time to keep historic data in database. This parameter is for avoid infinite growing of data in the database.
licenseFile	String		Path to license file
licensePubring	String		Path to pubring (public key file) for decoding license file
lnaGain	int	0 = max (default) 1 = medium 2 = minimum 3 = auto	RNHDv1 only low noise amplifier settings
maxActivePositioningRequestsPerMobile	int	1 (default)	Additional position request will be rejected if device is already busy with previous requests
maxPendingJobsPerMobile	int	1 (default)	Additional pending ranging jobs will be rejected if device is already busy with previous ranging requests
maxRangingJobsAfterLastMovement	int	0 = infinite 6 (default)	Device will stop ranging if not moving anymore
multipleTimesRangingPerBeaconEnabled	boolean	false (default)	RNHDv1 only Multiple Ranging descriptors to a single mobile within a beacon period.
positionValidTimeMS	int	time in Milliseconds (default is 0)	Time interval in which the mobile position is assumed to be valid and no ranging is needed.
positioningProcess	int	0 = PMU 1 = Super RSSI 2 = PMU and Super RSSI combined 3 = UWB 10 = Auto	For RNHDv2: Set triggerAutomaticMode=1 0 PMU: Mobiles and Anchors range within dedicated slots. This mode is for highest precision, but limited in number of mobiles within a beacon period. 1 Super RSSI (RNHDv1 only): All anchors are sending a RSSI signal and all mobiles are listening for them. This is for a fast but roughly positioning of many mobiles.

			2 Combined (RNHDv1 only): This mode switches the previous modes automatically 3 UWB: UWB ranging for UWB devices (experimental feature) 10 Auto select most accurate available (RNHDv2 only): UWB > PMU > RSSI
positioningAccuracyTarget	float	target accuracy radius in meters (set to 0.8)	only positions which have an accuracy better than positioningAccuracyTarget are marked as valid
reconnectPeriodEndDevice	int	time period in seconds	RNHDv1 only check if connected end device is still in the network
reconnectPeriodRouter	int	time period in seconds	RNHDv1 only check if connected router is still in the network
routerSelectAlgorithm	int	0 = static (experimental) 1 = dynamic (default)	Use static routers defined in RNHD/STATIC_ROUTERS/router or dynamic routing tree.
sleepTime	int	multiples of beacon intervals to sleep	All devices will sleep the set number of beacon intervals, if there is no task to do. During sleep, the network will not response until end of sleep.
slotGap	int	0 (default)	RNHDv1 only Adds additional empty slots after the ranging between mobiles and anchors in order to be more robust while receiving the ranging result packets.
staticBeaconEnabled	boolean	false (default)	RNHDv1 only Switch between predefined or dynamic Beacon structure. The mobiles for static beacons are defined in RNHD/ MOBILES_IN_STATIC_BEACON/ mobile
transmitPower	byte	0 = 3.5 dBm 1 = 1.8 dBm 2 = 0.5 dBm 3 = -2.5 dBm 4 = -6.5 dBm	Transmit power used for ranging. This parameter is part of the ranging descriptor.

		5 = -11.5 dBm 6 = -16.5 dBm 7 = auto	
triggerRangingAutomaticMode	int	Mode number	Modes 2, 3 and 4 are RNHDv2 only. 0: Disabled No automatic positioning 1: All active mobiles (Default) Request positions of all active mobiles with invalid positions (max ~1Hz) 2: Fast UWB 1 (experimental) Periodic UWB ranging with max 4 tags. Parameter: Period = 10, Mean = 5, Rep= 5 3: Fast UWB 2 (experimental) Periodic UWB ranging with max 2 tags. Parameter: Period = 4, Mean = 3, Rep= 15 4: EvaKitMode Max 5 tags, 4 anchors for 10Hz UWB ranging. Parameter: Period = 5, Mean = 1, Rep = 13
triggerRangingOnPositionRequestEnabled	boolean		Deprecated, RNHDv1 only Triggers ranging after request from REST API.
tryAgainAfterFailedPositioning	boolean		Range again after ranging error.
subSettings	String	path to ini	Ini-File Contains subsections

3.12.1 RNHD/ STATIC_ANCHORS

Parameter	Type	Values	Description
mobile_XX	short	short address of anchor	XX is short address of mobile, value is short address of anchor. Put multiple parameters in INI for multiple anchors.

3.12.2 RNHD/MOBILES_IN_STATIC_BEACON

Parameter	Type	Values	Description

mobile	short	short address of mobile	Put multiple parameters in INI for multiple mobiles per static beacon.

3.12.3 RNHD/STATIC_ROUTERS

Parameter	**Type**	**Values**	**Description**
router	short	short address of router	The order of multiple router parameters defines the order of the static routing tree.

3.13 SERVERSOCKET

The TCP server is deprecated and should not be used anymore. Set SERVERSOCKET/tcpServerEnabled = false.

3.14 SIMULATION

The simulation is used to test the server without any network. Enable this for experimental use only.

Parameter	**Type**	**Values**	**Description**
enabled	boolean	false (default)	enable / disable simulation mode
database	String	position_db_simulation (default)	database name for simulation test data

3.15 TRACKING

Parameter	**Type**	**Values**	**Description**
dimension	int	2 = 2D 3 = 3D (default)	should be always 3D
trackingFilter	int	0 = disabled 1 = KalmanCP 2…9	select tracking filter

3.16 WIFI

The Wi-Fi category is used for tags which connect via Wi-Fi to the gateway. This functionality is experimental.

4 REST and WebSocket API

The RNHD Server provides a REST and a WebSocket server interface. The REST interface is designed to response to data requests of a client. The WebSocket interface is designed as notification interface where the client needs to register to topics of interest first. Both interfaces provide their payload as JSON formatted string.

4.1 JSON formatted data model

This section describes the payload objects for the REST and Web Socket API. All numeric types are signed. **Long integer values are mostly converted to a String representation to avoid rounding errors in JavaScript during casting process.**

4.1.1 Device

A Device contains the properties of a Device in the network. Each device has a unique address.

Attributes	**Type**	**Values**	**Description**
timestamp	long		Creation Timestamp
networkId	short	-1 = invalid	Network Id (usually IEEE 802.15.4 PAN ID in decimal)
anchorId	byte	-1 = invalid	Unique ID of IEEE 802.15.4 anchor
shortAddr	short	-1 = broadcast -16 = invalid	2 Byte IEEE 802.15.4 network address
address	long as String	"-1" = invalid	Global unique 8 Byte IEEE 802.15.4 address
parentAddr	short	-16 = invalid	2 Byte IEEE 802.15.4 network address
networkRole	Enum as String	UNDEFINED COORDINATOR ROUTER END_DEVICE	IEEE 802.15.4 Network role. Is set as undefined for other network types.
networkType	Enum as String	UNDEFINED 📍? GENERIC 📍 IEEE_802_15_4 📍 WIFI 📍WIFI UWB 📍UWB BLUETOOTH 📍ᛒ LoRa 📍LoRa Bespoon 📍Bsp	The network type of the Device
appRole	Enum as String	UNDEFINED 📡 MOBILE 📍 / 🪪 ANCHOR 📶 COMMUNICATION 📡	Mobile devices have variable Positions, Anchor devices have fixed positions. Mobiles and Anchors are ranging with each other measure distances. Communication devices don't range.
deviceState	*byte*	*-1 = undefined* *0 = default* *2 = updating*	*might be removed in future*
activated	boolean	📍 (true) 📍 (false)	Deactivated devices are not used for positioning / ranging.
connected	boolean	📍 / 📍 (true) 📍 (false)	Non-connected devices are not reachable anymore and are not used for rangings.
customName	String	"new Mobile (123)"	User defined name

customType	String		User defined item type
hardwareName	String		Name of the Hardware (Set by the System)
softwareVersion	Int	-1 = undefined	Software version
battery	double	-1 = undefined	Battery Voltage
rssi	Int	-5 = undefined	RSSI to parent in dBm (Measured during association)
rangingCapabilites	Enum-Array	[]	Array of supported ranging technologies: GENERIC, RSSI_802_15_4, RSSI_BLUETOOTH, RSSI_WIFI, PMU_RTB, PMU_RBL, TOF_UWB, TDOA_UWB, UNKNOWN

Example:

```
{
        "activated": true,
        "addressAsHexString": "88880000000000B0",
        "address": "-8608630687718702928",
        "anchorId": -1,
        "appRole": "ANCHOR",
        "battery": -1,
        "connected": true,
        "customName": "Uwb Anchor 176",
        "customType": "",
        "deviceState": 0,
        "hardwareName": "",
        "networkId": -1,
        "networkRole": "UNDEFINED",
        "networkType": "UWB",
        "parentAddr": -16,
        "parentAddrAsHexString": "FFFFFFF0",
        "rangingCapabilities": [
            "RSSI_802_15_4",
            "PMU_RBL",
            "TOF_UWB"
          ],
        "rssi": -1,
        "shortAddr": -16,
        "shortAddrAsHexString": "FFFFFFF0",
        "softwareVersion": -1,
        "timestamp": 1416836165342
}
```

4.1.2 Position

A Position is a relative position with x, y, z coordinates related to a device via the device address.

Attributes	Type	Values	Description
timestamp	long		Creation Timestamp
id	long as String	"-1" = no id	Unique position id in history
address	long as String	"-1" = invalid	Global unique 8 Byte IEEE 802.15.4 address of the device
x,y,z	float		Metric Coordinates z is used for the height
mappedPosition		-1 = undefined	Map position to areas (area id)
hasMoved	boolean	true / false	True, if device has moved since last positioning. That means position is not valid anymore.
positionAccuracy	*Object*		*This is Only used for grid based positioning and might be removed or changed in future.*
accuracyRadius	float	accuracy radius in meter	The real position is in the radius by a likelihood of 95%.
state	int	0 = default 1 = measured 2 = tracking filtered 3 = map matched 4 = data fused 5 = set by hand 10 = predicted	The type/origin of the position

Example:

```
{
        "type": "position",
        "accuracy": 0,
        "accuracyRadius": 1.19,
        "address": "-8613303245920329199",
        "hasMoved": true,
        "id": "1395658567622",
        "mappedPosition": -1,
        "positionAccuracy":
        {
                "accuracy": 0,
                "euclid": 0,
                "id": "-1",
                "pdop": 0,
                "x_acc": 0,
                "y_acc": 0,
                "z_acc": 0
        },
        "state": 0,
        "timestamp": 1395658607783,
        "x": 7.03203,
        "y": 7.33926,
        "z": 2.16457
    }
```

4.1.3 GeoPosition

The GeoPosition is a position extended with alt, lat, lng properties.

Attributes	Type	Values	Description
<see Position>	…	…	…
alt, lat, lng	float		Altitude, latitude and Longitude (Altitude = z)

```
Example:

    {
        "type": "geoPosition",
        "accuracy": 0,
        "accuracyRadius": 1.19,
        "address": "-8613303245920329199",
        "hasMoved": true,
        "id": "1395658567622",
        "mappedPosition": -1,
        "positionAccuracy":
        {
            "accuracy": 0,
            "euclid": 0,
            "id": "-1",
            "pdop": 0,
            "x_acc": 0,
            "y_acc": 0,
            "z_acc": 0
        },
        "state": 0,
        "timestamp": 1395658607783,
        "x": 7.03203,
        "y": 7.33926,
        "z": 2.16457,
        "alt": 2.16457,
        "lat": 51.03648,
        "lng": 13.739965
    }
```

4.1.4 Distance

Attributes	Type	Values	Description
timestamp	long		Estimated measurement time
id	long as String		
addressA	long as String		Usually the mobile address
addressB	long as String		Usually the anchor address
foreignKey	long as String		Reference to the position id
type	RangingType-Object - AlgorithmId - NetworkType		Algorithm: - ID of ranging algorithm - 0 means no algorithm

	- RangingTechnology - TechnologyProvider		specified Network Type: UNDEFINED, IEEE_802_15_4, WIFI, UWB, BLUETOOTH Ranging Technology: GENERIC, RSSI_802_15_4, RSSI_BLUETOOTH, RSSI_WIFI, PMU_RTB, PMU_RBL, TOF_UWB, TDOA_UWB, UNKNOWN Technology Provider (Ranging): GENERIC, ZIGPOS, ATMEL, …
beaconNum	byte		Beacon interval number of the measurement
beaconSlot	byte		Slot number within beacon interval
antennaPair	Int	-1 = undefined 1 … 4	Used for diversity
rssi	float	127.0 = undefined	RSSI value between devices (if available)
accuracy	float	0 … 1	Accuracy/Thrust value (1 is best)
value	float	-1 = undefined	Distance in Meters

Example:

```
{
      "type":
      {
            "algorithmId":0,
            "networkType":"IEEE_802_15_4",
            "rangingTechnology":"PMU_RBL",
            "technologyProvider":"ATMEL",
      },
      "beaconNum":16,
      "beaconSlot":2,
      "antennaPair":-1,
      "timestamp":1448962275206,
      "accuracy":0.74,
      "value":2.77,
      "id":"0",
      "rssi":-80.0,
      "addressA":"9340351261199732",
      "addressB":"9340351261204038",
      "foreignKey":"1448957620732"
}
```

4.1.5 Layer

A layer may contain a background map and can be used for adding different levels of a building, for geofencing or obstacle avoidance.

Attributes	**Type**	**Values**	**Description**
id	Int	0 (default)	Unique identifier
name	String		Name of the layer
type	Int	NONE (default) FLOORPLAN OBSTACLE GEOFENCING UNKNOWN	Layer type
level	Int	0 (default)	Building level
maxZ, minZ, defaultZ	Double		DefaultZ is the height of the layer. MinZ and MaxZ span a range where z-coordinates belong to the layer.
mapId	int	-1 (no map)	Reference to a background map

Example:

```
     {
       "name" : "default_layer",
       "id" : 0,
       "type" : "NONE",
       "level" : 0,
       "maxZ" : 100.0,
       "minZ" : -100.0,
       "mapId" : 1,
       "defaultZ" : 0.0
     }
```

4.1.6 Background map

A background map contains an image with scaling information which is used as floor plan.

Attributes	**Type**	**Values**	**Description**
id	Int	0	Unique identifier
name	String		Name of the map
foreignKey	Int	0	Placeholder for future use
offsetX, offsetY	Double		Origin in meters, where the upper left corner of the image is.
Scale	Double		Scaling in meter per pixel
rotate	Double		Not supported

imgUrl	String	“background/img/<name-of-image>”	URL to background image relative to the REST-URL. The image needs to be uploaded via the REST-API.

Example:

```
{
  "id" : 1,
  "foreignKey" : 0,
  "offsetX" : 5.1,
  "offsetY" : 5.2,
  "scale" : 0.010835026747794478,
  "rotate" : 0.0,
  "imgUrl" : "background/img/office_raecknitz_snipped.png",
  "name" : "Office"
}
```

4.1.7 Area

An area defines a region on the map which can be used to trigger geo fencing events.

Attributes	Type	Values	Description
id	Int		Unique identifier
name	String		Name of a room or the area
layer	Int	0 (default)	Reference to the layer where the area belongs to
inOutHysteresis	float	0 (default)	Hysteresis distance to avoid frequently in out jumping on the boarder of an area. This value should depend on the expected position accuracy.
shapeType	Int	0 (deprecated) 1 (GeoJson)	Type of the shape. Default is type 1 for GeoJson.
shape	GeoJson object	GeoJson polygon	A GeoJson formatted polygon with outer ring only and x,y coordinates instead of long, lat. See [1]

Example:

```
{
  "id": 1,
  "inOutHysteresis": 0.8,
  "layer": 0,
  "name": "Room A",
 "shape": {
    "type" : "Polygon",
    "coordinates" : [ [
        [ 6.138445441456322, 11.004049953067232 ],
        [ 13.379301516836067, 11.06047220820006 ],
        [ 13.22884216981519, 16.627468047972542 ],
        [ 5.9315638393026155, 16.72150513986059 ],
        [ 6.138445441456322, 11.004049953067232 ]
    ] ]
  }


  "shapeType": 1
}
```

4.1.8 GeofencingEvent

Geofencing events are triggered when entering or leaving an area.

Attributes	Type	Values	Description
areaId	int		ID of the area
address	long as String		Address of the device
customName	String		Custom name of the device at event tiime
eventType	Enum as String	IN OUT	IN or OUT event
timestamp	long		Unix timestamp when the event was triggered
message	String		Optional human readable message

Example:

```
{
    "message" : "Device 'Danny' leaves area 'Students'",
    "timestamp" : 1470393041329,
    "eventType" : "OUT",
    "areaId" : 4,
    "address" : "9340351261199870",
    "customName" : "Danny",

}
```

4.1.9 Custom Item

Custom Items can be bound to a device. Each item can be linked to a parent item to build a hierarchical tree structure. Each custom item may have several item properties attached.

Attributes	Type	Values	Description
timestamp	long		Creation timestamp
id	String		Item id as string
parentItemId	String		Parent id of an item. Put “root” or “null” if item has no parent.
address	Long as String	“-1”	The device to which the item is bound to or “-1” if not bound.

Example:

```
	{
		"timestamp": 1517563997753,
		"parentItemId": "root",
		"id": "Test",
		"address": "-1"
	}
```

4.1.10 Item Property

Item properties can be used to add additional information to a custom item.

Attributes	Type	Values	Description
timestamp	Long		Creation timestamp
itemId	String		Id of the item to which the property belongs
name	String		Name of the property (identifier, unique per item)
type	String		Optional property type to group properties
value	String		Property value

Example:

```
	{
		"timestamp": 1517565187949,
		"type": "",
		"itemId": "Test",
		"name": "info",
		"value": "test"
	}
```

4.1.11 System Report

System reports are generated by the system diagnosis feature.

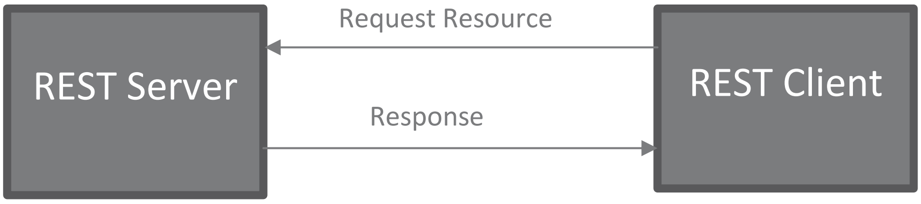
Attributes	Type	Values	Description
id	long		Unique id
timestamp	long		Creation timestamp
source	String		Indicator which component caused the error, for instance the address of a device.
type	String	VERBOSE, DEBUG, INFO, WARNING, ERROR, FATAL	VERBOSE: additional debug info (hidden to the user) DEBUG: debug information (hidden to the user) INFO: uncritical information to the user WARNING: system may not run as expected due to this ERROR: system is only partly running due to this error FATAL: system cannot run due to this error
category	String		Category to group similar reports
message	String		Human readable message

Example:

```
{
      "timestamp": 1517819385273,
      "source": "6507827280000059845",
      "type": "WARNING",
      "category": "network.devices.firmware",
      "message": "Device 'Anchor 10 (9C5)' firmware version is not up
      to date 1.1.0_1. Max version is 1.1.3_1",
      "id": 1517554922263
}
```

4.2 REST Interface description

Thre REST interface is designed to request data of different resources and to send commands to the system.



Name	Resource path	Description
BASE_URL	http://<ip>:<port>/rest/	Base URL for REST Service (Port is usually 8083)
Devices	BASE_URL/devices	Device data and device related sub-interfaces
Positions	BASE_URL/positions	Position data
Geopositions	BASE_URL/geopositions	Position data extended with global position (latitude, longitude)
Geofencing	BASE_URL/geofencing	Geofencing areas and functions
Distances	BASE_URL/distances	Distance data
PMUs	BASE_URL/pmus	RNHDv1 only PMU data
RSSIs	BASE_URL/rssis	RNHDv1 only RSSI data
IMUs	BASE_URL/imus	Inertial data
Firmware	BASE_URL/firmware	Firmware update
Command	BASE_URL/command	RNHDv1 compatibility command, control, configuration of the devices
Background	BASE_URL/background	Background map and layers
Management	BASE_URL/management	System data and settings data
Ranging	BASE_URL/ranging	For receiving ranging data (distances, rssis)
Values	BASE_URL/values	RNHDv1 compatibilty optional values (configurations, sensor data, etc.) for devices
Custom Items	BASE_URL/custom	Manage custom items and item properties

4.2.1 Devices

The devices interface provides access to device specific data, configurations and control functions.

4.2.1.1 Get and set devices

The following REST commands describe how to get and set device data

Get all devices.

URL	BASE_URL/devices
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of devices

Get single device.

URL	BASE_URL/devices/<address>
Method	GET
URL Params	address: MAC address in decimal
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: device
Error Response	Device not found Code: 404 NOT FOUND

Get single device by short address.

URL	BASE_URL/devices/query?shortAddr=<short address>
Method	GET
URL Params	short address: 2 bytes short address in decimal (signed)
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of devices

Add or update single device. Devices are update when a device with the given address already exists, otherwise they are added.

URL	BASE_URL/devices or BASE_URL/devices/<address>
Method	PUT
URL Params	address (optional): MAC address in decimal
Data	Content-Type: application/json

	device
Success Response	Code: 200 OK

Add or update multiple devices. Devices are update when a device with the given address already exists, otherwise they are added.

URL	BASE_URL/devices
Method	POST
URL Params	none
Data	Content-Type: application/json device
Success Response	Code: 200 OK

Delete device.

URL	BASE_URL/devices/<address>
Method	DELETE
URL Params	address: MAC address in decimal
Data	none
Success Response	Code: 200 OK

Delete all devices without positions (experimental).

URL	BASE_URL/devices/clean
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: “done”

4.2.1.2 Get and set single device attributes

The attributes of a device can be set without updating the complete device object. The following attributes are supported:

- customName

Get attribute.

URL	BASE_URL/devices/<address>/attributes/<attribute>
Method	GET
URL Params	address: MAC address in decimal

	attribute: attribute Name
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: value of attribute
Error Response	Device not found or attribute not found Code: 404 NOT FOUND

Set attribute.

URL	BASE_URL/devices/<address>/attributes/<attribute>
Method	PUT
URL Params	address: MAC address in decimal attribute: attribute Name
Data	Content-Type: text/plain attribute value
Success Response	Code: 200 OK Content-Type: text/plain data: “done”
Error Response	Device not found or attribute not found Code: 404 NOT FOUND

4.2.1.3 Control device

Reset device.

URL	BASE_URL/devices/<address>/reset
Method	GET
URL Params	address: MAC address in decimal
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: “done”

Set device LED.

URL	BASE_URL/devices/<address>/led/<LED-number>
Method	PUT
URL Params	address: MAC address in decimal LED-number: 1 = red LED, 2 = green LED
Data	Content-Type:text/plan mode as integer modes: 0 = off, 1 = on, 2 = automatic (default)
Success Response	Code: 200 OK Content-Type: text/plain data: “done”

4.2.1.4 Display update

The e-paper display of ZIGPOS badges can be configured with the display update interface which is located at

BASE_URL/devices/<address>/display.

This interface is documented separately in [2].

4.2.1.5 EEPROM

The EEPROM interface is for expert users only. It allows to manipulate low level data in the devices EEPROM which is documented in [3].

The EEPROM data is represented as hexadecimal string of raw bytes. The byte order is little-endian.

Read EEPROM data.

URL	BASE_URL/devices/<address>/eeprom/<EEPROM-address> ?length=<number of bytes>
Method	GET
URL Params	address: MAC address in decimal EEPROM-address: EEPROM start address length (optional): Number of bytes (default is 8 bytes, if not set)
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: hexadecimal string
Error Response	Code: 400 BAD REQUEST Code: 500 INTERNAL SERVER ERROR

Write EEPROM data.

URL	BASE_URL/devices/<address>/eeprom/<EEPROM-address>
Method	PUT
URL Params	address: MAC address in decimal EEPROM-address: EEPROM start address
Data	Content-Type: text/plain data: hexadecimal string of data to write
Success Response	Code: 200 OK Content-Type: text/plain data: hexadecimal string of data which has been written
Error Response	Code: 400 BAD REQUEST Code: 500 INTERNAL SERVER ERROR

4.2.2 Positions

The Positions resource provides access to current and historic position data.

4.2.2.1 Get and set positions

Get all current positions.

URL	BASE_URL/positions
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of positions

Get the current position of a single device.

URL	BASE_URL/positions/<address>
Method	GET
URL Params	address: MAC address of device in decimal
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of positions

Add or update single position. Set position state to 5 (set by hand) when updating anchor positions.

URL	BASE_URL/positions or BASE_URL/positions/<address>
Method	PUT
URL Params	address (optional): MAC address of device in decimal
Data	Content-Type: application/json position
Success Response	Code: 200 OK

Add or update multiple positions. Set position state to 5 (set by hand) when updating anchor positions.

URL	BASE_URL/positions
Method	POST
URL Params	none
Data	Content-Type: application/json array of positions
Success Response	Code: 200 OK

Delete single position.

URL	BASE_URL/positions/<address>
Method	DELETE
URL Params	address (optional): MAC address of device in decimal
Data	none
Success Response	Code: 200 OK

Delete all positions.

URL	BASE_URL/positions
Method	DELETE
URL Params	none
Data	none
Success Response	Code: 200 OK

Delete all positions without devices.

URL	BASE_URL/positions/clean
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK

4.2.2.2 Get historic positions

Get position history of a single device.

URL	BASE_URL/positions/<address>/history ?limit=<max number of positions> &since=<time span> &mintimestamp=<unix time stamp> &maxtimestamp=<unix time stamp> &state=<position state> &smoothing
Method	GET
URL Params	address: MAC address of device in decimal limit (optional): limit number of returned positions since (optional): don’t return positions which are older than current time minus given time span in Milliseconds mintimestamp (optional): minimum timestamp maxtimestamp (optional): maximum timestamp state (optional): filter by state attribute of position object • 0: undefined • 1: measured position • 2: tracking filtered position

	- 5: set by hand smoothing (optional, experimental): generate smoothed path from history by calculating smoothed interpolated positions
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: positions

Get position history of single device in csv format

URL	BASE_URL/positions/<address>/history/csv ?limit=<max number of positions> &since=<time span> &mintimestamp=<unix time stamp> &maxtimestamp=<unix time stamp> &state=<position state> &smoothing
Method	GET
URL Params	see GET BASE_URL/positions/<address>/history
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of positions

Delete historic positions.

URL	BASE_URL/positions/<address>/history/csv ?limit=<max number of positions> &since=<time span> &mintimestamp=<unix time stamp> &maxtimestamp=<unix time stamp> &state=<position state> &smoothing
Method	DELETE
URL Params	see GET BASE_URL/positions/<address>/history
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of positions

4.2.3 Geoposition

The interface is the same like the position interface, but using geo formatted positions instead.

4.2.4 Geofencing

4.2.4.1 Get and set areas

Get all areas.

URL	BASE_URL/geofencing/areas
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of areas

Get single area.

URL	BASE_URL/geofencing/areas/<id>
Method	GET
URL Params	id: area id
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: area
Error Response	Code: 404 NOT FOUND

Add or update area.

URL	BASE_URL/geofencing/areas or BASE_URL/geofencing/areas/<id>
Method	GET
URL Params	id (optional): area id
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: area

Delete area.

URL	BASE_URL/geofencing/areas/<id>
Method	DELETE
URL Params	id: area id
Data	none
Success Response	Code: 200 OK

4.2.4.2 Occupied areas

Get all devices inside of an area.

URL	BASE_URL/geofencing/areas/<id>/devices
Method	GET
URL Params	id: area id
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of devices
Error Response	Code: 404 NOT FOUND (if area not found)

Get all occupied areas.

URL	BASE_URL/geofencing/occupied
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of areas

Get all areas occupied by a specific device.

URL	BASE_URL/geofencing/occupied/<address>
Method	GET
URL Params	address: MAC address as decimal
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of areas

4.2.4.3 Events

Get Geofencing event history

URL	BASE_URL/geofencing/events ?limit=<limit number of events> &address=<filter by address> &name=<filter by custom name> &mintimestamp=<unix time stamp> &maxtimestamp=<unix time stamp>
Method	GET
URL Params	limit (optional): maximum number of events address (optional): filter events by address (MAC address as decimal) name (optional): filter events by custom name mintimstamp, maxtimestamp (optional): filter events by time interval
Data	none

Success Response	Code: 200 OK Content-Type: application/json data: array of geofencing events

Delete Geofencing event history

URL	BASE_URL/geofencing/events ?address=<filter by address> &name=<filter by custom name> &mintimestamp=<unix time stamp> &maxtimestamp=<unix time stamp>
Method	DELETE
URL Params	address (optional): filter events by address (MAC address as decimal) name (optional): filter events by custom name mintimstamp, maxtimestamp (optional): filter events by time interval
Data	none
Success Response	Code: 200 OK Content-Type: text/plain “done”

4.2.5 Distances

Get all stored distances.

URL	BASE_URL/distances ?limit=<limit number of distances> &addressA=<filter by addressA> &addressB=<filter by addressB>
Method	GET
URL Params	limit (optional): maximum number of distances addressA (optional): filter distances by addressA (MAC address as decimal) addressB (optional): filter distances by addressB (MAC address as decimal)
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of distances

Get distances used for positioning

URL	BASE_URL/distances/<position id>
Method	GET
URL Params	position id: id of the position where the distances belong to
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of distances

Delete all distances

URL	BASE_URL/distances
Method	DELETE
URL Params	none
Data	none
Success Response	Code: 200 OK

4.2.6 Values

This interface is for development and internal use only. Therefore it is not documented.

4.2.7 Firmware

The firmware interface is used for firmware updates of devices using SREC file format.

4.2.7.1 Upload firmware files

Get available firmware files.

URL	BASE_URL/firmware
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: new line (\n) separated names of firmware files

Upload firmware file.

URL	BASE_URL/firmware/upload/<filename>
Method	GET
URL Params	filename: name of SREC file
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: “true” (if success)

Delete firmware file.

URL	BASE_URL/firmware/upload/<filename>
Method	DELETE
URL Params	filename: name of SREC file
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: “true” (if success)

4.2.7.2 Control update process

Start update process of multiple devices.

URL	BASE_URL/firmware/update/<filename>
Method	POST
URL Params	filename: name of SREC file
Data	Content-Type: application/json array of device objects which should be updated
Success Response	Code: 200 OK Content-Type: text/plain data: “true” (if success)

Get update status of all updating devices.

URL	BASE_URL/firmware/status
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: new line (\n) separated status messages for each device

Get update status of single device.

URL	BASE_URL/firmware/status/<address>
Method	GET
URL Params	address: MAC address of device as decimal
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: status messages for device

Cancel update process.

URL	BASE_URL/firmware/cancel
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: “true” (if success)

4.2.8 Command

This API is deprecated and should not be used anymore in RNHDv2.

4.2.9 Background

This API provides access to the background layer. Each layer can define a Map which contains a background image with a certain scaling factor.

4.2.9.1 Layers

Get all layers

URL	BASE_URL/layers
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of layers

Get single layer.

URL	BASE_URL/layers/<id>
Method	GET
URL Params	id: layer id
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: layer

Add or update single layer.

URL	BASE_URL/layers or BASE_URL/layers/<id>
Method	PUT
URL Params	id (optional): layer id
Data	Content-Type: application/json layer
Success Response	Code: 200 OK

Add or update multiple layers.

URL	BASE_URL/layers
Method	POST
URL Params	none
Data	Content-Type: application/json array of layers
Success Response	Code: 200 OK

Delete single layer.

URL	BASE_URL/layers/<id>
Method	DELETE
URL Params	id : layer id
Data	none
Success Response	Code: 200 OK

Delete all layers.

URL	BASE_URL/layers
Method	DELETE
URL Params	none
Data	none
Success Response	Code: 200 OK

4.2.9.2 Maps

Get all maps

URL	BASE_URL/maps
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of maps

Get single map.

URL	BASE_URL/maps/<id>
Method	GET
URL Params	id: layer id
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: layer

Add or update single map.

URL	BASE_URL/maps or BASE_URL/maps/<id>
Method	PUT
URL Params	id (optional): map id

Data	Content-Type: application/json map
Success Response	Code: 200 OK

Add or update multiple maps.

URL	BASE_URL/maps
Method	POST
URL Params	none
Data	Content-Type: application/json array of maps
Success Response	Code: 200 OK

Delete single map.

URL	BASE_URL/maps/<id>
Method	DELETE
URL Params	id : map id
Data	none
Success Response	Code: 200 OK

Delete all maps.

URL	BASE_URL/maps
Method	DELETE
URL Params	none
Data	none
Success Response	Code: 200 OK

4.2.9.3 Images

Get available image names.

URL	BASE_URL/img
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: new line (\n) separated image names

Get background image.

URL	BASE_URL/img/<name>
Method	GET
URL Params	name: image name
Data	none
Success Response	Code: 200 OK Content-Type: image/png

Upload background image.

URL	BASE_URL/img/<name>
Method	POST
URL Params	name: image name
Data	Content-Type: multipart-form-data image file
Success Response	Code: 200 OK Content-Type: text/plain data: status message

Delete background image.

URL	BASE_URL/img/<name>
Method	DELETE
URL Params	name: image name
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: status message

4.2.10 Management

4.2.10.1 System

Get current server version.

URL	BASE_URL/management/system/version
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: version string

Get license information.

URL	BASE_URL/management/system/license
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: text/plain data: license information string

Start system diagnosis and return system reports.

URL	BASE_URL/management/system/diagnosis/check
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of system reports

4.2.10.2 Settings

Interface for getting or updating the server settings.

Get settings node.

URL	BASE_URL/management/settings/<path> ?expert=<true or false>
Method	GET
URL Params	path: settings path to the settings node in the settings hierarchy expert: show also hidden settings for expert user
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: settings node

Set parameter in settings

URL	BASE_URL/management/settings/<path>
Method	PUT
URL Params	path: settings path to the settings node in the settings hierarchy
Data	Content-Type: text/plain value as string
Success Response	Code: 200 OK Content-Type: application/json data: “success”

4.2.11 Custom Items

4.2.11.1 Items

Get all items.

URL	BASE_URL/custom/items
Method	GET
URL Params	none
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: array of all items

Get single item by ID.

URL	BASE_URL/custom/items/<id>
Method	GET
URL Params	Id: item id
Data	none
Success Response	Code: 200 OK Content-Type: application/json data: item

Add item.

URL	BASE_URL/custom/items
Method	PUT
URL Params	none
Data	Content-Type: application/json item
Success Response	Code: 200 OK

Delete item.

URL	BASE_URL/custom/items/<id>
Method	DELETE
URL Params	Id: item id
Data	none
Success Response	Code: 200 OK

4.2.11.2 Items History

Get Items History.

URL	BASE_URL/custom/itemshistory ?itemid=<item id> &address=<device address> &mintimestamp=<min timestamp> &maxtimestamp=<maxtimestamp> &limit=<limit results>
Method	GET
URL Params	itemid (optional): item id address (optional): device address mintimestamp (optional): min timestamp, default is 0 maxtimestamp (optional): max timestamp, default is current time limit (optional): limit the number of results
Data	none
Success Response	Code: 200 OK

Delete Items History.

URL	BASE_URL/custom/itemshistory ?itemid=<item id> &address=<device address> &mintimestamp=<min timestamp> &maxtimestamp=<maxtimestamp>
Method	DELETE
URL Params	itemid (optional): item id address (optional): device address mintimestamp (optional): min timestamp, default is 0 maxtimestamp (optional): max timestamp, default is current time
Data	none
Success Response	Code: 200 OK

4.2.11.3 Item Properties

Get all properties of an item.

URL	BASE_URL/custom/items/<id>/properties
Method	GET
URL Params	id: item id
Data	none
Success Response	Code: 200 OK Content-Type: application/json Array of all properties of the item

Add property.

URL	BASE_URL/custom/items/<id>/properties

Method	PUT
URL Params	id: item id
Data	Content-Type: application/json property
Success Response	Code: 200 OK

Add multiple properties.

URL	BASE_URL/custom/items/<id>/properties
Method	POST
URL Params	id: item id
Data	Content-Type: application/json array of properties
Success Response	Code: 200 OK

Delete all properties of an item.

URL	BASE_URL/custom/items/<id>/properties
Method	DELETE
URL Params	id: item id
Data	none
Success Response	Code: 200 OK

Get specific property.

URL	BASE_URL/custom/items/<id>/properties/<name>
Method	GET
URL Params	id: item id name: property name
Data	none
Success Response	Code: 200 OK Content-Type: application/json property

Delete specific property.

URL	BASE_URL/custom/items/<id>/properties/<name>
Method	DELETE
URL Params	id: item id name: property name
Data	none
Success	Code: 200 OK

Response	

Get property value.

URL	BASE_URL/custom/items/<id>/properties/<name>/value
Method	GET
URL Params	id: item id name: property name
Data	none
Success Response	Code: 200 OK Content-Type: text/plain Value as string

Set property value.

URL	BASE_URL/custom/items/<id>/properties/<name>/value
Method	PUT
URL Params	id: item id name: property name
Data	Value as string
Success Response	Code: 200 OK Content-Type: text/plain “done”

4.2.11.4 Item Properties History

Get Item Properties History.

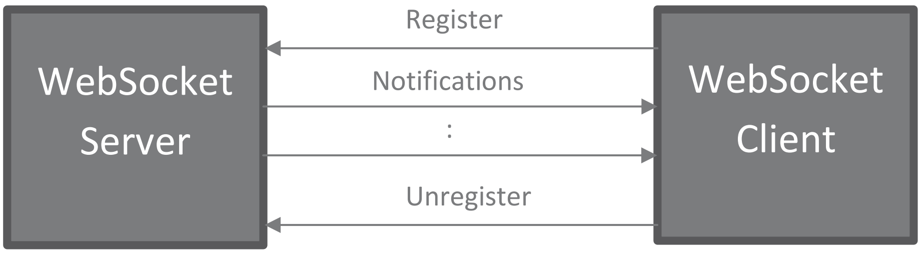
URL	BASE_URL/custom/propertieshistory ?itemid=<item id> &name=<name> &mintimestamp=<min timestamp> &maxtimestamp=<maxtimestamp> &limit=<limit results>
Method	GET
URL Params	itemid (optional): item id name (optional): property name mintimestamp (optional): min timestamp, default is 0 maxtimestamp (optional): max timestamp, default is current time limit (optional): limit the number of results
Data	none
Success Response	Code: 200 OK

Delete Item Properties History.

URL	BASE_URL/custom/propertieshistory ?itemid=<item id> &name=<name> &mintimestamp=<min timestamp> &maxtimestamp=<maxtimestamp> &limit=<limit results>
Method	DELETE
URL Params	itemid (optional): item id name (optional): property name mintimestamp (optional): min timestamp, default is 0 maxtimestamp (optional): max timestamp, default is current time limit (optional): limit the number of results
Data	none
Success Response	Code: 200 OK

4.3 WebSocket Interface

A WebSocket Client can register to several topics of the WebSocket Server. The WebSocket server notifies all registered clients about published data on the topic.



Name	Resource path	Description
Websocket URL	ws://<ip>:<port>/rest/socket	Base URL for REST Service (Port is usually 8083)

To register to a topic, the client sends a JSON formatted string like the following to the server

```
{
        "topic":"REGISTER",
        "payload":["POSITION","DISTANCES"]
}
```

To Register or unregister, the topic-field must be “REGISTER” or “UNREGISTER”. The payload-field contains an array of all topic names the client wants to register to.

The server will then notify the client like this (Example of position notification):

```
{
      "topic":"POSITION",
      "payload":
      {
            "x":7.131521,
            "y":6.126457,
            "z":2.4512286,
            "mappedPosition":-1,
            "hasMoved":false,

      "positionAccuracy":{"x_acc":0.0,"y_acc":0.0,"z_acc":0.0,"
euclid":0.0,"pdop":0.0,"accuracy":0.0,"id":"-1"},
            "timestamp":1448961911638,
            "state":1,
            "accuracy":0.0,
            "id":"1448957620729",
            "address":"9340351261199732"
      }
}
```

The topic-field contains the name of the topic. The payload contains the data published on the topic like described in section 4.1.

Be aware that after connection loss all topics are automatically unregistered and need to be registered after each reconnect to the WebSocket.

4.3.1 Data Topics

The data topics provide the main information of the system. Usually they are reported when new data is available and stored in the database.

Topic	Payload	Description
DEVICES	Device array	Devices updated or added
DEVICES_REMOVED	Device addresses array	Devices were removed
POSITIONS	Position array	Positions updated or added
POSITIONS_REMOVED	Device addresses array	Postions of devices were removed
GEOPOSITIONS	GeoPosition array	Geopositions updated or added
DISTANCES	Distance array	new Distances which belong to a Position
RSSIS	RSSI array	Signal strength data
IMUS	IMU array	Inertial data
LAYERS	Layer array	Layers updated or added
LAYERS_REMOVED	Layer array	Layers removed
AREAS	Area array	areas updated or added
AREAS_REMOVED	Area array	areas removed
HAS_MOVED	Device	the device has moved
FALL_DETECT	Device	fall detect event received
SHAKE_DETECT	Device	device shake detected
PUSH_SOS_DETECT	Device	SOS button pressed on the given device (experimental)
UPDATE_STATUS	UpdateStatus	Update Status changed

4.3.2 Geofencing Topics

Topic	**Payload**	**Description**
GEOFENCING_EVENT	GeofencingEvent	Event to notify if mobile device enters or leaves an area

4.3.3 Ranging Topics

Topic	**Payload**	**Description**
RANGING_DISTANCES	Distance array	new distances from network and calculated distances before positioning
RANGING_RSSIS	*RSSI array*	*not implemented*
RANGING_PMUS	*PMU array*	*not implemented*

4.3.4 Positioning Topics

Not implemented yet.

References

[1] "RFC7946: The GeoJSON Format," IETF, 2016.

[2] ZIGPOS, Badge Display Update, 2017.

[3] ZIGPOS, *RNHD EEPROM Documentation,* 2014.