Wellenvogel AvNav

AvNav Documentation

Generated: 2024/6/16

Navigation in the Browser

AvNav is a free of charge, open source navigation software for recreational boaters. Like similar applications in this area, AvNav displays nautical charts and marks the boat's position using GPS devices attached to the system. Markers, waypoints, routes, tracks, AIS, external sensors and many more features are supported. The server concept is one of it's key features: AvNav installs on any Raspberry Pi, Windows or Android device acting as 'navigation server'. It collects relevant data from various sources and administers nautical charts. Server access and graphical represention can be realised on any device running a modern web browser, irrespective of it's operating system. AvNav is consequently optimised to be used on touchscreens. Different layouts can be defined to adapt the user interface e.g. to meet special user requirements or to fit specific screen dimensions.

Charts

Any raster charts (RNC), not encrypted by their author, can be used. Vector charts of type oeSENC can also be used. They are, however, not free of charge and need to be acquired at the o-charts shop.

Variants

There is more than AvNav's server variant: as an alternative a stand alone version is offered for Rapberri Pi (AvNav Touch) as well as an Android app. On top of that: 'AvNav for OpenPlotter' integrates seamlessly into the OpenPlotter 2 image with all necessary connections to SignalK server configured out of the box.
All software is available for download under an open source license.

Quickstart

15.12.2020

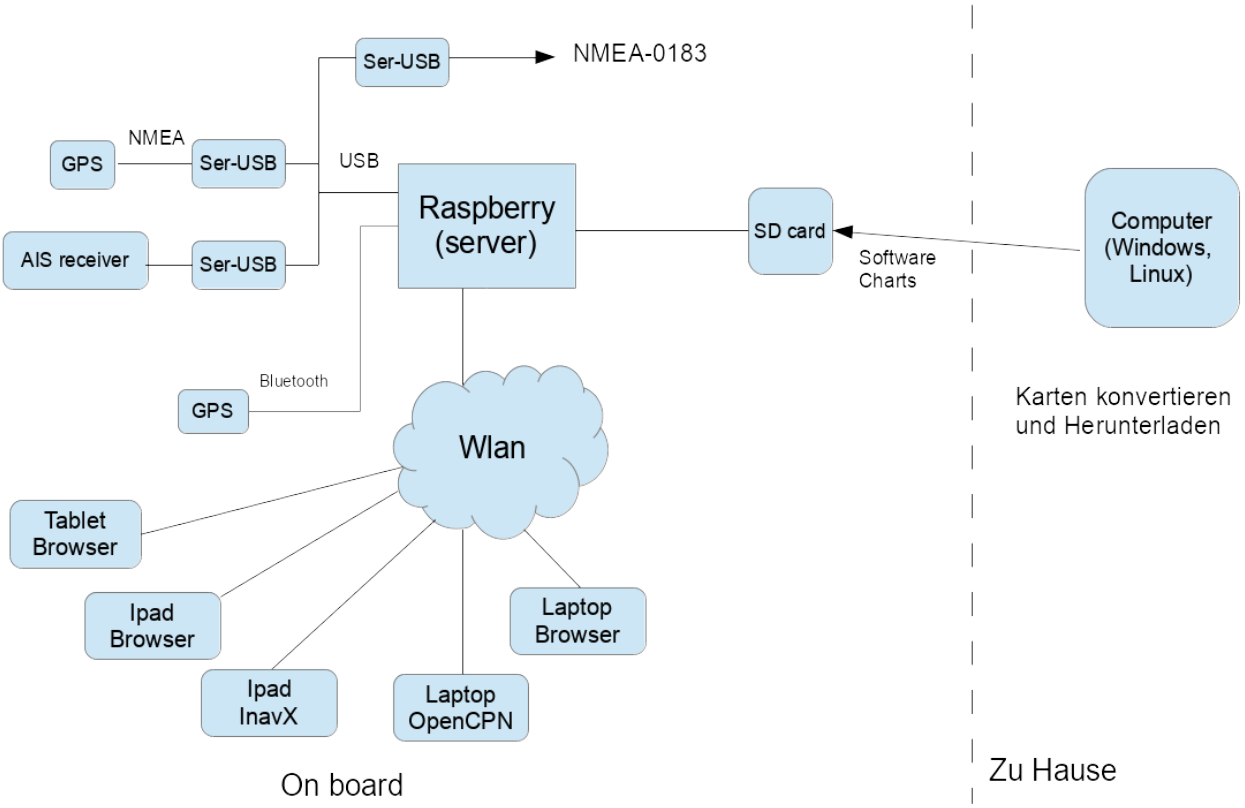
- YouTube Videos (german)
- AvNav at Boot 2020 Düsseldorf - open-boat-projects
- Privacy: english,german
- Thread in 'Segeln Forum' - german
- Source Code at GitHub
- User-Documentation
- Android App
- Downloading and Converting Charts
- Installation
- Release-Notes
- Demo
- Slide set (PDF) -german

Important Note:

In no case I promise or can be held responsible for correct function of AvNav - especially using it for navigation is at your own risk. Before using it I recommend to carefully test the precision of the display and the used charts.

Overview

avnav-raspi-2020



The whole solution consists of different parts:

- A Raspberry Pi with the server software reading the connected devices (e.g. NMEA via serial-USB converter – like PL 2303), Bluetooth GPS, multiplexing and storing the data and providing them via WLAN
- A variant of this software for the desktop (Windows/OSx/Linux) to convert and prepare the charts

The raspberry pi creates an access point and various devices can access it's data:

- Variant 1: the client (lie laptop or iPad) runs some (other) navigation software (tested: InavX, OpenCPN), those access the NMEA data via TCP or UDP.
- Variant 2: the client device just runs a browser. All the navigation is handled by the AvNav WebApp provided by the raspi. There is no need for any other installed software on the device, only a current HTML5 browser (tested: Chrome Windows, OSX, Safari, Android starting 4.4 –

Chrome/Stock/Boat Browser, IOS, Blackberry stockBrowser, WebBrowser mini).

The server part is written in python and can be configured via an xml file. The normal set up is ready to go - i.e. there is no need for any configuration to get started. Beside the software itself there are images you can download to get started - see installation.

The java script app provides all basic navigation functions on raster charts (gemf, mbtiles) or oeSENC vector charts including AIS display, waypoint navigation, routes, tracks,... The charts to be used by the web app must be installed on the raspberry.

oeSENC charts can be bought in the o-charts shop.

Chart formats that cannot be handled directly in the app must be converted to gemf using AvNav at the desktop (or directly on the pi within the App). The converter can process the following formats:

- All chart sources that can be read by GDAL software (especially BSB)
- Charts downloaded with Mobile Atlas Creator

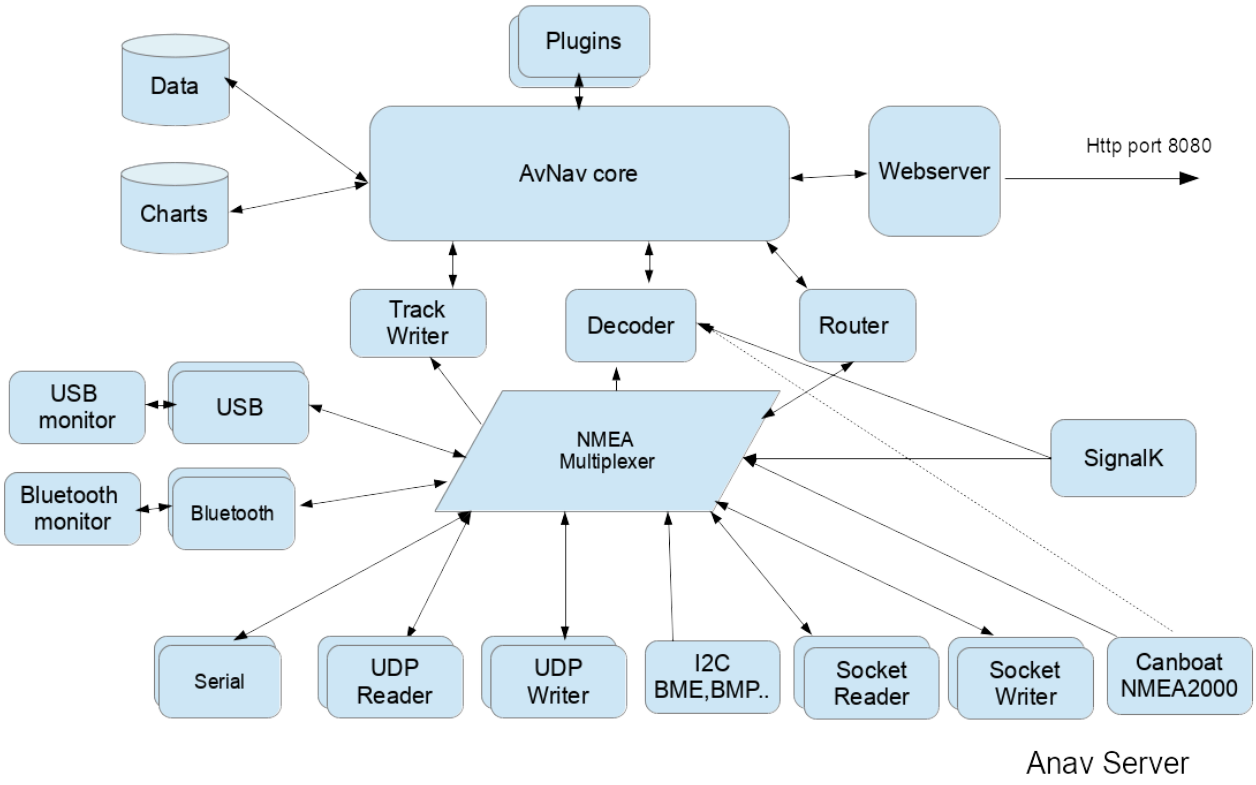
Beside the described set up there is also an Android-App providing similar functions.The server part is implemented natively in Java, the display part is identical.

The following chapters describe the parts a bit more in detail.

The Server Software (avnav_server.py)

The pi holds a normal Debian image (app. 2GB). A couple of additional packages get installed together with AvNav (see list below).

Structure of the server software:



The server tries to detect all devices connected via USB-serial and via bluetooth-serial. You have to take care that your USB-serial device is correctly handled by the pi - like e.g. this one. The server scans connected devices (via dbus) and discovers the device nodes. It then tries to do some auto-bauding (i.e. determine the baudrate for the device) between 4800 and 34000 baud and tries to recognise NMEA data. If no data is detected it starts over again by closing and reopening the devices. This way it can cope with disconnecting and reconnecting devices. I have connected a AIS VHF receiving GPS data from the pi and sending AIS data to it. If bluetooth serial devices are detected AvNav will also try to read from them.

You can also configure devices to output the data and this way have a NMEA multiplexer.

All NMEA data internally are stored in a queue and made available for output. By default a TCP socket is open where you can connect to (default port: 34567).

Additionally you can configure more outputs (TCP,UDP,serial).

Internally the data (including AIS) are forwarded to a decoder storing them in the server ("NMEA decoded data"). Those data are available as JSON for the webapp's HTTP - API requests. Additionally the server writes track data, computes routing information, handles anchor watch and alarms.

To control an auto pilot the server computes RMB data and sends them to the internal queue so that it can be received by any connected device.

If there is a valid time information in the GPS data it sets the system time of the raspberry.
For automated start up there is a (systemd) service so you can control start and stop via systemctl.

Most of the server functions you can configure via an xml file (avnav_server.xml). In the installed template there are examples for many use cases. Additionally (if a bluetooth device has been detected) AvNav tries to connect to bluetooth serial devices and receive NMEA data from them.

Normally there is no need for additional configuration.

You can find the software on github. For installation instructions see the installation description.

Software on the raspberry

On the raspberry the following directories are used;

Directory	Content
/usr/lib/avnav	software
/home/pi/avnav/data/	base data directory
.../data/charts	charts - see converting charts.

.../dava/log	logfiles
.../data/tracks	trackfiles (gpx), one file per day, nmea logs
.../data/routes	routes - xxx.gpx and the current segment (leg) currentLeg.json
.../data/import	input directory for the converter: charts stored here will be converted to 'gemf'
.../data/user/viewer	location of user files (user.js, user.css,...)
.../data/user/images	location for user image files
.../data/layout	location if user layout files

Apart from the systemd scripts (and some command scripts) the software is running as user pi (on the raspberry). You can also start the software from the command line using the command "avnav". On a desktop system you may run this as any user.

The Web App

The app is a single page app built with ReactJs.
It communicates with the server using Ajax (javascript). The app entry point is http://avnav.avnav.de/viewer/avnav_viewer.html. It is optimized for mobile devices - especially tablets starting at 7". But you can use it also on desktop systems or larger displays. Meaningful usage starts at around 900x540 pixel.

URL Parameters

The WebApp support a couple of URL Parameters that you can add to control some functions.

parameter	description
defaultLayout	The name of an existing layout that will be used as the initial layout.
defaultSettings	The name of an existing settings file (regex supported) that will be used as default settings when AvNav is started for the first time in the Browser with this URL. Example: defaultSettings=.*localFirefox
fullscreen	You can provide a parameter in the form of "server:<command>". Command must be an existing command that has been configured at the AVNCommandHandler. This will be executed when the Fullscreen Button is pressed (instead of using the browser fullscreen feature) Example: fullscreen=server:fullCommand
dimm	You can provide a parameter in the form of "server:<command>". Command must be an existing command that has been configured at the AVNCommandHandler. This will be executed when the dimm Button is pressed (instead of using a dimm function that is available in the Browser - like AvNav on Android) Example: dimm=server:dimmCommand
noCloseDialog	Prevent the Dialog that will warn you if you are going to leave the AvNav page. Example: noCloseDialog=true

splitMode	Start AvNav directly in split mode. Example: splitMode=yes
preventAlarms	Do not show any Alarms. Example: preventAlarms=true

For a user documentation of the web app refer to this description.

Demo

The demo shows you the main program functions.
You cannot store any data on the server - so e.g. the "connected" mode will not work.

The charts are retrieved (online) from http://kartor.eniro.se/, from a "meshup" from BSH https://www.geoseaportal.de and from http://www.openseamap.org/ (consider their copyrights).
BSH server are very slow - the display will be delayed.
The gps data is from a tour in 2013 from Klintholm to Vitte. They also include some AIS data.
The timeline for the data is somehow compressed - approximately double speed. If the demo reaches the end it will jump back to the beginning - this will lead to some strange track line. Just close your browser window and start over.

As a starting point just click on the Boat button to lock the chart and start zooming in.
Continue playing...
Upload and download will not work in all browsers.

newest version (preview)

AvNav Quickstart

Important Hint: ***In no case I promise or can be held responsible for correct function of AvNav - especially using it for navigation is at your own risk. Before using it I recommend to carefully test the precision of the display and the used charts.***

A detailed description of the concepts can be found in the chapter introduction.

General

AvNav has been designed to be usable on touch devices (also with relatively small screens). The idea was to allow ease of use also under restricted "on board" conditions.

Of course you can operate AvNav by mouse and keyboard as well.

Installation and Set Up

AvNav is available in 2 variants:

1. Client-Server
   The server will be installed onto a Linux or Windows system (like a Raspberry Pi). As "client" you will use an arbitrary browser to interact with the system (e.g. on a tablet or smart phone)
2. Android App
   The complete functionality is bundled within the app. Additional client devices can connect with a browser.

Client Server

AvNav is available as package for various Linux distributions (Debian packages, Rpm) and as a Windows installer.
The Debian packages are hosted in a repository but they can be downloaded from the release page as well.
Additionally we maintain images for the Raspberry Pi. A detailed documentation is available in the chapter installation.

After installing and starting up you can use your browser to connect to AvNav and open the WebApp.
If you are using our images the raspberry will establish a Wifi network (name and password can be adapted). For details on how to connect to your server refer to the image documentation.
If you are connected to the server differently you can adress http://avnav.local:8080 (does not work on android) - or you can use the IP of the server.
For IOS and Android devices I would recommend using a Bonjour Browser. This tool can find the AvNav servers in the local network and will start a browser without the need of entering an address.

- IOS: 
- Android: 

Android App

The App is available in the Play Store

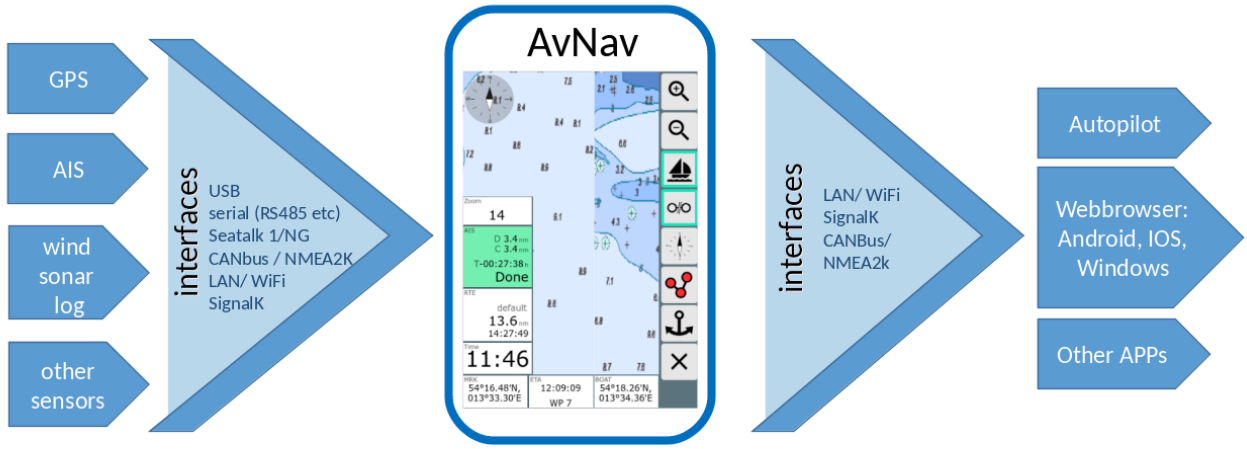
Configuration

After installing AvNav (or when using a ready to go image) you can normally start without any additional configuration. AvNav will scan USB interfaces for serial adapters and will try to determine the appropriate baud rate. Depending on the kind of installation a udp receiver is active on port 34667.

To adapt the AvNav server configuration you can use the server/status page. Additionally you can adapt the look and feel with settings, layout adaptation and with user defined code/CSS and plugins.

With the Android installation, all settings are directly integrated in the App.

NMEA Data



Client Server

AvNav processes NMEA0183 Data from connected USB devices, serial ports, bluetooth devices, TCP (client and server) or UDP. A NMEA multiplexer is

integrated allowing for flexible configuration of the flow of received and sent data.

A couple of NMEA sentences are decoded within AvNav (position data, AIS,...) and used for it's display and routing functions.

In combination with Canboat and SignalK you can also handle NMEA2000 data. Additionally you can use all your own vessel's SignalK data to be displayed within AvNav.

The following NMEA sentences are decoded:

- !AIVDM
- $xxGGA
- $xxGSV
- $xxGLL
- $xxVTG
- $xxRMC (mag. variation since 20240520)
- $xxMWV (since 20220225 added waterSpeed)
- $xxDPT
- $xxDBT
- $xxXDR (since 20210114)
- $xxHDG (since 210106xx , mag. variation since 20240520)
- $xxHDT (since 20210619)
- $xxHDM (since 20210619)
- $xxVHW partial (since 20210619)
- $xxVWR (since 20220225)
- $xxMTW (since 20220225)
- $xxZDA (since 20220421)
- $xxVDR (since 20240520)

Depending on the configuration AvNav is able to generate following NMEA sentences:

- $GPRMB
- $GPAPB
- $AVXDR

- $AVMDA
- $AVMTA

In combination with Canboat and SignalK NMEA2000 data can be received and processed too. All own vessel data available in SignalK can be displayed in AvNav.
Since 20220421 AvNav can directly receive it's navigation data from SignalK.

Android

On Android you can utilize the internal GPS. Additionally you can have a TCP or bluetooth connection for receiving GPS or AIS data. If your Android device has USB-OTG available you can connect an USB-serial adapter as well.

The following NMEA sentences will be decoded:

- !AIVDM
- $xxGGA
- $xxGSV
- $xxGLL
- $xxGSA
- $xxRMC
- $xxMWV (since 20220225 added waterSpeed)
- $xxDBT
- $xxXDR (since 20210114)
- $xxHDG (since 210106xx, mag. variation since 20240520)
- $xxHDT (since 20210619)
- $xxHDM (since 20210619)
- $xxVHW partial (since 20210619)
- $xxVWR (since 20220225)
- $xxMTW (since 20220225)
- $xxVDR (since 20240520)

AvNav is able to create and send out $GPRMC and $GPRMB (since 20220225 also $GPAPB) if configured.

Charts

AvNav generally handles raster charts. You can download them from various internet sources (e.g. OpenSeaMap or NOAA) - or you can use software like MobileAtlasCreator or SASPlanet to download/create charts.

AvNav can directly process charts in gemf and mbtiles formats. Additionally you can convert other types of raster charts like BSB (kap). This can be done directly on the server (raspberry) - or better beforehand on a desktop system.

Additionally AvNav can process oeSENCcharts from o-charts - on Android with the avocharts app (ochartsng). The company offers charts for Open Source Software at reasonable prices.

With the ochartsng plugin AvNav can also display free S57 charts (after conversion).

Charts need to be uploaded to AvNav before being usable - at the Files/Download page, section charts  . o-charts must be uploaded via the o-charts plugin / ochartsng plugin

Using the mapproxy-plugin AvNav can include various online chart sources. Areas of those charts can be downloaded with the plugin for offline usage.

More details in the chapter charts.

Displays

At the Navigation page the boat position, it's course, the route to the next waypoint, the current route, AIS targets and their courses, navigation circels and defined overlays will be shown on the chart.

At the Navigation page, inside the Route Editor and at up to 5 Dashboard pages you can display the values of all available navigational data. These

include values from SignalK.

You can use simple numeric displays, analog gauges or graphical displays.

The displays can be adapted to your preferences. There is a Layout Editor and you can create / modify displays with some lines of Java Script Code and CSS.

You can define different sets of display "layouts" and store them on the AvNav server. For each display device you can select the set ("layout") that you would like to use.

There is also an adaptation to different screen sizes available in the layout ("small").

Routes

You can easily create and edit routes within AvNav. You will use the chart view at the Route Editor. Normally you just drag the center of the chart (cross) to the next point you would like to add. With a button click you add this to the route. You can easily move, edit or delete points within the route.

Waypoints or other routes that are shown as overlays can be added to the route.

You can invert the route or empty it. Within AvNav routes are stored as gpx files. You can export or import them at the Files/Download page. From within the Route Editor you can immediately start the route navigation. When following a route an alarm will be raised as you reach the next waypoint (if you are within an "approach" distance) and AvNav will automatically switch to the next waypoint.

Since version 20220819 AvNav can handle two different routing modes.

great circle

A route is computed as the shortest distance between two points on the earth. The drawback is a course that will change permanently throughout the route. In the map display this route will be a curve.
In older versions AvNav always computed routes as great circle - by accidently did draw them as straight lines.
For shorter distances (< 100nm) this basically does not matter at all.

rhumb line

A route will be computed for a constant course. The display on the map will be a straight line.
To switch the modes use the Router at the  server/status page. For the  measure tool you can set the mode separately in the settings of the App (at Navigation/Measure Display RhumbLine). This way you can easily compare the two pathes.

Next Waypoint Switching

To trigger the automatic switch to the next waypoint two conditions must be meat simultaneously:

1. The boot must be located inside the approach radius of the waypoint (can be defined in the app settings before you start a route). This will become visible by a red display of the Route Widget and the triggering of the waypoint alarm.
2. Depending on the selected mode (Router on the  server/status page: nextWpMode) - new since 20220819
   "late" (the default and the one used in older versions): The distance to the current waypoint must not decrease any more and the distance to the next waypoint must start to decrease.
   "90": The waypoint is "abeam" - more exactly the boat has crossed a line +/- 90° to the original waypoint course.
   "early": The switch will be triggered by a delay (that you can configure) after the waypoint alarm has been initiated (without any further conditions).

It is important to know the the automatic switch will only occur if really both conditions are meat. If you would like to trigger the switch by hand you can bring up the waypoint buttons by a click on the lower left widgets and use the  Button.

Tracks

AvNav is records the current track displaying it on the chart. To minimize the number of trackpoints they will just be recorded after major changes of position or after a given time interval (see Configuration ANVTrackWriter). The tracks will be output as gpx file at regular intervals. They can be exported and imported at the Files/Download page (on that page you can also see their metadata like length and time). Every day a separate gpx file is created. Available tracks can be displayed as overlay on the chart.

You can as well convert a track into a route. There is some program logic implemented to reduce the number of points.

AIS

On your chart there will be a display of the AIS targets within a defined range (default: 20nm) with their positions and courses. CPA (closest point of approach) of AIS targets will be computed and a warning issued if a defined minimum distance is not kept.

For details refer to the navigation page.

Alarm

AvNav can trigger alarms for:

- Anchor Watch
  On the Dashboard pages you can activate the anchor watch. Depending on the layout, the displays will change. Whenever your boat leaves the defined circle or if the gps signal is lost, an alarm will be issued.

- Approaching the next waypoint
- Man over Board

  On all pages there is a separate  Man-over-board button. By clicking this button the current position will set as routing target, all other routings will be stopped. Additionally the alarm is set.

All alarms will be handled on the AvNav server. So you may switch off all display devices, still any alarm handling (like e.g. anchor watch) will continue to work.

You can assign a sound (both at the server and at display devices) to the alarm. At the server you can trigger definable commands whenever an alarm is raised (see example). The configuration is handled within the configfile - AVNAlarmHandler.

Nightmode

At the Main page you can active the night mode ◐. All pages will be adapted accordingly.

Remote Control

AvNav is able to control display functions on one device from another one - or from the server. For details refer to the description .

Adaptations

You can adapt AvNav to your personal needs by various means. As all pages are handled within a browser you can use CSS to customize your display.

The server configuration is defined by the file avnav_server.xml. Typically there is no need for editing this file directly. Instead you should use the server/status page to change the settings.

Beside adapting the displays with the Layout editor you can easily setup your own custom displays with some Java Script.

You can also include the display of other web pages (external ones or pages you created within AvNav). This will be handled as "User Apps".

The symbols that are used for various displays can be customized with a json file - like the keybord short cuts.

With Python, Java Script and CSS you can write plugins.

AvNav Installation

Releases

A description of the releases and download links can be found in the releases document.

If you are in a hurry - those are the current image, the daily builds and the release downloads.

To provide a "ready to go" solution there are some images for the raspberry pi you can use.
Since version 20220421 the images do support both the "headless" working mode - i.e. no keyboard or monitor attached to the pi - as well as the "touch" mode - a monitor (with our without touch) is locally attached, optionally keyboard and mouse.

AvNav is optimized for running on touch devices but you can use it on desktop systems with monitor, keyboard and mouse, too.

So the usage of the images depend on your use case. For "headless" usage the Pi is only used as a server, the display logic will run in the browser e.g. on some mobile device. In this case a Raspberry Pi 3B(+) with 1GB should be sufficient. If you would like to run a local display you should go for a Pi4 with at least 2GB of memory.

If you are looking for a complete desktop system with a lot of other applications OpenPlotter could be your choice. For running OpenPlotter I suggest a Pi4 with 4GB of RAM. 2GB will do but without a lot of room for later extension requirements.

In the past there were special AvNav Touch images. But they are not maintained any more.

AvNav Images (named Headless in the past)

Those images are maintained by BlackSea (many thanks...). A description is available on his page.
The images can be downloaded from free-x and installed on a SD card as described at http://www.raspberrypi.org/downloads (raw images).
The images come with

- avnav
- avnav-update-plugin
- avnav-ocharts-plugin
- avnav-mapproxy-plugin
- avnav-history-plugin
- SignalK
- Canboat
- Support for MCS
- optionally and X-Server with openbox and firefox in kiosk mode
- support for various HATs

They are preconfigured to route NMEA0183 traffic from all interfaces to AvNav and from there to SignalK. AvNav will additionally fetch all data from SignalK and is able to display it.
NMEA2000 data will flow via Canboat to SignalK and in parallel to AvNav.
For details refer to CanBoatAndSignalK.

Image Preparation

new as of 20210322, extended with 20220421

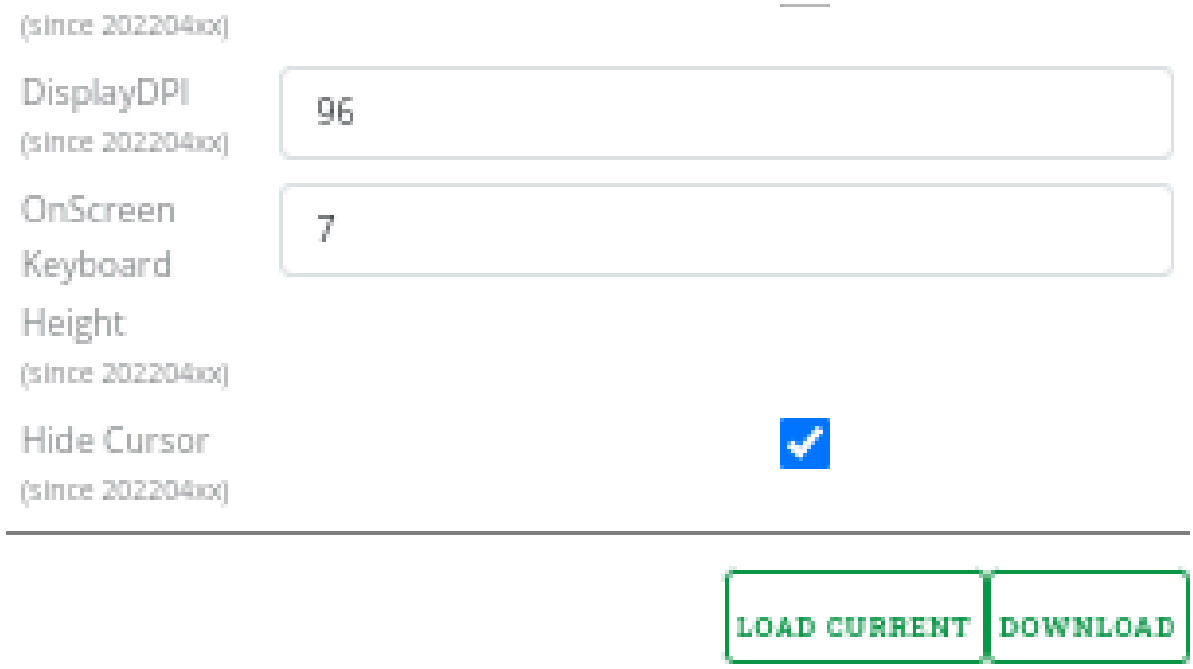
Before inserting the SD card into your Raspberry (after you wrote the image to it) you should adapt a couple of settings (especially passwords).
The images hold a configuration file "avnav.conf" in the first partition of the SD card (boot partition). This file can be adapted using a text editor.
This is the place to configure whether you would like to run a local display ("touch variant").

More easily you can use a simple GUI that is provided here.

avnav.conf Generator

You can create a file avnav.conf intended to be stored in the first partition of your SD card.
All data will remain in your browser and not being transfered to the server.
You should change at least the passwords.

Config Sequence	1
Wifi SSID	avnav
Wifi Password	avnav-secret
User pi Password	raspberry
Hostname (since 20210429)	avnav
Base Board (since 202303xx)	NONE
HAT (since 202303xx)	NONE
Module RTL8188EU (since 202304xx)	☐
Module RTL8192EU (since 202304xx)	☐
TimeZone (since 20210429)	Europe/Berlin
Wifi Country (since 20210429)	Germany
Internal Wifi as Client (since 20210429)	☐
KeyboardLayout (since 20210429)	German
KeyboardType (since 20210429)	Generic 105-key PC (intl.)
TouchSupport	☐



The meaning of the fields:

name	default	description
Wifi SSID	avnav	The name of the Wifi network created by the raspberry. The images are prepared to create multiple networks - if you e.g. decide to add an additional Wifi stick, the software will append a 1 digit number to this name
Wifi Password	avnav-secret	The password for the Wifi network. In any case you should change this. Just

		keep in mind that everybody that can connect to this Wifi network is able to influence your navigation.
User pi password	raspberry	This is the password for the user pi whenever you connect e.g. with ssh or by attaching a monitor and keyboard. Also change this one!
Base Board	None	You can select one of the supported base boards. • **MCS:** If this option is activated the software for the Marine Control Server from GeDad will be activated. • **OBPPLOTTERV3:** This will set up the software for the Open Boat Projects Plotter (V3) .

HAT	None	You can select one of the supported HATs. AvNav will create the necessary entries for overlays in /boot/config.txt and will set up the can network interfaces. • WAVESHAREB: waveshare RS485 CAN HAT (B) • WAVESHAREA8: waveshare RS485 CAN HAT (8Mhz) • WAVESHAREA12: waveshare RS485 CAN HAT (12 Mhz) • WAVESHARE2CH: waveshare 2CH CAN HAT • PICANM: PICAN-M • MCARTHUR: MacArthur HAT
Module RTL8188EU	off	If activated the kernel driver module for wifi adapters with the

		chip set RTL8188EU will be activated via DKMS . If you would upgrade the kernel of your system (from the command line) the kernel module will be compiled to fit the newly installed kernel.
Module RTL8192EU	off	If activated the kernel driver module for wifi adapters with the chip set RTL8192EU will be activated via DKMS . If you would upgrade the kernel of your system (from the command line) the kernel module will be compiled to fit the newly installed kernel.
TimeZone	Europe/Berlin	The time zone that the image will use
WifiCountry	Germany	Must be set for legal reasons before start

		using the Wifi adapter
InternalWifi as Client	off	If switched on the internal Wifi adapter will not be configured as access point - instead you can use it to connect to other Wifi networks. Remark: This requires some other way to connect to the Pi - as you need to configure the Wifi first.
KeyboardLayout	German	The layout of a keyboard (including the X keyboard)
KeyboardType	Generic 105-key PC(intl.)	Type of a connected keyboard
TouchSupport (since 20220421)	off	If enabled a local X server with firefox in kiosk mode is started - see touch.
Display Dpi (since 20220421)	96	Only if touch support is enabled. Resolution of the connected screen

		(pixel/inch). A small calculator will come up to allow you to provide the screen dimensions in mm and in pixels. This will scale some of the display elements on the connected monitor.
OnScreen Keyboard Height (sind 20220421)	7	Only if touch support is enabled. The height of a keyboard row of the on screen keyboard. Should be a compromis between the ability to touch the correct key and the occupied space.
HideCursor (sind 20220421)	yes	Only if touch support is enabled. Will hide the cursor on the connected screen. Switch this off if you are going to use a mouse.

After filling the values you need to click the download button and store the file avnav.conf into the first partition of your SD card (overwriting the existing example there!). To be able to do this, your SD card must be visible on your

computer (on Windows you will only see the first partition). Potentially it could be necessary to eject, remove and reinsert the SD card after you have written the image before the card becomes visible.

It would also be a a good idea to additionally backup the downloaded file at a safe place to be able to use it again whenever you need to create a new SD card.

You can now insert the SD card into your raspberry and power it on. The first boot will take some time as it is rebooting once to enlarge the file system on your card. Depending on the selected configuration it will potentially reboot twice.

Now you will be able to connect to your pi.

Connecting to your Raspberry Pi

If you configured a local monitor connecting is fairly easy of course. But for the headless mode (and as a fallback also recommended with local screen) the following options are available to connect to the Pi.

1. with an ethernet cable
   You could either connect to a router or point to point e.g. to a laptop
2. via the internal wifi
   By default the Pi will open up an access point with the SSID and password you have selected in the configuration. It will append a number to the SSID as it could create more access points if you would connect additional wifi adapters.
3. via USB from an Android device
   Modern Android devices mostly have an "USB-Tethering" function that will share the device wifi or mobile connection over USB. Unfortunately this typically is only available if your device has mobile support at all.
4. Via a different wifi network
   This requires a different connection initially as you need to configure the access. It also requires an additonal wifi adapter connected to the correct

USB socket. Optionally you could switch the internal Adapter to client mode in the configuration ("Internal Wifi as Client").

Connection via Ethernet Cable

If you connect your pi to a router (e.g. in your home network) it will be assigned an IP address from there. You can connect to the Pi using this address.
Sometimes it is not that easy to find out this address - therefore the Pi will announce it's services via mDNS (Bonjour, Avahi).
Using mDNS it is simple to connect:

```
http://xxxx.local:8080
```

xxx ist the hostname you have choosen in the configuration.
You can alos use this kind of address for an SSH connection(e.g. putty). The access passowrd for the user pi has been set in the configuration.

If you create a point to point connection with your network calble (without a router) the Pi will create an IP address by its own after some time. This could take 1...2 minutes. This IP will belong to the Automatic Private IP Addressing range 169.254.x.x. Most desktop systems will also support this (at linux you would potentially need to enable this explicitely).
If your laptop also created such an Ip on it's ethernet interface you should now be able to connect using the mDNS names xxx.local.

If the access via xxx.lcal does not work you need to find out the IP of the Pi (e.g. using the admin UI of your router).

Connection via the internal Wifi

You can use the Wifi network that the Pi has created. The SSID and the passwort had been set in avnav.conf.

When being connected to the wifi network you still need to find out the address of the Pi. Like described for the ethernet access you should normally

be able to use the mDNS address

```
http://xxx.local
```

Ifthis does not work you can try with the fixed addresses 192.168.30.10, 192.168.40.10, 192.168.50.10, 192.168.60.10:

```
http://192.168.30.10
```

This should load the AvNav mainpage. It should also be possible to use xxxx.local for an SSH connection.(like putty on Windows).

There is one restriction: Unfortunately xxx.local will not work on Android devices. I recommend installing a tool that is able to resolve mDNS- BonjourBrowser . For IOS there is a similar tool - although xxx.local should work there in the browser. You will find your Pi with the AvNav image with the name "avnav-server". Normally there will be a second entry "avnav" - this is SignalK.
If you can see AvNav in the Bonjour Browser on Android but loading the page fails this could be caused by some strange network handling on Android. Sometimes it will help to disconnect mobile data.

Since version 1.12 the Android BonjourBrowser will also support SSH services. The AvNav image (ab 20220421) also will announce it's SSH access via mDNS. If you installed a SSH client on Android (like JuiceSSH) you can also connect to the pi for an SSH session. This is maybe not that comfortable for normal operation - but could help for some repiar actions to issue commands.

The password has been set using the config file above. When connecting via ssh login with the user name pi. If your selected password does not work, try again with the default ('raspberry'). Potentially you did not store the avnav.conf correctly.
You can enter a root shell with sudo -i.

Connection from Android via USB

You need an Android device that supports USB Tethering (normally at connection settings). After you have connected your device via USB cable to the Pi switch on the USB Tethering (normally it will switch off again when you disconnect).
Beside the option to connect the Pi to the internet this way you can alos access AvNav or connect via SSH. As you need to know the IP I still recommend to install the Bonjour Browser app see at Wifi. For SSH access again JuiceSSH.

This is also an option to access the Pi if the Wifi does not work for some reason. In the BonjourBrowser you will find 2 http: addresses (port 8080 for AvNav and port 3000 for SignalK). Additionally (since 20220421) the SSH access.

Connection via a different Wifi Network

If you created (like described below) a wifi connection to another network you can enable the access to your Pi on this network ("external access" when you configure it).

You should only allow this on a trusted network (e.g. the net of your own LTE router). **You should never enable this for a public network as there is no protection and everybody from within this network can access your Pi.**

The connection process is the same like described at Wifi connection.

Connecting the Pi with the Internet

For some function (e.g. software updates) the Pi needs an internet connection. This is not necessary for the basic navigation functions, of course.

Before the image version 20220421 you needed to consider that the Pi was not automatically setting it's local time without a connected GPS. This could

lead to trouble for many internet connections.
Since version 20220421 the Pi will synchronize it's local time with the net (NTP) if there is no GPS.

The following options are available to connect to the internet:

1. Ethernet cable to a router
2. Connection via a different wifi network
3. Connecting via an Android device being connected on USB

If the Pi has an internet connection it will share this to all devices connected to it's own wifi network.

Connection via Ethernet Cable

You will connect the Pi with an ether cable to a router.
There is nothing to be configured at the Pi.
On some Pi3 there is a chance that it will not detect the network if the cable was only connected after the boot. In this case just reboot the Pi with the cable connected.

Connection via a different Wifi Network

You need an additional wifi adapter (USB adapter). Please check the compatibility before buying one - e.g. here.

The stick must be plugged at the correct port (see picture). On the Pi4 use the blue USB Port at the board side).
The internal name of the network will be wlan-av1.

As an alternative you could set "InternalWifi as Client" in the image configuration. In this case the internal wifi adapter will not be used for an access point but will be avilable to connect to other networks. But you need a different way to connect to your Pi (e.g. cable or USB).

The wifi connection can be configured in the AvNav App.
For each network that you connect to you can select if you would allow access to your pi from this network("external access"). If you deselect it, the pi can access the internet on this network but nobody can access the Pi on it.
Please not the Hints for access.

Connection via and Android Device on USB

As described at access you can also utilize an Android device with USB tethering. Internally this will create a network interface usb0.
The Pi can use this interface to access the internet.

This could be an easy option if you only need temporary internet access and don't have a (working) wifi stick.
If you have been connected to the internet a different way before (cable, wifi) it could happen that the pi will not immediately use the USB connection for the internet access. In this case just reboot (don't forget to re-enable USB tethering on the android device...).

Technical Details

The pi will set up one (or more) wifi networks using the internal adapter or additional wifi sticks. These networks will have addresses like 192.168.20.0/24, 192.168.30.0/24, 192.168.40.0/24, 192.168.50.0/24. The pi itself has the address 192.168.x.10.

There is a DHCP server and a DNS server (dnsmasqd).

If the pi is connected to a wired network it tries to obtain an ip via DHCP. It has (NAT) forwarding from it's wifi to the wired network. This way you can easily connect to the internet if you are in the wifi network of the pi.
If you connect additional wifi sticks, more access points will be established.

For most actions ssh access to the pi is not required. To update you should use the preinstalled Update Plugin . The server configuration can now be accomplished within the AvNav app - at the Server/Status page.

Image with Display (AvNav touch)

In previous versions there has been a separate AvNav Touch image.
This is not maintained any more.
Since version 20220421 the support for a connected monitor (focus on touch) has been integrated into the "normal" AvNav images.

As described at Preparation you can switch on the support for a local monitor there.

If you enable this the system will start up a service "avnav-startx". This service creates a local X server, a user session for the user pi with openbox as window manager and firefox in kiosk mode.

Onboard will be used for an on screen keyboard.

On the AvNav mainpage (and on some other pages) there will be a "raspberry" button that will bring up a second virtual screen that contains a file manager, a terminal and some other applications.
Intentionally the system has not been set up as a complete desktop system to save resources.

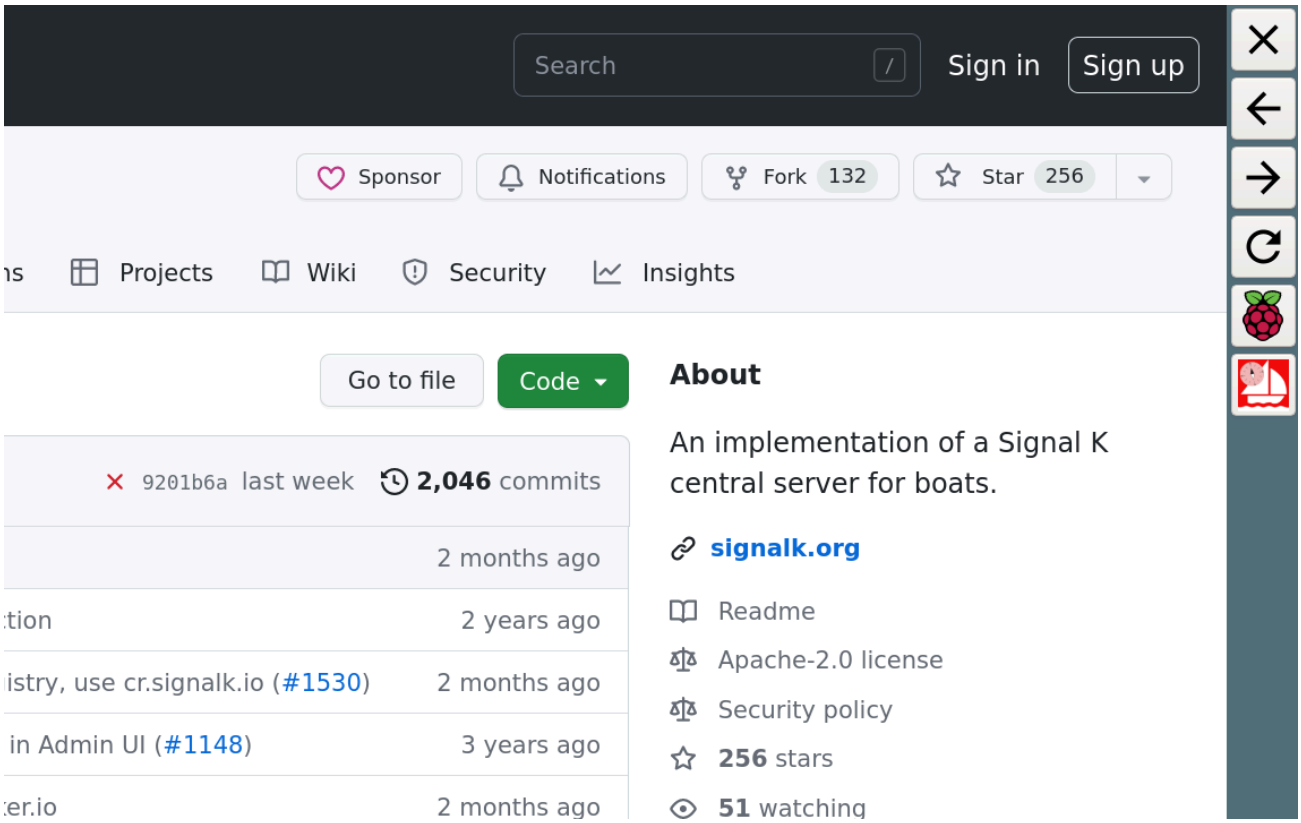
As you can only reach the system tools via the button inside the AvNav app it is recommended to additionally prepare one of the other options to access your system to be able to repair.
To restart the local UI from the command line use

```
sudo systemctl restart avnav-startx
```

If Firefox fails to start due to some problems with it's profile data you can just remove it. On the next restart it will be recreated.
Important: AvNav settings that not have been saved to the server will be lost in this case.

```
sudo systemctl stop avnav-startx
rm -rf /home/pi/.mozilla/firefox/avnav
sudo systemctl start avnav-startx
```

Since version 20230614 the main screen will show an additonal panel every time AvNav has not (or not completely) loaded.



Using this panel you will get some basic firefox navigation functions and you can switch to the second (system) screen. Additionally you can execute the reset function for the firefox profile described above ( ).

This will help to operate the system even if AvNav does not start up completely. For new installations you will also find the reset function on the second screen.

Package Installation

Repositories

Thanks to Oleg there are ready to go package repositories you can use in your debian based system. This will work on the pi as well as on any other debian (like ubuntu).

Information is available at his github wiki.
To set up the repository on your system, run the following commands (only required if you don't use the headless or the AvNav touch images):

Debian Buster (amd64,armhf,arm64)

```
wget https://www.free-x.de/debian/oss.boating.gpg.key
sudo apt-key add oss.boating.gpg.key
wget https://www.free-x.de/debian/boating-buster.list
sudo cp boating-buster.list /etc/apt/sources.list.d/
```

Debian Bullseye (amd64,armhf,arm64)

```
wget -O - https://www.free-
x.de/debian/oss.boating.gpg.key | gpg --dearmor | sudo
tee /usr/share/keyrings/oss.boating.gpg
echo "deb [signed-
by=/usr/share/keyrings/oss.boating.gpg]
https://www.free-x.de/debian bullseye main contrib non-
free" | sudo tee -a /etc/apt/sources.list.d/boating.list
```

Ubuntu Jammy (amd64)

```
wget -O - https://www.free-
x.de/ubuntu/oss.boating.gpg.key | gpg --dearmor | sudo
tee /usr/share/keyrings/oss.boating.gpg
echo "deb [signed-
by=/usr/share/keyrings/oss.boating.gpg]
https://www.free-x.de/ubuntu jammy main" | sudo tee -a
/etc/apt/sources.list.d/boating.list
```

For the installation on a linux system you need to execute:

```
sudo apt update
sudo apt install avnav
```

Startup

Afterwards you can start up avnav (as every user) from the commandline

```
avnav
```

(Note: this will only start the server part, you need to use a browser to connect to it - port 8080).
With

```
sudo systemctl enable avnav
sudo systemctl start avnav
```

you can let AvNav start with the user "avnav" automatically on system start up.

User Service

Starting from 20240520 you can run AvNav as a user systemd service for your user. To enable this you need to run

```
avnavservice enable [port]
```

This will enable a user systemd service "avnav" that will automatically start when the user logs in and stop when the user logs out. To have the service running even when the user is not logged in you can set the "linger" mode for the user:

```
loginctl enable-linger
```

For details refer to the systemd documentation.

To check the status of the running service use

```
systemctl --user status avnav
```

AvNav will use the default data directory at $HOME/avnav to put it's data there.

To disable the service again use

```
avnavservice disable
```

Download

Alternatively you can download all packages from my download pages:

- Releases
- Daily Builds

After downloading you can install with

```
sudo apt install ./avnav_xxxxxxxx_all.deb
```

If you would like to set up your raspberry like we build our images you can additionally install the "avnav-raspi" packages.
This will change the network configuration, will create a start configuration for AvNav with the user pi and will enable setting the system time and managing client networks.
Hint: This can easily conflict with the settings of your base system - so please use with care.

In any case you should additionally install the AvNav Update-Plugin with

```
sudo apt-get install avnav-update-plugin
```

or by direct download from GitHub.
Remark: The start and stop functions of the update plugin do not (yet) work with avnav services created as a systemd user service - but you can easily restart AvNav from within the app.

If you did not install the "avnav-raspi" package but you still want to auto start AvNav with a different user then avnav you should use the systemd user handling described above.

As an alternative you could create a service override - example for the user pi:

Create /usr/lib/systemd/system/avnav.service.d and create a file avnav.conf in this directory with the following content:

```
#Overrides for the avnav service
[Service]
User=pi
ExecStart=
ExecStart=/usr/bin/avnav -q -b /home/pi/avnav/data -t
/usr/lib/avnav/avnav_template.xml -n /etc/default/avnav

[Unit]
After=avnav-check-parts.service
```

Afterwards you can enable and start avnav as system service with

```
sudo systemctl daemon-reload
sudo systemctl enable avnav
sudo systemctl start avnav
```

If you did not create this file avnav will not run as user pi but as user avnav.

OpenPlotter

For OpenPlotter we have a complete integration for AvNav (thanks to e-sailing). In the repository https://www.free-x.de/deb4op/ (that is already active in OpenPlotter) the necessary packages are already available. So you can install them simply with

```
sudo apt update
sudo apt install openplotter-avnav
```

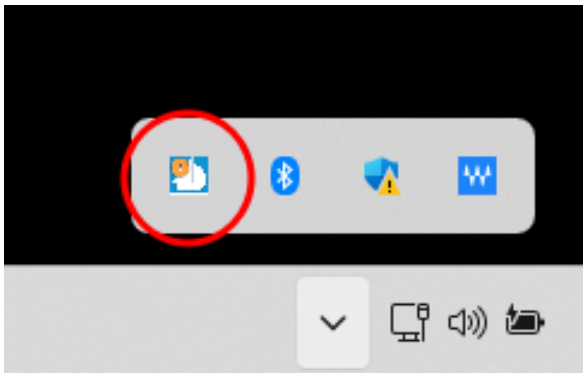
Since 2021/03 AvNav is already officially available in OpenPlotter. Sou if you update OpenPlotter you should already have openplotter-avnav being included.

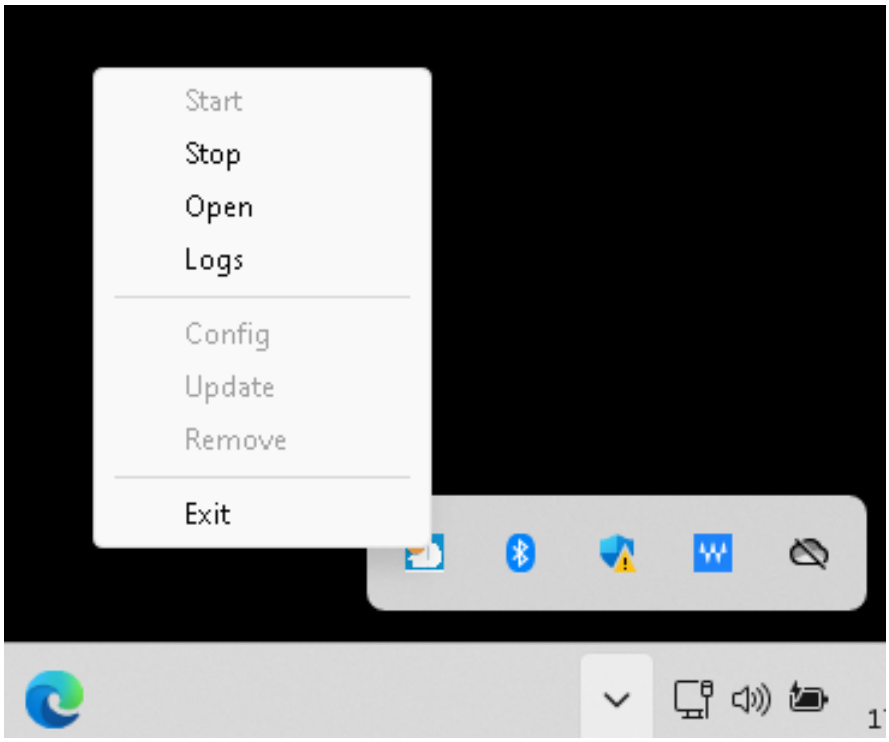
Do not install avnav-raspi_xxx.deb on OpenPlotter as this will interfere with the OpenPlotter network configuration. Within the OpenPlotter AvNav configuration you can change the port for AvNav (default 8080 and 8082 for ocharts) if this conflicts with other applications.

When installing AvNav this way it will receive all NMEA data from SignalK (and will not open up any USB devices on its own). So you can make any interface configuration changes in OpenPlotter or SignalK.

Windows

For Windows there is an installer (new from 20240520). The current version is available for download here. This installer will create an app "avnavservice" that (by default) is started automatically for you (User autostart). This service creates an icon in the notification area that can bring up a menu to handle installation, update start and stop of avnav. This small service does not yet include the AvNav software itself (or the necessary python installation). But from the menu you can trigger this installation.
For deinstallation use the normal way (ControlPanel/Software).





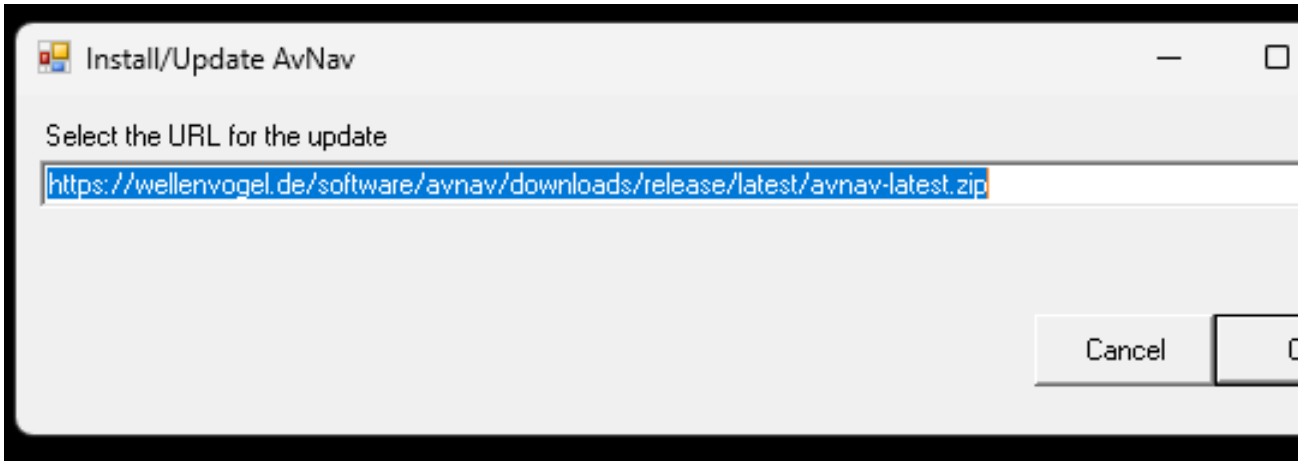
The service Menu has the following entries:

Caption	Function
Start	Start the avnav server. Only available if the avnav software is installed and avnav ist not running. The service will remember the last start action - so if you later restart and AvNav was running before it will automatically be started again.
Stop	Stop the AvNav server.
Open	Open the default browser and connect to the AvNav server.
Logs	Open an explorer in the AvNav log directory (PROFILEDIR/AvNav/logs). There is the normal avnav.log and additionally the outputs from the startup (service-err.log). To reach the AvNav config file you can simply navigate one directory up in the explorer.
Config	Allow to set the used HTTP Port (default: 8080)

Update	This will be "Install" if the AvNav software is not installed yet. It will bring up an install dialog (see below)
Remove	This will remove the installed AvNav software (but all your data at PROFILEDIR/AvNav will be kept). Before deinstalling the avnavservice (via system control/software) you should use this "Remove". Otherwise you need to clean up PROFILEDIR/AppData/Local/avnav later on by hand.
Exit	Stop AvNav and exit the avnavservice (the system notification will go away). You can start the service again via the start menu. Normally you can leave avnavservice up and running. As long as the AvNav server is stopped it will consume only very few resources. Only after an installation of avnavservice itself you need to stop and start it.

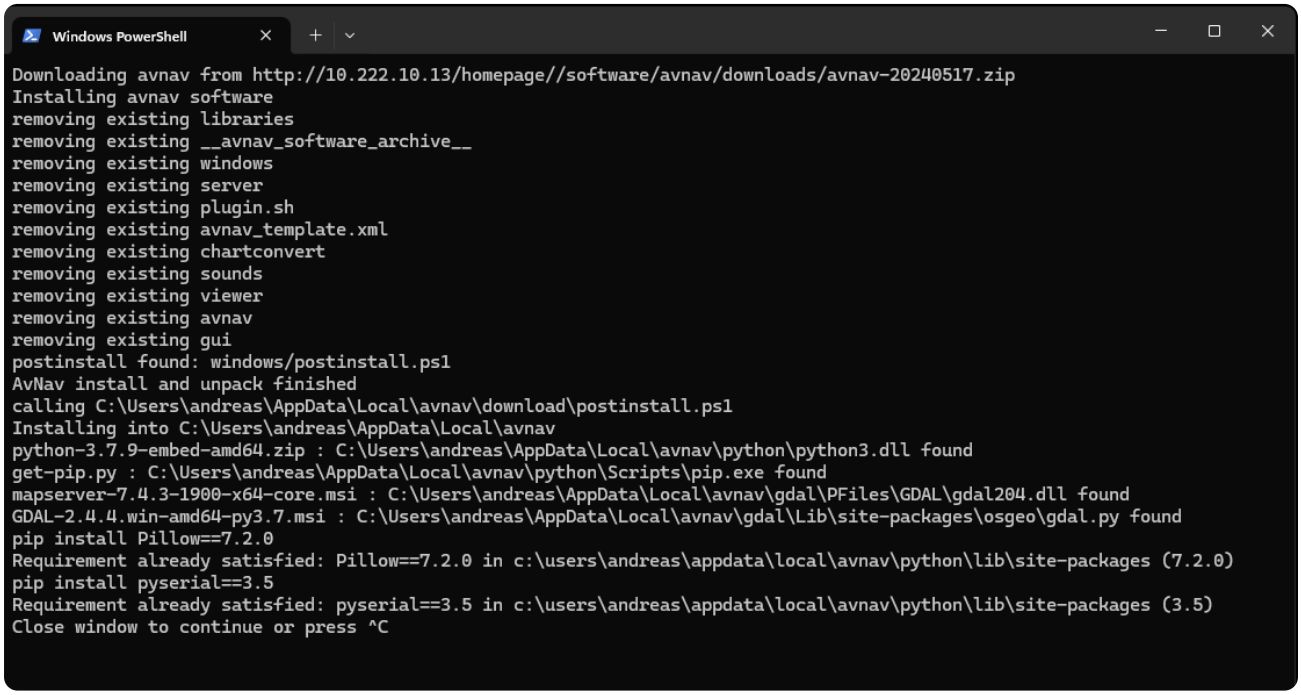
Installation

After clicking the Install/Update menu a small dialog will be brought up.



The url provided here is the default installation url for the latest AvNav software. But you can input any URL that points to a valid avnav software

package - e.g. from the daily or release pages.
After clicking OK an installation window will pop up and show you the installation progress.



After you close this window you can start the AvNav software again in the menu.

Windows Hints

The avnavservice requires Powershell (>= 5.x) - that should be available on all modern windows systems. If the installation later on fails there is a chance that the system misses the latest C/C++ libraries. In this case download them from Microsoft. The direct link normally is here.

This new windows installation replaces the old AvNavNetSetup with an on GUI. Releases starting from 20240520 will not be compatible with the old installation. It is recommended to completely remove any old installation before using this new installer.

The chart conversion functionality is now fully integrated in AvNav any way and can be used as described in the App Documentation.
Additionally you can use devices connected to your windows system in the normal way (e.g. a GPS stick).

As the AvNav server is running in the background you can also use it easily as a NMEA multiplexer and logger.

Avnav Releases

The directory with all releases can be found here.

Hints

The installation instructions have a detailed description of the different options to install AvNav. On this page you will find information about the various releases.
Not every update will be delivered by new images (although available for most). Intermediate updates are either developer versions (daily) or releases.
To install such an intermediate update you need to enter a command line on the pi (or you have the AvNav update plugin installed). From windows you could e.g. use putty. Just determine the address of your pi and connect, use pi, password: raspberry (or the values you have customized for your image).
Information on how to install the packages you will find in the installation instructions .
Afterwards you potentially need to adapt the configuration

```
/home/pi/avnav/data/avnav_server.xml
```

This could be done by

```
sudo systemctl stop avnav
nano /home/pi/avnav/data/avnav_server.xml # enter
modifications and save
sudo systemctl start avnav
```

If you have installed the AvNav Update plugin you can edit the configuration file from this plugin without the requirement to enter the command line.

On this page you will only find notes for released versions. For daily builds refer to the commits on GitHub .

Versions

20240616 link

AvNav base

- #255 Allow to start with the last split screen setting
  use Settings->Layout->"start with last split mode" to enable this
- #263 Allow to start with last used chart on navigation page
  use Settings->Map->"start with last map" to enable this

AvNav raspberry

- handling for bookworm and Pi5
- only start avnav during installation of avnav-raspi if it was running before (avoid issues with time changes during installation)

20240525 link

AvNav base

Bugfix for 20240520

- #342 Unable to activate routes

20240520 link

Please skip 20240520 and install 20240525 - Unable to activate routes

AvNav base

- new importer handling (not for Android)
  - allow to change import names
  - upload zip files
  - download converted files (zip)

- status display, disable, restart
- plugin api extension
- includes S57 converter for ochartsng

- new windows service and installer
  Please remove any old Windows installation before. This version is not compatible with the old AvNavNet installer any more!.
- removed separate windows/linux for chart conversion (is now fully integrated)
- simplified startup of AvNav from the command line, allow to start w/o any avnav_server.xml, http port option on command line
- allow avnav to run as a user service
- better status on NMEA channels including send/receive rates, error rates
- #324: allow AvNav to handle burst of received data with correct age handling in internal queues
- internally use steady timers - no need any more to discard data when system time changes
- #292: add keys
- #303: icons in title bar for anchor watch and disconnected mode
- #303: edit anchor watch, extra stop dialog
- #327: lock to zoom levels when using +/- buttons
- #334: changed VMC and HDG computation
- prevent sending out own received messages on channels (with a config switch to enable/disable)
- workaround for broken signalk-to-nmea2000 plugin for waypoint data
- #325: show relative motion vectors for AIS targets and turn indicator
- #320: remove unclear SK to AvNav mappings (i.e. no fallback mapping any more)
- #333: add more NMEA decoders
- #331: decode magVariation
- #247: add some help text
- allow for user icons and text in geojson, unifiy style parameter handling for overlays
- #312: accept XDR values without type or unit

- #310: include vmg into the navigation data sent to signalK, adapt the pathes to the signalk-to-nmea2000 plugin
- #294: search option on Ais List page
- #305: lock button on Ais List page
- better interruption of larger uploads

smaller bugfixes:

- allow AIS decoding of malformed messages (missing AIS channel ID)
- #332: invalid blacklist handling
- #321: logs filled with exceptions
- #319: fix broken waypoint dialog
- #318 adapt bsh demo source
- #313: correctly check list properties when saving
- #307: broken canvas gauges with negative values
- #301: correctly delete layouts

AvNav Raspberry (avnav-raspi)

- #330: prepare for bookworm (not finally tested)
- #297: support for MacArthur HAT (raspi)

AvNav Android

- allow for upload and download handling in the android app (important for ochartsng)
- always ensure IPv4 when starting browser (important for user app/plugin pages)

20230705 link

All Versions - important Bugfix:

- #288: You cannot add new points in the route editor

20230702 link

Important: Please do not use this release, update to 20230705

Main Focus: AvNav Android :

- released for Android SDK 33, new in play store (contains all changes from 20230426)
- source priorities for input channels (like on the linux variant)
- #284: location of version information

AvNav base (avnav) :

- improve course up handling when moving map in locked mode
- show values in bold on aisinfopage for better readability
- #283: handling too many buttons for two button rows
- #282: autohide 2 column buttons on dashboard in landscape mode not ok
- #286: add higher baudrate. e.g. moitessier hat

20230614 link

AvNav Images (avnav-raspi):

- Improved handling for touch images (OBP Plotter)
  - new panel on the start page if AvNav does not come up completely (see documentation)
  - reset for AvNav GUI (firefox profil)
  - **important Hint for OBP Plotter**: Please do also update avnav-obp-plotter-v3-plugin to 20230601 and reboot after all updates!

- New canboat version >= 4.12 to avoid n2kd forking every 30s (and eating up system resources)
- #274 update RPI4 (CM) Revisions in uart_control (GPS on obp plotter v3)

AvNav base (avnav):

- #272: Improved AIS Handling (Documentation)

- different icons depending on navigational status
- use HDG for display if available
- display of atons
- show estimated position

- #276: Move Map in lock mode (documentation)
- #275: Installation on bookworm (remove pip in postinstall)
- #277: correct requirements for fedora
- make installation possible on OpenSuse
- correctly show formatter Parameters in EditWidget dialog for widgets with flexible formatters
- allow to use online (AIS) streams that follow IEC 62320-1 - new Parameter stripLeading for readers
- #279:Fix permissions of avnav system user for serial/usb devices
- be more relaxed when checking kml overlays (allow more kml sources)
- do not ask for sound permissions on startup if sound is disabled

AvNav Android:

- #273: handle AIS on android for southern latitudes

20230426 link

Support for the OpenBoatProjects 10 inch plotter V3 - see project page.

- New plugin obp-plotter-v3
- Support in avnav.conf and in image preparation UI

Support for some Raspberry Pi HATS

- for AvNav images we can automatically configure a couple of HATs by creating appropriate config.txt entries, network interfaces,...
  Refer to the image preparation UI for a list

Extensions in the plugin API(Server)

- plugins (on the raspberry pi) can now have scripts to be able to set up system configurations based on settings in the avnav.conf
- new API functions sendRemoteCommand, registerSettingsFile, registerCommand

Include some additional Kernel Driver (raspberry)

- a new package will be installed on the AvNav images that includes drivers for the RTL8188EU and RTL8192EU wifi chip sets

Smaller Improvements and Bug Fixes
All:

- #249: use an event translation list to correctly handle event names for user widgets
- #251: allow to remove key bindings in user key.json
- #252: handle button key bindings correctly even if the buttons are currently hidden or in the second column
- #261: prevent an error on value and dict in store, limit logs to app. 110MB
- avoid filling the log with bluetooth errors
- add key definition z for toggle dimm mode
- add a reload page button on the settings page
- add some start URL parameters
- keep the scroll position on the AIS page and on the WPA page (allow to handle many AIS targets or many wifi networks)

Raspberry:

- #270: avoid shutting down wpa_supplicant (client wifi) on restart (e.g. from updater)
- #266: avoid blocking when setting the system time
  correctly handle initial time setting without gps, correctly switch between gps time and ntp time
- do not run the startup-check (handling of avnav.conf) when installing avnav-raspi (prevent reboot during installation)
- have a template of avnav_server.xml in /etc

- change all wlan ap network files to mode manual to avoid conflicts with avahi-autoipd (sometimes the wifi access point will not come up correctly)
- set restart-ms for all CAN interfaces (see forum) to improve NMEA2000 robustness
- correct some country codes for Image Preparation
- remove the obsolete spi-bcm2835-overlay
- Update install docu for packages to newer versions (bullseye)
- restart wlan-av1 when changing the system time (as wpa_supplicant will potentially stop working if the time changes)
- make wpa firewall command working again (external access for wifi networks) , better wpa logging
- add the uart_control script into /usr/lib/avnav/raspberry
- correctly configure CANboat (n2kd)

Android:

- some initial implementations for a plugin handling

20220819 link

Split Screen

- Change to and from split screen via button
- partially separated settings per tab - see documentation

Different Routing Modes: RhumbLine, GreatCircle

- default: GreatCircle
- switch in RoutingHandler
- correctly display great circle courses
- documentation

Improvements

- set wp data that is not computed to undefined (display: ---)
- #224: AIS improvements length, beam, draught

- #213: JS API for drawing on the map (SailSteer Widget)
- Some renaming of the captions for wind widgets, select true, apparent,...
- #237, #238: use the first kml file in kmz archives if there is no doc.kml
- only redraw the overlay itself if changed, avoid redrawing the complete map
- Scale parameters for kml and geojson overlays
- info dialog if server version has changed
- JS code (including widgets) will not be loaded for disabled plugins
- Default icon for GPX overlays, allow to set color and size of points
- display time, course, speed after clicking on a point on a track (feature info)
- Restart leg button (restart the XTE computation)
- export the JS LatLon modul at the API
- #243, #217: show remote conrtrol buttons only if remote control is enabled
- #170: different switch modes for way points: early,90,late

BugFixes

- #232: correctly send the AIS target SOG in m/s from android
- make conversion of track to route from downloadpage working again
- #222: ensure to reload the list of files at the downloadpage after delete
- #228: avoid to replace with the current hostname when editing a user app
- #228: keep the newWindow setting for a user app even on AvNav restart
- #225: make the chartconversion working again on gdal2 (Windows)
- #223: correctly merge storeKeys from a widget definition and from it's editable parameters
- #221: make translateFunction also work for simple user widgets
- #219: avoid the sk time socket connection to fill up the AvNav log
- #220: avoid hanging nmea decoder on invalid xdr records
- replace gir1.2-appindicator3-0.1 by gir1.2-ayatanaappindicator3-0.1 for xui set up
- prevent minification of user.js
- avoid accidently emptying the track file when the android app stops

- correctly compute route approach in the viewer
- #231 Fix Track GPX for OpenCPN compatibility
- #244: wrong XTE direction

20220421 link

Extensions of the SignalK Integration

- AvNav is now able to directly fetch it's navigation data from SignalK (including AIS)
- Waypoint data can be sent to SignalK
- Alarms can be received from and sent to SignalK
- The SignalK handling has been moved from a plugin to the "core"
- For details refer to the description

Images with Touch Support

- The AvNav images now include support for local monitors (touch optimized)
- This replaces the AvNav-Touch images that are not maintained any more.
- For details refer to the installation description

Setting of the System time without GPS

- The AvNav images now have a fallback handling for setting the system time
- If there is no valid GPS signal for a certain time AvNav will try to fetch the time from NTP
- This should prevent issues during the internet access (SignalK, mapproxy,...)

Other new Functions

- Separation of true and apparent Wind in the internal data. The display of the wind widgets can now explicitly assigned to one of them.

- When parsing RMC there is now a check whether there is a valid course and valid speed
- #169: Map scale display (Settings/Layer)
- #87: Alarm and red headline if server connection is lost
- #206: Set and reset anchor watch with keyboard commands
- #200: Different boat smybols for COG and HDM/HDT as well as "zero SOG", better computation of course averages for map rotation
- #202: Parsing of ZDA
- #188: Change/Switch off the remote channels directly on the navigation and dashboard pages
- Pitch and Roll for BME280, skPitch,skRoll widgets can now also handle deg as input
- Internal corrections

Bug Fixes

- correctly average course values
- improve error handling for avahi
- ensure restart button visibility on the status page
- android: correctly handle overlay-change-detection
- #212: Speed Gauge: maxValue < 10 crashes app (endless loop)
- chart converter now also working on bullseye (GDAL3)

Migration Hint

- if you did not configure special alarm commands just remove all AVNAlarmHandler entries from your avnav_server.xml. This will allow you to change the alarm sounds at the configuration and upload your own sounds to the user directory.

20220306 link

- Bug Fix #196: Android AvNav stops receiving GPS when in background
- Bug Fix #195: AvNav does not load a layout from a settings file from the initial dialog

- Small internal bugfixes

20220227 link

Windows Only!

- Correct bug in 20220225 that prevents server start

20220225 link

Save and Load Settings

- allow to save settings on the server and load them on the same or other devices
- extend the plugin api to be able to register settings (files)
- when starting for the first time allow to select a settings file if more then one setting is available. Later settings files can be loaded from the settings page.
- show changed settings on the settings page page with different text styles
- remove the ability to have settings in a layout file (use the new save/load settings instead)

AIS improvements

- #185: show AIS targets around the ship position and around the map center
- #187: allow to separately decide which AIS classes to be displayed (class A, class B,other)
- optimize AIS list performance by avoiding many dom objects
- allow to reduce the details in Ais list, allow to change list update interval
- allow to scale down class B ais targets in display
- use a separate color for tracked AIS target
- do not start tracking an AIS target when we click on it, only after "center to"  on AIS info page
- #149: allow to customize the data that is shown at an AIS target on the map
- #145: allow to hide AIS targets that do not move

-

Other new features:

- #135: allow to measure distances by dropping a mark (button , show course and distance to measure near map center and in center display widget
- make the edit route dialog a bit clearer
- #141: make WP buttons usable when anchor watch is active
- #141: add more options to set the anchor watch. Add anchor watch to the layout parameters for the nav page
- avoid sending old records when a new connection is establisehd on the server
- #146: sort charts on mainpage, prevent center map dialog
- #148: AvNav will autodetect every change of an overlay and will redraw the map
- #151: allow to open a user app in a new window
- #155: allow to disable the USB monitoring , bluetooth reader and NMEA logger
- #160: decode water temp from MTW and speed trough water from VHW add widgets for STW and water temp
- #163: allow sending data in the socket reader
- allow to run on a system without installed python-bluez (show in status if not installed)
- send out APB on android
- decode VWR NMEA0183 sentence
- #164 Add higher baudrates then 115200
- #180: warn if Android power saving will switch off GPS in background
- new SignalK widgets: skpitch/skroll (thank's Tom)
- better navigation between aislist and aisinfo

server startup changes:

- the logfile does now contain all the output also from the start up
- allow to set loglevel when starting, use -q to only control stderr logging
- do not start in silent mode when running as service

- start with loglevel error by default with -q

bugfixes:

- #192, #193: correctly provide formatter parameters to widgets that have a formatter being a function
- #191: remove useless caption in edit widget for combined widget, make css class working
- #190: add minValue/maxValue as parameters for radialGauge as they are mandatory
- #189: use own fork for canvas gauges to work around broken animationTarget plate
- adapt sizes for wind widget and depth widget
- #186: avoid sporadic crash
- #184: correctly handle store keys and formatter parameters in layout editor
- correctly deregister a layout or setting when the plugin is disabled
- #182: prevent setting empty formatter parameters if they are not defined in the layout
- correct bug in user defined formatters doc
- correctly handle timestamp for zip and kml files in head requests
- avoid duplicate user app when a plugin tries to register multiple times
- correctly show the default for formatTemperature to be kelvin
- do not reset colors to defaults when editing overlays, ensure opacity to be a number
- #161: add fallback handling if the cfg file is corrupt and no ok file is there, early logging switch to log cfg errors, show cfg file info in status
- handle keyboard interrupt for shutdown
- correct linux default config
- #178: make translate function working for all widgets
- correct APB
- #157: correctly switch to the next way point if no client is connected
- #154: update package install docs
- #150: directly link to list of mbtiles from OpenSeaMap
- #153: do not redraw dialogs when window size changes
- #152: make delete route working again on android

- work around gdal not correctly converting kap header data to utf-8
- prevent android crash if no file manager found
- #143: allow for a complete server stop from the UI

internal changes:

- new build tools and libraries
- force nodejs version 16 for build
- update to newer version of gradle node plugin
- introduce split chunks to optimize loading

20210619 link (Android Version: 20210618)

New Functions

- show night mode button during layout editing
- #88: decode HDG,HDT,HDM,VHW (part) on server and android
- add HDT,HDM widgets
- allow to use HDT or HDM for boat direction display (settings/navigation/boat direction)
- #138 lock the boat on any point on the screen (settings/map/lock boat mode)
- #139 show night mode on NavPage
- float map behind buttons on NavPage (settings/map/float map behind buttons)
- auto hide buttons on NavPage and Dashboard Pages (settings/buttons/auto hide buttons...)
- #132 show +/- 180 in wind widgets (configure in layout editor), small wind graphics improvements
- add the layout name as a css class to the app to allow to style for different layouts
- remote control
  - control one display from another or from the server
  - 5 channels, each display can send and/or receive on one of them
  - UDP port to receive remote control commands
  - plugin to interwork with obp-rf-remote

- add interface metrics to allow for easy switching between internet access via ethernet or wifi client (avnav-raspi)

Bug fixes:

- freeze when using linear gauges
- correctly sort buttons on older browsers
- #140: night mode for xte widget, wind graphics, various gauges
- make switching between access point and wifi client for the internal rpi adapter working againg (Config Gui)
- when both apparent and true wind are available prefer apparent in wind widgets, show which one is being used (avoid toggle between them)
- correctly handle filters separated by comma in the plugin API
- correct read filter handling for SocketWriter

20210502 link

New Functions

- Extending the configuration for headless images (hostname, keyboard, different Wifi modes for the internal adapter)
- Faster connection to other Wifi networks (faster getting of an IP address)

Bug fixes:

- Show button icons correctly in chrome lite mode on Android (#128)
- Some MCS modules for one wire have not been loaded correctly (headless images only)
- The name resolution for NMEA0183 services only happened once, now it gets correctly repeated on error

20210424 link

Main Focus: Android App

- The Android App now integrates a full NMEA0183 multiplexer - very similar to the server variant
- you can:
  - connect to multiple TCP/UDP NMEA0183 data sources
  - send out NMEA to TCP/UDP
  - use multiple USB-serial devices to send and receive NMEA data
  - send and receive NMEA via Bluetooth
  - send out own GPS data
  - run multiplexer in background and provide the NMEA data to other apps
  - use mdns (Bonjour/Ahavi) to easily connect to NMEA data sources even in changing networks
  - one click connect using mdns (Bonjour/Ahavi) e.g. to the NMEA0183 output of a SignalK server
  - announce own NMEA outputs via mdns
  - filter NMEA data in and out with filterstrings and blacklists
  - log NMEA data to files
  - view the status of all connections on the Status/Server page

- Android now integrates the WebServer in the normal mode (was named "external browser mode" bevor)
  - access from local device or from other devices
  - default port now 8080

- compute RMB records when routing (can be send out to control an auto pilot)
- the configuration of the Android App to most parts is similar now to the server version - using the Status/Server page
  - HINT: some configuration is migrated automatically but not all

For the Raspberry Version (when using the Headless images):

- support USB tethering to connect an android device via USB and have internet access this way on the raspberry - or connect to the AvNav server

(easily when using the BonjourBrowser)

For all Server Versions (Headless Images, OpenPlotter,...):

- announce TCP NMEA outputs via Avahi (mdns/Bonjour) - another AvNav server or the Android app can easily connect
- AVNNmea0183ServiceReader to connect to Avahi (mdns/Bonjour) NMEA services (like SignalK)
- Extension in the plugin API - initFunction does now receive the widget parameters

BugFixes:

- #123: bluetooth disconnects
- adapt to newer gdal versions for importer
- correctly always use utf-8 for file reading and writing in chart converter
- adapt CSS for better compatibility to older browsers
- better robustness againts NMEA messages with missing elements
- sometimes changing one widget in the layout editor also changes the one it was copied from

20210323 link

Bugfixes for 20210322

- Windows: AvNav does not start after the update
- Windows: SocketWriter cannot be modified
- #119: Crash in the app with linear Gauges
- #120: Crash in the app if a formatter is not existing (any more)
- Defaults for widgets in layout editor are not set correctly (e.g. colors at gauges)
- Crash in the app with Date/Time formatters if there is no or invalid data

new Windows Installer

20210322 link

The biggest change is the ability to directly edit the server configuration (e.g. adding new interfaces) from within AvNav. This requires all plugins to be updated to their newest versions (use the updater plugin).

AvNav Core:

- You can now edit the configuration of the server in the AvNav App (Server/Status page), changes will become effective without restart
- When using charts with empty layers (like many mbtiles files) AvNav will now zoom up lower resolution tiles to avoid empty areas in the display
- You can now scale the Map display (Settings/Map). This can help to improve the visibility on High DPI displays
- There is a "technology preview" for a split window mode - access with http://address:port/viewer/viewer_split.html, see demo
- Correct handling of symbol sizes on High DPI displays
- direct registration at MDNS (Bonjour), you can change the name
- Extensions in the plugin API

Raspberry Setup (Images)

- support for the MCS from GeDad, see image description
- allow for some customization of the image with a config file avnav.conf including a GUI to create it
- ensure Visibility of both AvNav and SignalK in MDNS/Bonjour
- more stable Wifi by switching of wifi power management
- automatically set up an IP address on the ethernet interface if not connected to a DHCP server
- Forward port 80 from all interfaces to AvNav (8080)

Smaller Bugfixes and Addons

- handle utf-8 in download names correctly
- keep status on status page until we get server error, correctly show and hide server error

- increase network timeout
- work around strange height bug on ios 12,13 safari
- always use shutdown when closing sockets, re-enable timeout for socket reader
- #113: merge pr, correctly handle null values in SK
- #117: prevent browser 2 finger zoom
- restart avahi when changing the system time
- avoid hiding second buttons on status page
- decrease default font sizes
- create an error gemf for failed conversions, add importer log display on download page, timeout importer status items
- allow opensuse install
- show plugin info in status if startup fails
- restart server
- log display
- server download log
- cleanup name handling and thread id for logging
- better timeout handling for serial (avoid reopen after 1s)
- rename AVNGpsdFeeder to AVNFeeder, completely remove gpsd related code
- #106: avoid writing errors to the log if empty positions lead to decode errors
- merge #105
- adapt to new handling of com port names on windows
- correct windows download links

20210115 link

- #73 migration to python3 (the current python2.7 is out of maintenance)
- plugin API java script extension for featureFormatter
- decoding of NMEA XDR records
- formatter for presssure and temperature
- extension of the java script API - registration of formatters
- improved handling of formatterParameters in the layout editor

- NMEA filter for AVNSocketReader,AVNBluetoothReader and AVNUdpReader
- the avnav history plugin is now available in the package repository and can be installed with

  ```
  sudo apt-get install avnav-history-plugin
  ```

- Bugfixes:
  - #97: NMEA checksum in lowercase
  - correct handling of chart dim for night mode

- **Installation Hints**
  When installing packages new dependencies are necessary.

  ```
  sudo apt-get update
  sudo apt-get upgrade
  ```

  If you are using the headless images you need to set the time before installing.

  ```
  sudo date -s "2021/01/15 15:00"
  ```

  It is important to install the current version for the ocharts-plugin (20210115).
  On Windows a new Installer should be used.
  If you update using the existing installer you need to run it twice! On the first update there is still no installation of the new python version.

20201227 link

- Plugin API extension for handling of USB devices
- Bug fixes
  - **RMB computation was broken with the last versions, repaired**
  - minor bugs in the plugin widget handling
  - minor bugs in button displays
  - some improved logs at the server

20201226 link

- Fullscreen Button
  If supported by the used browser there will be a button to toggle fullscreen on Mainpage, Navigationpage, Dashboardpage
- Separate settings category for buttons
- Bug fixes
  - OSM online charts
  - compatibility for older browsers
  - Android dim handling

20201219 link

- Chart Overlays
  - show gpx, kml, kmz and geojson files on the chart
  - stack multiple charts on top of each other
  - show available tracks and routes
  - user defined symbols and links to HTML pages
  - information dialog when clicking on chart object

- Display of object information (Feature Query) at o-charts (requires new avnav-ocharts version)
- Extensions in track handling
  - information display(length, time,speed)
  - #67: display of tracks on the chart (as overlays)
  - Converting tracks to routes
  - Importing of gpx tracks

- Improvement in route handling
  - New dialog for rename, copy, delete, empty, invert
  - Combining of routes
  - #9 add waypoints from a gpx file to routes
  - build routes "backwards" ("insert before")

- Improved error handling
  - Show an error dialog with the ability to save the error
  - Show errors in the user.js

- Settings extensions
  - "increase fonts on hires" - Scale up various elements on high resolution displays
  - "Overlay Info on Click" - Show an information dialog when clicking on chart objects (default: on)
    this also enables the object info for o-charts
  - "Always Info on Chart Click" - Also show a dialog on areas without objects (default: off)

- Improved documentation
  - Keyboard support documented
  - Index

- #79: Start parameter noCloseDialog to prevent close dialog when leaving the AvNav page
- Bugfixes:
  - Go back to mainpage after clicking on AIS object
  - "cancelTop" is working in firefox
  - Correctly send out MTA NMEA sentences (missing $)

20201202 link

Bugfix version for 20201105

- Bug: The signalk websockets connection(new in 20201105) does not work. Unchanged data will be removed after 30s. Additionally you could not unconfigure the use of websockets
- Bug: The documentation of the parameter useWebsockets for signalk was wrong.

20201105 link

- Feature: Power saving button (android and BonjourBrowser) #69
- Feature: Implement course vectors (depending on speed) for boat and AIS #59, see Navigation Page (some new Settings)
- Feature: Allow to set AIS symbol size and Border #58
- Feature: Allow to set images for different AIS targets, own boat, marker #53, see Description
- Feature: Show overflow button if more then 8 Buttons on a page, allow for 2 button columns #68, see Navigation Page
- Feature: Setting to force back button on top
- Feature: Remove package dependency to gpsd
- Feature: Allow plugins (widgets) to have event handlers at their objects and to communicate with the python part #75, see Plugin Description and User Java Script
- Feature: Usage of SignalK charts #76, see SignalK Charts
- Bug: Avoid n2k not working any more after long time
- Bug: Unexpected go back from various pages when clicking (itemlist clicking)
- Bug: Correctly go to previous waypoint when deleting last in route
- Bug: Become robust against empty values in DPT,DBT #60
- Bug: Correctly handle filters in avnav_server.xml with blacklist only
- Bug: Become more robust against shortened AIS type 5 message (no callsign, name,... for some AIS targets)
- Bug: Reload a chart in the browser if it has been modified on server (mbtiles, ocharts)
- Bug: (Android only) accept unknown NMEA talker ids
- Bug: Make center to Ais target working again #74
- Bug: Correctly show anchor watch distance in m #62
- Bug: Fix broken user app dialog for long urls #66
- Bug: Change handling for mbtiles scheme #63, see Description Charts
- **Hint**: When installing with dpkg -i (will give an error) you need to install missing dependencies with:

```
sudo apt-get install -f
```

20200609 link

- correct handling for chart files having spaces in their names
- Android corrections (20200605): correctly handle external browser mode on older devices

20200515 link

- GPS status display on mainpage (thanks free-x)
- Wind graphics on nav page does not shrink any more
- Depth display is working again
- CSS class for widgets in the layout editor is working again
- NMEA logger is working again
- No multiple loading of plugins
- remove eniro from demo sources

20200401 link

- fix connection to open external wifi networks (without password)

20200325 link

- Layout Editor to adapt the layout to your needs
- mbtiles format for charts
- Embedded graphic displays (using canvas-gauges)
- Management of user files and "user apps" to integrate own web pages
- Extension and adaption of AvNav with java script and css
- Rewritten Documentation (also in english)

20200204 link

- Support for NME2000 (via canboat) and SignalK - see description
- Layout changes

Older Versions on the german page only....

AvNav Charts and Overlays

Technical Background

To use charts in AvNav they have to be available in a tile format. This is a format used by services like OpenStreetMaps or GoogleMaps. A tile has (normally) a size of 256x256 pixel. The world is projected to a flat surface (imagine a paper cylinder wrapped around the earth at the aequator). Each point with it's coordinates (longitude/latitude) will pe projected at this cylinder. How this is performed in detail, which units will be used and whether the earth is considered as an ellipsoid or a sphere is described by the projection. AvNav is using the so called google mercator projection (earth considered a sphere) with the EPSG code 900913. The projected units are always meters (but can be converted to longitude and latitude). If you have charts using a different projection they have to be reprojected beforehand. The whole projection area is split into tiles. The zoom level determines the number of tiles for the complete area. Zoom level 0 means: complete earth (from -85°...+85°, outside these limits the projection is undefined) is shown in one tile of 256x256 pixel. With each other level there will be more tiles - level 1: 2x2 tiles, level 2: 4x4 tiles. The most relevant zoom levels for our usage typically range from 7 to 18..19. That means at level 19:2^19x2^19 tiles.

The openlayers library is used to display the charts. This library loads the tiles based on zoom level from the server (raspberry) and displays them on the screen. It is often used in OpenStreatMap applications.

You can easily imagine the huge numbers of tiles required for higher zoom levels (if you try e.g. zoom 19 for all). Therefore it makes sense to follow the same approach we take for paper charts: for an overview a small zoom level is used, detail charts a bit more and e.g. port approaches with the highest levels 18 or 19 (60cm/pixel and 30cm/pixel). To still maintain a nice display experience the different detail levels can overlay the other tiles of less detail. If there is a better (higher) zoomlevel for an area this one will be displayed - otherwise the one with a lower zoom level (potentially up scaled). To not overload the display devices, the number of layers should be limited to 3...5 (depending on the device).

Additionally AvNav can display vector charts from o-charts (using the ocharts or ochartsng plugins) and S57 charts (after conversion using the ochartsng plugin). Those vector charts are converted to the described tiles "on the fly" by the plugins.

You can get charts from various source - either already in formats that AvNav can directly use (gemf, mbtiles) or in formats that have to be converted (like BSB - .kap files).

There are also tools available to download such charts outside of AvNav or inside AvNav.

When converting charts the goal is to assign available charts to layers, to potentially reproject them and finally create the chart tiles (maybe together with a description). Typically this requires a lot of computing power (and time) and should be better run on a desktop system. But typically this should be no problem as you will do this only once. After you will be rewarded with a continous chart view with no gaps inbetween.
The same has to be considered when it comes to downloading charts. You need to carefully select areas and zoom levels.

Chart Formats

AvNav uses GEMF as its primary format. It is a compact binary format merging all tiles into one file with the necessary meta data added. This format permits special functions like automatic zoom level adaptation.
Since version 20200325 AvNav can directly use the mbtiles format. To use this format you potentially have to select the correct variant of internal tile sorting. The default format is "xyz" but there is also a "tms" format available. You can switch the format at the Files/Download page. You can download MbTiles directly e.g. from OpenSeamap.

Hint: Up to version 202011xx AvNav was reading an entry "scheme" in the mbtiles metadata. Unfortunately the usage of this entry is not well defined and different sources are using differently (see Issue #63). In earlier editions AvNav internally inverted the meaning of xyz and tms. Since version 202011xx AvNav ignores this value (and assumes tms). If the value is set in the metadata, the chart will be displayed in red and you can visit the Download-Page to choose the type to be applied for this specific chart.

For the handling of the vector charts refer to the ocharts/ochartsng descriptions.

Chart Sources

You can get charts from various sources - either in a format that is directly usable by AvNav (.gemf, .mbtiles, o-charts) or a format that needs conversion before (.kap, s57).
There are tools available to download such charts - either within AvNav or outside.

A list of chart sources:

- Downloading ready to go raster charts (e.g. from OpenSeamap, NOAA - mbtiles)
- Downloading charts using the Mobile Atlas Creator.

- Downloading charts and converting them inside AvNav (e.g. BSB kap files)
- Downloading charts inside AvNav using the mapproxy plugin
- Buying charts at o-charts and using them via the ocharts/ochartsng plugins
- Downloading S57 charts and converting/using them with the ochartsng plugin
- Using charts from the signalK chart provider (if the signalK integration is active).
- Using online chart sources, provided they support the default url format. You have to configure this with an xml file. An example can be found in the provided online source for OpenSeaMap.
- ...

Installation of Charts

After installing AvNav there is one/a couple of online demo charts available. For real usage you have to install charts in AvNav.

How to do this depends on the chart type.

Directly usable chart types (gemf,mbtiles)

Upload them on the Files/Download page.
On Android you can also copy them directly to the external chart directory (gemf files only).
For mbtiles consider the option to switch the internal scheme at the Files/Download page .

Charts with conversion (kap, s57)

In normal versions (not Android) you can upload charts that need conversion at the Importer (starting from 20200325).

s57 charts require an installed ochartsng plugin.
You should keep in mind that converting can require a lot of CPU power and can last for hours on the raspberry pi. Probably it is more efficient to run this on a desktop system.
As the conversion functions are not available for Android, you could additionally install AvNav on a Linux- or Windows-System and run the conversion there. Converted charts can be downloaded from the importer and uploaded to the Android app.

Vector Charts (o-charts, s57)

Those charts require an installed ocharts or ochartsng plugin (on Android the avocharts app - see ochartsng). You cannot use those chart types on windows as the plugins are not available (except the converter from s57 to ocharts). For details refer to the ocharts/ochartsng documentations. S57 charts that have been converted will directly become visible in the ochartsng plugin.

Downloading Charts inside AvNav

If the MapProxy plugin is installed its charts will directly become visible in AvNav without any further installation.

Overlays

Since version 20201219 AvNav can display further information as overlay (or underlay). You can as well combine different charts.

For details see Overlays.

Download using Mobile Atlas Creator

To use the Mobile Atlas Creators you only need Java and the MOBAC itself. It is important to follow a certain procedure when selecting the ranges for

download in order to fit the charts to the described layer concept and to limit the amount of data.

Typically, I recommend to use 3 layers: overview(zoom level 7-10), navigation (level 10-15), details (Level 16-18). Afterwards just proceed in MOBAC layer by layer. Select the zoom levels for the layer, select the areas you want and add them with an arbitrary name. Repeat this for all layers. Now save your selection using a meaningful name (the xml file could be reused later). Select OsmDroid GEMF (File->convertAtlasFormat) as output format and start atlas creation. In the output directory a xxx.gemf file will be created. Install this on the pi.
At the mapsources page I present a collection of useful map sources.

Avnav Ocharts(NG)

AvNav ochartsng is a new implementation of the AvNav ocharts plugin.
This implementation has a couple of new features:

- Now also available on Android (see below)
- Has a direct (online) Shop interface
- Can now also handle unencrypted S57 charts (after some initial conversion)
- Has more options for adapting the display

- Avoids the large disk caches that created a lot of SD card write load and a lot of CPU load
- Does not need to restart to handle newly uploaded charts
- Can utilize more cores of the processors and should run faster on many systems
- Has fewer external dependencies - especially it does not need a working X11 environment any more
- Utilizes less memory
- ...

Currently this implementation is in a beta status. So there are / could be some restrictions:

- Handling of oeRNC charts (raster charts) is still not implemented
- There could be bugs in the display or in some handling flows
- On Android you cannot use the same license for OpenCPN and for AvNav

Overview

AvNav is able to handle charts in various raster formats. So far it was unable to handle any commercial charts. The o-charts company supplies charts for various regions of the world for usage in OpenCPN.

After some agreements with o-charts, those charts can now also be used for AvNav (starting with version 2024xxx with this new plugin - see below). At present you can use oeusenc vector-charts .

Additionally the plugin is able to handle free S57 charts (after conversion).

To display those charts in AvNav they must be rendered into raster images. This is handled by a new plugin for AvNav(avnav-ochartsng). The rendering takes place on the fly whenever tiles are to be displayed. A certain amount of rendered chart data (tiles) is kept in memory on the server to serve them fast if you switch between certain zoom levels or if you run multiple displays.

All oeSENC charts handling is performed by the plugin - including installation (you cannot install them directly via the download page). For this purpose a dedicated GUI is offered by the plugin. You enter it from the main page via (User Apps) and Ocharts-Provider .

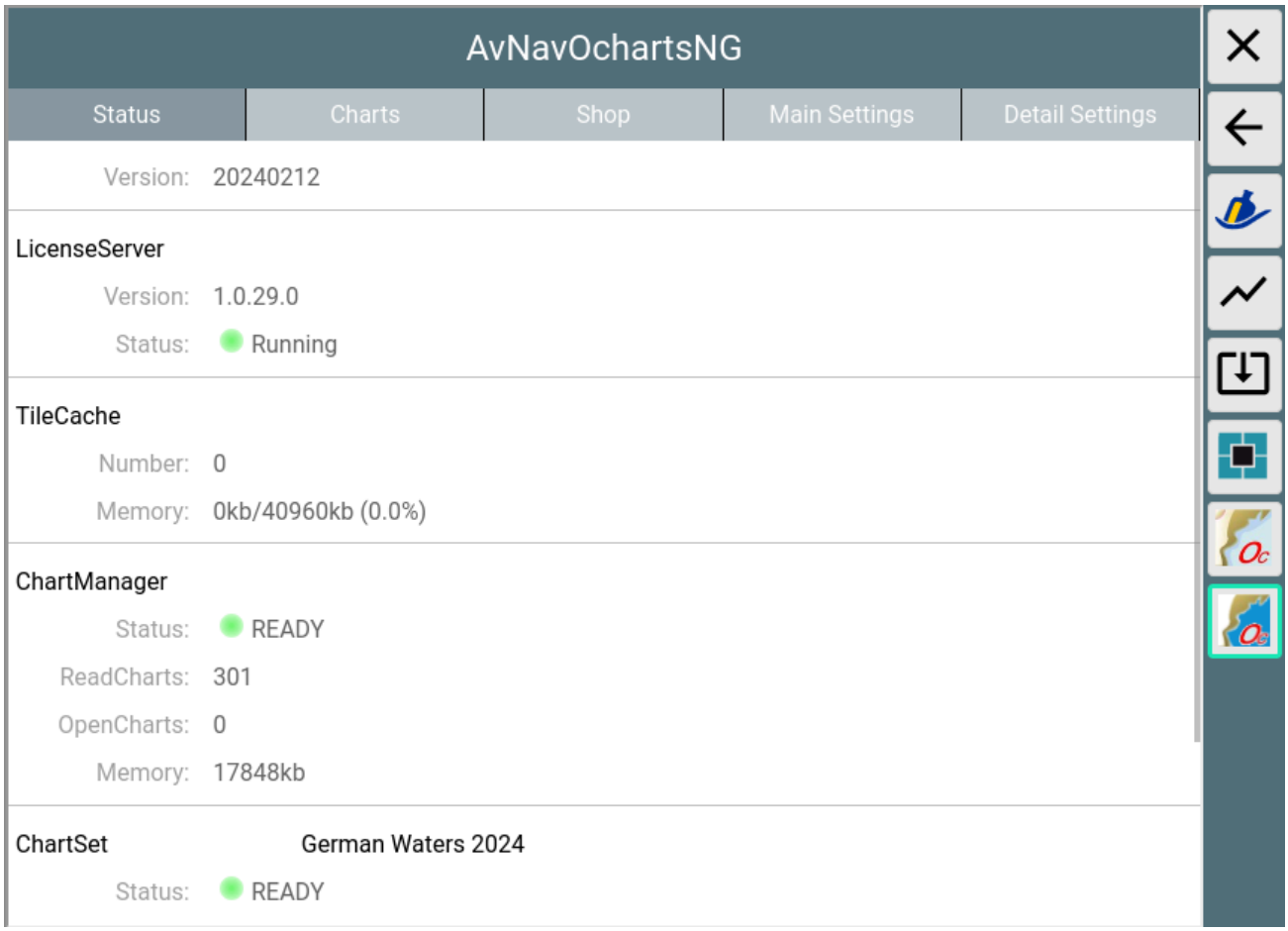


Chart Types

AvNav ochartsng can handle the o-charts vector charts (oeusenc) as described in this document.
Additionally it can also handle unencrypted vector charts - like S57 charts available for download for various regions.
To use S57 charts they need to be converted into the OpenCPN format that AvNav-ochartsng understands.
For details on how to convert charts refer to Chart Conversions.

Buying and Installing the Charts

Important Hint: If you do not have a dongle from o-charts, your chart license is bound to your system. So in case of any trouble **do not** set up a new system (by writing an image to an SD card) but instead try to repair the system. If you set up a new system you will lose your license. I will by happy to support in case of trouble - contact e.g. via email.

To be able to buy charts at o-charts you have to create an account at their site first.
Afterwards you have to register the systems you would like to buy charts for at the o-charts site.
You have two different ways to do this - named the "online process " and the "Offline" process .
The online process requires that your system (the one running the AvNav server - not necessarily the one running the browser!) is connected to the internet.
For Android the registration of the system is only possible with the online process.

The Offline Process

This process consists of the following steps:

1. Create a "fingerprint" for the system you would like to use the charts at. In AvNav you will create it in the GUI for the plugin and download from there.
2. Upload the fingerprint to o-charts and create a system (basically assigning a name).
3. Buy charts
4. Assign charts to the system you created
5. After a short time you will receive an email from o-charts with a download link (zip file).
6. Upload charts to AvNav (via the plugin GUI).

For updates repeat steps 4, 5 and 6 (only requesting the notification mail at step 4).

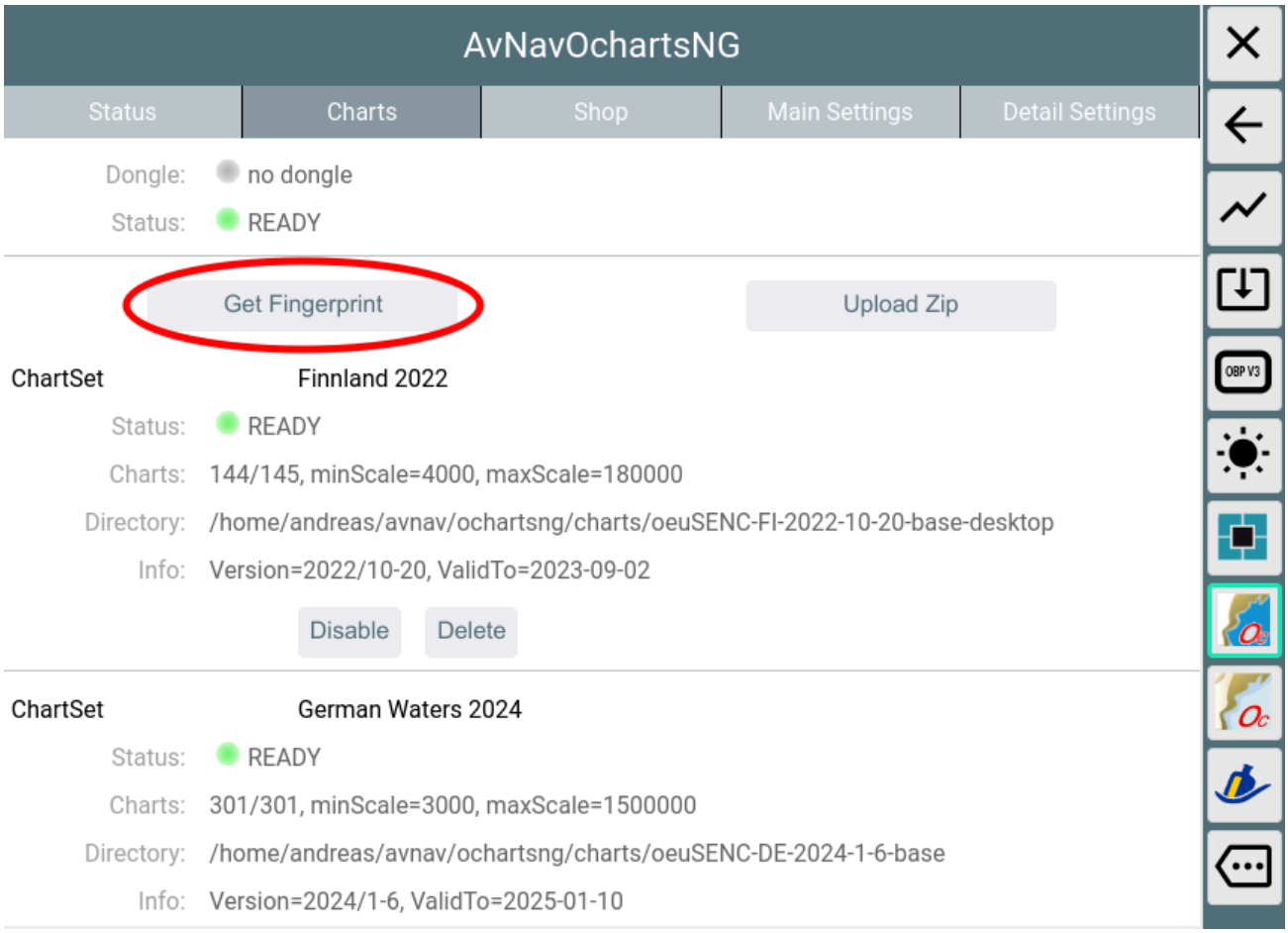
For further chart sets steps 3-6.
For steps 2,3,4 and 5 you need a system with internet connectivity. You can e.g. use a laptop or an android device.
Important hint for Android: The registration of the system (Steps 1 and 2) cannot be handled using the offline process. You have to use the online process for those steps. Buying charts, assigning them, downloading and installing them can be also done with the offline process.

I created a video to demonstrate the offline process. Additionally here is a short description.

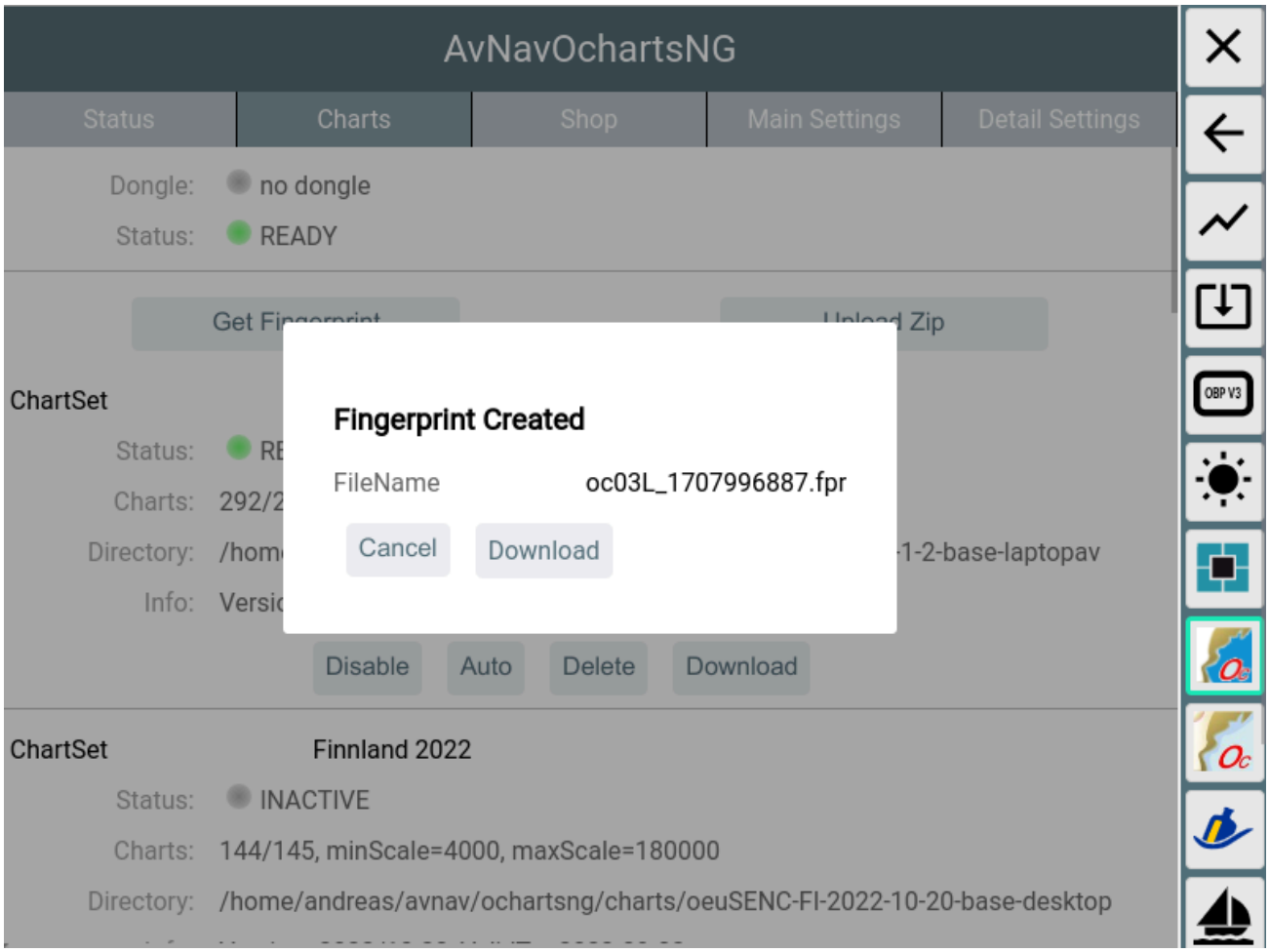
Hint: If the charts are already registered on the same system (for OpenCPN) you can directly continue at step 6 (but not for android - see restrictions above).

Creating the fingerprint

Via  ->  you enter the GUI of the plugin, select the "Charts" tab.



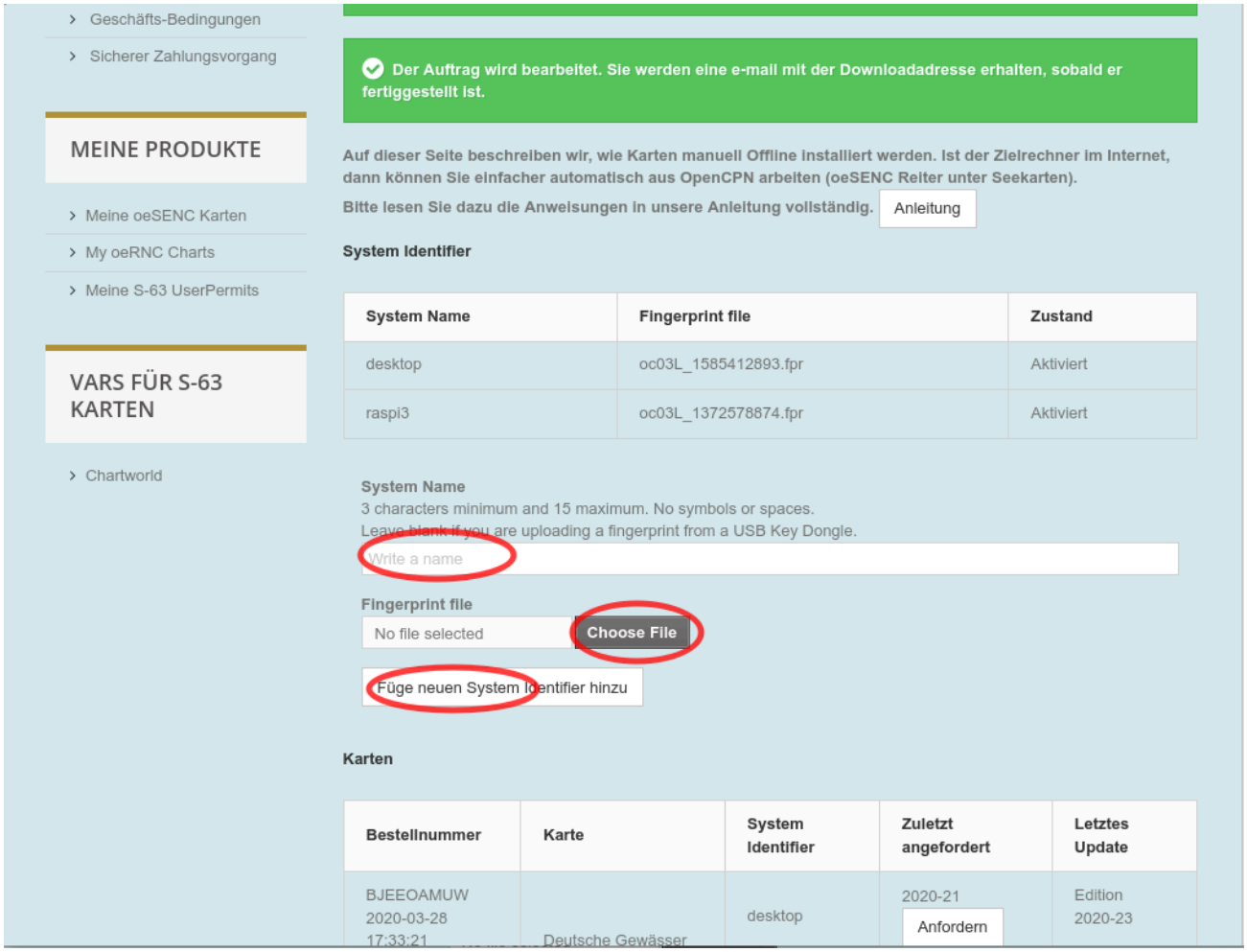
Use "Get Fingerprint" to create the fingerprint file. If you are using an o-charts dongle - just select "Get Fingerprint(Dongle)". The dongle button will only be visible if a dongle has been detected (dongle line is green).



Choose Download to save the created file on your device.

Uploading the fingerprints to o-charts

Enter the o-charts page and upload the fingerprint.

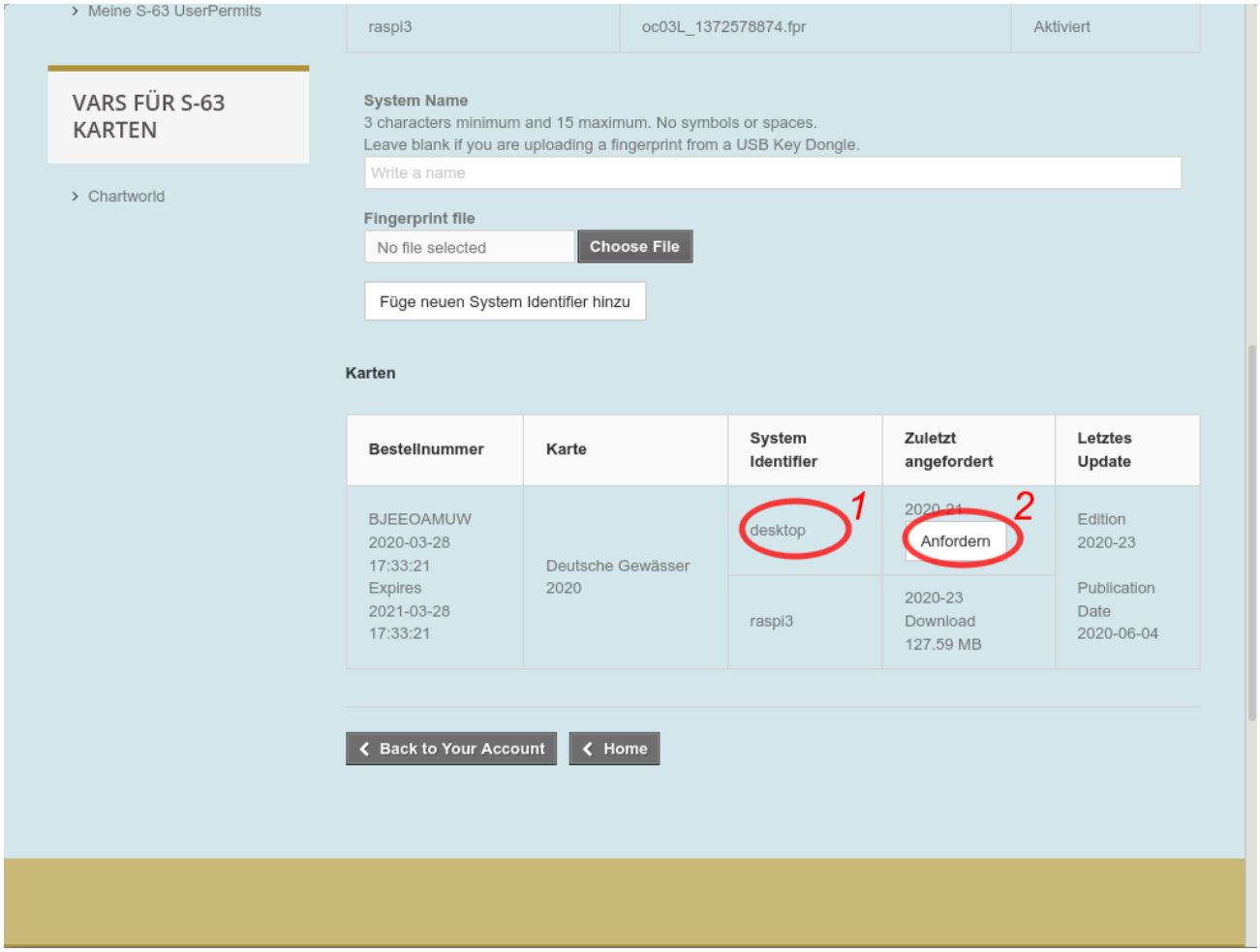


With "Choose File" you select the file stored at step 2. Assign a meaningful name to the new system - this will be part of the mails you will receive later.

Buying the Charts

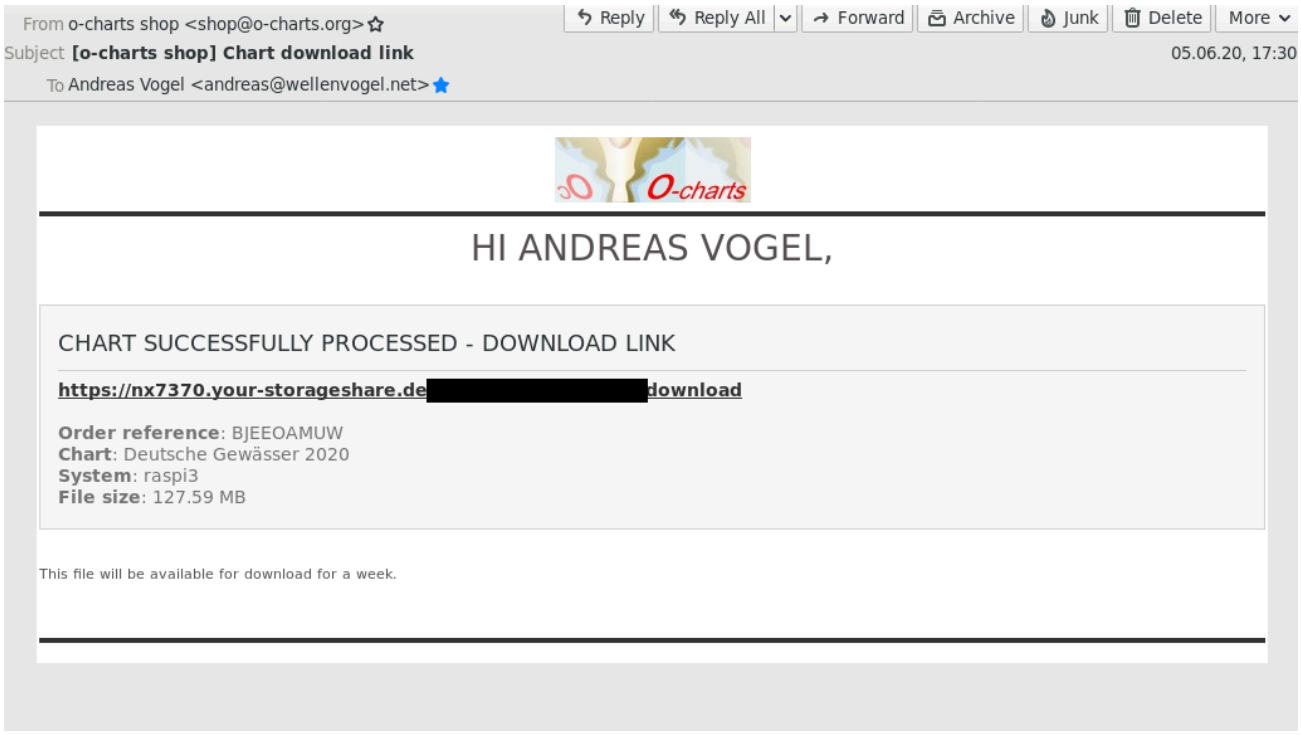
Select the desired charts from oeSENC charts.

Assigning to your System



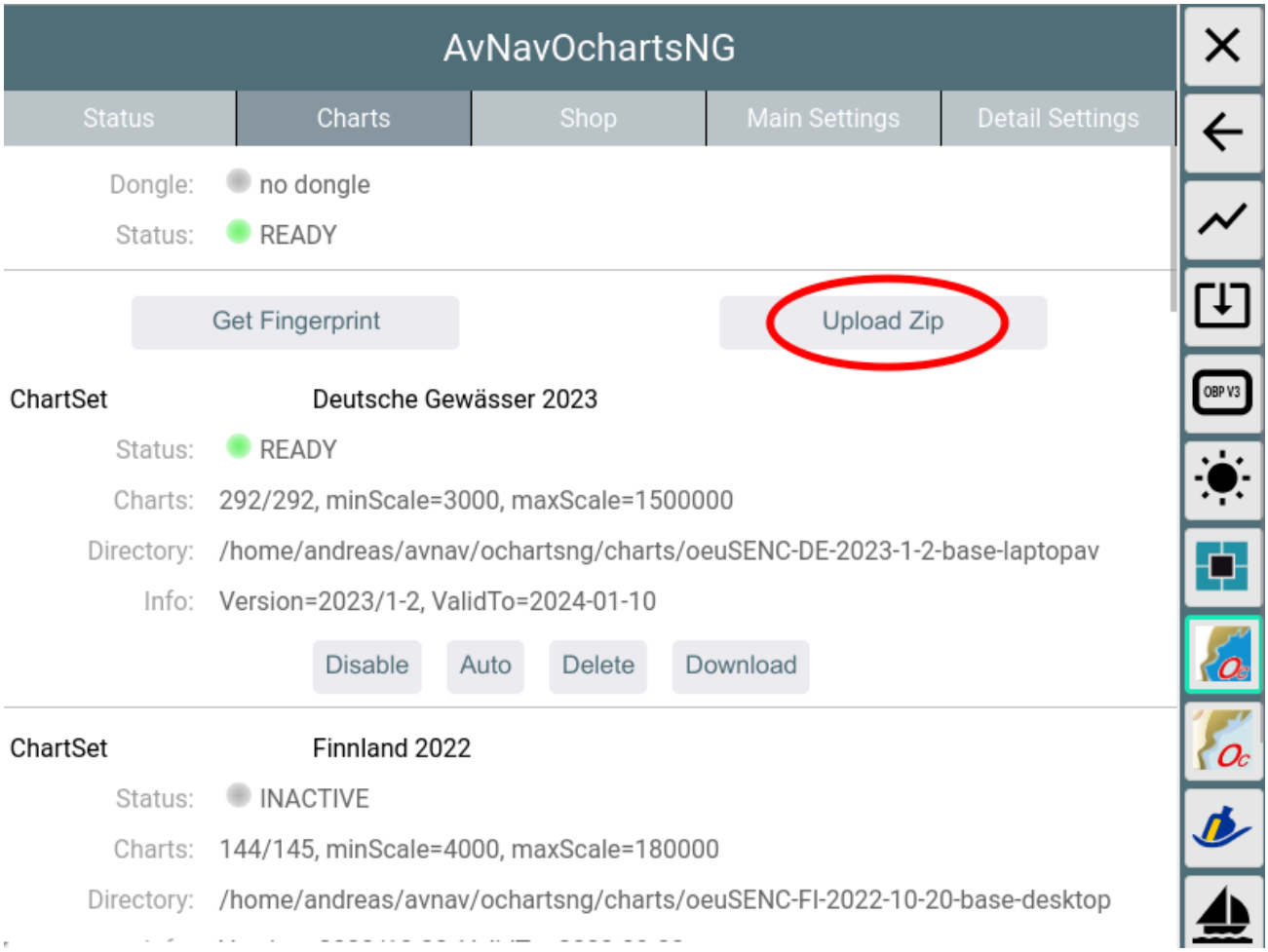
At 1 you can assign charts to your system (in this screenshot this is not available anymore as the max amount of 2 systems are already assigned). At 2 you request the mail holding the download link (you would do the same for updates - in the screenshot: last version I have downloaded is 21, latest available is 23)

Downloading the Charts



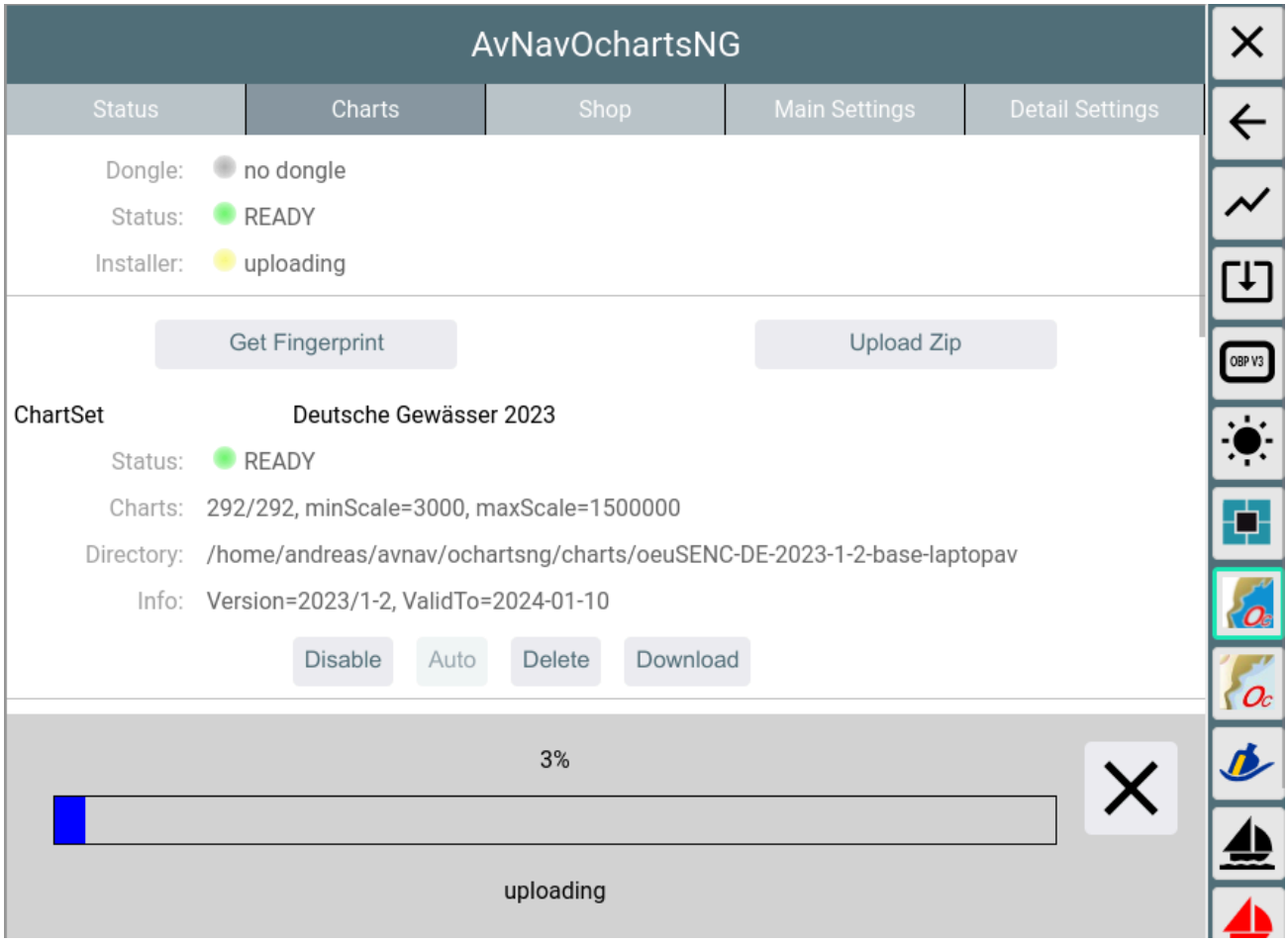
After a short time you will receive a mail containing the download link for your charts. Download the zip file.

Uploading the Zip File to AvNav



In the plugin's GUI select "Upload Zip" to transfer the zip file downloaded in step 5 to AvNav.

There will be a progress bar during upload.



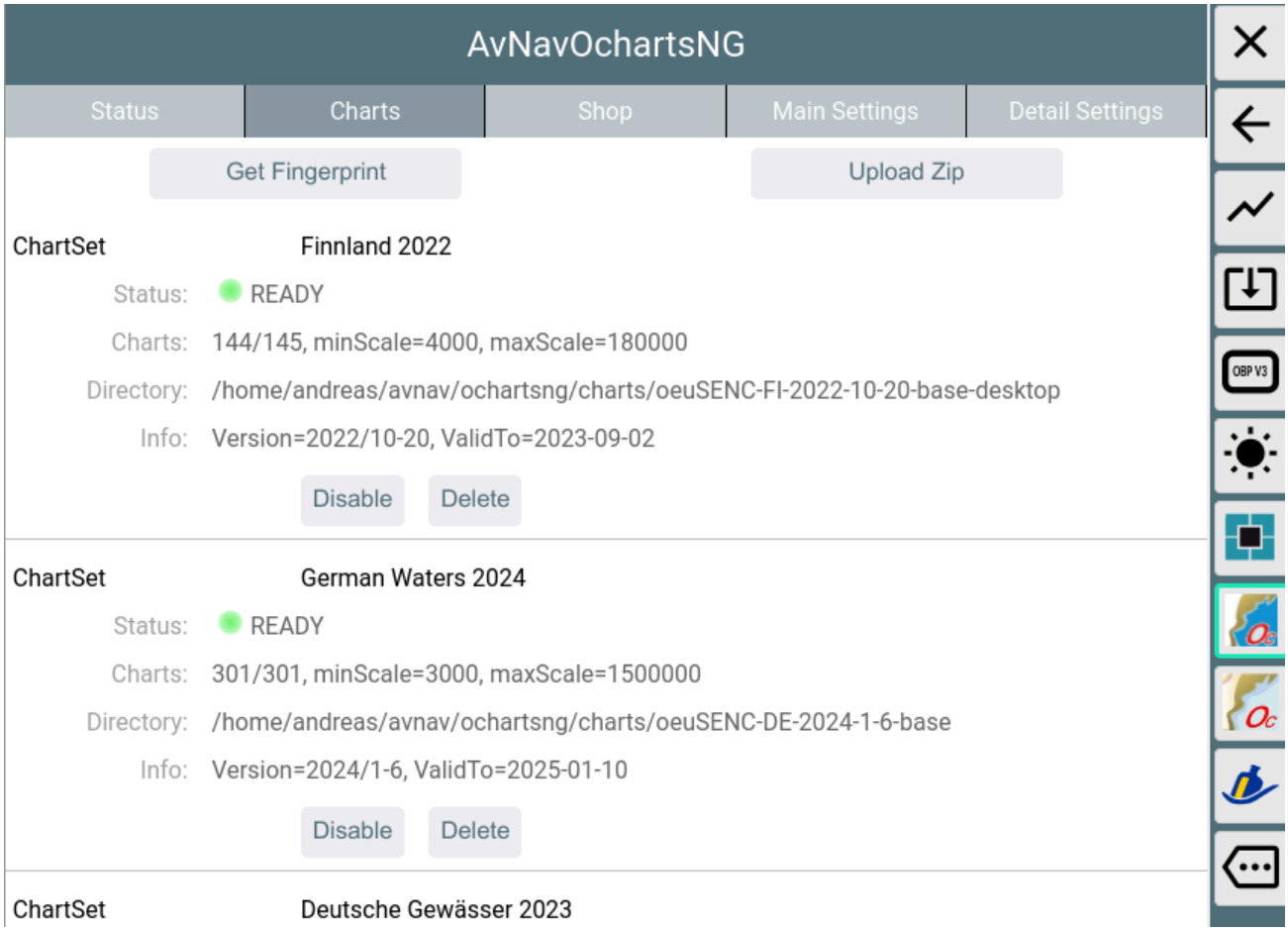
After the upload is completed the zip file will be unpacked. Some initial checks can take some time.

Unless configured otherwise, the charts will be uploaded to /home/pi/avnav/data/ocharts/charts.

If the newly uploaded chart contains updates to an already existing set, the existing set will be deactivated. You may change this later on by using Enable/Disable/Auto. Chart sets not required anymore can be deleted.

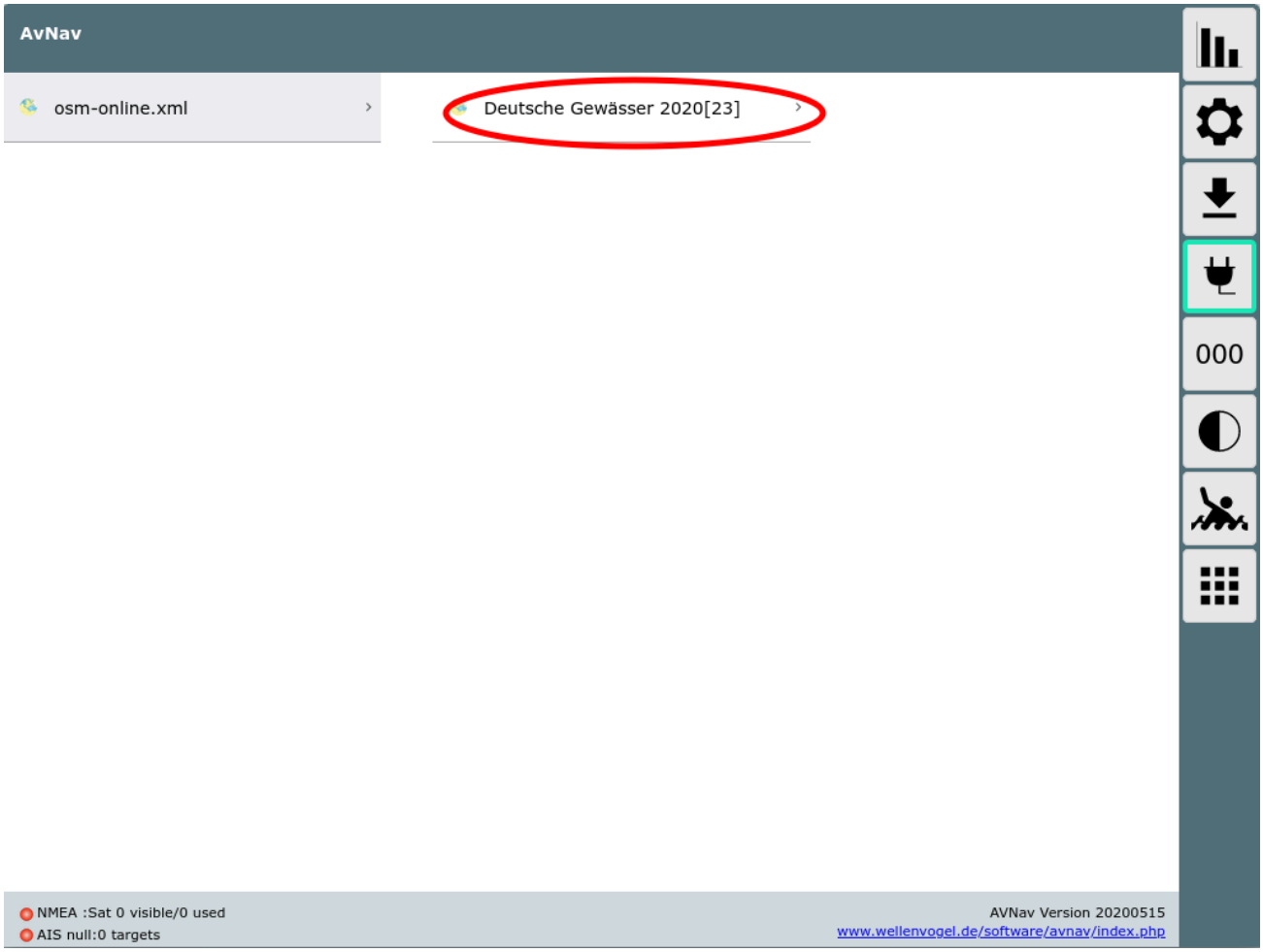
Important hint: As the charts will only be available at the o-charts page as long as they are valid just keep the downloaded zip file at a safe place. You can still upload and use those charts at AvNav ochartsng even after the validity period (but you will not receive any updates).

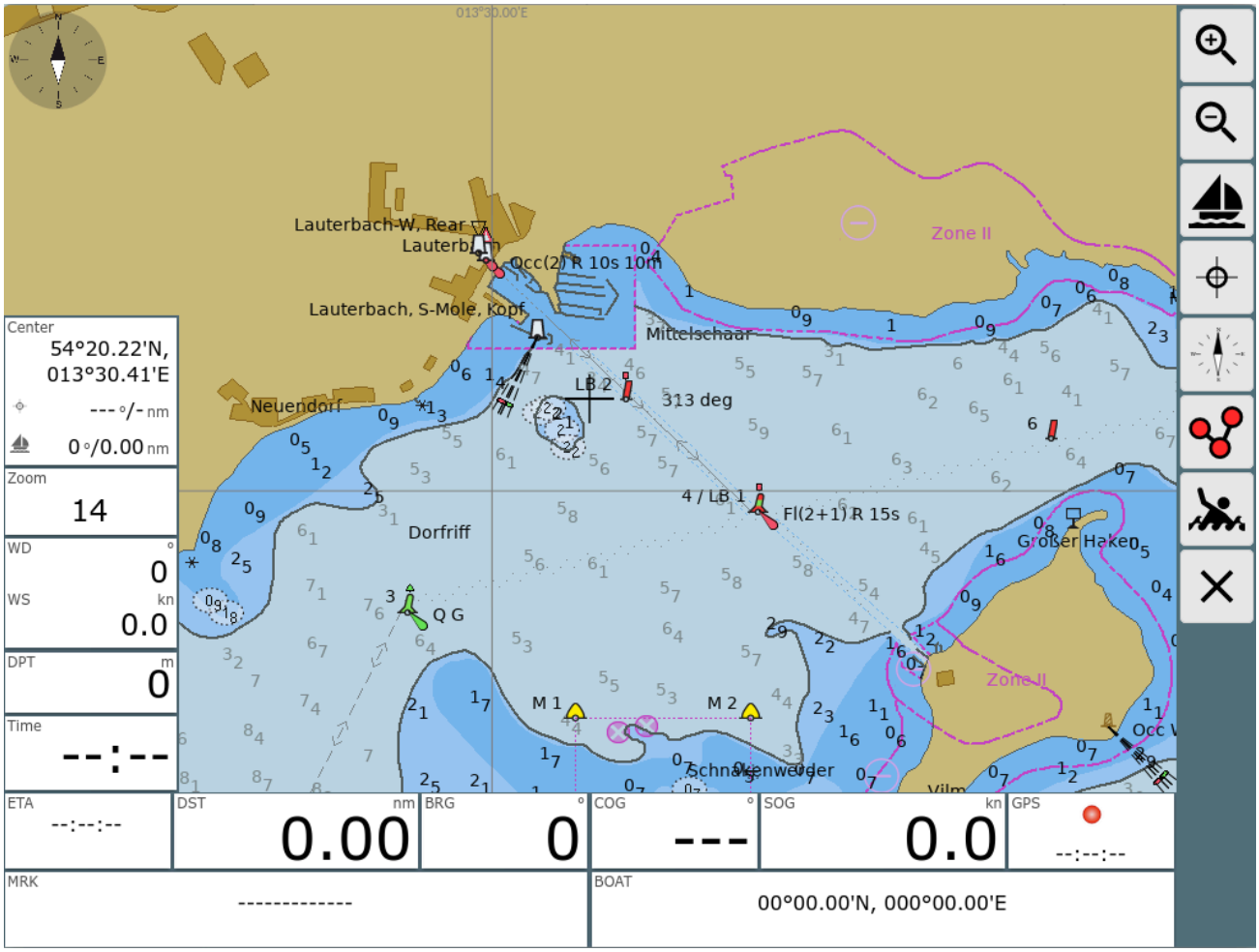
After all charts are successfully read the status should change to green (READY).



If the status turns to "ERROR" (red) you maybe uploaded a zip that was not built for your current system. You can check details in the log file at /home/pi/avnav/data/ocharts/provider.log.

Now the charts are available and can be used.


AvNav
osm-online.xml
Deutsche Gewässer 2020[23]
000
NMEA :Sat 0 visible/0 used
AIS null:0 targets
AVNav Version 20200515
www.wellenvogel.de/software/avnav/index.php


Lauterbach-W, Rear
Occ(2) R 10s 10m
Lauterbach, S-Mole, Kopf
Mittelschaar
Zone II
LB 2
313 deg
Neuendorf
4 / LB 1
Fl(2+1) R 15s
Dorfriff
Q G
M 1
M 2
Zone II
Schnaapenwerder
Center
54°20.22'N,
013°30.41'E
Zoom
14
WD
0
WS
kn
0.0
DPT
m
0
Time
--:--
ETA
--:--:--
DST
nm
0.00
BRG
0
COG

SOG
kn
0.0
GPS
--:--:--
MRK

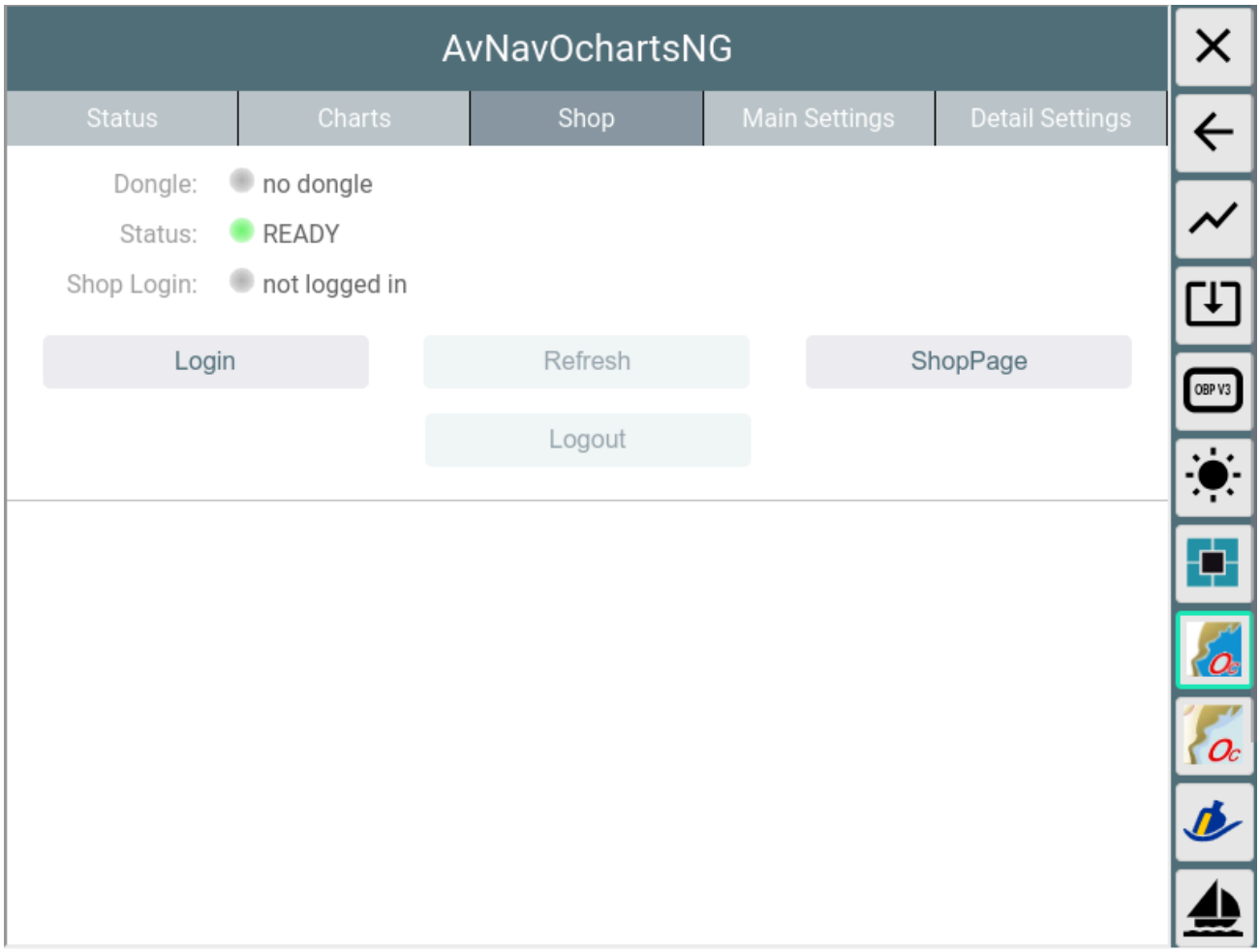
BOAT
00°00.00'N, 000°00.00'E

The Online Process

For using the online process your system (the server that runs AvNav - not necessarily the one that you run your browser on) needs internet connectivity.

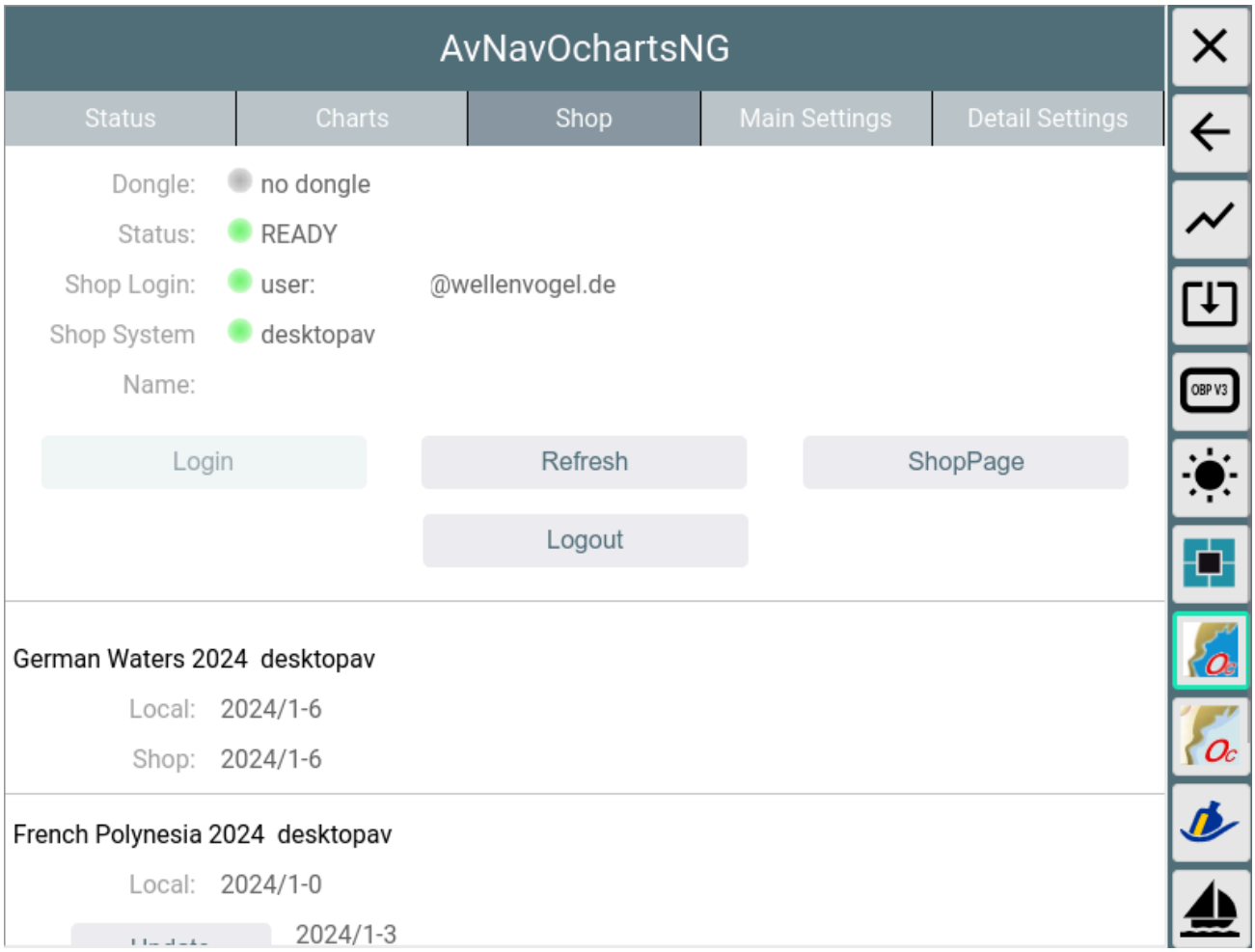
It will allow you to register your system, check if it is known in the ocharts shop, list charts or updates being available for download, download and install them.
Currently the buying process and the assignments of the charts have to be done on the o-charts web site. But the GUI will contain a link button to go there.
The AvNav ochartsng will not store any shop credentials. But as you run inside a browser you can easily use your browser to store the credentials (like you do on any other web site).

Security hint: Although the communication with the ocharts shop is secured by SSL the communication between your browser and the AvNav server is not. So when you are using the online process just be sure to work within a secured network (e.g. behind an LTE router or a mobile phone hotspot) - not within a public open Wifi network.

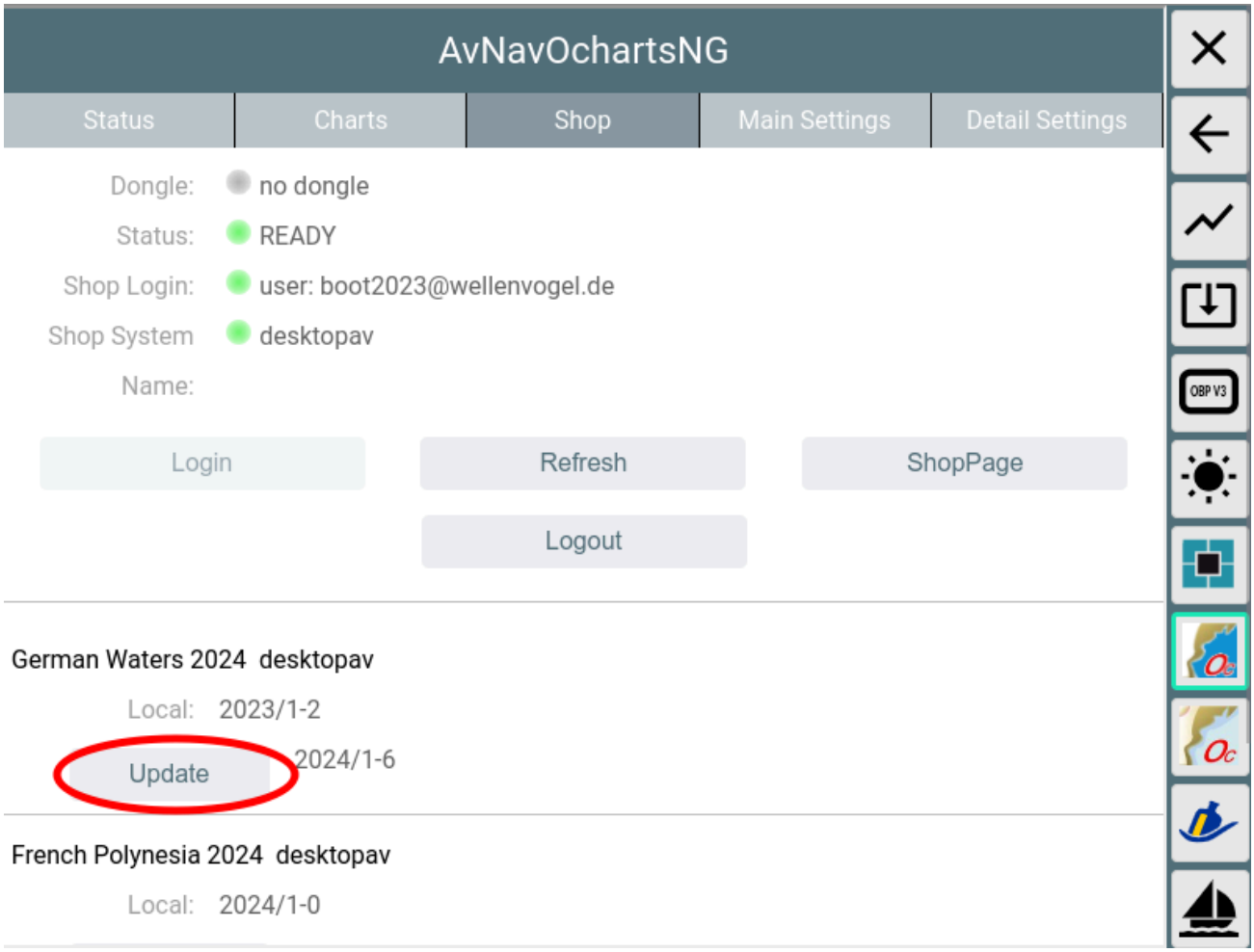


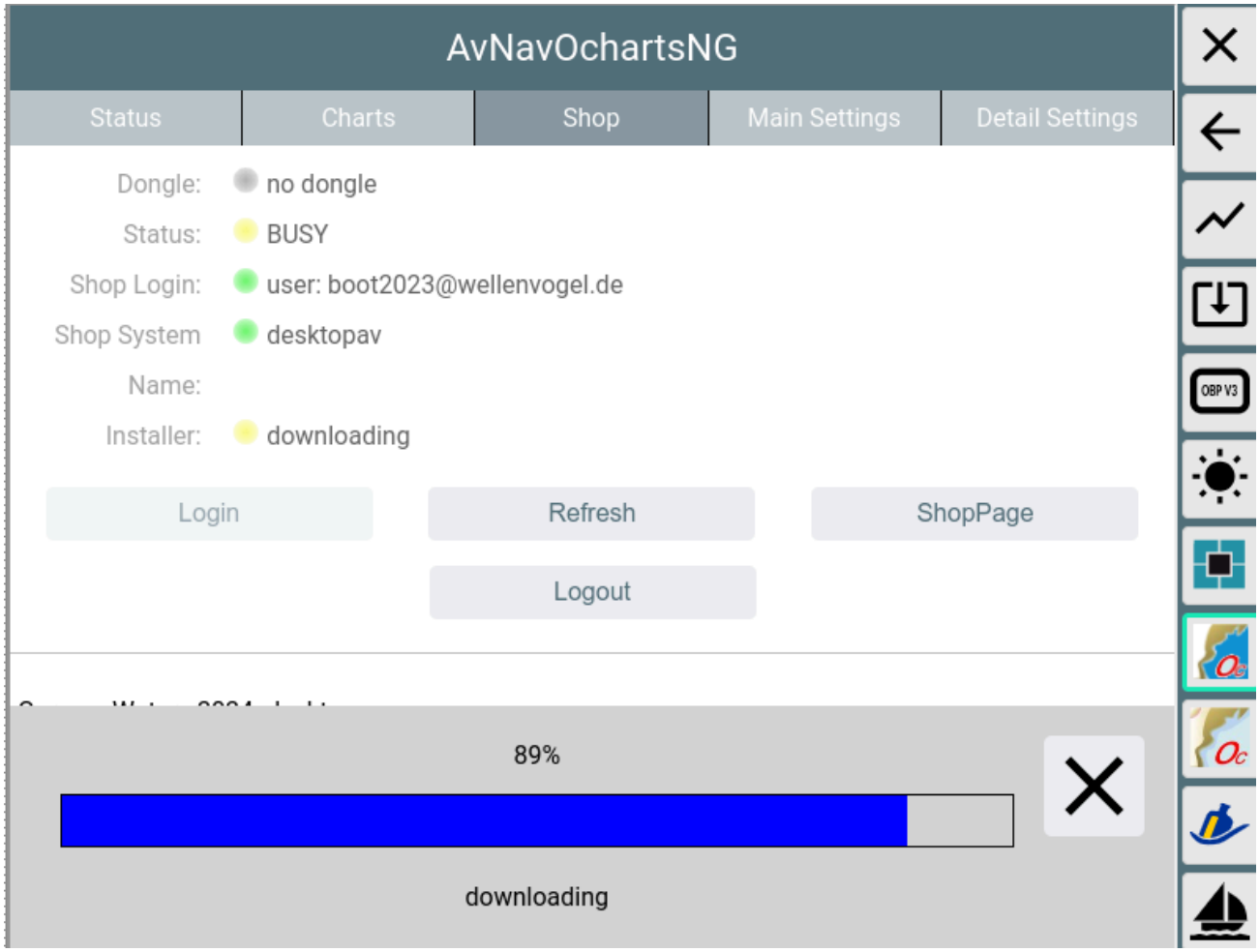
When opening the "shop" tab you will not be logged in and you will see an info whether a dongle is active on your system or not. Via the "ShopPage" button you can directly access the o-charts web page (but better log in first...).

With tapping "Login" you will login to the o-charts shop.



When the shop already knows your current system the "Shop name" will be shown and the list of available charts/updates that have been assigned to your system (see OfflineProcess) will be visible.
If a version in the shop is newer then your local version (or the chart is not locally available) you can immediately install it.

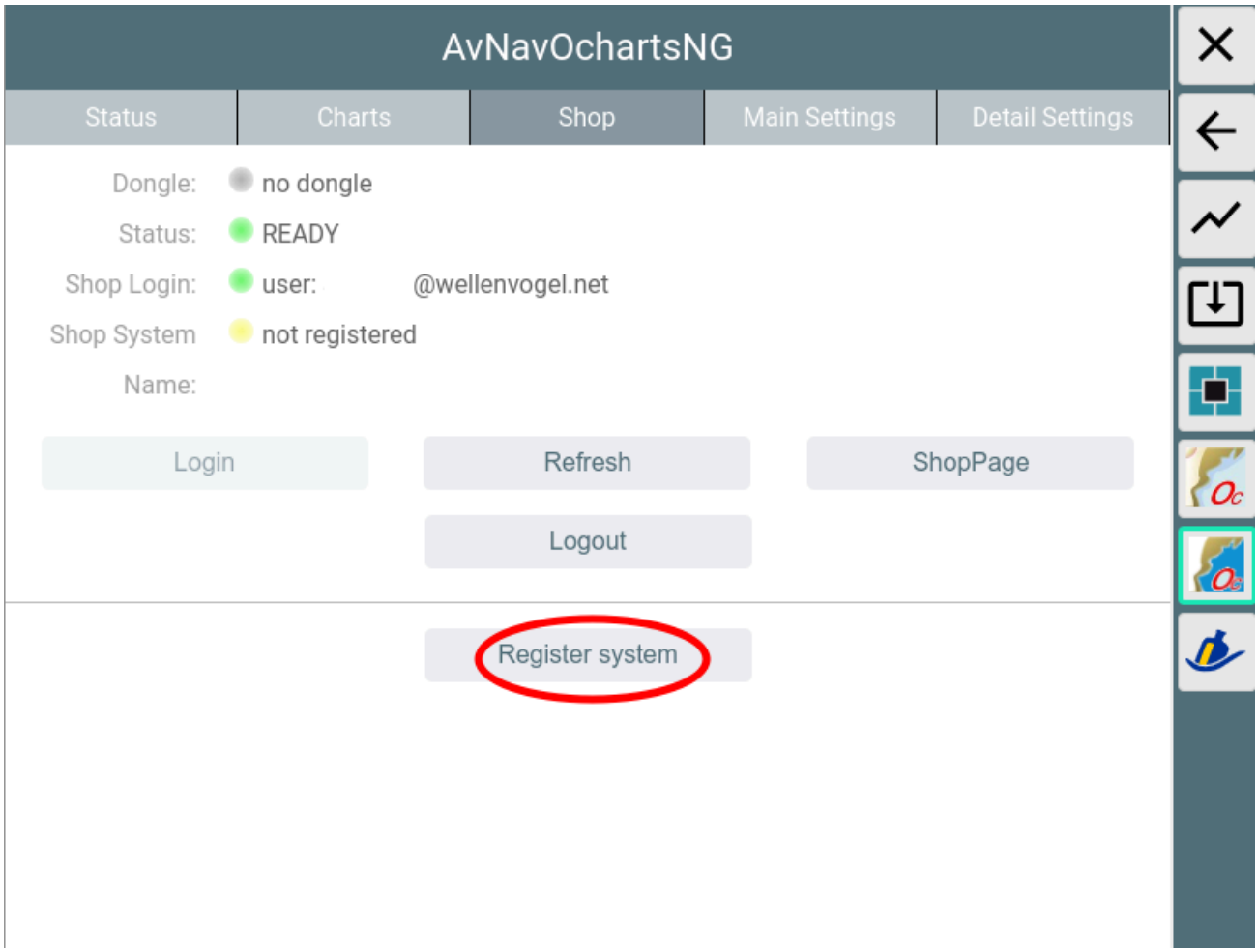

AvNavOchartsNG
Status
Charts
Shop
Main Settings
Detail Settings
Dongle: no dongle
Status: READY
Shop Login: user: boot2023@wellenvogel.de
Shop System desktopav
Name:
Login
Refresh
ShopPage
Logout
German Waters 2024 desktopav
Local: 2023/1-2
Update
2024/1-6
French Polynesia 2024 desktopav
Local: 2024/1-0


AvNavOchartsNG
Status
Charts
Shop
Main Settings
Detail Settings
Dongle: no dongle
Status: BUSY
Shop Login: user: boot2023@wellenvogel.de
Shop System desktopav
Name:
Installer: downloading
Login
Refresh
ShopPage
Logout
89%
downloading

After downloading the chart will be unpacked and can immediately be used. If it is an update for an existing chart set, the new version will become active but the old one will still be available (similar handling like uploading a chart zip file). On the chart tab you can later on delete the older version if you don't need this any more.

Important hint:
As the charts in the shop will only remain available for download as long as they are valid (one year) please store the charts additionally at a safe place. You can either do this from the shop website by requesting the charts (like in the offline process). Or you can download the chart set from the "charts" tab (Download button).

If your system is still not known to the shop you are able to register it there.



On linux systems you will be prompted for a system name but on android the system name is fixed and cannot be changed.

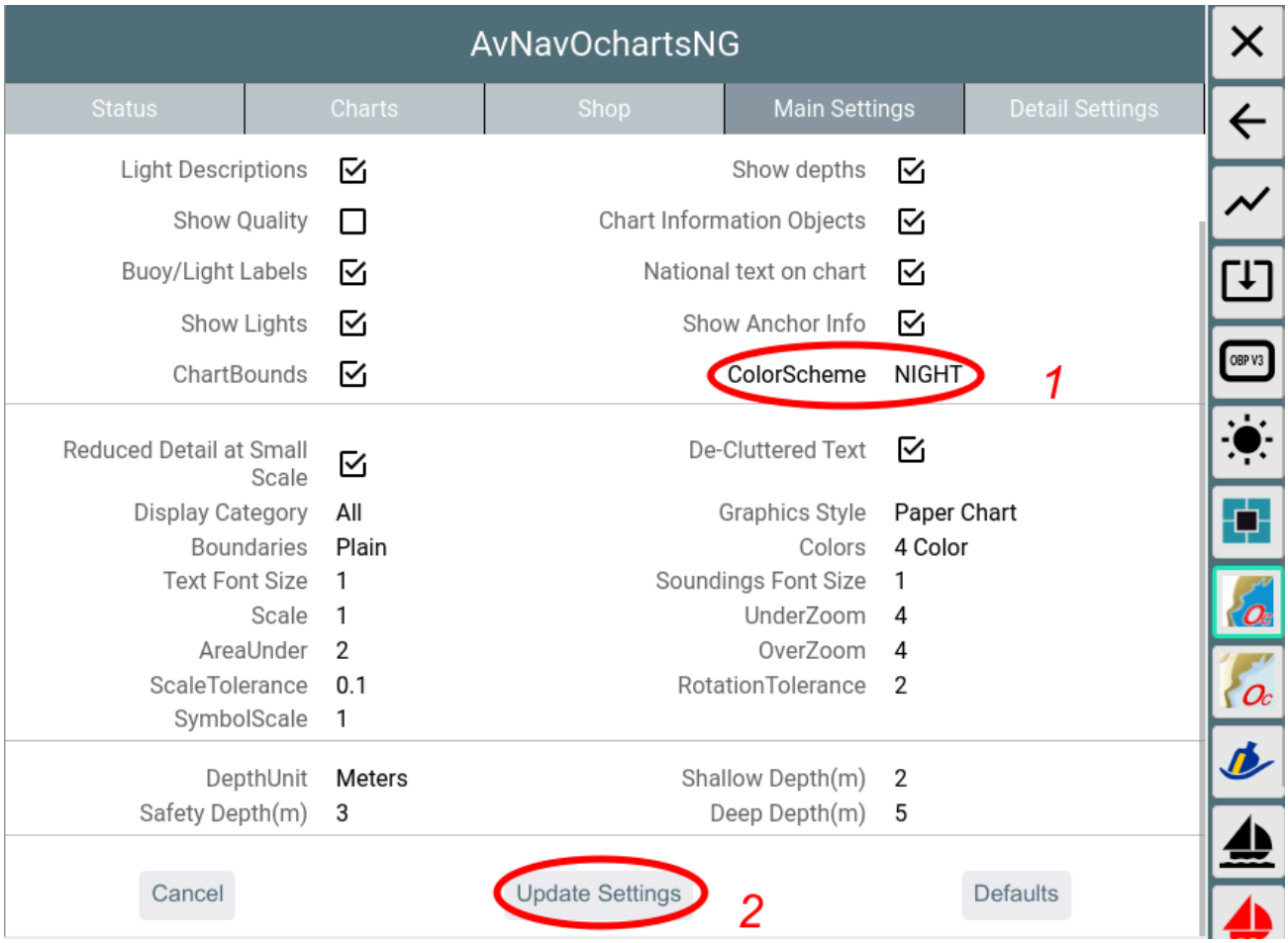
After you registered your system you can click the "ShopPage" link and start assigning charts to your system. When coming back to the shop tab you need to tap "Refresh" to get the list of charts that have been assigned.

Adapting the Look and Feel

As the o-charts are vector charts you can adjust their look and feel. However some limitations must be considered:

1. The modifications apply to the server's chart settings - so they will become effective for all displays.
2. If you change the look and feel all data in the internal cache is wiped and the chart tiles must be rendered from scratch. So this will (potentially) introduce some delays on systems with less computing power.

Changing display parameters is done in the plugin GUI(-> ), tab "Main Settings".



If you change a setting (1) it will be displayed bold. Changes will only become effective when you click "Update Settings"(2).
When you changed the settings the chart display will be rebuild. This will typically cause a small delay with no display at all.

By selecting "Cancel" you can revert your changes. "Defaults" will reset to the built-in defaults. Most parameters are similar to the ones you find at OpenCPN settings.

The following parameters are available.

Name	Meaning	Default
Show Text	show text for chart objects	true
Important Text Only	hide less important text	false

Light Descriptions	show descriptions for lights	true
Show Depths	show soundings	true
Show Quality	show extra objects indicating the quality	false
Chart Information Objects	show informational objects	true
Buoy/Light Labels	show labels for buoys and lights	true
National text on chart	show national text	true
Show Lights	show lights	true
Show Anchor Info	show anchoring information	true
Color Scheme	The color scheme to be used for the chart display	DAY_BRIGHT
Chart Bounds	Show the chart bounding boxes	true
Reduced Detail at Small Scale	reduce details at lower zoom levels	true
De-Cluttered Text	improve text positioning	true

Display Category	Base, Standard, All, User Standard	Standard
Graphics Style	Paper Chart, Simplified	Paper Chart
Boundaries	Plain, Symbolized	Plain
Colors	4Color, 2 Color	4 Color
Text Font Size	Scaling for text on charts	1 (ca. 12px)
Soundings Font Size	Scaling for soundings	1 (ca. 12px)
Scale	Base scaling. Higher values for more details on lower zoom levels	1
UnderZoom	Number of zoom levels to downscale higher resolution chart tiles if no chart tile is available at the requested zoom level.	4
AreaUnder	If with in the under zoom range still no charts have been found, just try some lower zoom layers - but only display the areas (land, depth) - no other information. Be careful when increasing this as this could impact the performance a lot.	2
OverZoom	Number of zoom levels to upscale a lower resolution chart tile if is no chart tile is available with better resolution.	4

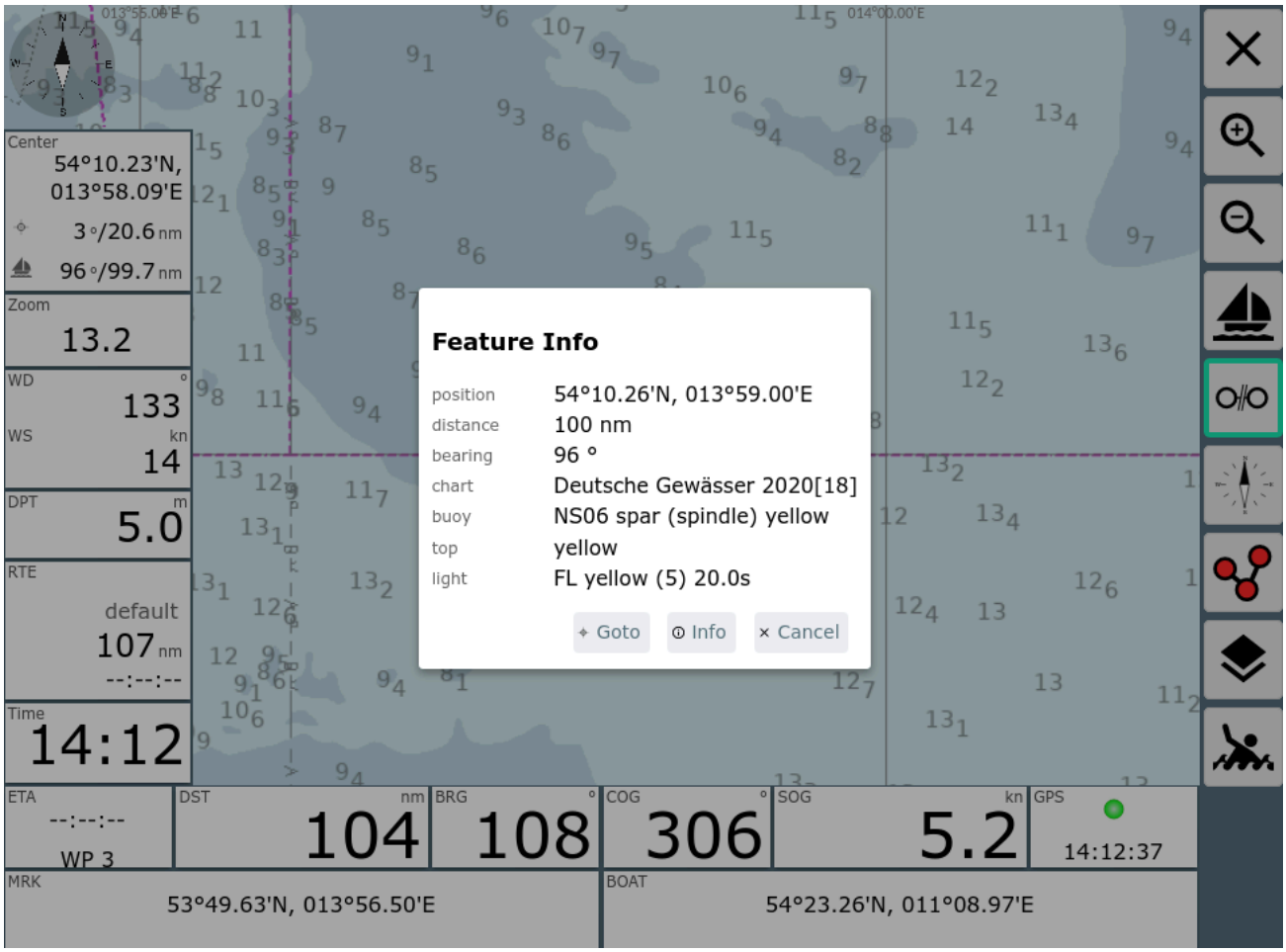
	Hint: Scale, UnderZoom and OverZoom, AreaUnder heavily influence the cost of the rendering process as they determine the number of charts to be processed to generate a single chart tile. Lower values normally mean less charts (i.e. being faster) - but there could be white areas between chart tiles. The defaults should be a good compromise.	
Scale Tolerance	When scaling the map or the symbols (symbol scale) all the symbols have to be rendered on those scaled values. To avoid too much additional symbols you can allow a small tolerance here.	0.1
Rotation Tolerance	When Objects (symbols, lines, lights... have to be rotated on the map for each different angle a new symbol has to be created. To optimize performance and memory usage we can allow a small tolerance here so that a couple of different angles can use the same rotated symbol (in degrees)	2
SymbolScale	Scale the display of symbols. Some of the symbols are build out of raster data - so a scale > 2 could already make theme look very ugly.	1
Depth	Unit for soundings(Meters, Feet, Fathoms)	Meters
Shallow Depth	Adjust to your needs	2

Safety Depth	Adjust to your needs	3
Deep Depth	Adjust to your needs	5

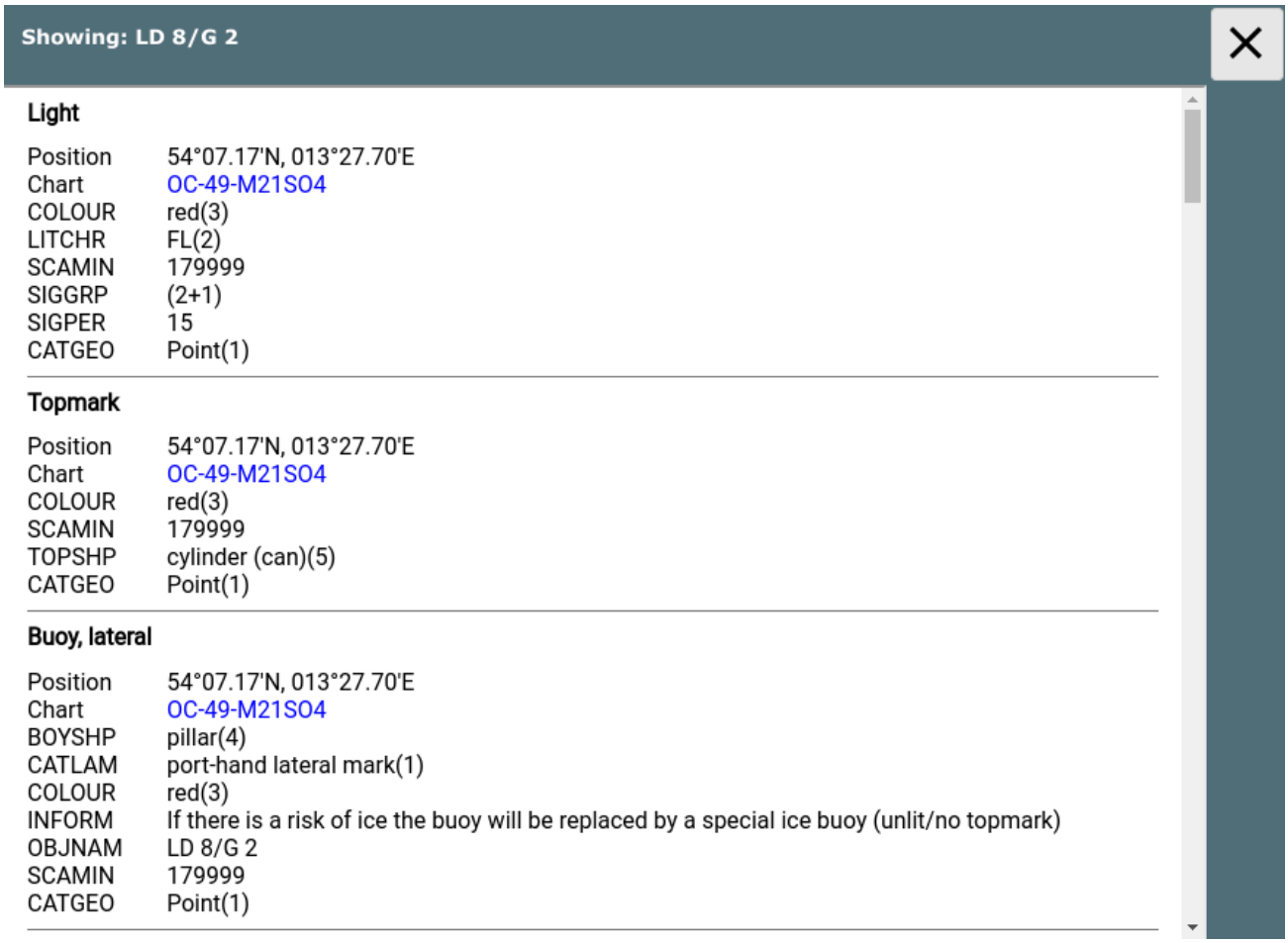
At the tab "Detail Settings" you can switch on/off particular chart features. Those detail settings will only become effective when using the display category "User Standard".

Feature Info (Object Query)

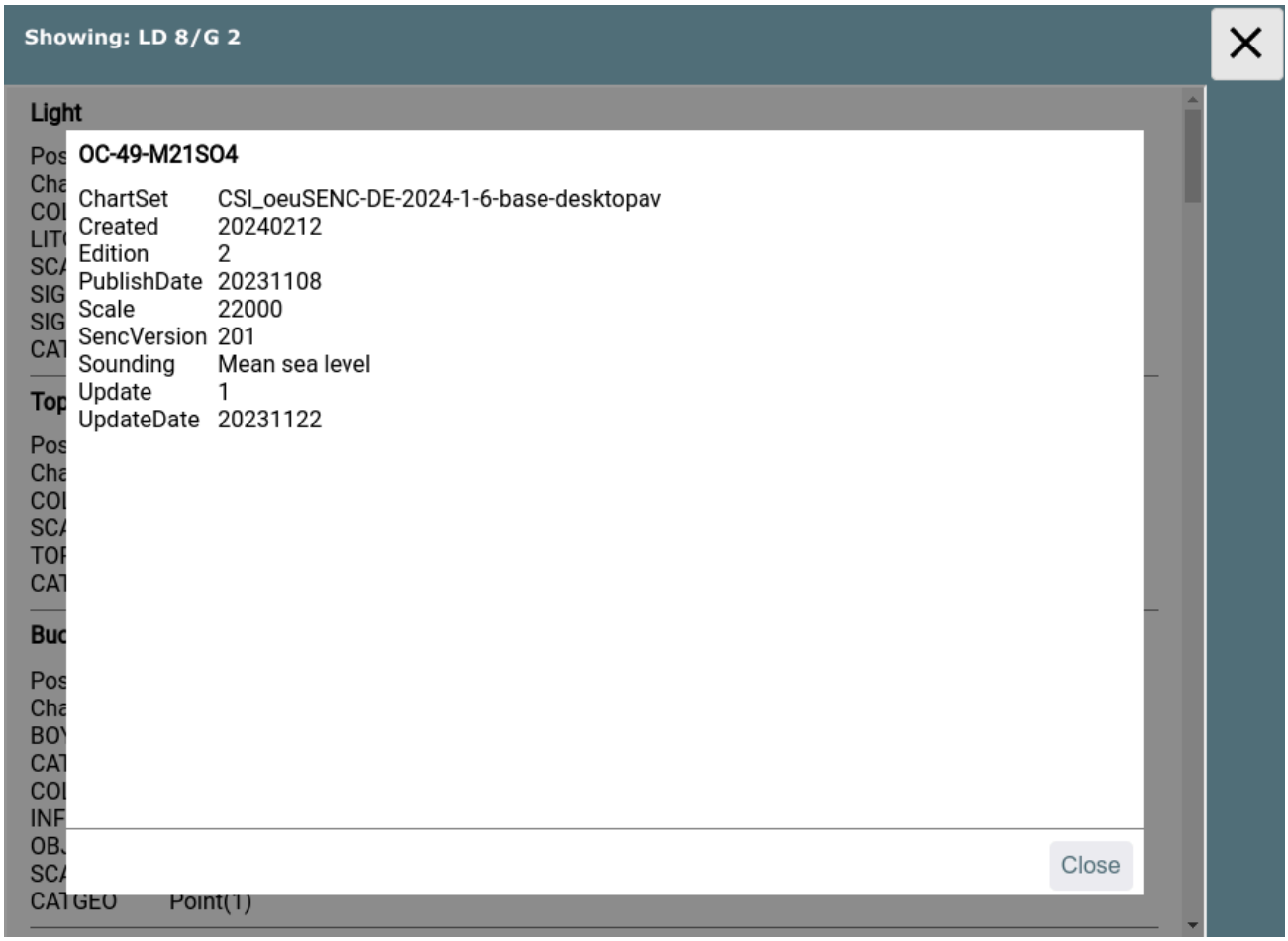
When you click on a chart you will have AvNav's Feature info. This will be extended by some important information found around the click position.



By clicking "Info" you can view the raw information from the chart.



By clicking the blue chart name you can get some more information about the chart that this object belongs to.



Installation

Linux/Raspberry

The plugin is available as a debian package.
As the plugin is currently in a beta phase, the packages are still not part of the official repositories.

For the AvNav Images the necessary packages can be found in the beta repository. To enable it you can edit the file /etc/apt/sources.list.d/extra.list to enable the beta repository.
Start the editor with:

```
sudo nano /etc/apt/sources.list.d/extra.list

#deb [signed-by=/usr/share/keyrings/oss.boating.gpg]
https://www.free-x.de/debpreview bullseye main contrib
non-free
```

```
    deb [signed-by=/usr/share/keyrings/oss.boating.gpg]
https://www.free-x.de/debian bullseye main contrib non-
free
```

Just remove the hash mark at the debpreview line. Afterwards you can install the package normally with the avnav updater:

- avnav-ochartsng

If you want to do this from the command line execute:

```
sudo apt-get update
sudo apt-get install avnav-ochartsng
sudo systemctl restart avnav
```

If you are working on other images you should add the repository from free-x (adapt "bullseye" to your debian release):

Refer to the AvNav package install description.

```
deb https://www.free-x.de/debpreview bullseye main
contrib non-free
```

Alternatively you can download the packages them from the daily builds directory.

To use one of those packages just download and install the package (replace the version by the one you want):

```
cd /home/pi/avnav
wget -O avnav-ochartsng_20240214-raspbian-
bullseye_armhf.deb
https://www.wellenvogel.net/software/avnav/downloads/daily
ochartsng/20240214/avnav-ochartsng_20240214-raspbian-
bullseye_armhf.deb
sudo apt install ./avnav-ochartsng_20240214-raspbian-
```

```
bullseye_armhf.deb
sudo systemctl restart avnav
```

You can also download to a PC, transfer by scp/WinScp to the pi and install then.

AvNav-ocharts and AvNav-ochartsng Together

When you install the new avnav-ochartsng-plugin you will get some conflict with the existing ocharts plugin. At the end both will try to use the same port - so at least one of the two will fail to start.
The suggested solution is to disable the ocharts-plugin on the  server/status page.
Additionally enable the "useLegacy" switch in the PluginConfig of the ochartsng plugin. This way the new plugin will also use all the charts that you installed with the old one.

If you would like to run both plugins in parallel change the port of the ochartsng plugin to e.g. 8083. Running both in parallel should only be done if your system has sufficient RAM (at least 2GB).

Android

On android there is an extra app that will contain the ochartsng.

See AndroidApp below.

Releases

You can find all releases and intermediate developer builds (daily builds) at:

- Releases
- Daily Builds

Release Versions

-- none yet --

License Notes

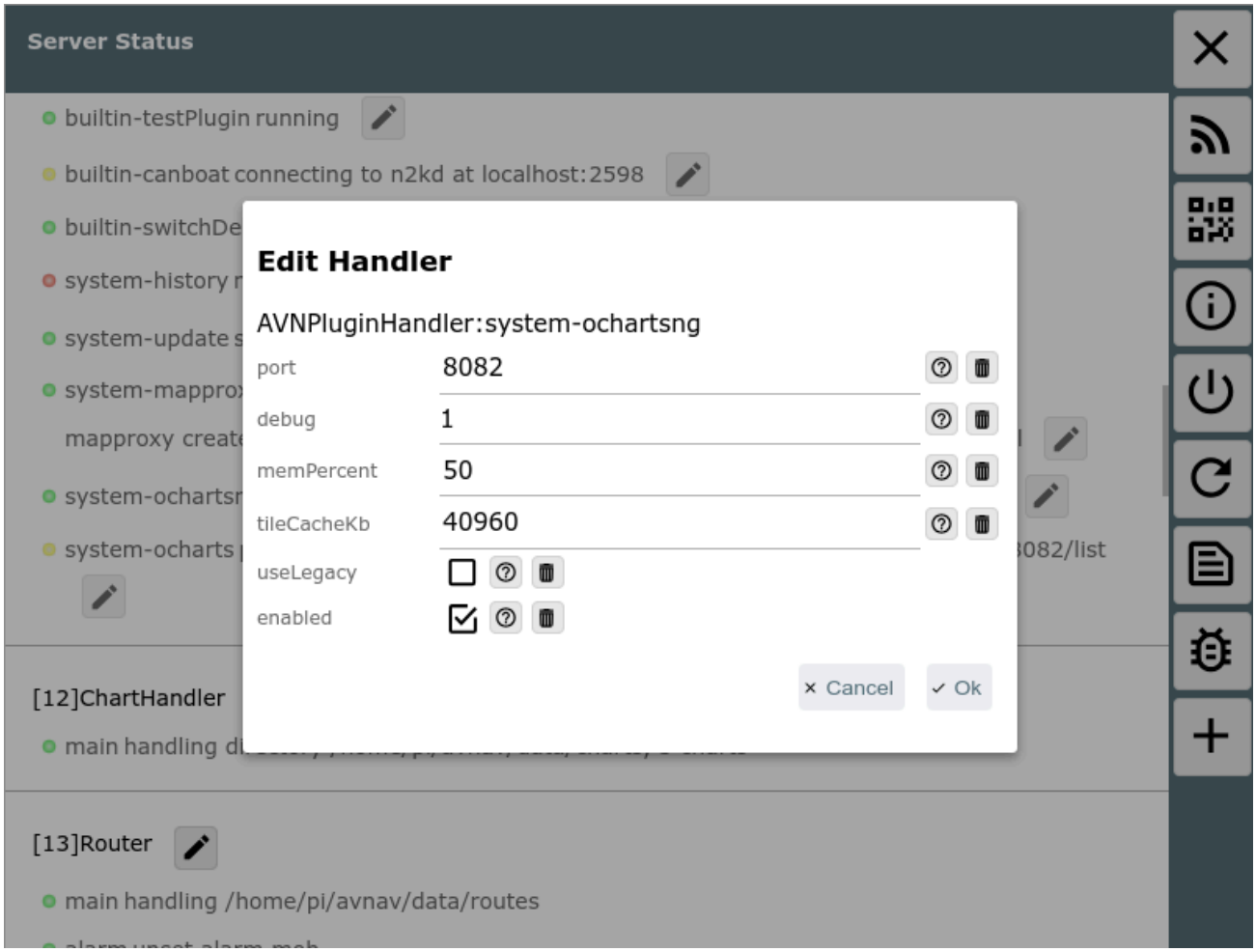
Using the charts in AvNav with the oesenc-pi plugin has been agreed with o-charts and therefore is inline with their license conditions.
You have to agree to the license conditions of o-charts
Especially it is not allowed to copy the charts or use them on other than the licensed systems.

Access to the charts in AvNav is only possible from within the local net. You can connect at most 5 devices (Clients) at the same time.

For software licenses see the Readme.

Plugin Configuration

Linux only. For Android refer to Android/Settings.
Some of the plugins settings can be changed on the server/status page  at "plugins/system-ochartsng" .



Those are:

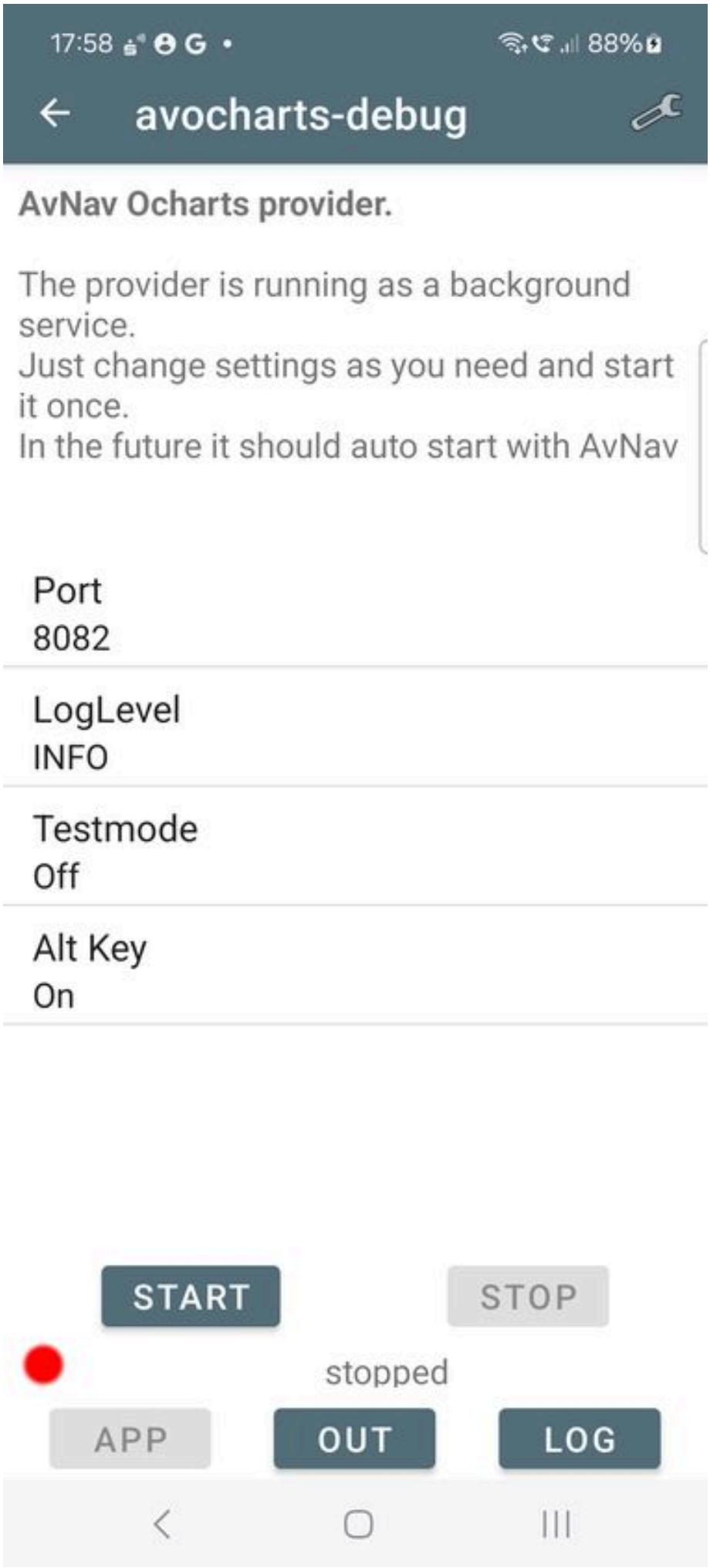
Name	Meaning	Default
port	Http port	8082
debug	log level for <datadir>/ocharts/provider.log. <datadir> on a raspberry is /home/pi/avnav/data	1
memPercent	The amount of memory (percent of the system memory) that the plugin (correctly: the provider process) will use. If this value is not set (or set to low) the provider will use an internal minimum. This potentially could be rather low - especially when using raster charts. If it is very low the	50

	provider must open and close chart files very often and this will slow down it's operation. If you have enough memory (e.g. 2GB) you can increase speed by setting the memory to 1GB .	
tileCachekb	KB of memory to be used for the internal tile cache (set to 0 to disable it for some testing)	40960
useLegacy	Also use the charts that have been installed with the old avnav-ocharts-plugin. By activating this you can disable the old plugin in the settings and just use all the charts you already installed.	off

Android App

AvNav OchartsNG also runs on Android and provides charts for the AvNav android app.
The Ocharts support is built into a separate Android App - avocharts.

Currently the app is not (yet) available in the google play store as it is still in beta status. So you need to download it from the daily builds directory and install it (allow external sources).

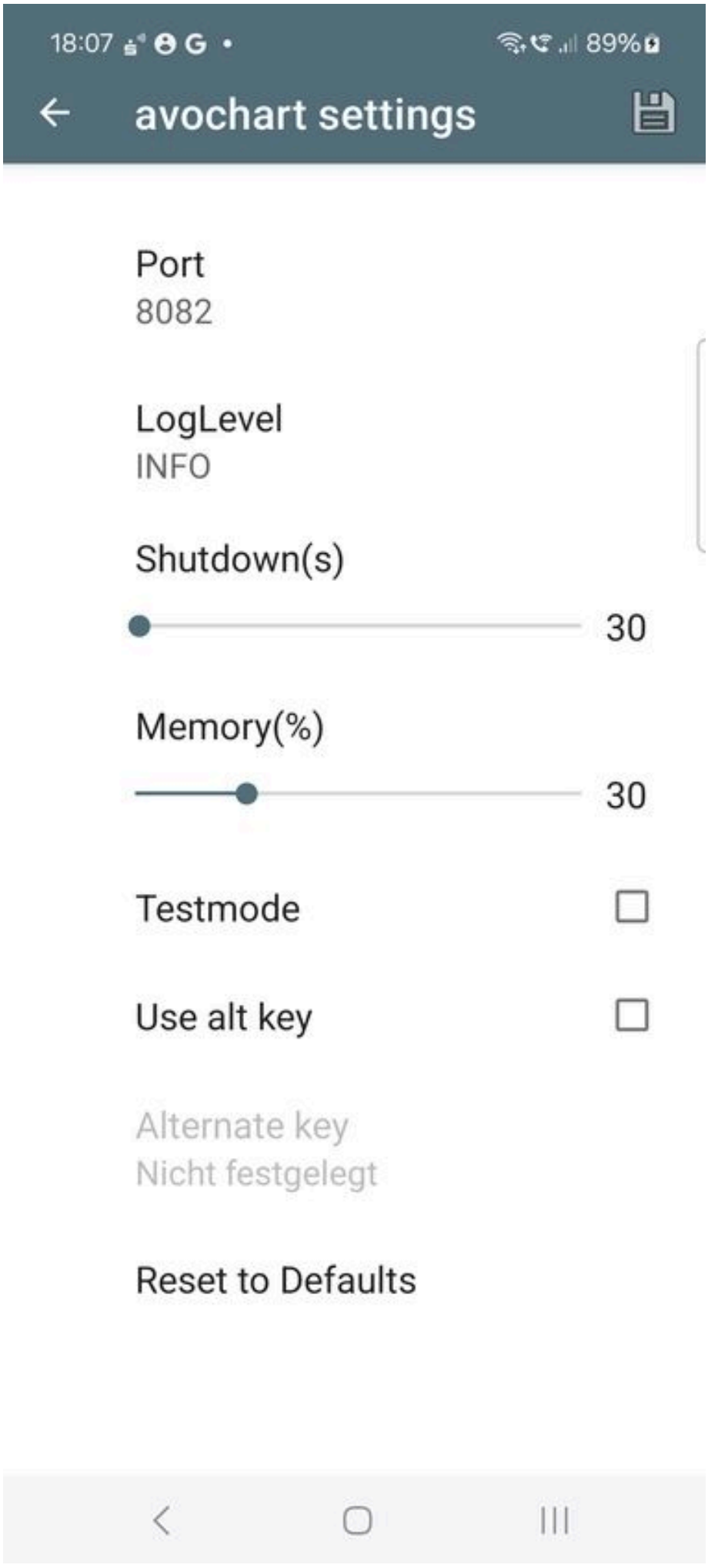

17:58
88%
avocharts-debug
AvNav Ocharts provider.
The provider is running as a background service.
Just change settings as you need and start it once.
In the future it should auto start with AvNav
Port
8082
LogLevel
INFO
Testmode
Off
Alt Key
On
START
STOP
stopped
APP
OUT
LOG

After you started up avocharts for the first time you can adapt some settings. When you see this picture the real ocharts provider process is still not running - it will start up when you tap "start".

Later it will not be necessary to start the avocharts app by hand - it should normally being started automatically when AvNav starts up. But directly starting the avocharts app can help to troubleshoot as you can get access to the output and log of the chart provider.
Please consider that the auto start will only work for apps of the same type - i.e. an AvNav release app can only start an avocharts release app and an AvNav beta app can only start an avocharts beta app.
So when you install the avocharts beta together with the "normal" AvNav app from the store you need to start avocharts by hand before you can use the charts in AvNav.

Settings

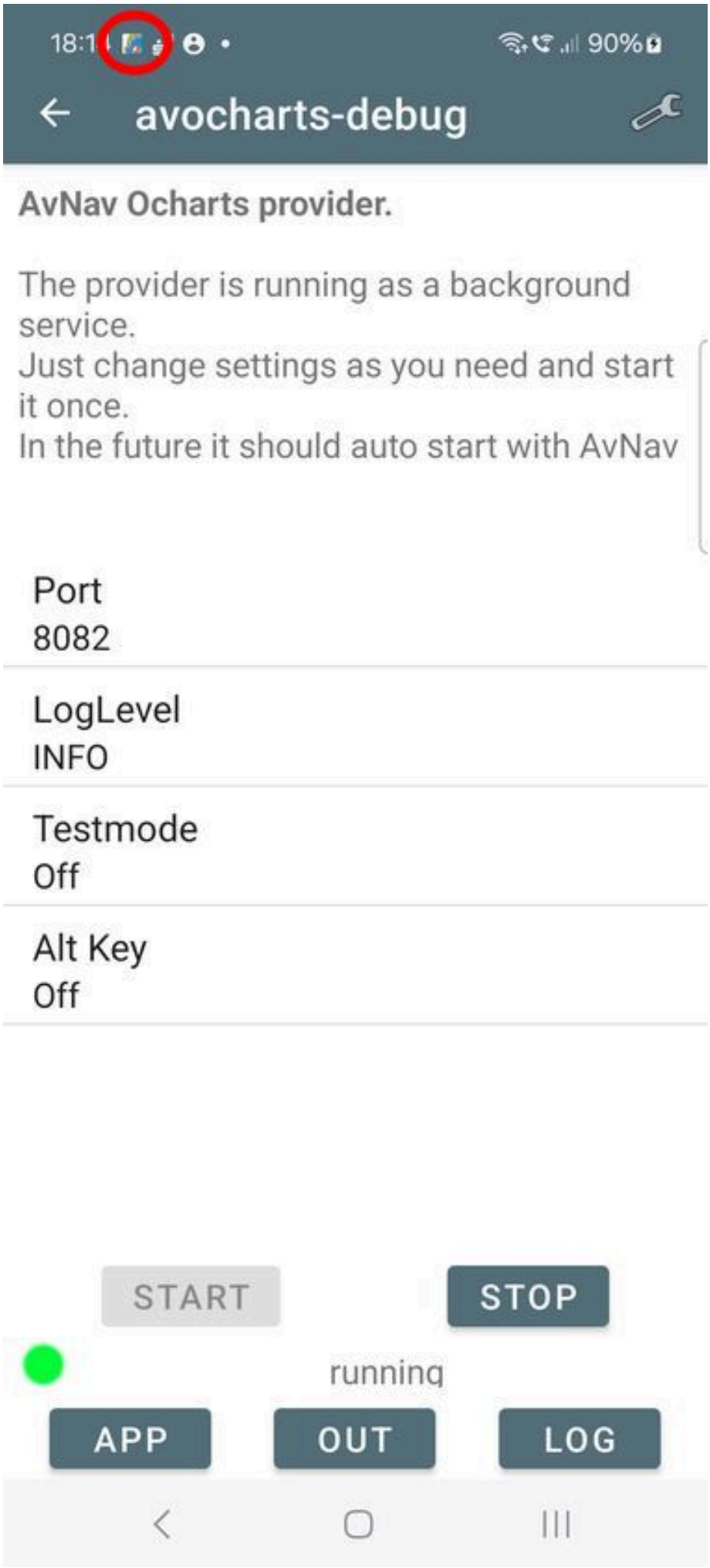
To adapt settings tap on the shown settings or on the settings icon on the upper right.


18:07
89%
avochart settings
Port
8082
LogLevel
INFO
Shutdown(s)
30
Memory(%)
30
Testmode
Use alt key
Alternate key
Nicht festgelegt
Reset to Defaults

Name	Explanation	Default
Port	The IP Port the provider process will use. If you get errors when you start the provider and see something like "address already in use" in the output (tapping the out button on the main page) you maybe need to change the port	8082
LogLevel	0 - errors, 1- info, 2- debug	1
Shutdown	Time (in seconds) the provider will wait for a heartbeat from AvNav before it stops (only active when started by AvNav)	30
Memory	Allowed Memory usage (%) of the system memory	50
Testmode	internal testing only	false
Use alt key	See below for the hints about Release/Beta Versions	
Disk symbol (upper right)	Save your key (See below for the hints about Release/Beta Versions)	

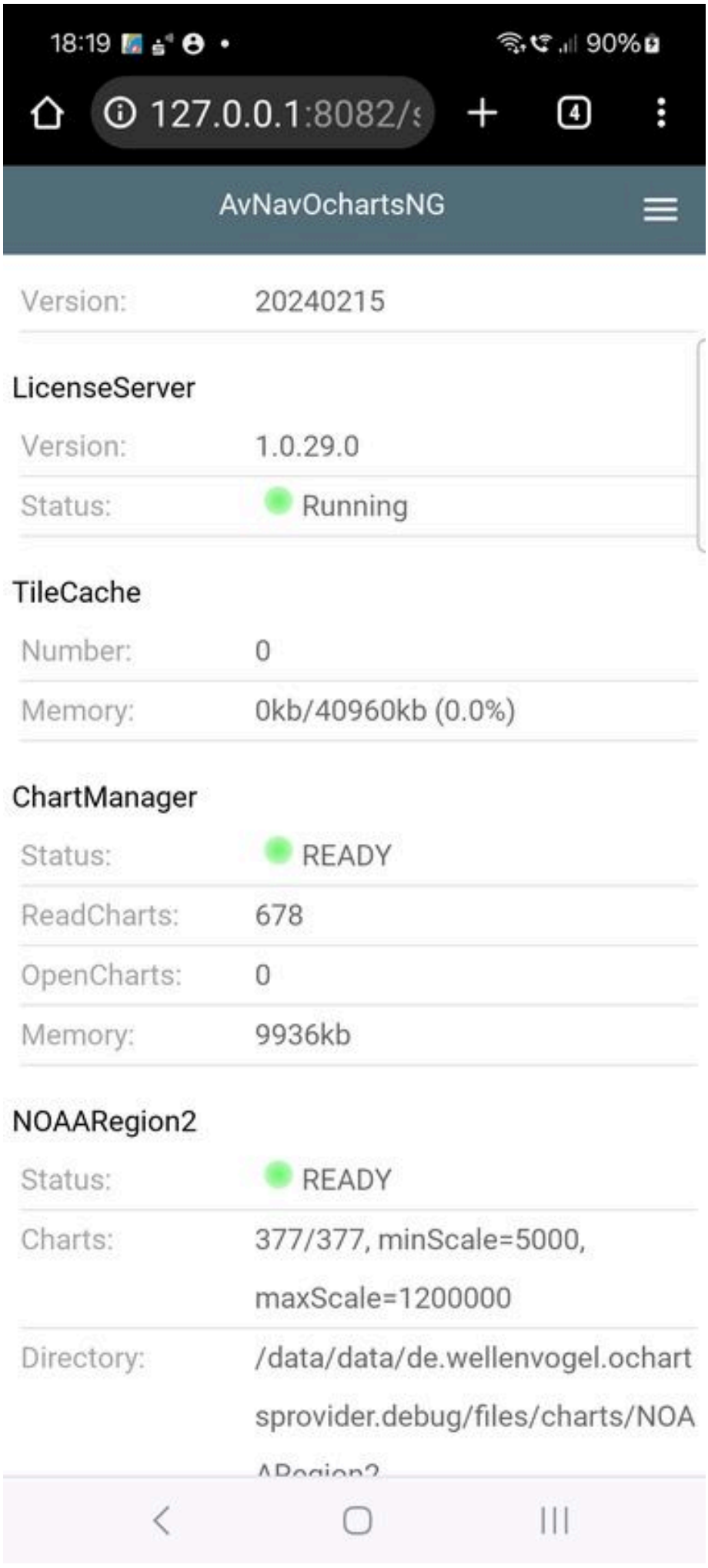
When done with the settings tap the back arrow.

You can now start the chart provider by tapping "START".


18:1
90%
avocharts-debug
AvNav Ocharts provider.
The provider is running as a background service.
Just change settings as you need and start it once.
In the future it should auto start with AvNav
Port
8082
LogLevel
INFO
Testmode
Off
Alt Key
Off
START
STOP
running
APP
OUT
LOG

When the provider process is up an running you will see the indication in the notification bar. If something goes wrong just try to have a look at the output (tapping "OUT") or the log (tapping "LOG").

Tapping "APP" will open up a browser window with the ochartsng app - the same view like you get from within AvNav.


18:19
90%
127.0.0.1:8082/
AvNavOchartsNG
Version: 20240215
LicenseServer
Version: 1.0.29.0
Status: Running
TileCache
Number: 0
Memory: 0kb/40960kb (0.0%)
ChartManager
Status: READY
ReadCharts: 678
OpenCharts: 0
Memory: 9936kb
NOAARegion2
Status: READY
Charts: 377/377, minScale=5000, maxScale=1200000
Directory: /data/data/de.wellenvogel.ochartsprovider.debug/files/charts/NOAARegion2

You now can e.g. register at the ocharts shop (on android this is only possible using the Online Process).

Beta and Release Versions

Due to the way that the ocharts license daemon is working you will normally not be able to use charts that you installed with the release version within the beta version and vice versa.
To overcome this problem you can export the necessary information from the app version that you used to assign the charts and import this in the different version. Hint: You still can only use this on the same android device!
To export a key tap the disk symbol on the settings screen within the app and store the file at a safe place. When you later use the alternative version, select "use alternate key" , tap on the key value and import the saved key file there.
You can easily check if your app has the correct key by going to the shop page and log in. When the shop recognizes your current key it will show the system name - see Online Process.

Please also consider the automatic startup relations between the Beta and Release version.

Technical Details

The charts are provided by an executable that normally serves at port 8082. Communication with AvNav is handled by an AvNav plugin on linux and by the avocharts app on Android.

The GUI is a reactjs app that is also provided by the executable. It is integrated into AvNav as a User App.

You will find the complete code at GitHub.

You install into /usr/lib/avnav/plugins/ochartsng. The data directory is /home/pi/avnav/data/ochartsng. You can set some additional parameters for

the AvNav plugin by editing avnav_server.xml. Normally this is not necessary at all.

You can separately set the directory for all settings and the charts:

```
<AVNPluginHandler>
...
<system-ochartsng dataDir="$DATADIR/ochartsng"/>
</AVNPluginHandler>
```

Chart Conversions

Unencrypted vector charts like S57 charts have to be converted into the internally used "SENC" format before they can be handled.
There are 2 Options to convert such charts.

Converting with AvNav (since AvNav 20240520)

The avnav-ochartsng-plugin adds a chart converter to AvNav's importer. This converter contains a python script to convert S57 data into a zip archive that can be uploaded.
To convert your S57 charts simply upload a zip with the charts to AvNav's importer after you installed the plugin.
If you run AvNav and ochartsng on linux the converted charts will be automatically uploaded and activated to ochartsng.

If you run ochartsng on a different system (e.g. on Android) you need an installation of AvNav either on a linux machine or on Windows.

Ochartsng itself cannot run on Windows but the conversion tooling can. So after you installed AvNav on Windows just open the "Update" Dialog and enter the Url of the ochartsng-s57 converter for Windows:

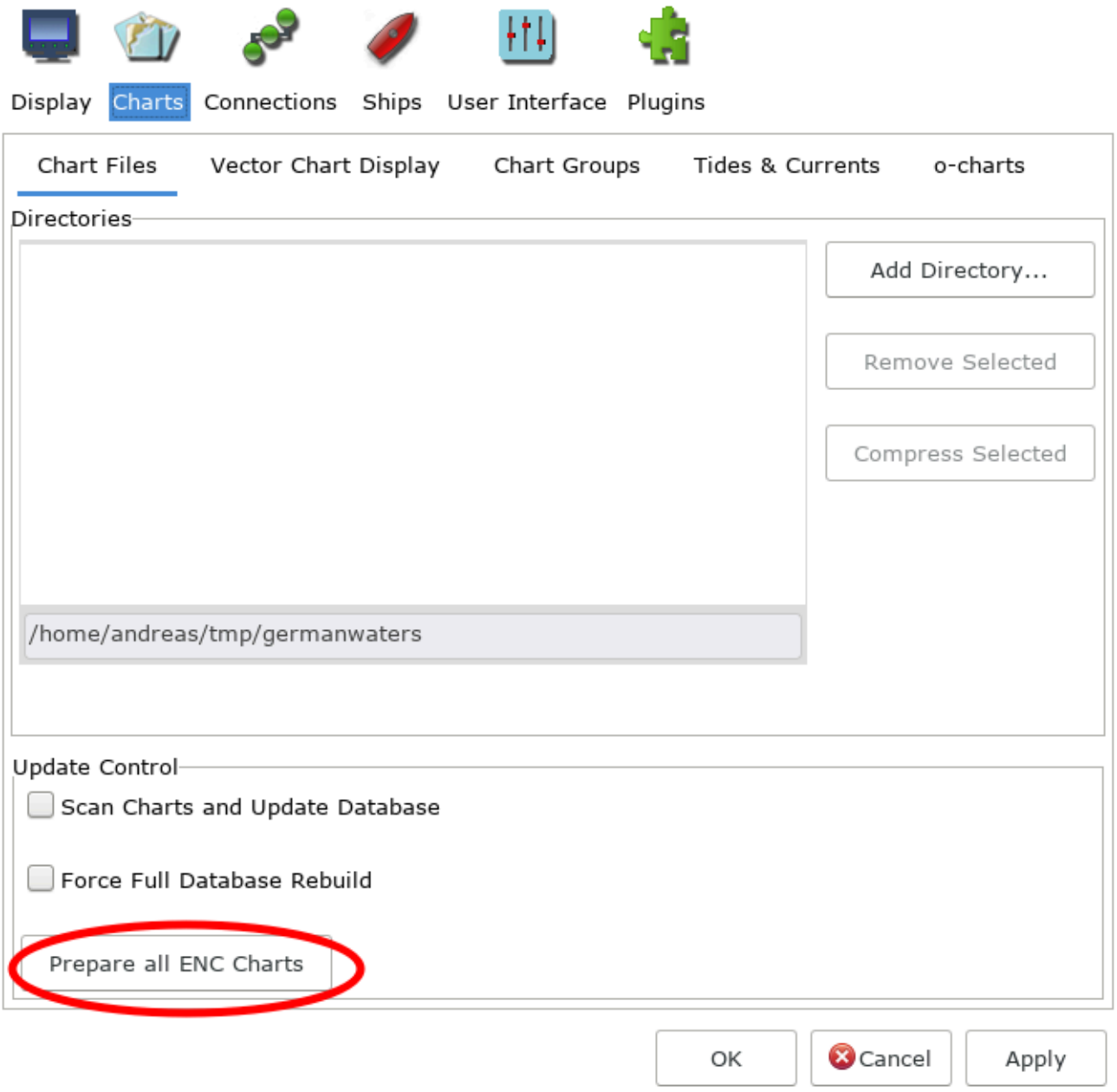
- release: https://www.wellenvogel.de/software/avnav/downloads/release-ochartsng/latest/ochartsng-latest.zip

- daily: https://www.wellenvogel.de/software/avnav/downloads/daily-ochartsng/latest/ochartsng-latest.zip

Afterwards let the conversion run and when finished download the converted chart (.zip). This zip can directly be uploaded to ochartsng both on linux and Android.

Converting using OpenCPN

If you already have a computer with OpenCPN up and running you can let OpenCPN convert the charts. To do so you need to activate the charts in OpenCPN. On the OpenCPN settings / tab "chart files" just click "Prepare all ENC Charts".



This will create a lot of files with the s57 extension in the OpenCPN directory (~/.opencpn on linux). Just pick the ones with similar names like your s57 charts, copy them in a new, empty directory with a name that explains the chart set. Afterwards create a text file "Chartinfo.txt" in this directory with just one line of text:

```
ChartInfo: nameOfTheChart
```

Create a zip archive from this directory so that the zip archive has this directory inside. This archive can be uploaded to AvNav ochartsng.
On Linux there is a script to automate this collection process.

Avnav Ocharts

Hint: since 2024/02/17 there is a new (Beta) version for the o-charts support. This new version also works on android. For a documentation refer to OchartsNG.

=== not for Android ===

Inhalt

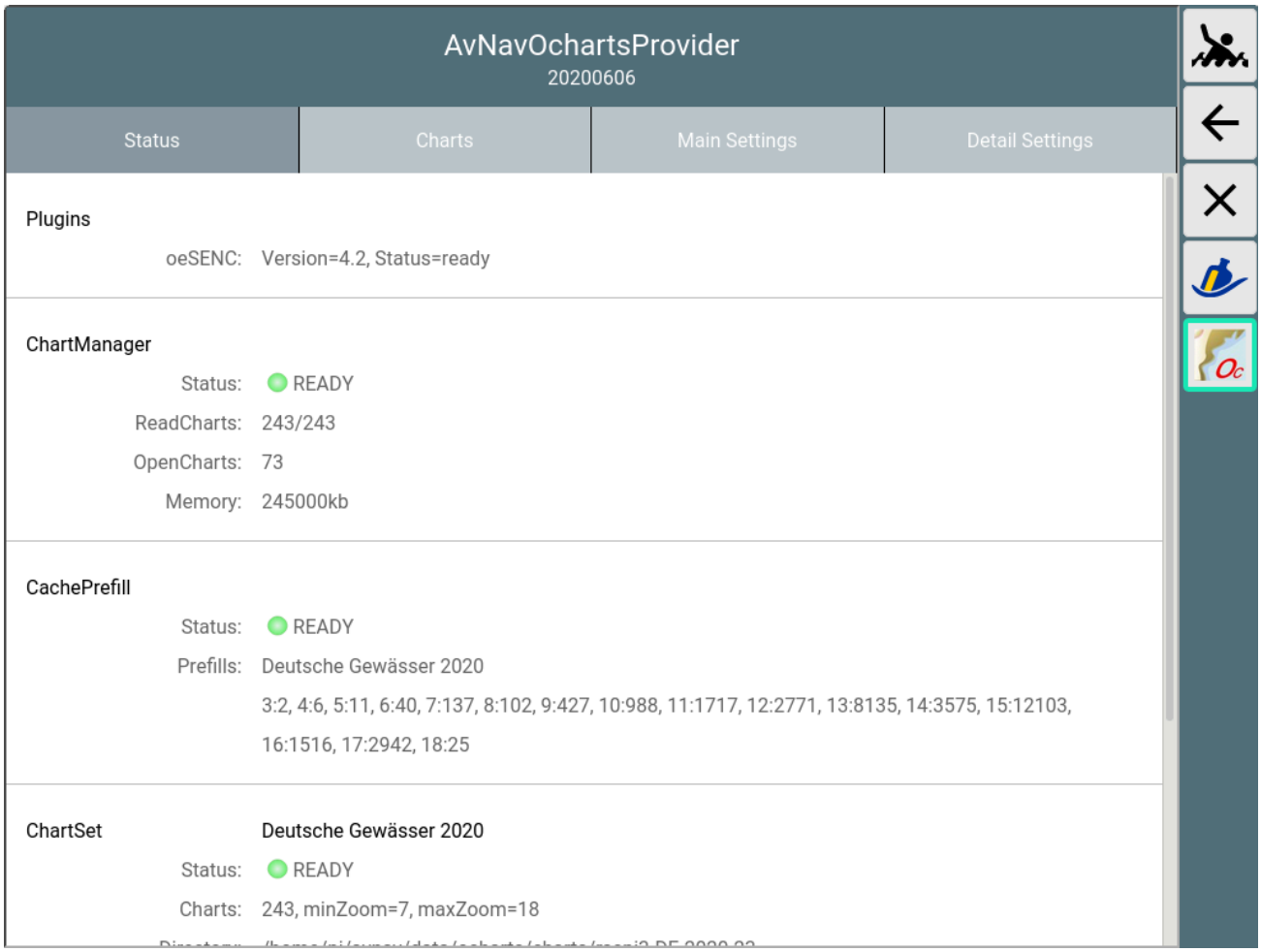
AvNav is able to handle charts in various raster formats. So far it was unable to handle any commercial charts. The o-charts company supplies charts for various regions of the world for usage in OpenCPN.

After some agreements with o-charts, those charts can now also be used for AvNav (starting with version 20200515 with a plugin - see below). At present you can use oesenc vector-charts and beginning from version 20220225 (and the related change in the o-charts shop - see releases below) you can also use oeRNC raster charts.

To display those charts in AvNav they must be rendered into raster images. This is handled by a new plugin for AvNav(avnav-ocharts). The rendering takes place on the fly whenever tiles are to be displayed. However, major areas of the chart will be prerendered to cached tiles upon chart installation.

This way those charts can be used the normal way - without worring about the rendering process.

All oeSENC charts handling is performed by the plugin - including installation (you cannot install them directly via the download page). For this purpose a dedicated GUI is offered by the plugin. You enter it from the main page via (User Apps) and Ocharts-Provider .



Buying and Installing the Charts

Important Hint: If you do not have a dongle from o-charts, your chart license is bound to your system. So in case of any trouble **do not** set up a new system (by writing an image to an SD card) but instead try to repair the system. If you set up a new system you will lose your license. I will by happy to support in case of trouble - contact e.g. via email.

To be able to buy charts at o-charts you have to create an account at their site first.

Afterwards you have to register the systems you would like to buy the charts for there. AvNav is using the "Offline" process .

This process consists of the following steps:

1. Create a "fingerprint" for the system you would like to use the charts at. In AvNav you will create it in the GUI for the plugin and download from there.
2. Upload the fingerprint to o-charts and create a system (basically assigning a name).
3. Buy charts
4. Assign charts to the system you created
5. After a short time you will receive an email from o-charts with a download link (zip file).
6. Upload charts to AvNav (via the plugin GUI).

For updates repeat steps 4, 5 and 6 (only requesting the notification mail at step 4)

For further chart sets steps 3-6.

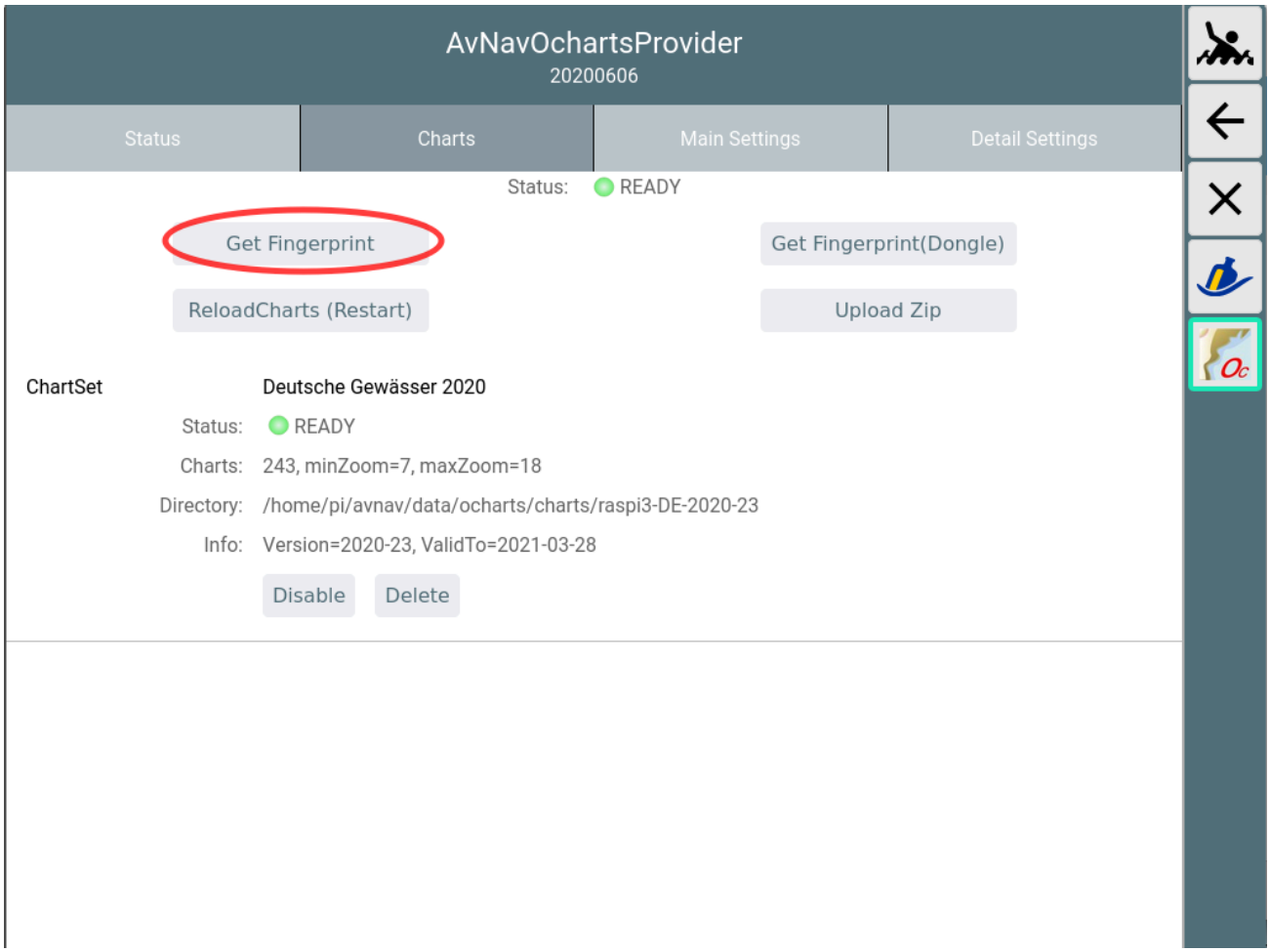
For steps 2,3,4 and 5 you need a system with internet connectivity. You can e.g. use a laptop or an android device.

I created a video to demonstrate this process. Additionally here is a short description.

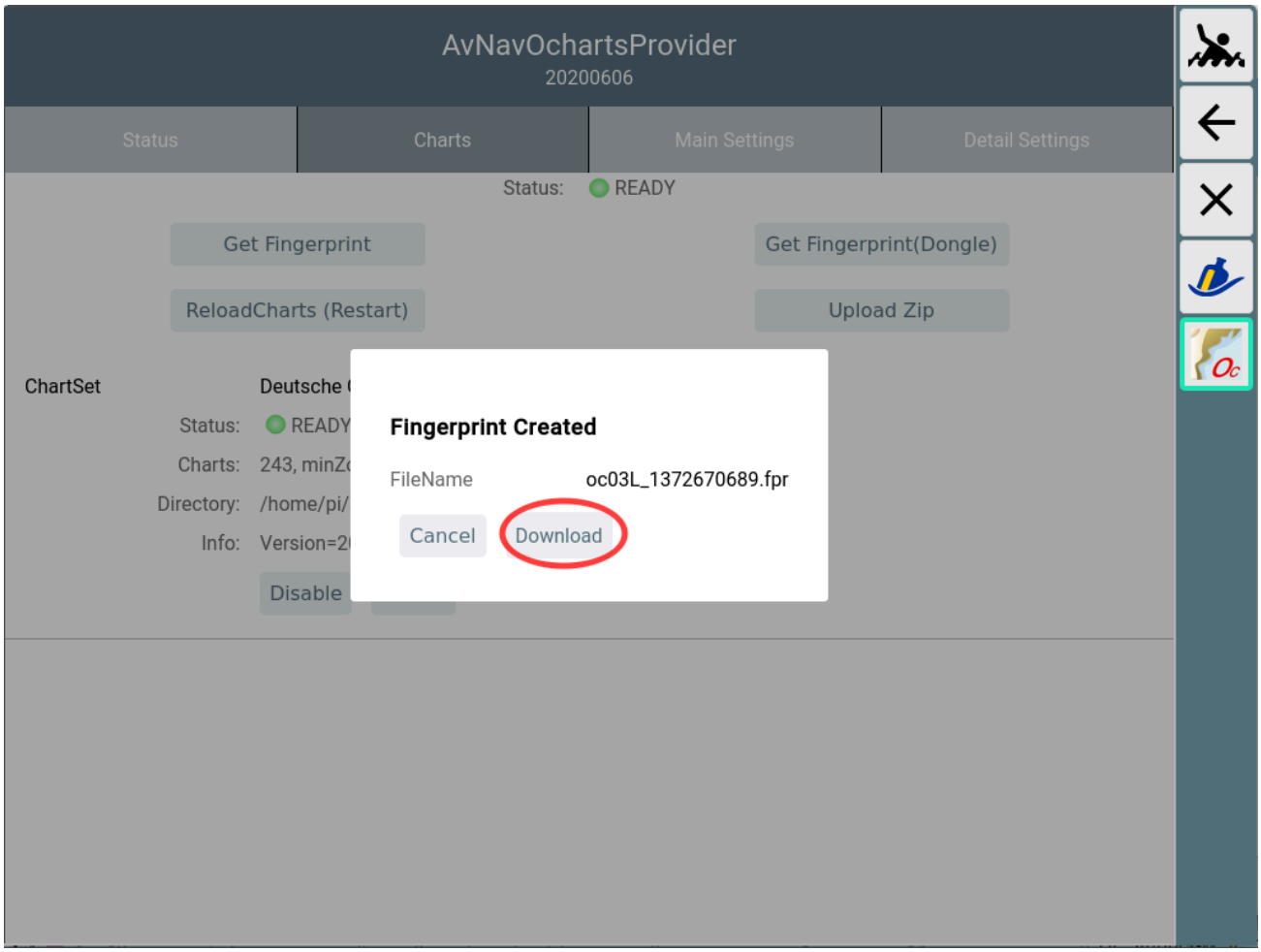
Hint: If the charts are already registered on the same system (for OpenCPN) you can directly continue at step 6. Alternatively you could also configure to access the OpenCPN chart directories at the plugin .

1. Creating the fingerprint

Via  ->  you enter the GUI of the plugin, select the "Charts" tab.



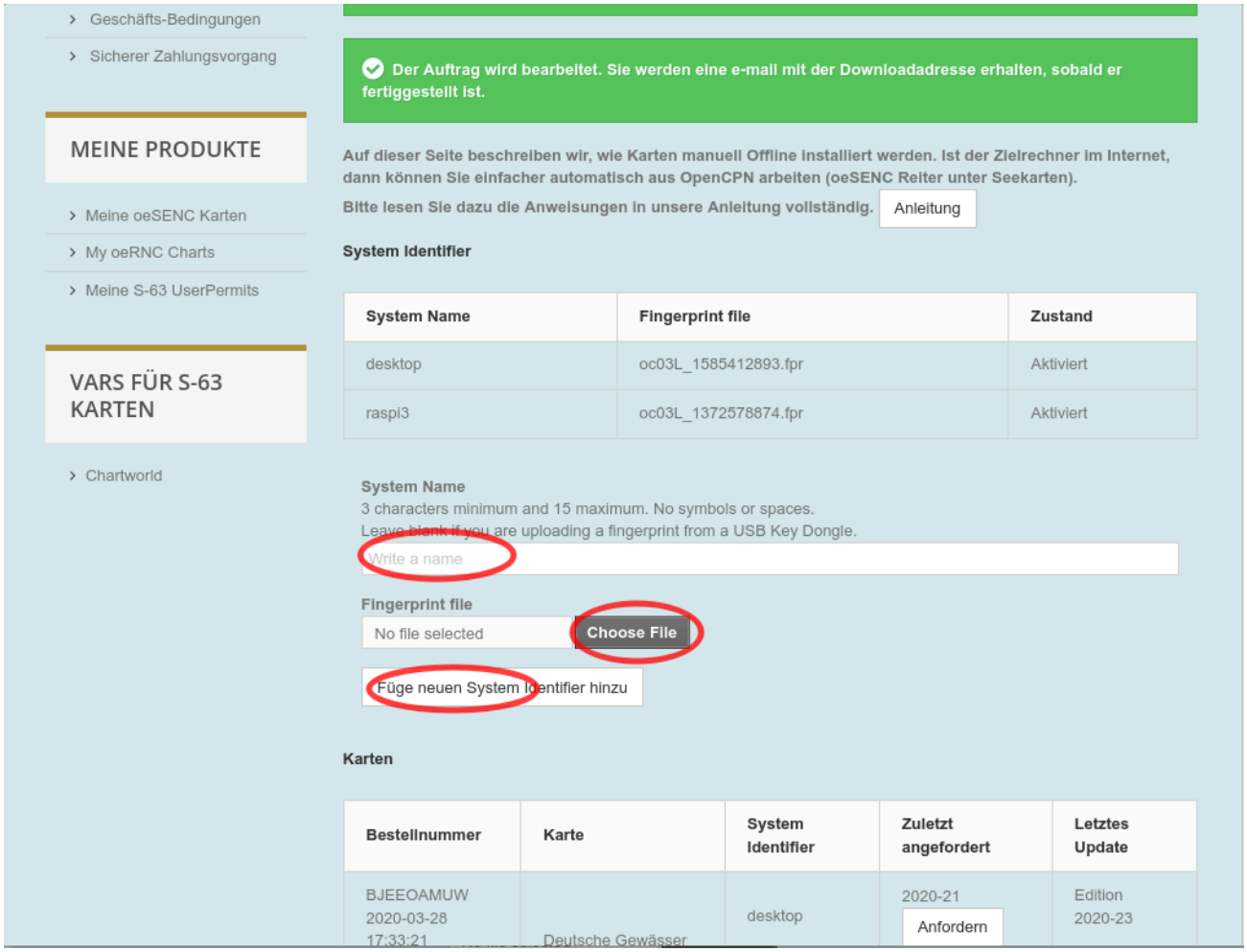
Use "Get Fingerprint" to create the fingerprint file. If you are using an o-charts dongle - just select "Get Fingerprint(Dongle)".



Choose Download to save the created file on your device.

2. Uploading the fingerprints to o-charts

Enter the o-charts page and upload the fingerprint.

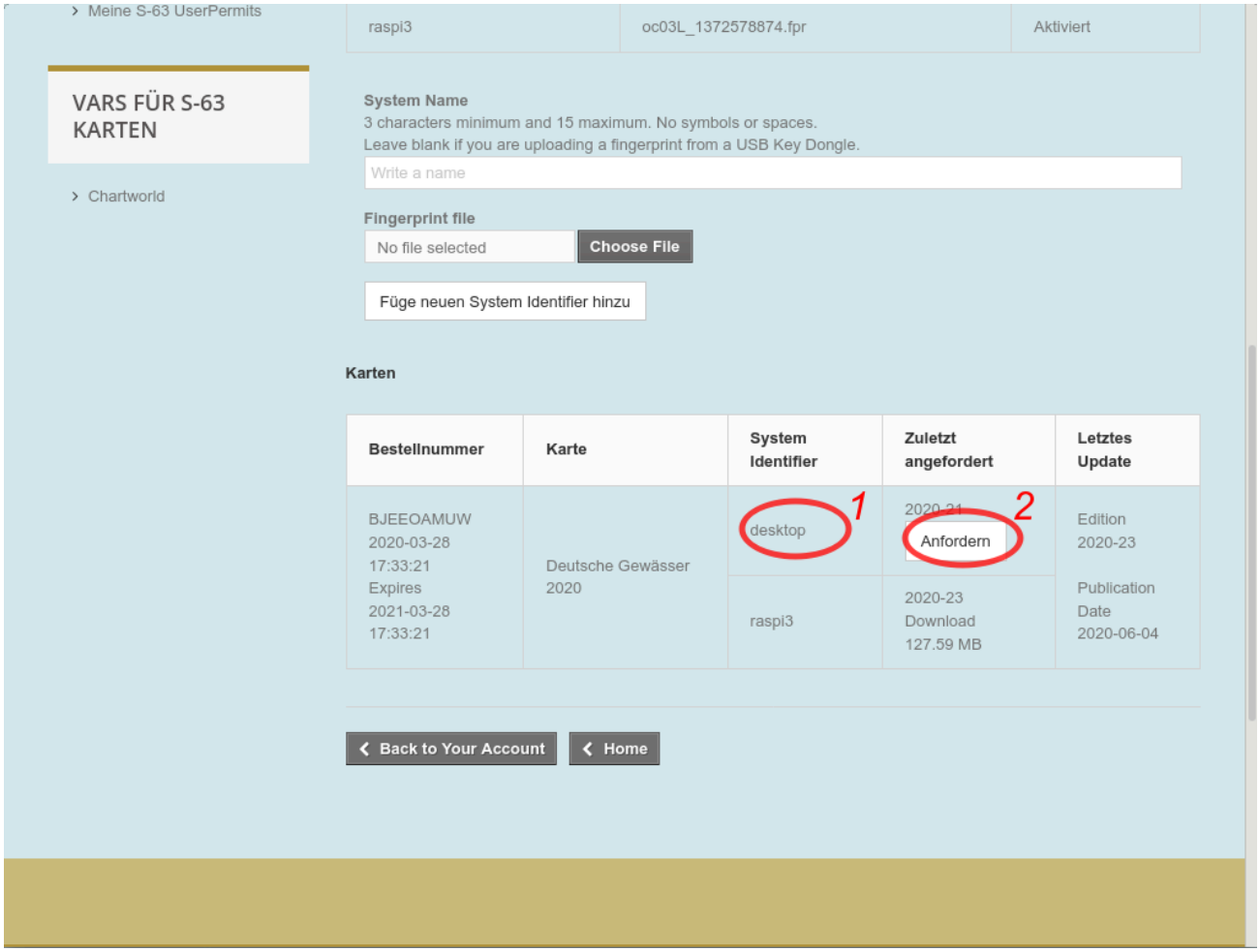


With "Choose File" you select the file stored at step 2. Assign a meaningful name to the new system - this will be part of the mails you will receive later.

3. Buying the Charts

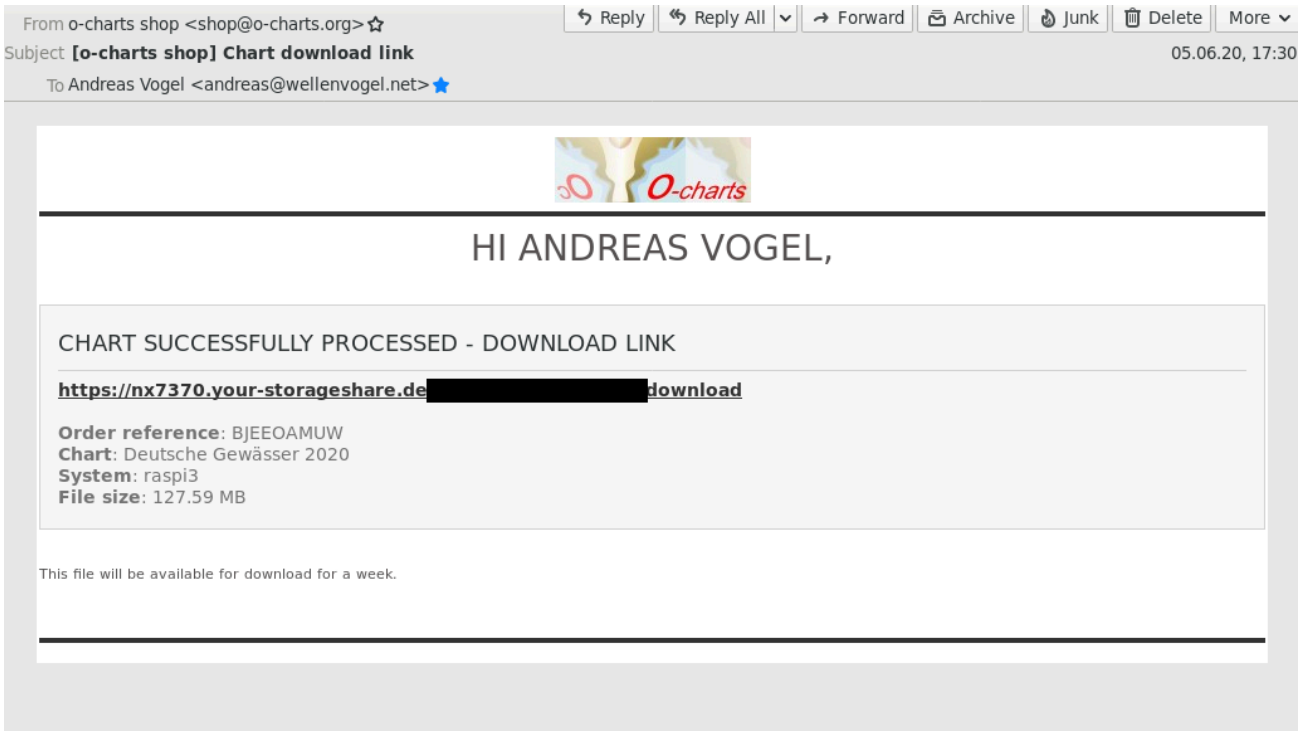
Select the desired charts from oeSENC charts.

4. Assigning to your System



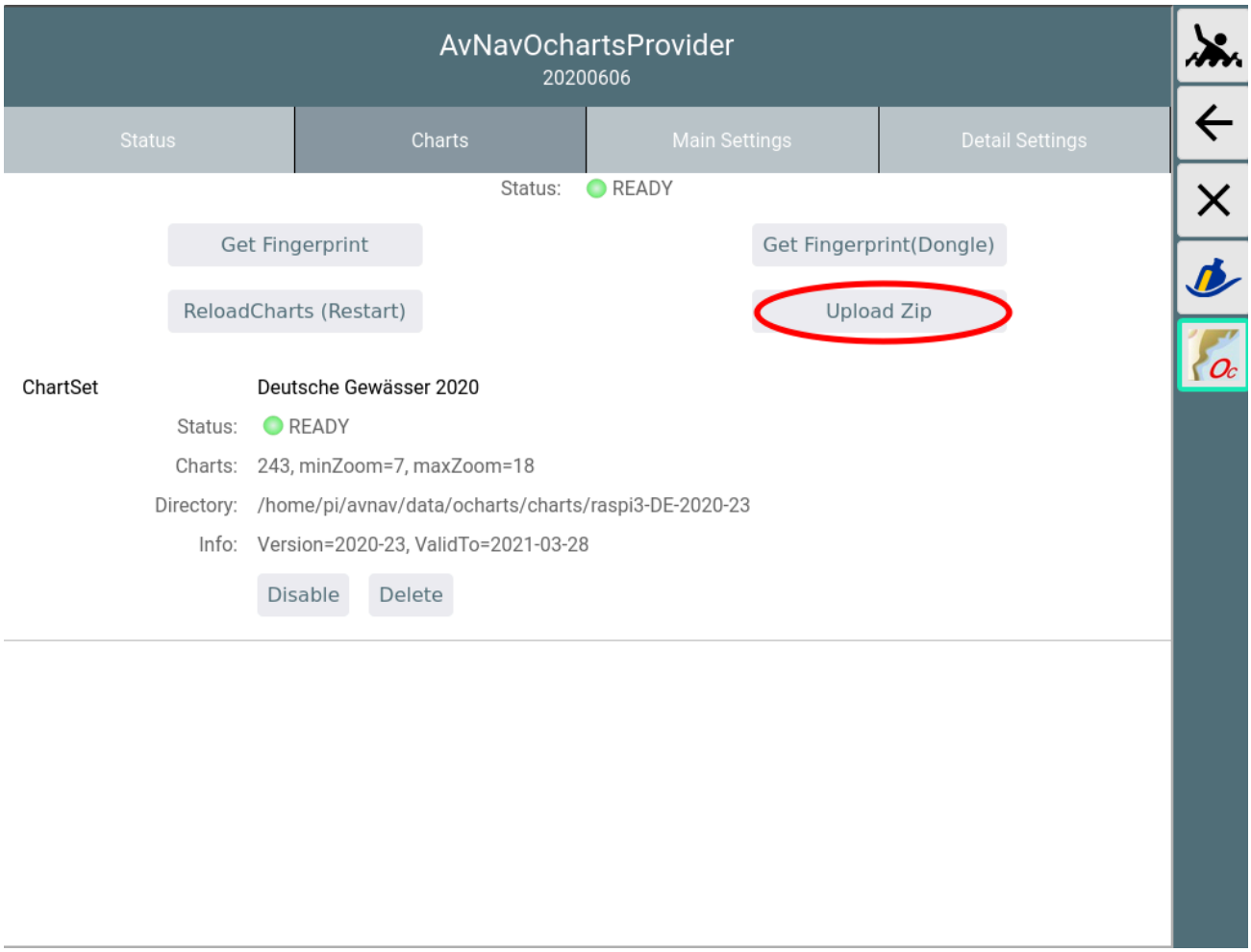
At 1 you can assign charts to your system (in this screenshot this is not available anymore as the max amount of 2 systems are already assigned). At 2 you request the mail holding the download link (you would do the same for updates - in the screenshot: last version I have downloaded is 21, latest available is 23)

5. Downloading the Charts



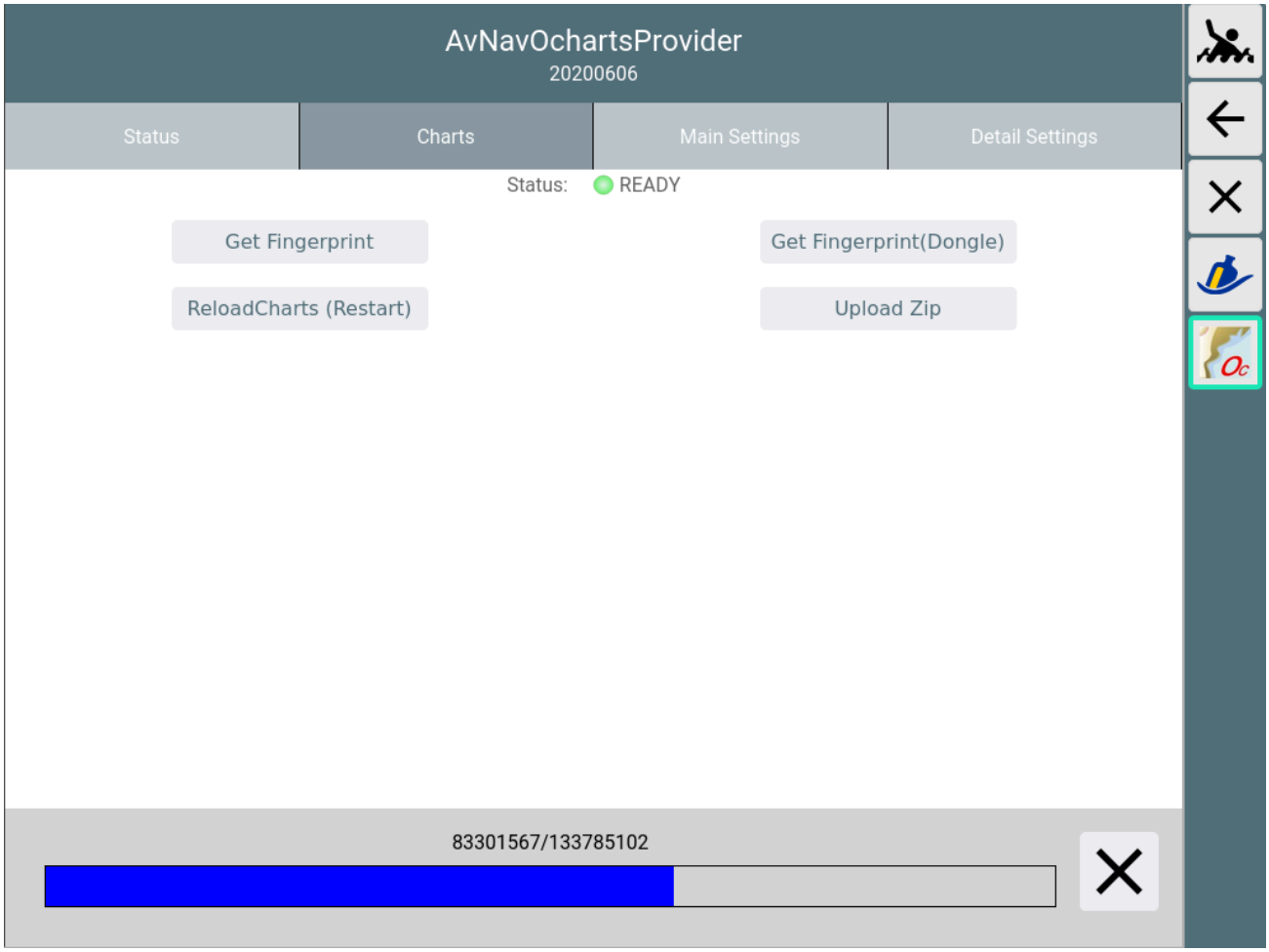
After a short time you will receive a mail containing the download link for your charts. Download the zip file.

6. Uploading the Zip File to AvNav



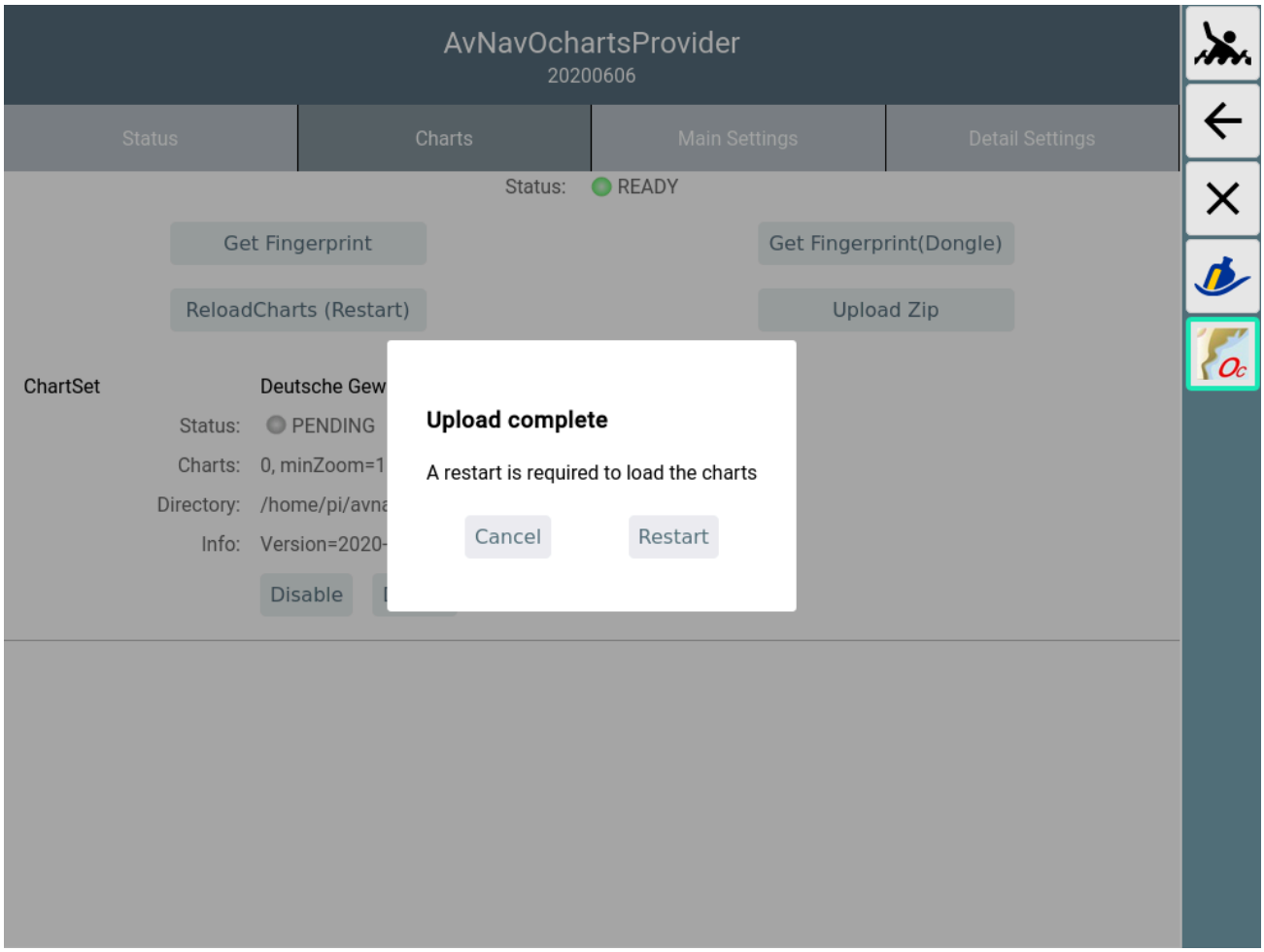
In the plugin's GUI select "Upload Zip" to transfer the zip file downloaded in step 5 to AvNav.

There will be a progress bar during upload.



After the upload is completed the zip file will be unpacked. Some initial checks can take some time.

Unless configured otherwise, the charts will be uploaded to /home/pi/avnav/data/ocharts/charts.

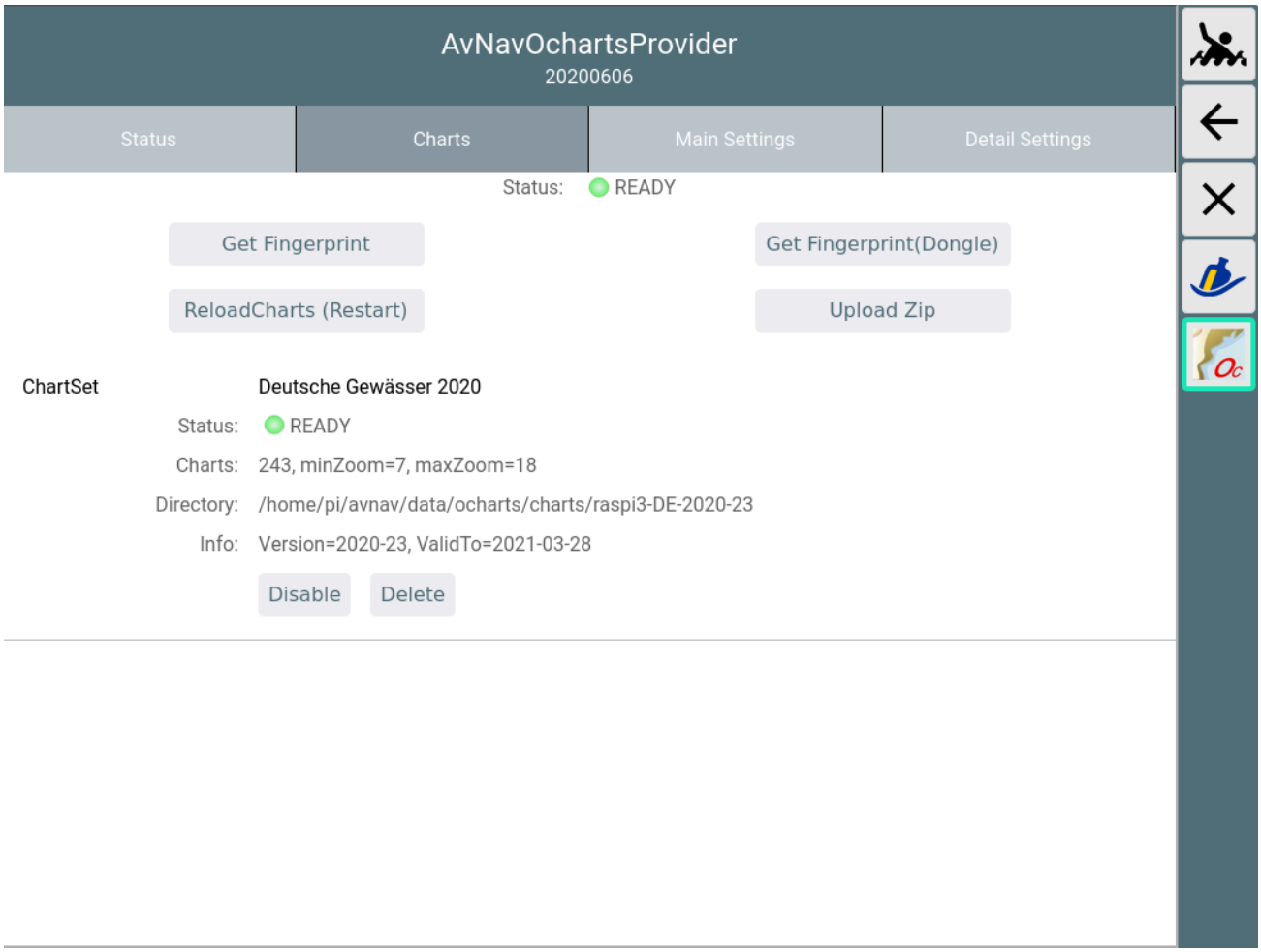


After all checks have passed successfully, the dialog will ask to restart the plugin to make the charts usable.

If the newly uploaded chart contains updates to an already existing set, the existing set will be deactivated at restart. You may change this later on within the GUI. Chart sets not required anymore can be deleted.

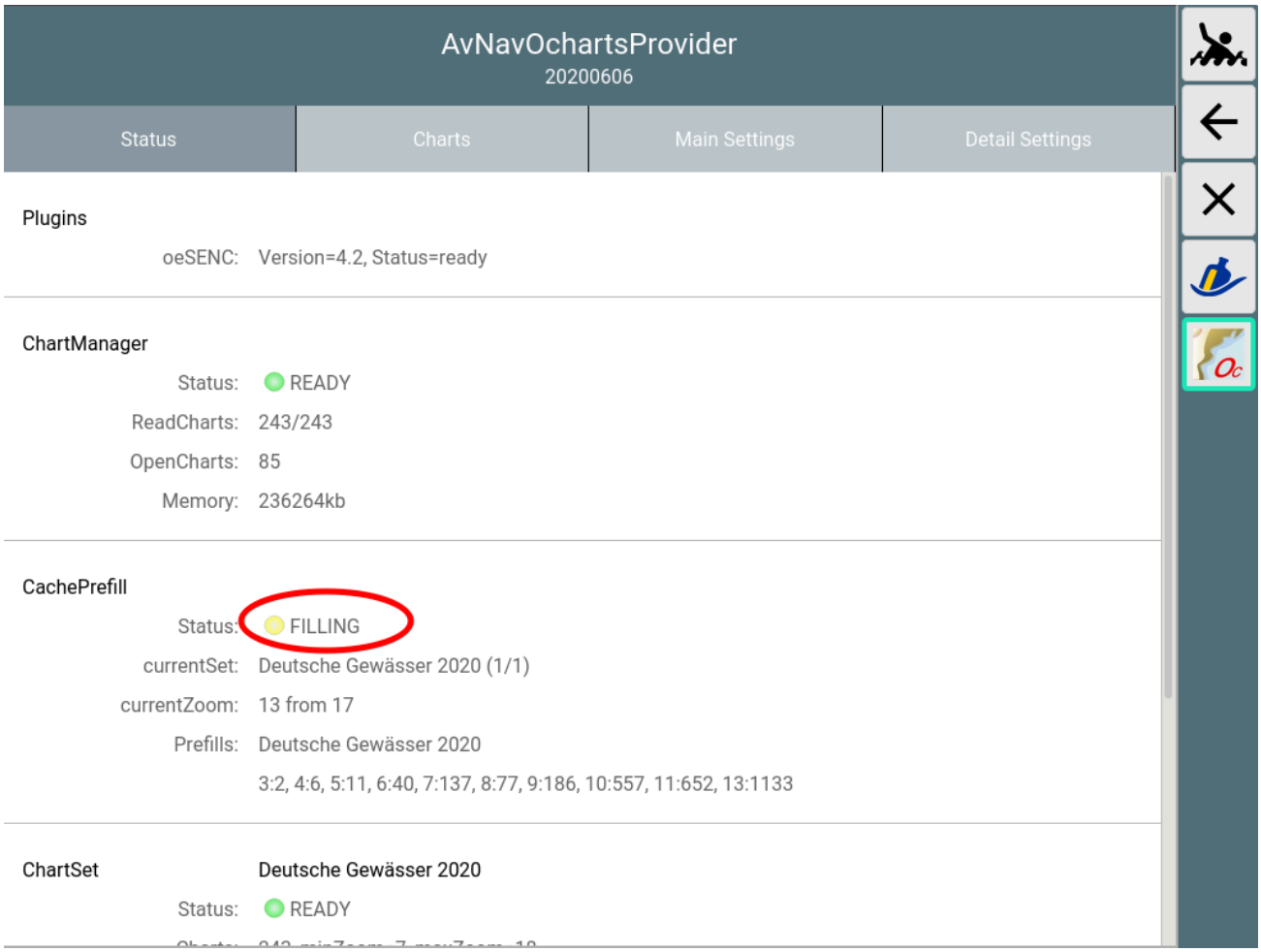
During restart a couple of error messages will be displayed briefly but within 30s the status should be at least set to yellow (the plugin is now reading all charts).

After all charts are successfully read the status should change to green (READY).



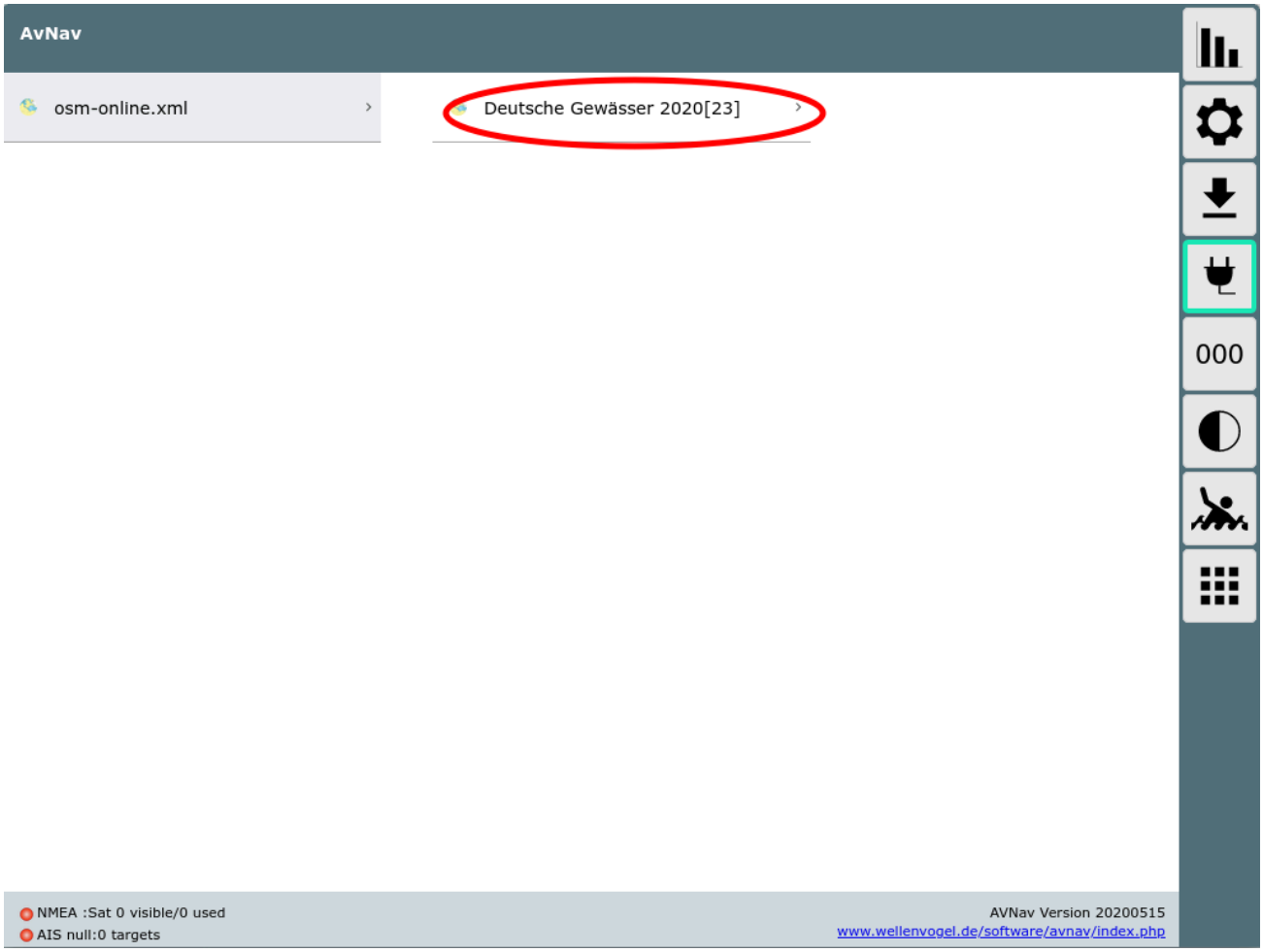
If the status turns to "ERROR" (red) you maybe uploaded a zip that was not built for your current system. You can check details in the log file at /home/pi/avnav/data/ocharts/provider.log.

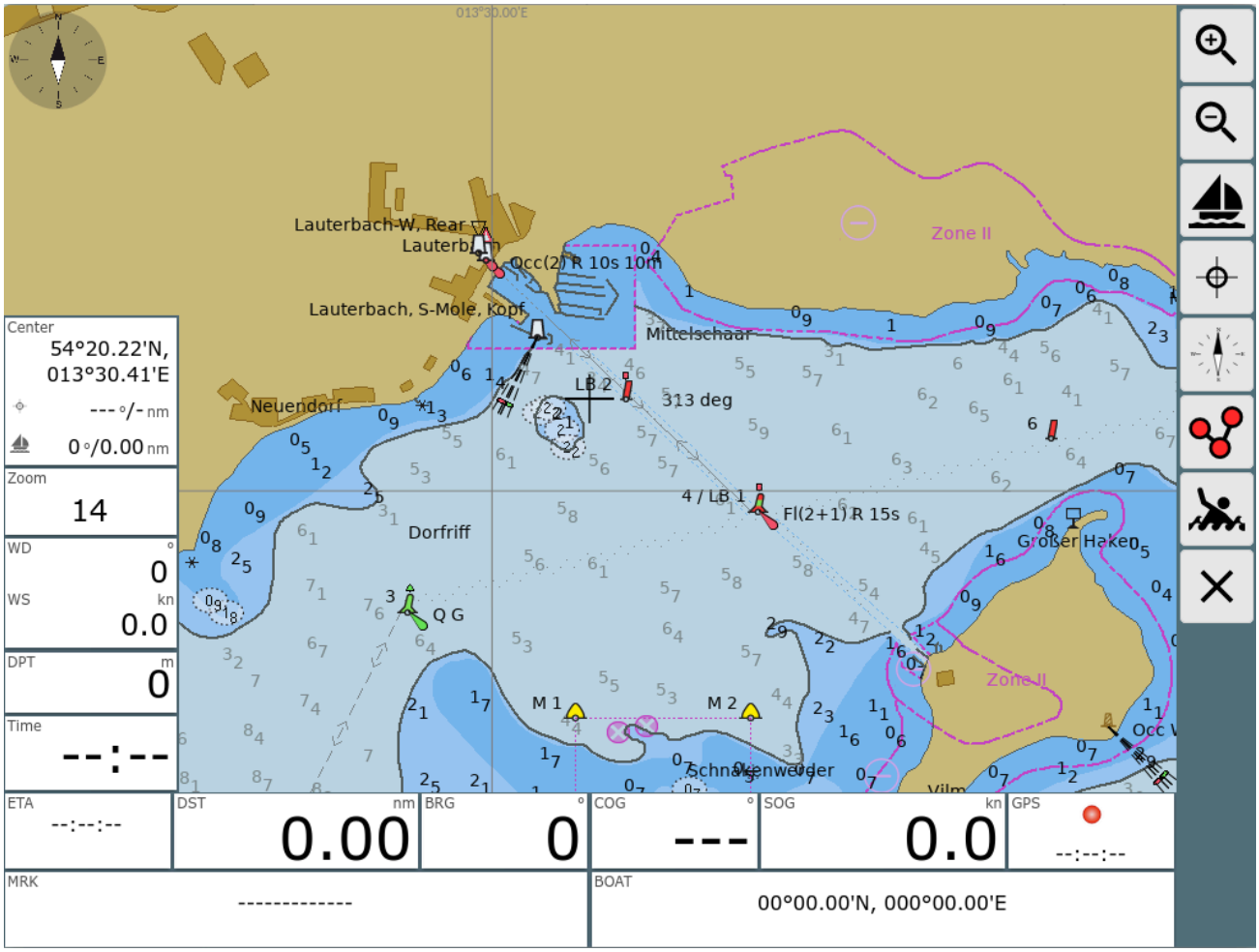
Now the charts are available and can be used.
At the "Status" tab you can see some more details.



In the screenshot you see the process ("FILLING") that was started after upload. It pre-renders a major set of chart tiles to a cache file. This will help to reduce on-the-fly rendering lags that otherwise would (potentially) occur due to the limited resources of the pi. This process can be active for several hours. The provider will use one CPU (out of the 4 available) on a raspberry pi during this time. Once prefill is completed cpu usage will dramatically decrease.

In any case you may immediately start using the charts (the prefill will only use the idle time).


AvNav
osm-online.xml
Deutsche Gewässer 2020[23]
000
NMEA :Sat 0 visible/0 used
AIS null:0 targets
AVNav Version 20200515
www.wellenvogel.de/software/avnav/index.php


Lauterbach-W, Rear
Occ(2) R 10s 10m
Lauterbach, S-Mole, Kopf
Mittelschaar
Zone II
LB 2
313 deg
Neuendorf
4 / LB 1
Fl(2+1) R 15s
Dorfriff
Q G
M 1
M 2
Zone II
Schnaapenwerder
Center
54°20.22'N,
013°30.41'E
Zoom
14
WD
0
WS
0.0
DPT
0
Time
--:--
ETA
DST
0.00
BRG
0
COG

SOG
0.0
GPS
MRK
BOAT
00°00.00'N, 000°00.00'E

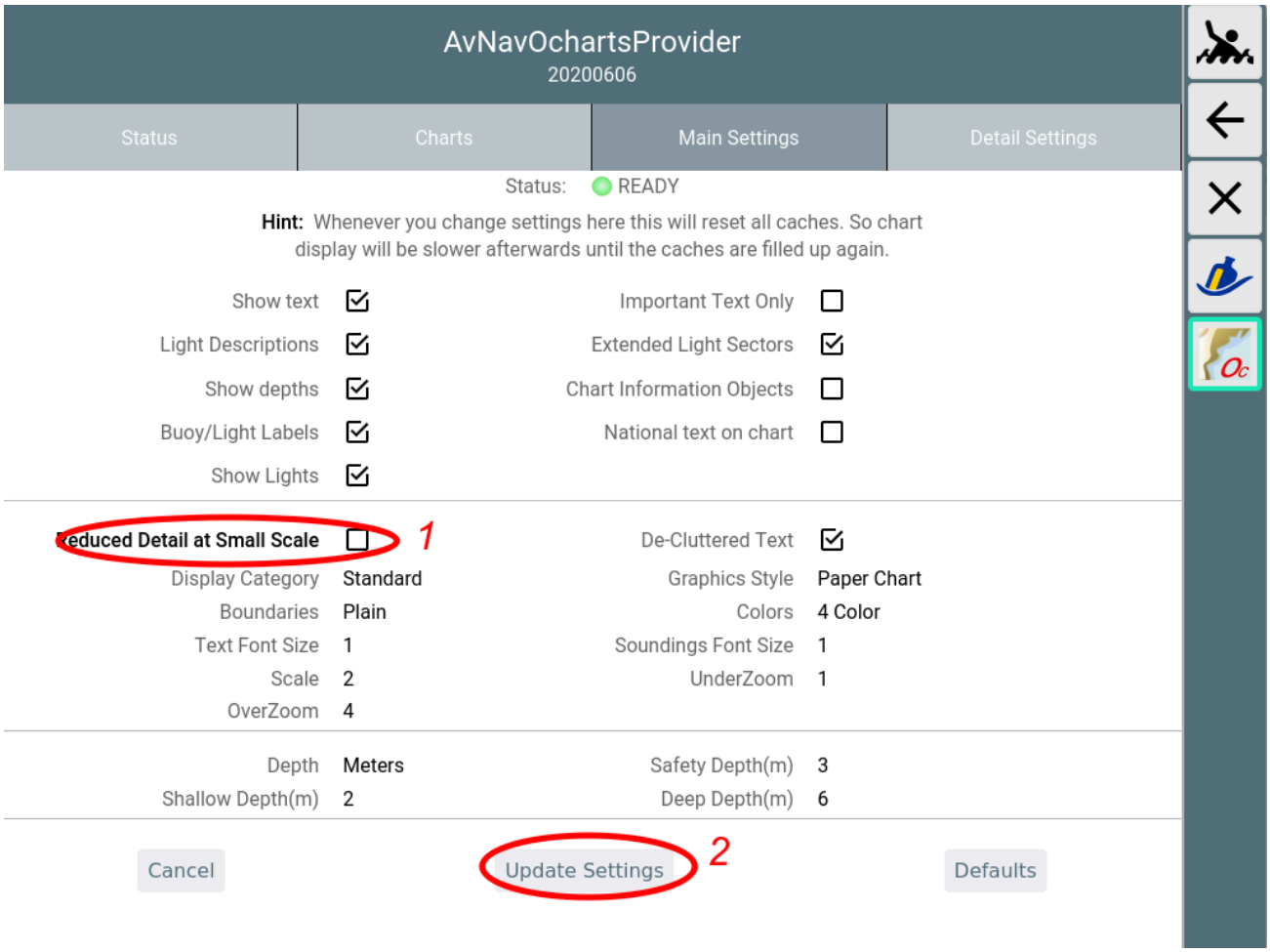
As already mentioned, there is a chance of lagging to occur when you are displaying a particular range/zoom of the chart for first time (you will notice this especially on the smaller/older pi's). After first usage of an area the necessary tiles will be held in the cache file and there should be no more lag.

Adapting the Look and Feel

As the o-charts are vector charts you can adjust their look and feel. However some limitations must be considered:

1. The modifications apply to the server's chart settings - so they will become effective for all displays.
2. If you change the look and feel all data in the cache have to be wiped and all chart tiles must be rendered from scratch. So this will (potentially) again induce lag on smaller/older systems. The automatic cache prefill process starts again to pre-render a lot of tiles.

Changing display parameters is done in the plugin GUI(-> ), tab "Main Settings".



If you change a setting (1) it will be displayed bold. Changes will only become effective when you click "Update Settings"(2).

By selecting "Cancel" you can revert your changes. "Defaults" will reset to the built-in defaults. Most parameters are similar to the ones you find at OpenCPN settings.

The following parameters are available.

Name	Meaning	Default
Show Text	show text for chart objects	true
Important Text Only	hide less important text	false
Light Descriptions	show descriptions for lights	true

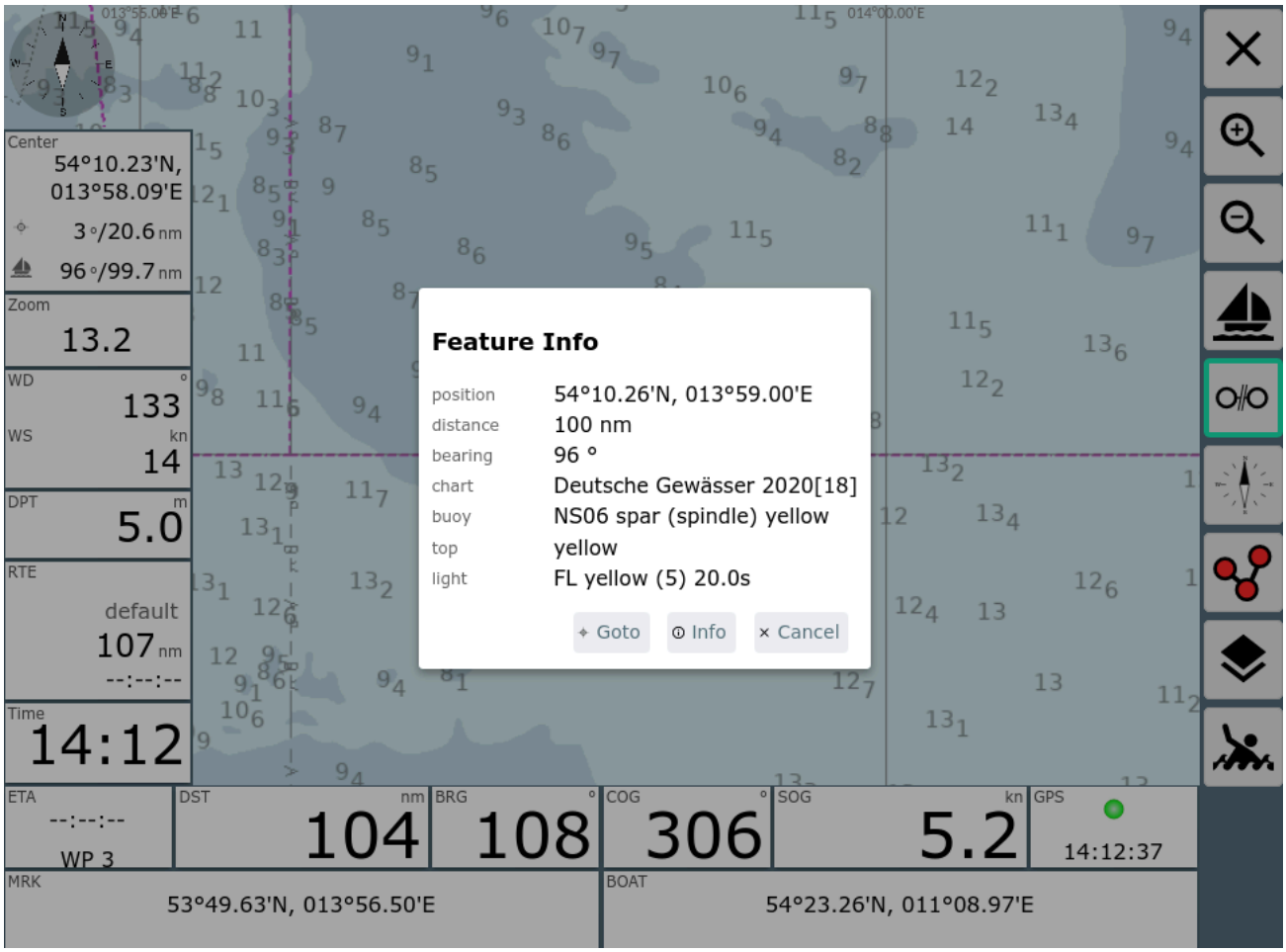
Extended Light Sectors	show sectors for lights	true
Show Depth	show soundings	true
Chart Information Objects	show special chart object infos	true
Buoy/Light Labels	show labels for buoys and lights	true
National text on chart	show national text	true
Show Lights	show lights	true
Reduced Detail at Small Scale	reduce details at lower zoom levels	true
De-Cluttered Text	improve text positioning	true
Display Category	Base, Standard, All, User Standard	All
Graphics Style	Paper Chart, Simplified	Paper Chart
Boundaries	Plain, Symbolized	Plain
Colors	4Color, 2 Color	4 Color

Text Font Size	Scaling for text on charts	1 (ca. 12px)
Soundings Font Size	Scaling for soundings (since oesenc-pi 4.2.x)	1 (ca. 12px)
Scale	Base scaling. Higher values for more details on lower zoom levels	2
UnderZoom	Number of zoom levels to downscale higher resolution chart tiles if no chart tile is available at the requested zoom level.	1
OverZoom	Number of zoom levels to upscale a lower resolution chart tile if is no chart tile is available with better resolution. **Hint**: Scale, UnderZoom and OverZoom heavily influence the cost of the rendering process as they determine the number of charts to be processed to generate a single chart tile. Lower values normally mean less charts (i.e. being faster) - but there could be white areas between chart tiles. The defaults should be a good compromise.	4
Depth	Unit for soundings(Meters, Feet, Fathoms)	Meters
Shallow Depth	Adjust to your needs	2
Safety Depth	Adjust to your needs	3
Deep Depth	Adjust to your needs	6

At the tab "Detail Settings" you can switch on/off particular chart features.

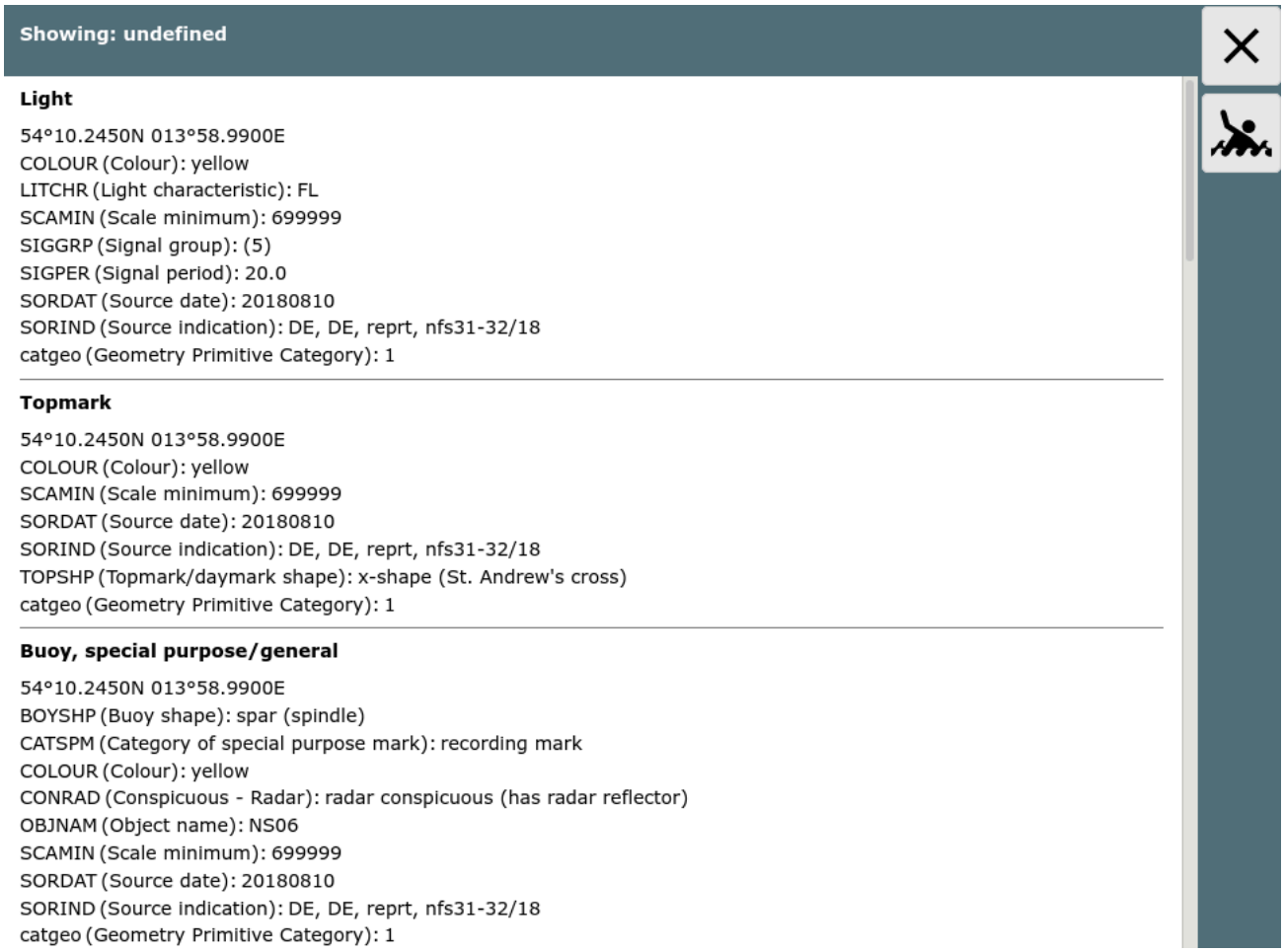
Feature Info (Object Query)

Since version 20201219 (versions of AvNav and plugin are required) chart object information is displayed upon click.



The dialog box displays compact information about "important" objects like lights, buoys and others.

By clicking "Info" you can view the raw information from the chart.



Installation

If you are running an AvNav image you can install the new plugin as package. For the Headless Images the necessary packages are already contained in the images. Additionally they are available in the repository. You need to install

- avnav-ocharts-plugin
- avnav-oesenc

For avnav-ocharts-provider you need at least version 20200606. The package avnav-oesenc is the oesenc-pi plugin - repackaged to install into /usr/lib/avnav/plugins/ocharts to avoid conflicts with a parallel OpenCPN installation.

```
sudo apt-get update
sudo apt-get install avnav-ocharts-plugin avnav-oesenc
sudo systemctl restart avnav
```

If you are working on other images you should add the repository from free-x:

```
deb https://www.free-x.de/debian buster main contrib
non-free
```

You can also see the packages in the release list below. To use one of those packages (if it is not in the repo yet or if you need an older one) - just download and install the package (replace the version by the one you want):

```
cd /home/pi/avnav
wget -O avnav-ocharts-plugin_20200606-raspbian-
buster_armhf.deb
https://www.wellenvogel.net/software/avnav/downloads/relea
ochartsplugin/20200606/avnav-ocharts-plugin_20200606-
raspbian-buster_armhf.deb
sudo dpkg -i /home/pi/avnav/avnav-ocharts-
plugin_20200606-raspbian-buster_armhf.deb
sudo systemctl restart avnav
```

You can also download to a PC, transfer by scp/WinScp to the pi and install then.

Releases

You can find all releases and intermediate developer builds (daily builds) at:

- Releases
- Daily Builds

Release Versions

- 20230706 packages
  - Bug fix: Make check boxes visible again in settings

- 20230702 packages
  - Bug fix #41: Handling of SENC overlays
  - Bug fix #47: Handling of obsolete charts (open error 3)
  - Improvement: allow to set a render timeouts in the plugin settings
  - Usage of OpenCPN charts should work better now

- 20220605 packages
  - Bug fix #36: Problems on OpenPlotter with OpenCPN flatpak

- 20220421 packages
  - Bug fix: provider not always restarted correctly
  - dependency to avnav-ocharts

- 20220307 packages
  - Bugfix #31: Missing checkbox for "use OpenCPN charts"
  - Bugfix: wrong version was shown
  - improved error handling during chart usage, show such errors in the status
  - correctly limit setting for memPercent

- 20220225 packages
  - New O-charts shop
    During the next days the shop at o-charts will move to a new encryption schema.
    This will require this new version of the AvNav plugin. **Additionally you need to install the new avnav-ocharts package** - at least 0.9.0.72 (use the avnav update plugin to handle this). The package avnav-oesenc is now obsolete and can be removed (but it can also remain on the system).

The handling of the charts does not change and all your old charts will continue to work. But once the shop is changed you will not be able to download new charts with the old plugin.
With the new plugin (and the new shop) **AvNav will now also be able to handle oeRNC** charts (like the imray charts for the Med).

- packages will now also be available for the raspberry debian bullseye OS versions (both 32 bit and 64 bit)
- the handling for "reduce details on lower zoom levels" does now work correctly
- the "under zoom" setting now works better so it's range could be extended and the default changed to 4. This avoids some white areas if no charts directly matched the required zoom level
- You can now enable the usage of OpenCPN charts on the same system directly in the AvNav plugin settings. This will only work for charts installed with the new ocharts_pi plugin in OpenCPN.

- the error handling has been improved

- 20210711 packages
  - bugfix: Cache rebuild not triggered when changing parameters

- 20210328 packages
  - Parameter handling in AvNav (requires AvNav >= 20210322)
  - Limit for zip file size enlarged to 3GB
  - restart error fixed

- 20210115 package
  - shift to python3 (requires AvNav since 20210115)
  - improved error handling for charts

- 20201219 package
  - Display of object information (requires AvNav >= 20201219)

- 2020115 package

- Improved memory handling. The Xvfb will be restarted if it reaches 120MB
- Faster start up. The chart information will be kept in a cache file. So reading the charts is only necessary if some charts have changed.
- Removed memory leak within the used OpenCPN plugin (workaround) - so the memory usage of Xvfb will not increase that much any more.
- Hint: When installing with dpkg -i (will give an error) you need to install missing dependencies with:

```
sudo apt-get install -f
```

- 20200710 package
  - You can now set the directory for the charts - see details
  - When uploading charts it is checked if those charts can be read. Otherwise the upload is rejected.
  - No errors in GUI any more when restarting the provider
  - correctly work on a vfat partition (as used by avnav touch)

- 20200705 package
  - correct a problem in the OpenCPN plugin that could potentially stop it from decoding charts after running for a long time.

- 20200606 package
  - first version

License Notes

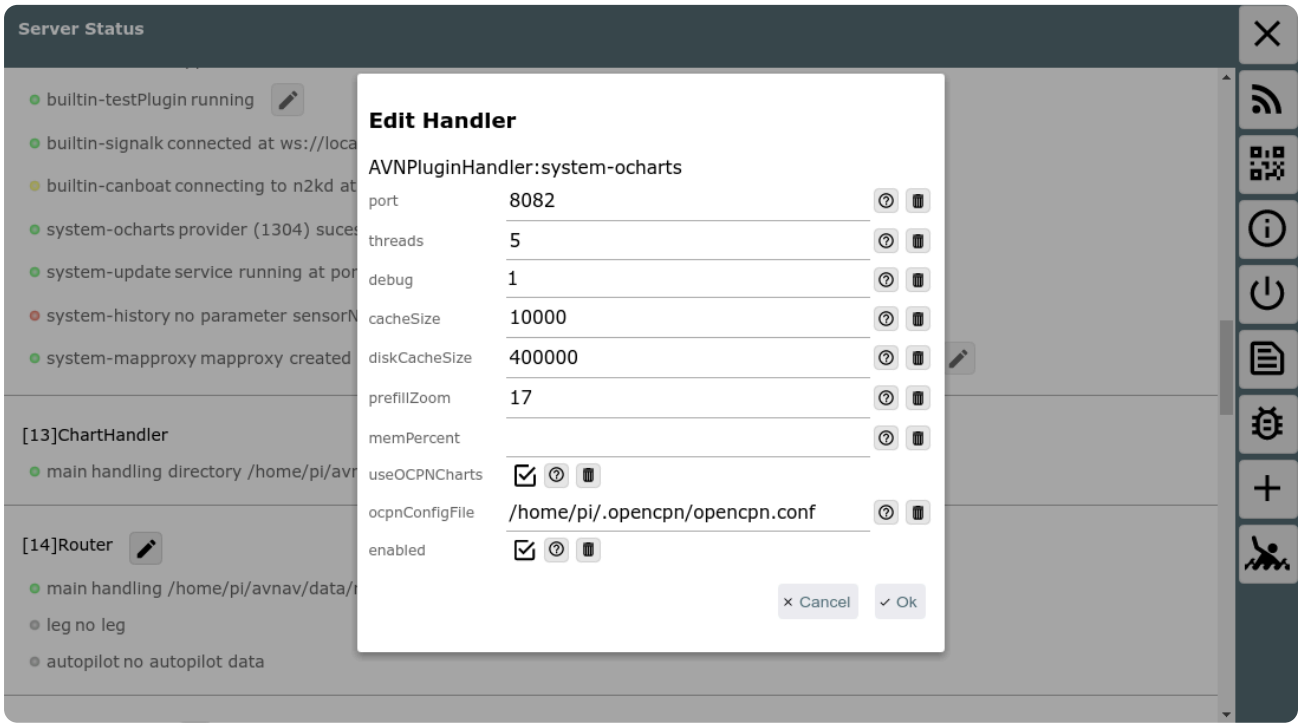
Using the charts in AvNav with the oesenc-pi plugin has been agreed with o-charts and therefore is inline with their license conditions.
You have to agree to the license conditions of o-charts
Especially it is not allowed to copy the charts or use them on other than the licensed systems.

Access to the charts in AvNav is only possible from within the local net. You can connect at most 5 devices (Clients) at the same time.

For software licenses see the Readme.

Plugin Configuration

Some of the plugins settings can be changed on the server/status page at "plugins/system-ocharts" (AvNav >= 20210322).



Those are:

Name	Meaning	Default
port	Http port	8082
threads	Number of threads that will be used	5
debug	log level for <datadir>/ocharts/provider.log.	1

	<datadir> on a raspberry is /home/pi/avnav/data	
cacheSize	Maximal number of chart tiles that will be cached in memory. The plugin will additionally consider the allowed memory and will potentially lower this setting.	10000
diskCacheSize	Maximal number of chart tiles (for one set) that will be cached in a file.	400000
prefillZoom	Up to which zoom level the prefill process should already render chart tiles into the cache. If you increase this value the prefill will take more time.	17
memPercent	The amount of memory (percent of the system memory) that the plugin (correctly: the provider process) will use. If this value is not set (or set to low) the provider will use an internal minimum. This potentially could be rather low - especially when using raster charts. If it is very low the provider must open and close chart files very often and this will slow down it's operation. If you have enough memory (e.g. 2GB)	---

	you can increase speed by setting the memory to 1GB .	
useOCPNCharts (seit 20220225)	If this fla is set you can use charts that have been installed with OpenCPN on the same system within AvNav . This will only work for charts being installed with the new o-charts_pi plugin (since 2022/03/01).	aus
ocpnConfigFile (seit 20220225)	Path to the OpenCPN config file (normally $HOME/.opencpn/opencpn.conf). The plugin needs to read this file to find the installed charts.	$HOME/.opencpn/o

Technical Details

The charts are provided by an executable on the raspberry pi that normally serves at port 8082. This executable loads the oesenc-pi OpenCPN plugin. Communication with AvNav is handled by an AvNav plugin.

The GUI is a reactjs app that is also provided by the executable. It is integrated into AvNav as a User App.

You will find the complete code at GitHub.

You install into /usr/lib/avnav/plugins/ocharts. The data directory is /home/pi/avnav/data/ocharts. You can set some additional parameters for the AvNav plugin by editing avnav_server.xml. Normally this is not necessary at all.

Since version 20200709 you can separately set the directory for uploading the charts:

```
<AVNPluginHandler>
...
<system-ocharts uploadDir="$DATADIR/charts/ocharts"/>
</AVNPluginHandler>
```

In this example the directory is set to /home/pi/avnav/data/charts/ocharts (normally it is located at /home/pi/avnav/data/ocharts/charts). This could be helpful on the touch image as sufficient disk space is available only in /home/pi/avnav/data/charts.

Avnav Overlays

Since version 20201219 AvNav is able to show additional information beside the chart itself in it's chart views. This could be waypoints from a GPX file or available tracks or other data (like e.g. current buoys from nautin.nl).

Additionally you can show multiple charts on top of each other (e.g. an OpenSeamap chart that covers a wider range below an o-charts chart - or just multiple o-charts for different ranges).

Within AvNav this is called Overlays.

You can use the following types of data for overlays:

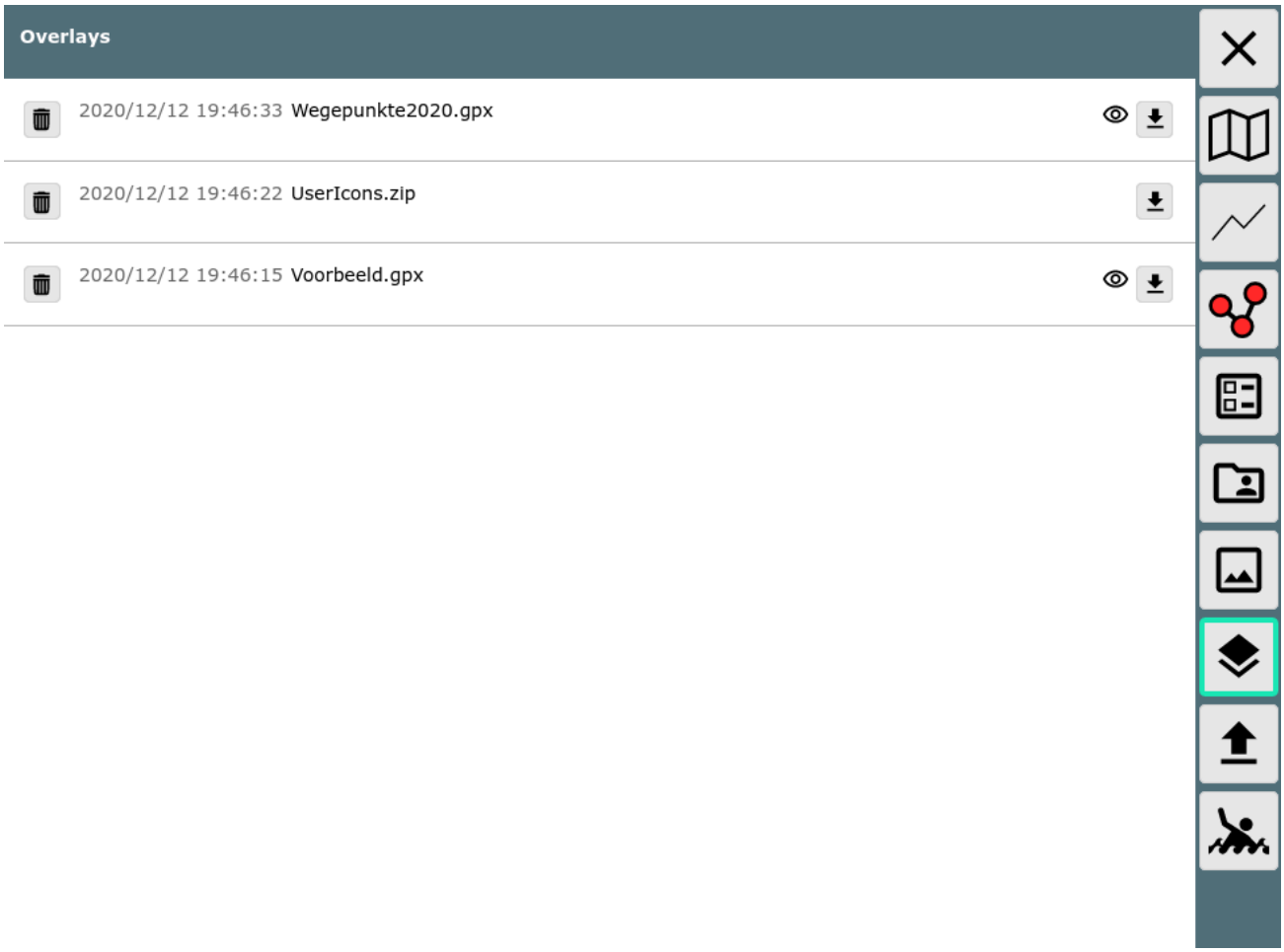
- GPX files (waypoints, tracks, routes)
- KML and KMZ files (google) - inside the KMZ file you must have a doc.kml.
- geojson files
- other AvNav charts
- AvNav routes
- AvNav tracks

For GPX and KML files you can provide user defined icons or e.g. html pages that can be accessed via link properties within a zip archive. KMZ files normally already contain such kinds of data.

If you have such overlays visible on a chart you can obtain additional information by clicking on the symbol on the chart.

Configuration

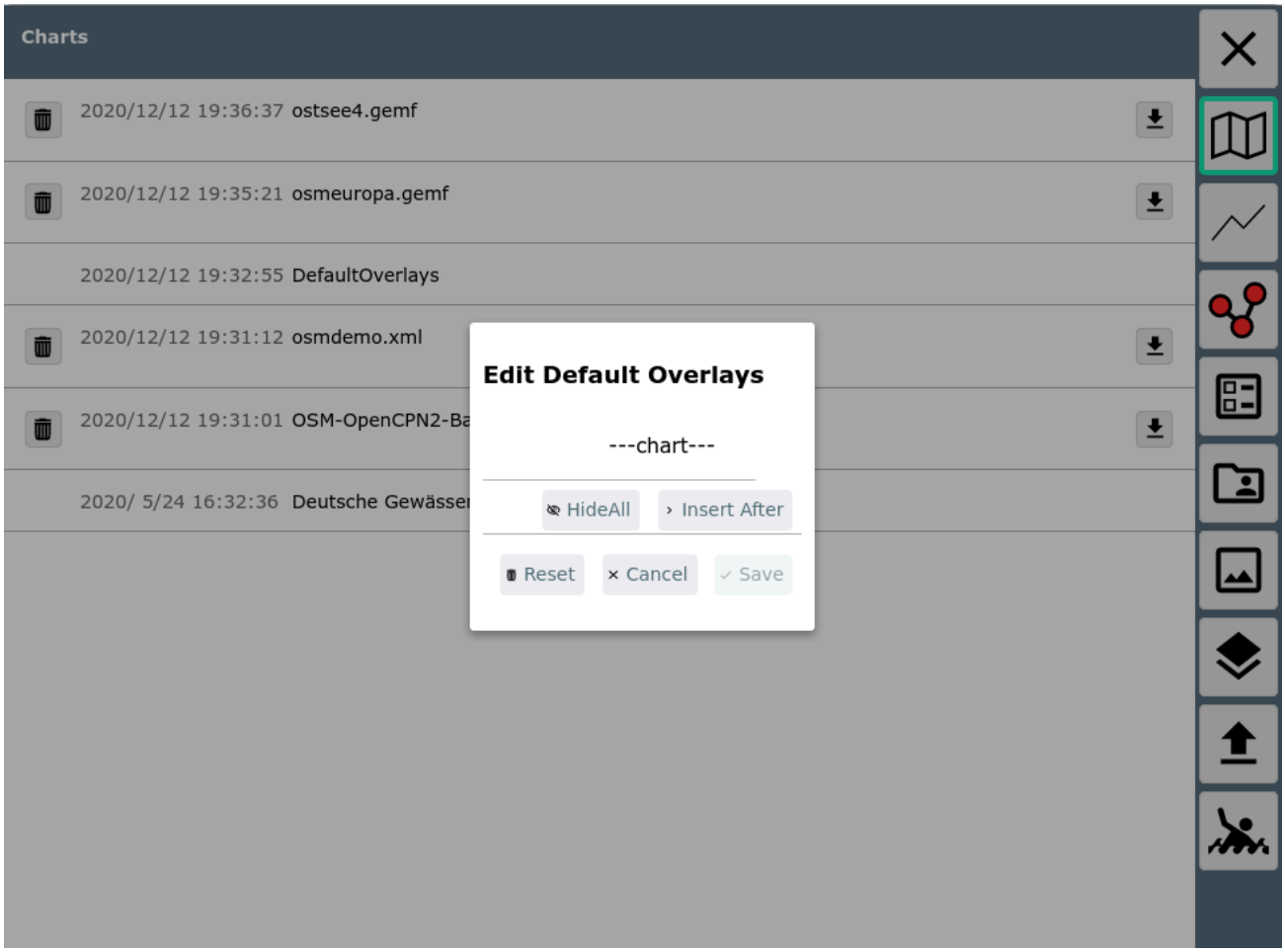
Before you can start using overlays they need to be uploaded to AvNav.



Within AvNav the symbol is used for overlays.

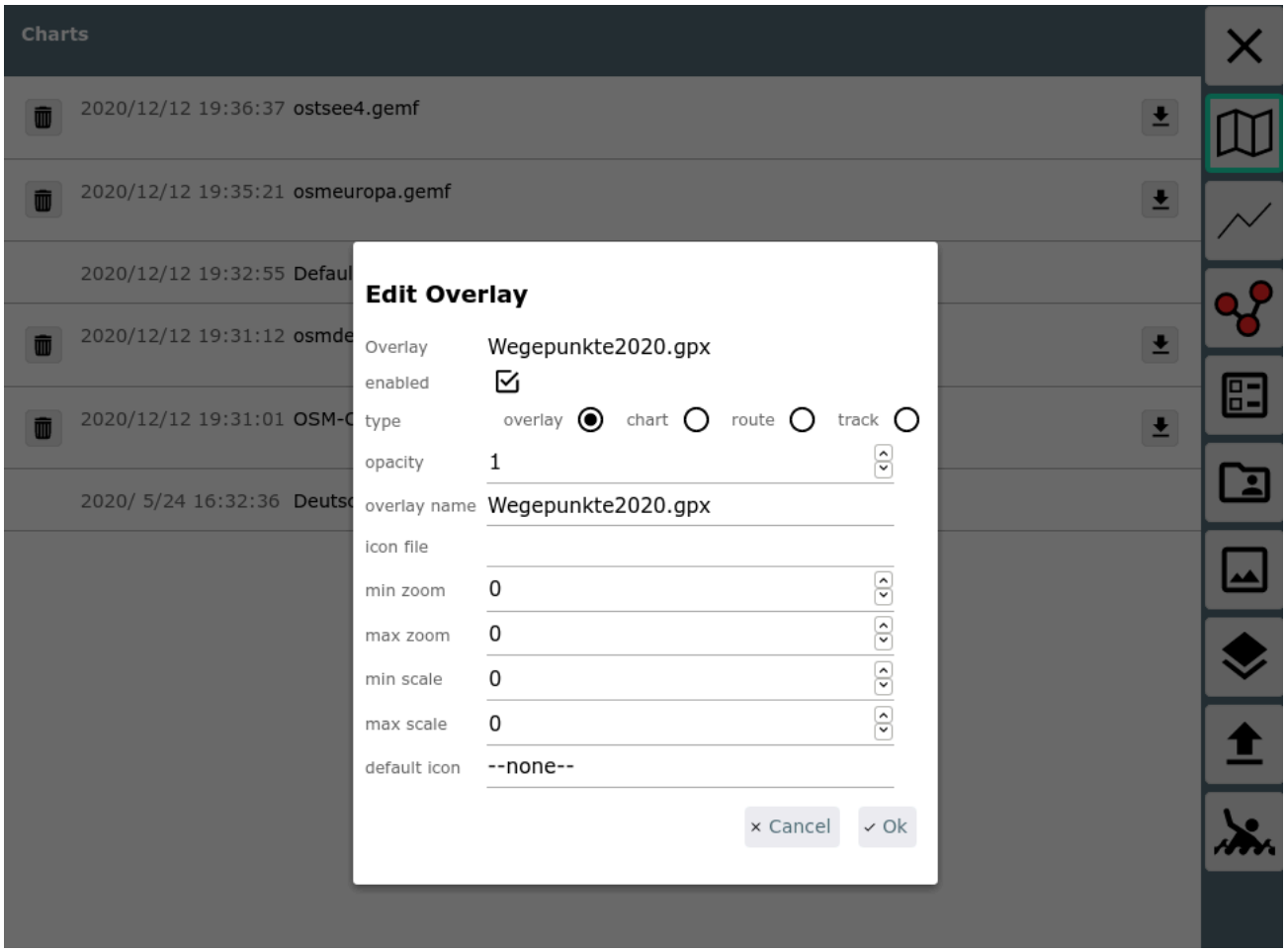
After you uploaded the files you can assign them to the available charts. You can define a set of overlays to be shown on every chart (default overlays). Additionally you can define per chart which overlays you would like to see on it.

You can make those assignments either on the Files/Download page at the sub section "charts" - or at the Mainpage.



In the screenshot you see the dialog showing up once you click on "DefaultOverlays" at the Files/Download page.

In this dialog you can add, edit or remove overlays.



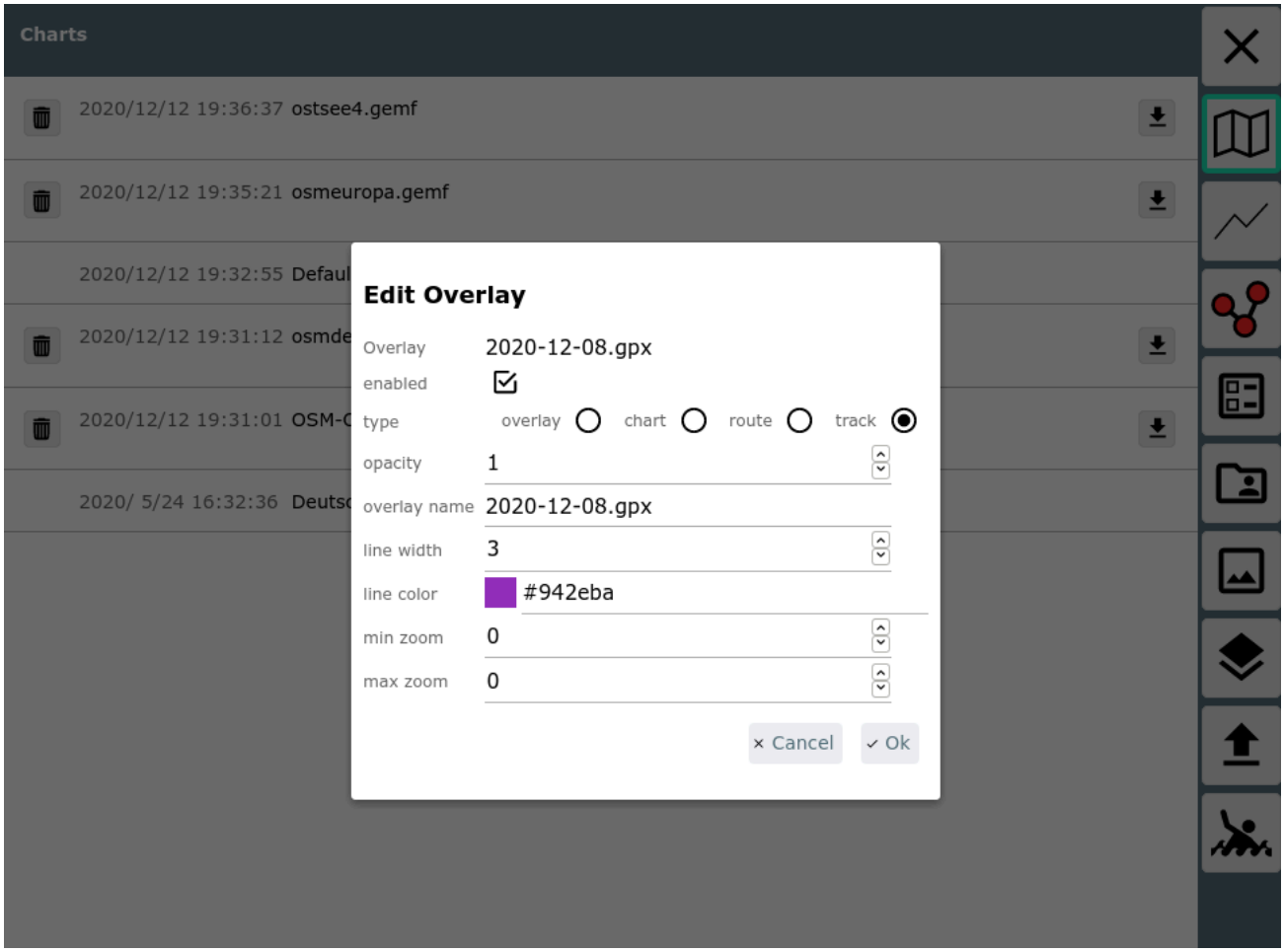
In the screenshot you see the dialog that opens on clicking "InsertAfter".
In the example I selected the file Wegepunkte2020.gpx (NV Verlag) that has been uploaded to the overlays directory.
You can now set various parameters (depending on the type of overlay).

Name	default	Description
enable	true	If set to false the overlay is configured - but will still not show up. You can have it displayed later on by a single click.
type	overlay	type of overlay overlay: a file that has been uploaded to overlays chart: a chart from AvNav route: a route from AvNav track: a track from AvNav

overlay name	---	you will get a list to select from available overlays depending on the selected type
opacity	1	the opacity for the display (0...1) - 0: invisible
user icons	---	You can select a zip file (that has been uploaded to overlays) containing symbol icons or other content (like html pages). If there is a "sym" property within a gpx file (e.g. at a waypoint) - like sym="b1" AvNav will try to read a file b1.png from the provided zip file to display it at the position. If a link property like link="data/1.html" is present, an entry data/1.html will be searched for within the zip file. The same applies for KML files. Normally however, you better use KMZ files as they already contain all necessary data.
min zoom	0	the minimal zoom level for displaying the overlay(0: always)
max zoom	0	the maximal zoom level for displaying the overlay(0: always)
min scale	0	if the zoom level of the chart is below this value, symbols will be scaled down (0: function disabled)
max scale	0	if the zoom level of the chart is above this value, symbols will be scaled up (0: function deactivated)
default icon	--none--	With this parameter you can select an icon file (png, svg) to be used if an icon is not found (or you did not fill the user icons parameter). You can select from files you have uploaded to overlays or to user

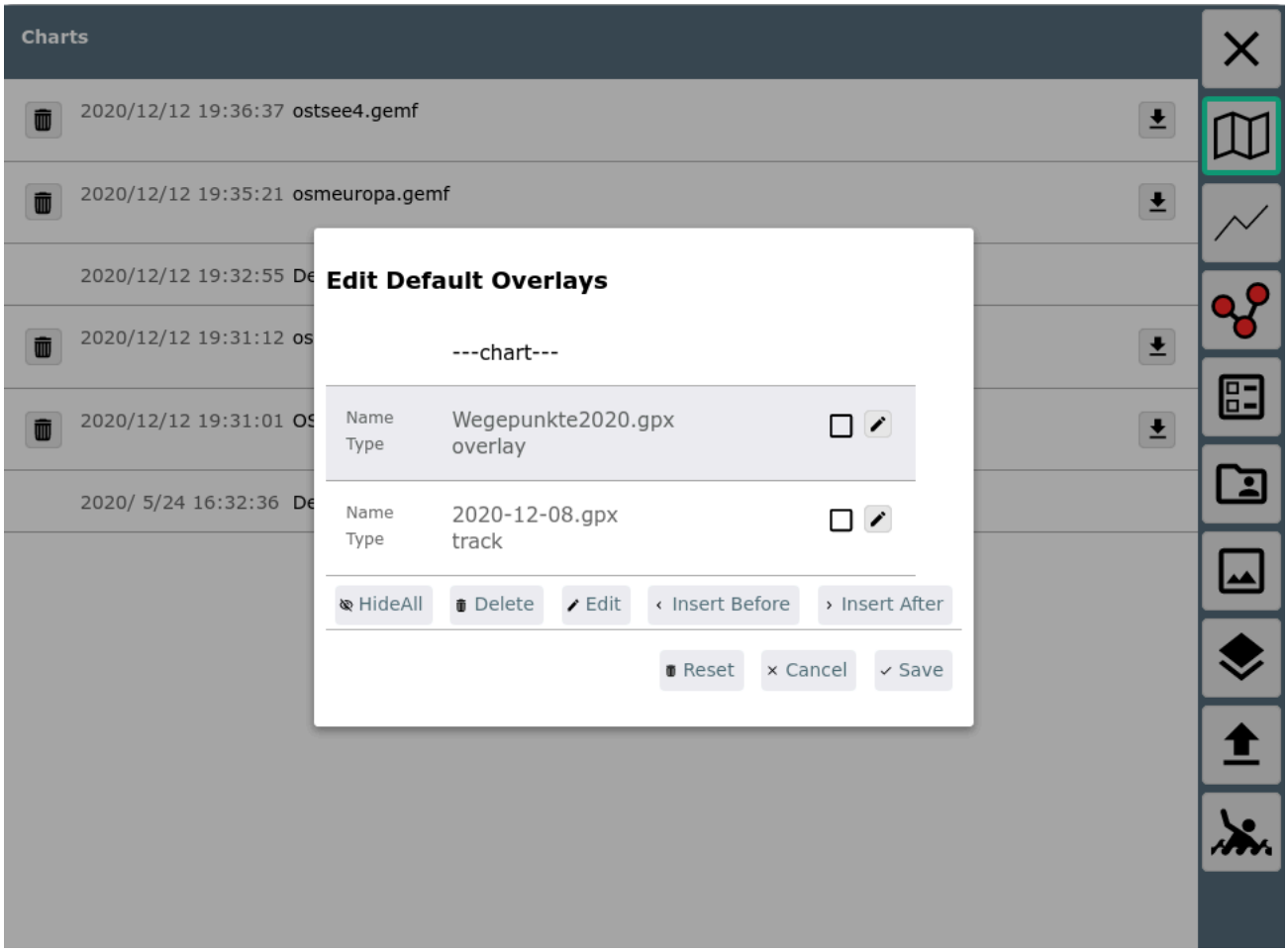
files or to images.
With --DefaultGpxIcon-- you can select a builtin icon.

If the overlay contains other data than way points, potentially more parameters can be displayed.



Name	default	Description
line width	track/route line width	line width for display
line color	track/route color	line color for display

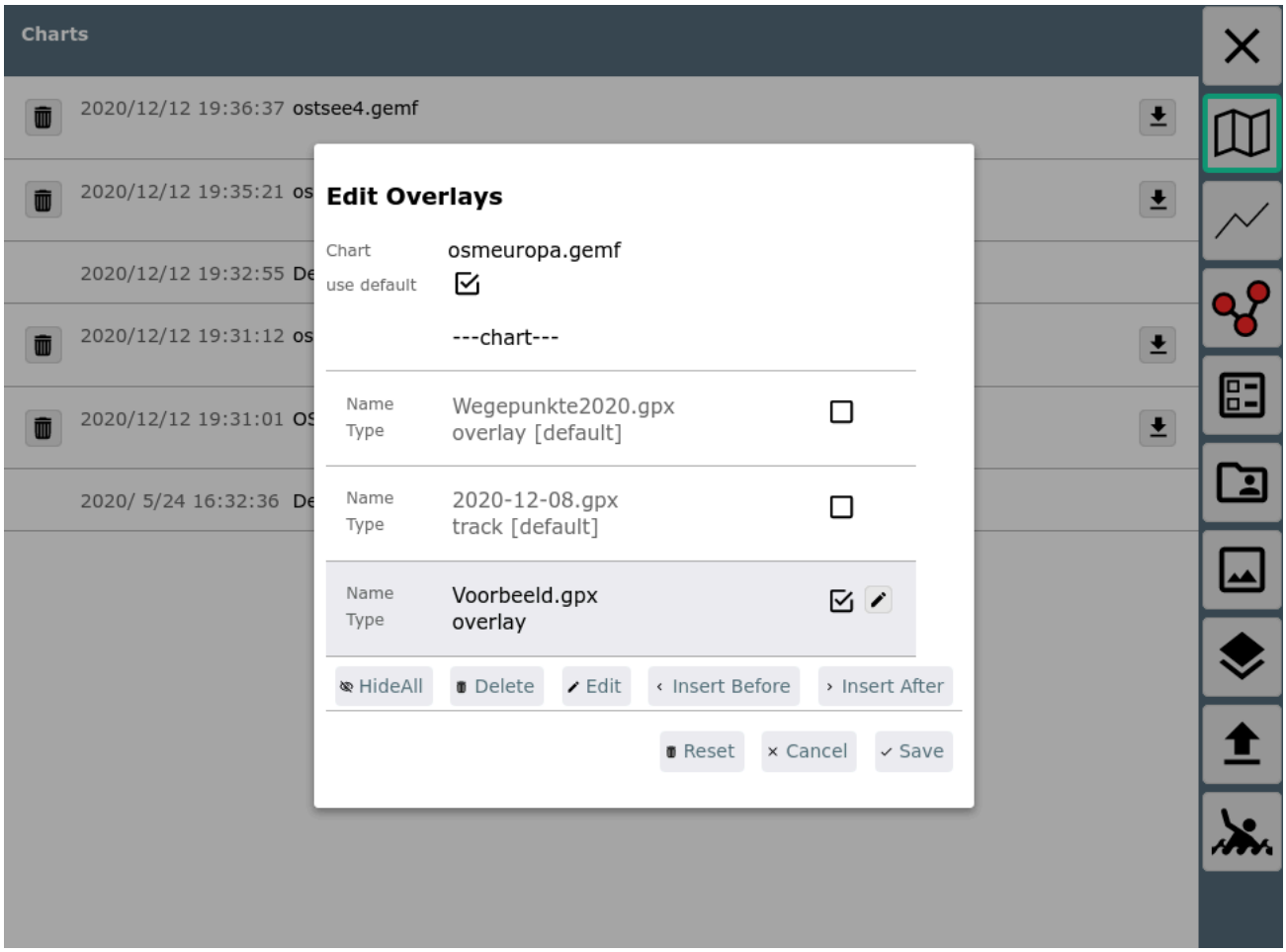
After clicking "Ok" the overlay has been added.



You can change the overlay order by drag&drop - if you want one below the chart you have to move it above the ---chart--- row.

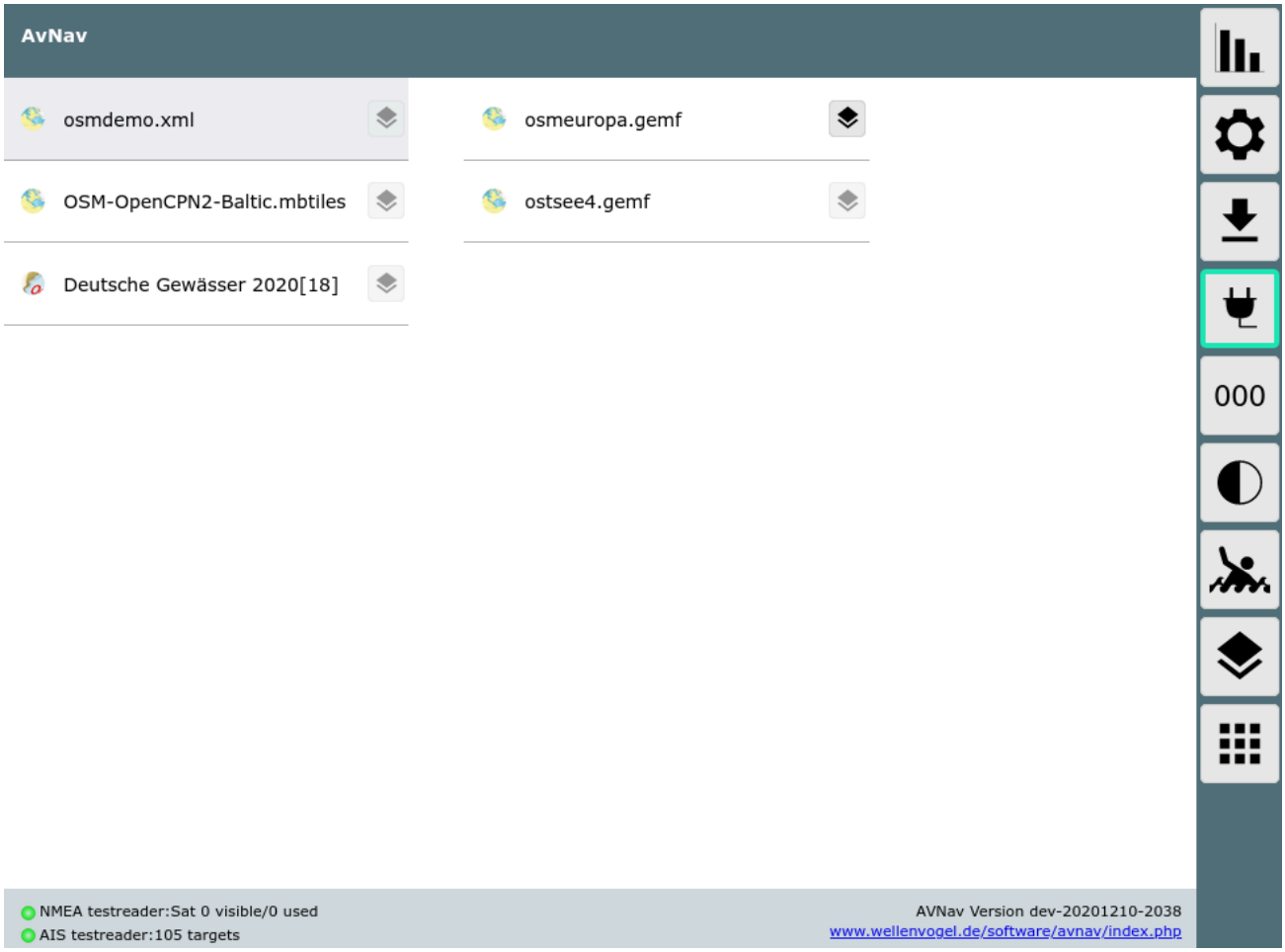
It could make sense to keep default overlays disabled first and just enable them on the charts where you really need them - or just on demand.

For a chart the overlay configuration looks like:



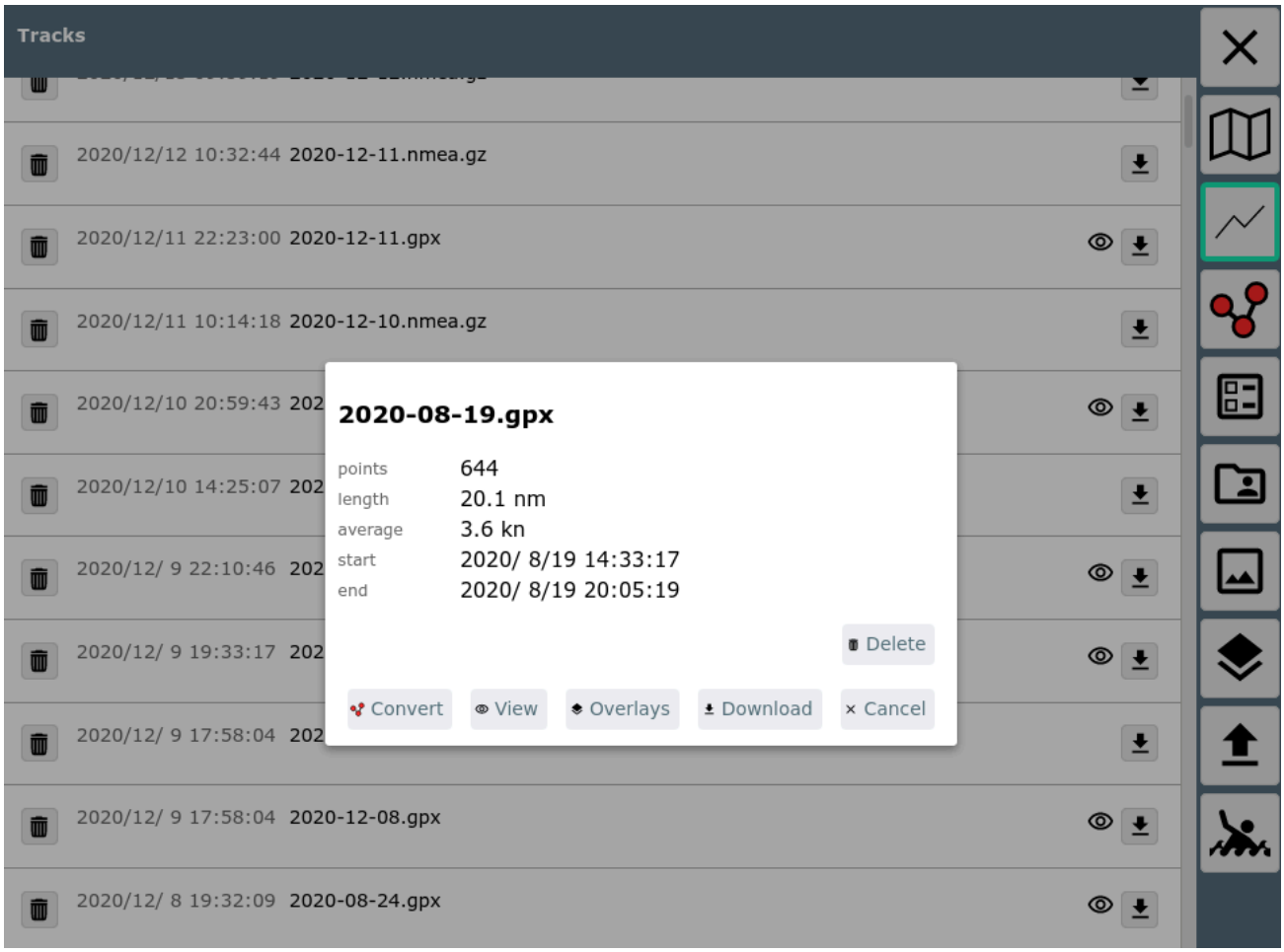
Overlays configured as defaults can only be moved (as a block) or enabled/disabled. The overlays assigned explicitly to the chart can be edited.

You can also access the overlay configuration page from the Mainpage via the button beside each chart.

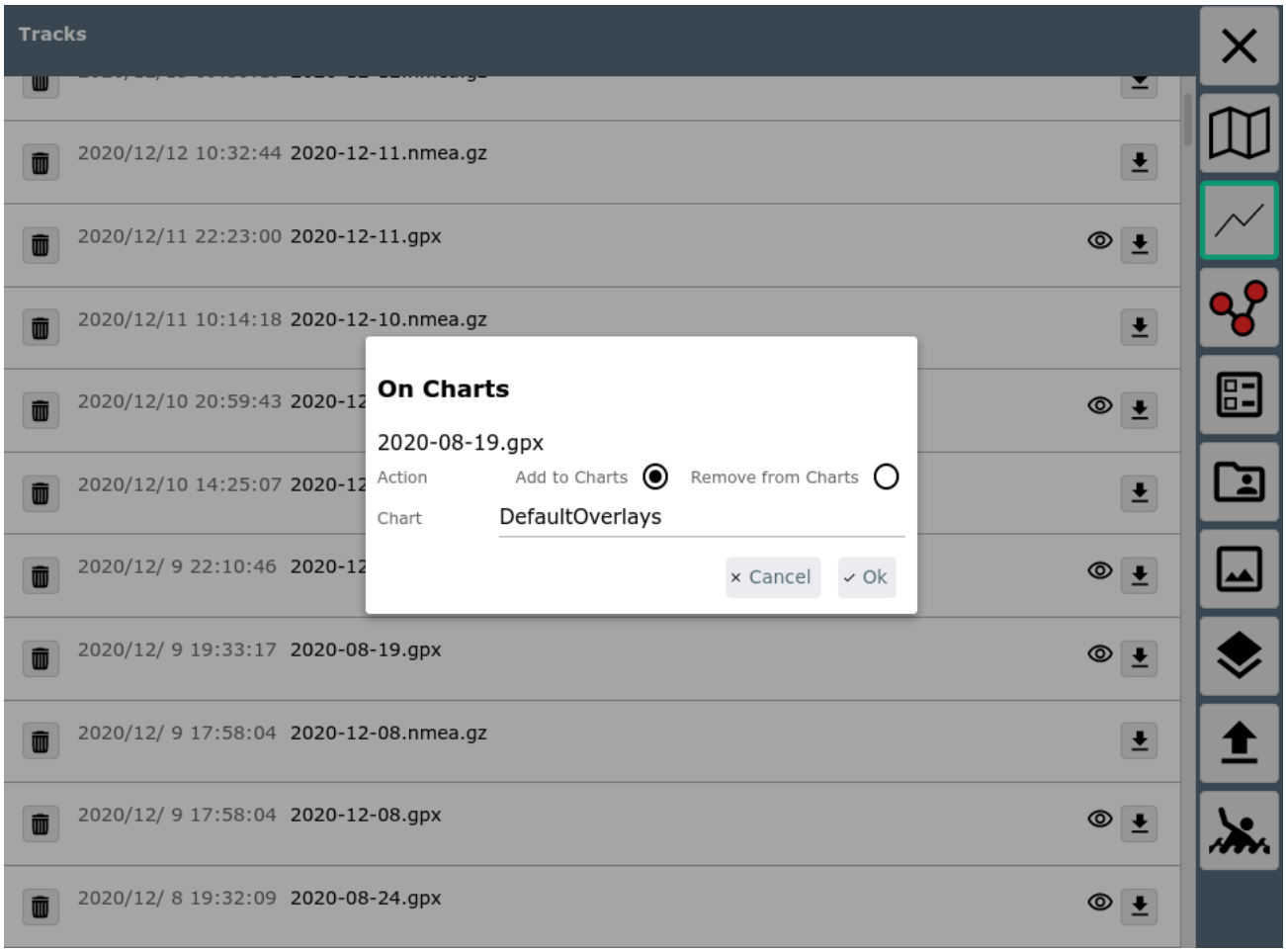


The button on the right side will show the configuration for the default charts.

For tracks and routes you can also invoke overlay configuration directly from a track (gpx file) or a route on the Files/Download page.



In the picture you see the info dialog that pops up after clicking a gpx track. By selecting "Overlays" you can assign this track to charts (or remove it).

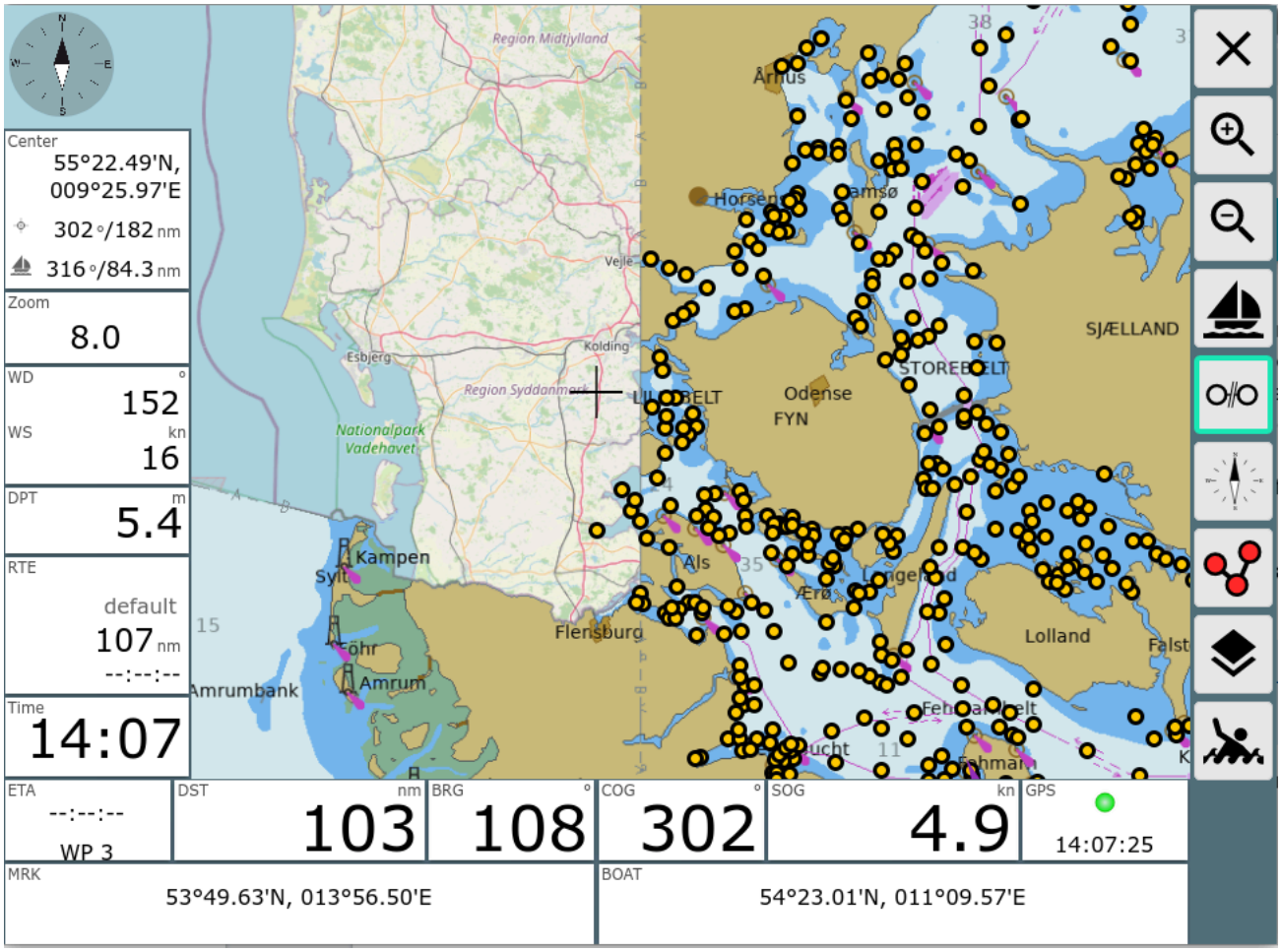


After deciding whether to assign it to the default overlays or to a particular chart, the normal overlay configuration dialog will pop up.

If you select "Remove from Charts" this track will be removed from all overlays.

Usage

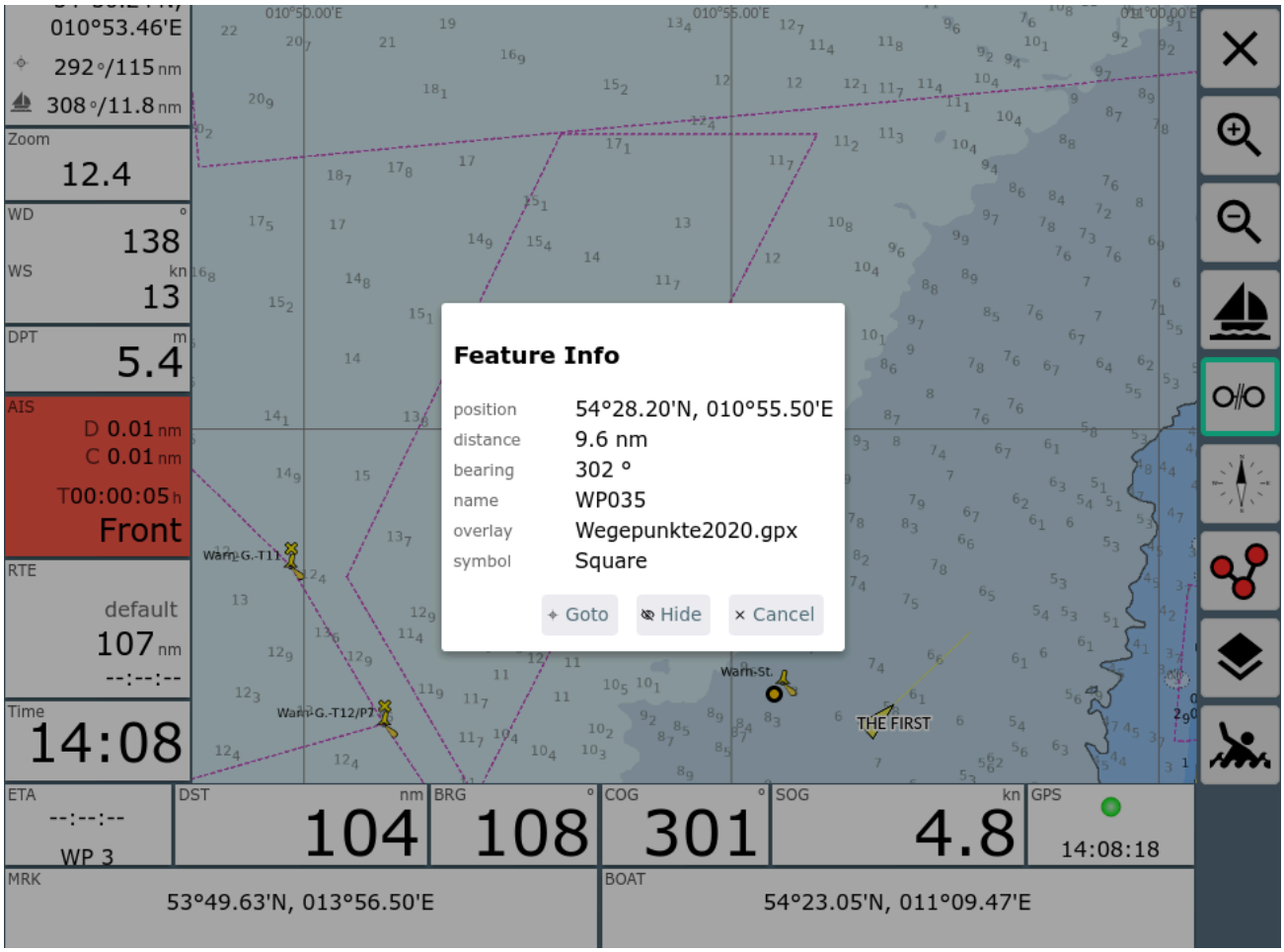
A chart selected for display will be shown together with the configured overlays.



This example shows an o-charts chart together with an OpenStreetMap chart (below) and a waypoint file on top.

As default icon the --DefaultGpxIcon-- has been selected.

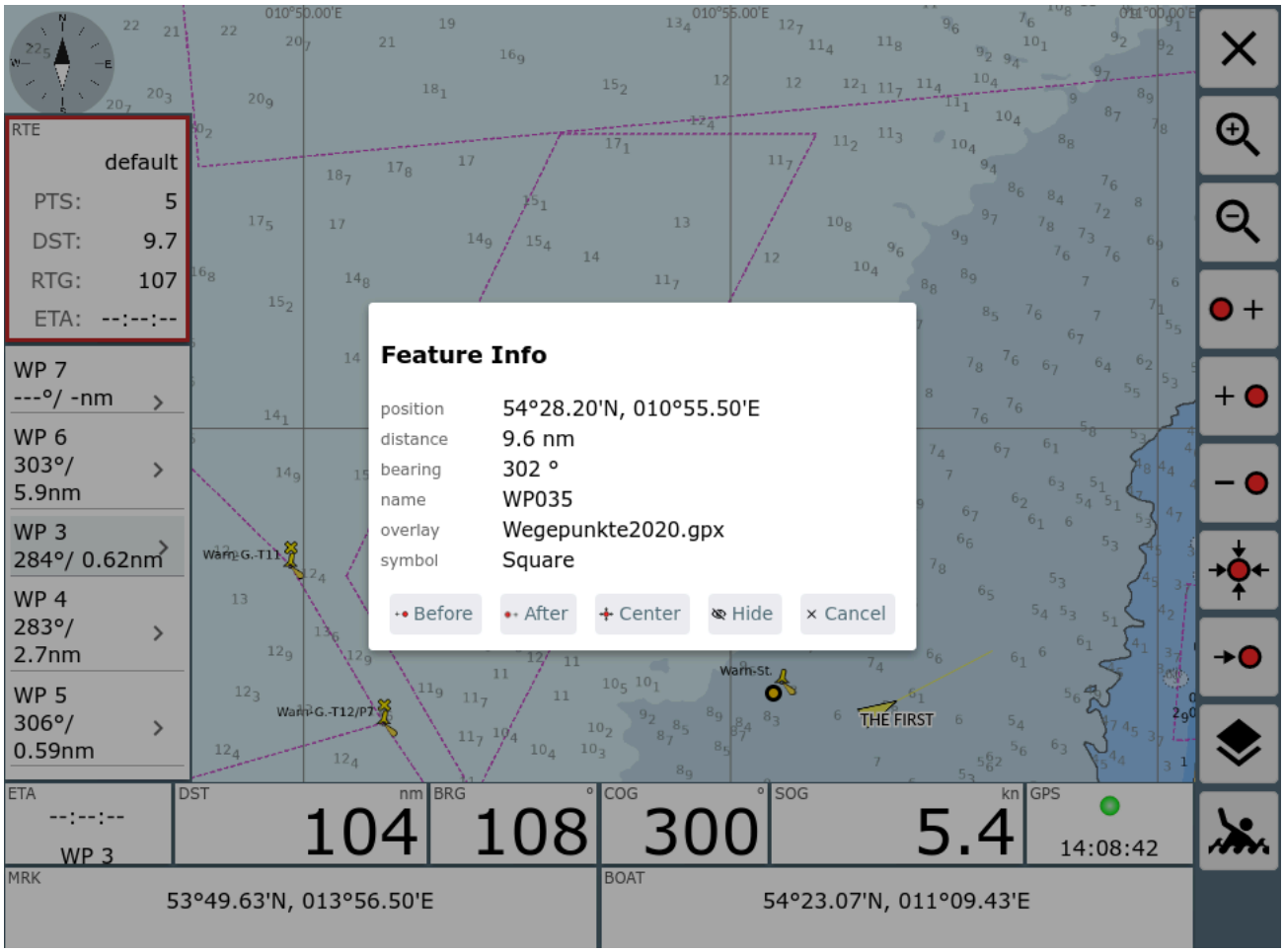
If you click one of the overlay's icons you will get its Feature Info dialog.



You will see the inforamtion available for this point and you can e.g. use this point as a routing target immediately (Goto).

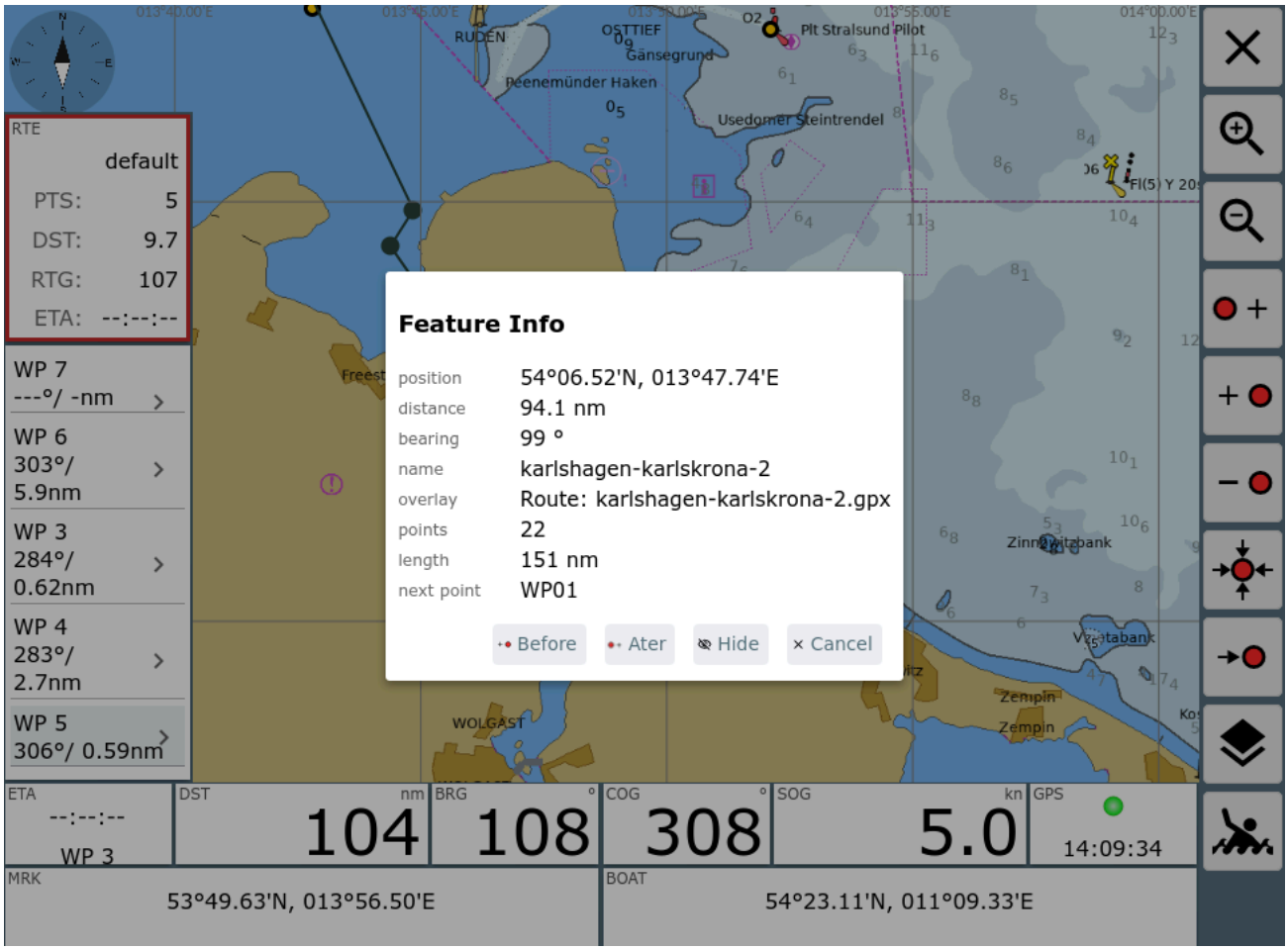
Using "Hide" you can temporarily disable the complete overlay.

From within the Route Editor some more functions are available at the info dialog.



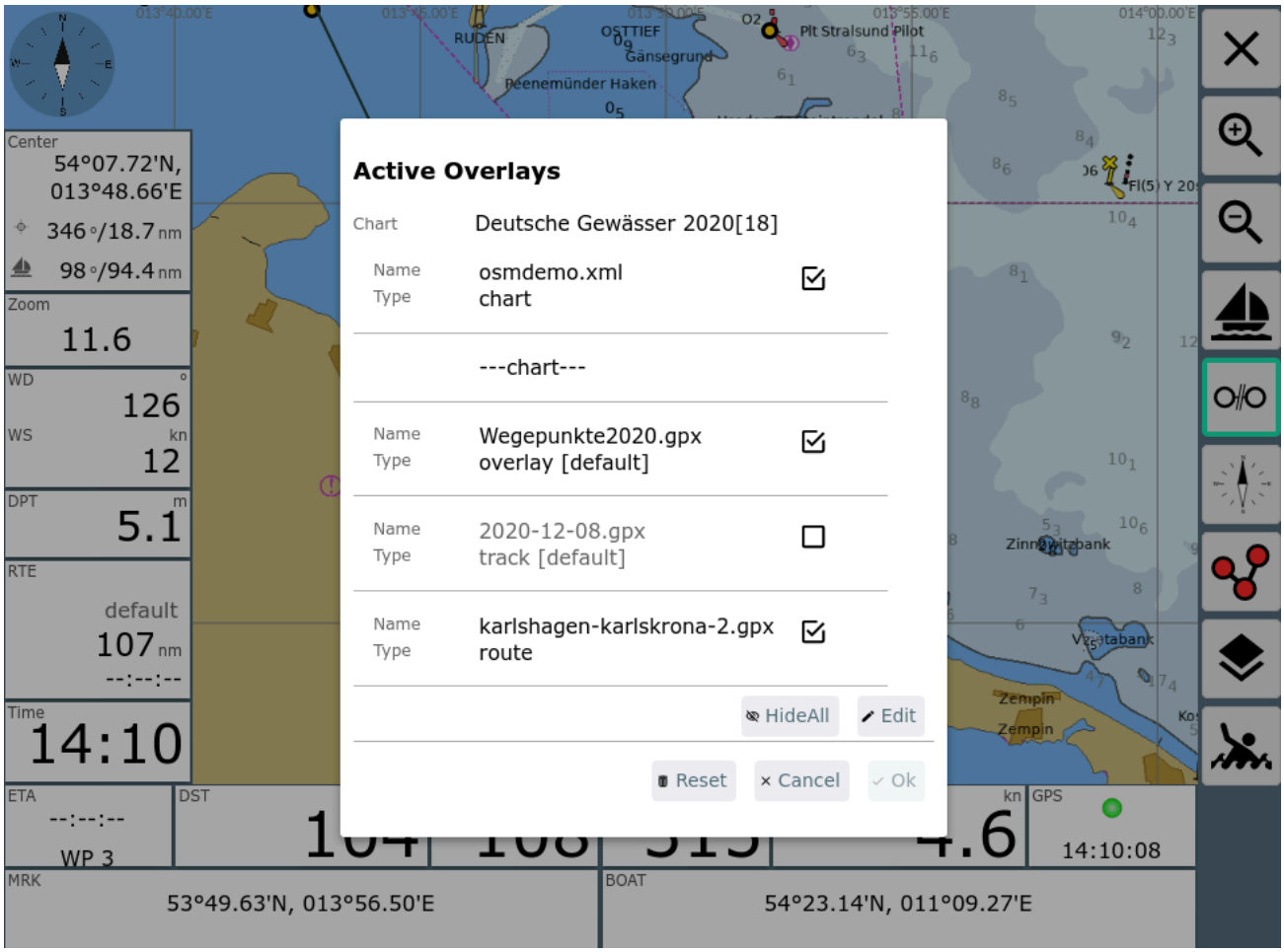
The selected point can be added to the route (before or after the currently selected point) or you can move the currently selected waypoint to the position of the shown point.

If the overlay is a route you can insert this route (starting from the selected point) into the current route. This way you can e.g. combine two routes.



At the Navigation Page and within the Route Editor there is an option to temporarily disable or enable overlays.

This is done using the button on the right side.



The enable and disable settings remain active until you change the chart. They are only valid for your current device (in contrary to all other overlay configuration functions that will always affect all connected devices as they are handled at the server).

By clicking the Edit button you can enter the normal overlay configuration.

Hint:

Using overlay files with a lot of elements can seriously impact the display performance. There is currently no fixed limit for overlay file sizes. Files with several 100MB will for sure lead to problems - 1000 points within an overlay are ok.

Changes of Overlays

Since version 20220225 AvNav will monitor the timestamp of the overlay files and will automatically update the chart display whenever an overlay file

changes.

Adaptations

Since version 20210114 you can adapt the information to be extracted from the overlay file and shown at "Feature Info". This could make sense as potentially overlay files contain information that otherwise cannot be handled by AvNav on its own.

For this purpose you can set a "featureFormatter" function when you configure the overlay. This featureFormatter is a java script function, either already built in into AvNav or implemented by you in your user.js It may also be provided by a plugin.

This java script function will retrieve all available parameters of the element from the overlay file that you have clicked on. It can return the properties to be shown in the "Feature Info" dialog.

You can return the following properties:

Name	Meaning
sym	The URL for an icon to be shown. This can be a relative URL referencing an icon file within the configured userIcons file. Alternatively it can be an absolute path like /user/images/myImage.png or an external url starting with http: or https: - this one can only be used with an existing internet connection, of course.
name	The name to be displayed
desc	The text for the "description" field.
htmlInfo	A html string, that will be shown when clicking the Info button.

time	A time value (string or java script Date)
link	An URL, that will be shown when clicking the ⓘ Info button (alternatively to htmlInfo). The same rules apply as for "sym".

The parameters retrieved by the function depend on the overlay file. Additionally you will always receive "lat" and "lon".

An example for the built in genericHtmlInfo function retrieving all available parameters and writing them into the htmlInfo property as HTML.

```
let genericHtmlInfo=function(properties,extended){
    if (! extended) return {};
    let htmlInfo='<div class="featureInfoHtml">';
    for (let k in properties){
        if (!properties[k]) continue;
        if (htmlInfo !== "") htmlInfo+="<br/>";

htmlInfo+=avnav.api.escapeHtml(k)+"="+avnav.api.escapeHtml
    }
    htmlInfo+='</div>';
    return {htmlInfo:htmlInfo};
}
```

The second parameter accepted by this function determines whether to compute all properties or only the "sym" property. If it is set to false you should not execute any time consuming operations, as this function will potentially be called for each of the overlay's elements.

To let AvNav know about your own formatter function you need to register it (see user.js).

```
avnav.api.registerFeatureFormatter('myHtmlInfo',myHtmlInfo
```

After registration you can select this function in overlay configuration.

AvNav WebApp User Doc

The web app can be started via the link http://avnav.avnav.de or http://avnav.local if you are connected to the Wifi provided by the raspberry pi. For some detailled hints on how to reach your AvNav server see the image description.

On Android calling the startpage is directly handled by the app.
As an alternate option you could also install a bonjour browser on your mobile device - this way you can connect without any typing of urls.
Apps for that case:

- IOS: 
- Android: 

The AvNav web app consists of several pages that can be entered either directly from the main page or via some sub pages.
After opening the App you will normally see the mainpage with the list of available charts. Since 20240616 you can enable with Settings->Map->"start with last map" to directly open the navigation page with the last used chart. If the chart is not (yet) available (e.g. provided by a plugin that did not complete the startup) it can take some time until the chart can be displayed. In this case AvNav will wait for the chart to appear and show a dialog informing you about this situation.

The links in the second column of the table will guide you to the descriptions of the pages.

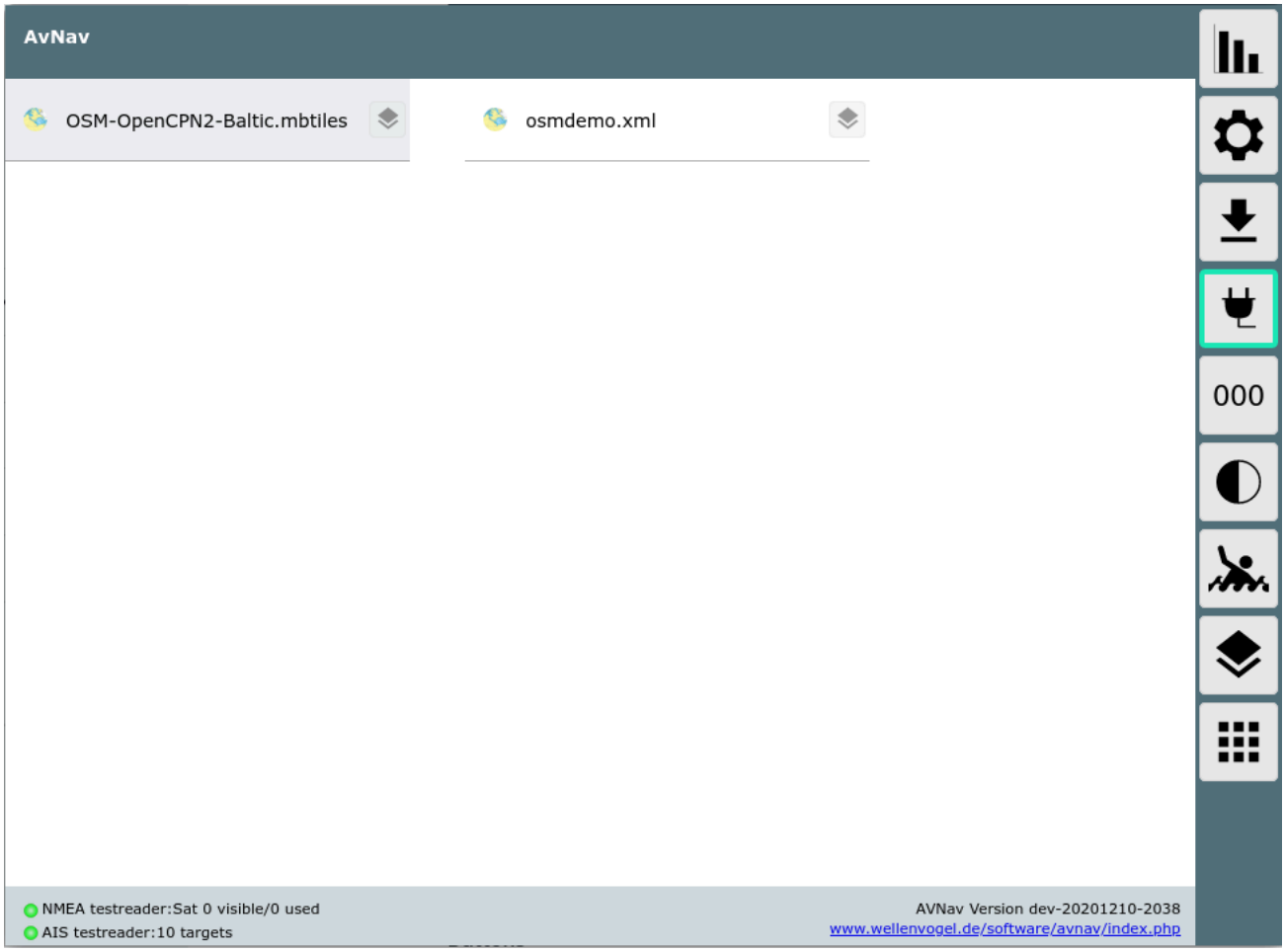
Icon	Page	How to get there	Function
	mainpage	opens at start	list of charts, NMEA status, go to further pages
	navigation page	click on a chart on the mainpage	base navigation functions, charts, instrument displays, waypoints, routes...
	server/status page	button on mainpage	status display for the server, changing the server configuration. Go on to wifi configuration, display of server addresses, shutdown, license and privacy info
	settings	button on mainpage	settings for the display in the browser, go on to layout editor, to user app configuration and to android settings (android only)
	files/download	button on mainpage	display, download, upload, edit of tracks, routes, charts, images, user files, layouts
000	dashboard	button on mainpage, click on	display of instrument data (up to 5 sub pages)

		some displays on navigation page	
	route editor	button on navigation page	creating and editing of routes
	user apps	button on mainpage, only visible if user apps are configured	display of internal or external HTML pages (e.g. the signalk web ui)
	route list	click on the display of the active route inside the route editor	display of all waypoints of a route, copy function, editing, selecting stored routes
	wifi configuration	button on status page ()	connection to en external wifi network (only if configured and a wifi stick is plugged in, not on android)
	display of server addresses	button on status page ()	display of the current server addresses, QR code to be scanned by external devices
	layout editor	button on settings page ()	editing of the displays on the navigation page and on the dashboard pages
	user app configuration	button on setting spage ()	configure external or internal HTML pages

			that should be shown as user apps
	ais info	click on the AIS display on the navigation page or an AIS target on the map	display of information about an AIS target, go on to the AIS list.
☰	AIS list	via ais info page	display of the AIS targets in vicinity as list, can be sorted
	Import Page	via Files/Download page, button ⇄	Chart import (conversion) - not on Android

AvNav Main Page

Overview



Buttons

Icon	Name	Description
	ShowStatus	status page for the server
	ShowSettings	settings page
	ShowDownload	files/downloads for download/upload/delete/edit of charts, tracks, routes, layouts, user files, images
	Connected	if active - sync waypoints and routes with the server, otherwise keep them locally only
000	ShowGps	dashboards
	Night	night mode
	MOB	man over board (only visible if connected)
	NavOverlays	editing the default default overlays
	FullScreen	Fullscreen on/off (supported browsers only)
	MainAddOns	display of configured user apps (e.g. signalk)
	RemoteChannel	Switch the remote control channel and mode
	Split	Switch Split Mode on or off

In the page main area there is the list of charts available on the server (files at /home/pi/avnav/data/charts on raspi, on android charts in the choosen

directory).

After first installation you will see some/one online demo charts. They can only be used when connected to internet, of course.

You can upload further charts via the files/download page or copy them directly into the correct directory (raspberry) or read them from a selected directory (android).

AvNav can read charts in the gemf format (preferred), since version 202003xx also in mbtiles format. You can also add online chart sources using an xml definition. For details refer to chart formats.

O-Charts need to be uploaded via the o-charts Plugin (reachable via the button).

If you have charts installed in SignalK, you will also see them here.

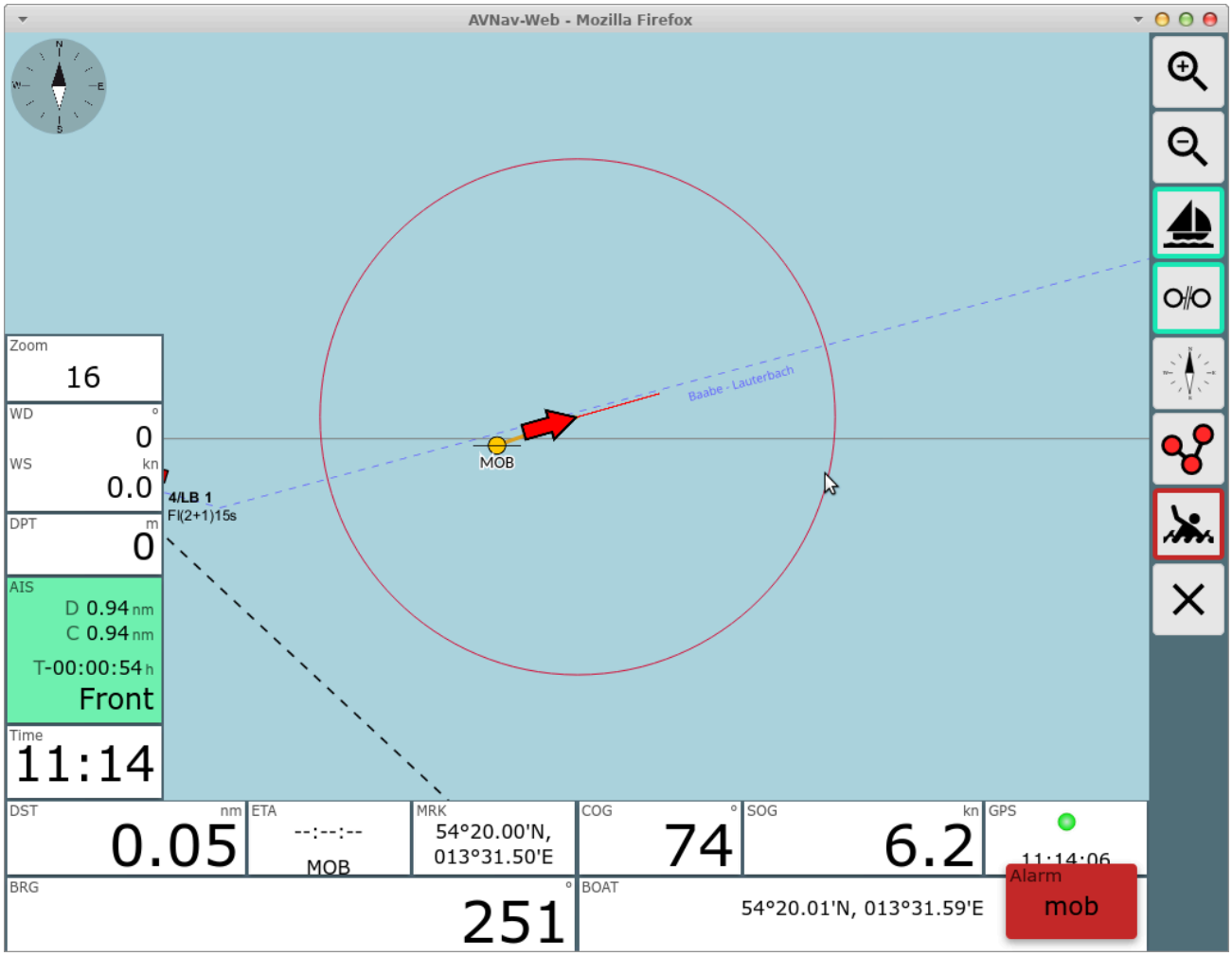
Next to each chart the button will open the overlay editor for this chart.

By clicking on a chart entry you will be taken to the navigation page with the selected chart set.

Special Functions

Man over Board

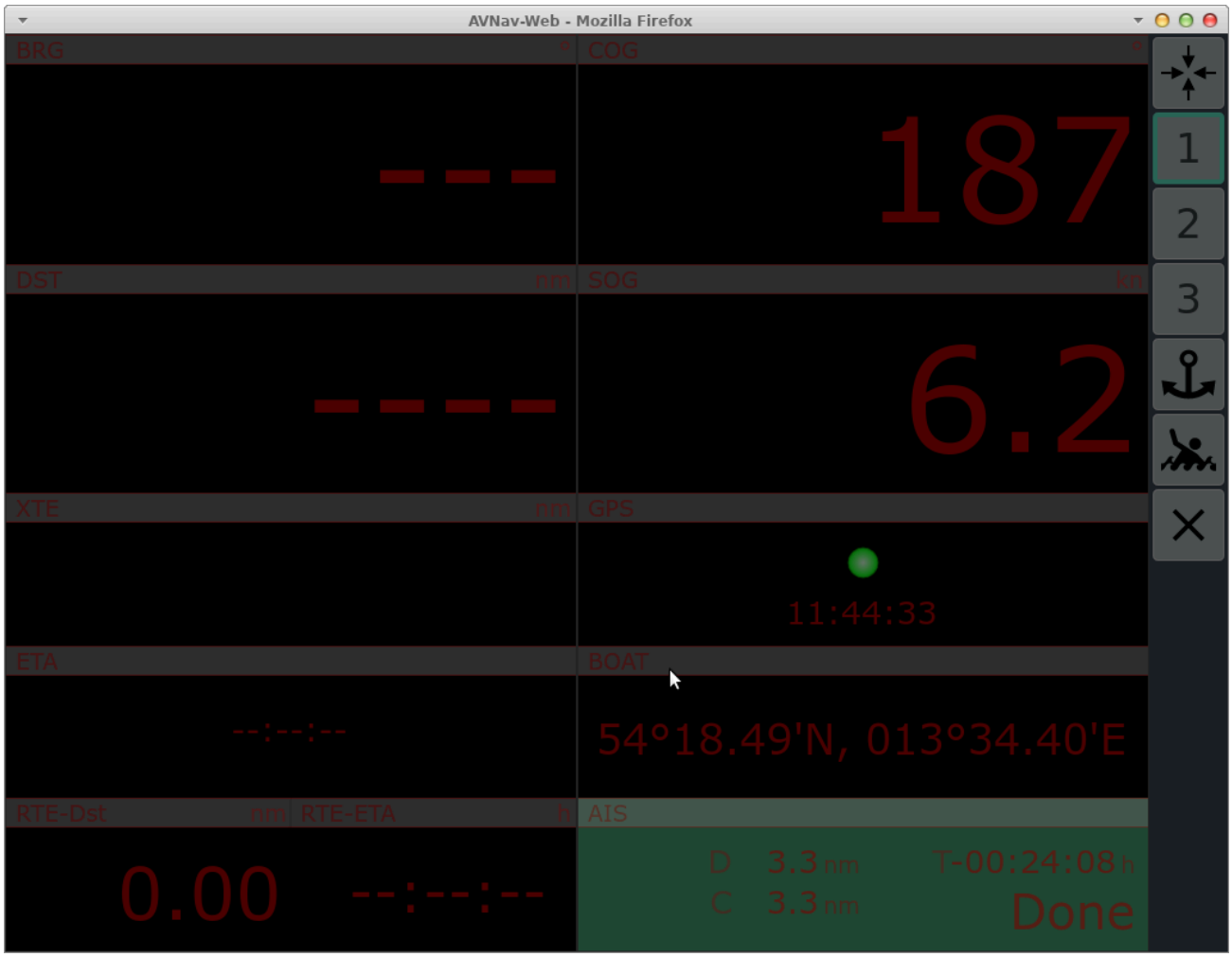
Visible on all pages - but only if we are in connected mode - button with green border.
By clicking on the MOB button the current position will turn into a dedicated waypoint named "MOB". Any active routing will be stopped and the route to the MOB waypoint will be activated. The display will switch to the navigation page with the last selected chart. The chart will be centered to the boat and a fixed zoom will be applied (can be preset in settings). Additionally a MOB alarm will be triggered. The alarm can be stopped. The MOB routing remains active until you click the MOB button again.



If there was no selected chart before the display will go to the dashboard .

Night Mode

By clicking the ◐ button you will activate the night mode. You can fine tune the dim factors in the settings.

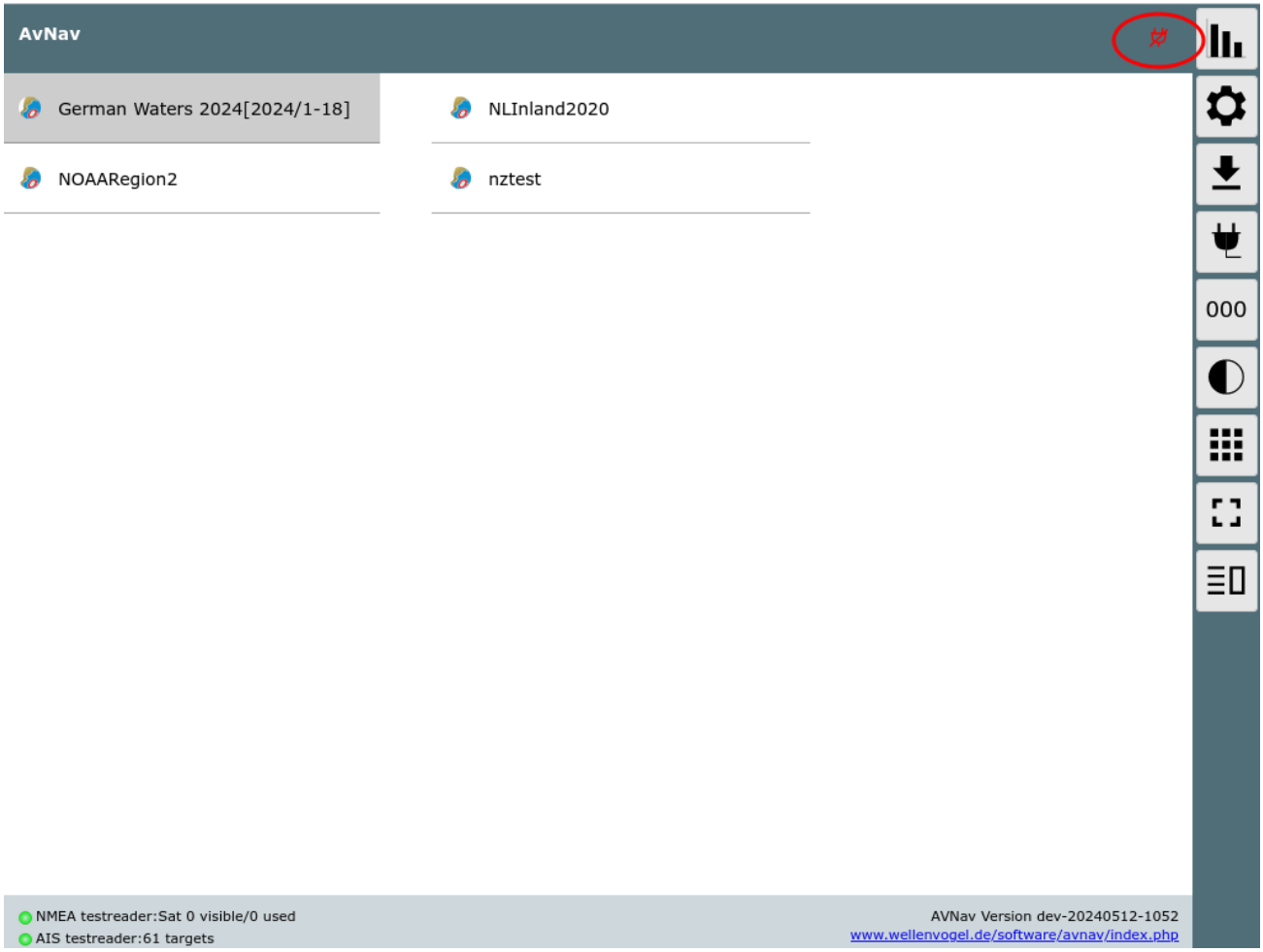


Connected/ Not Connected

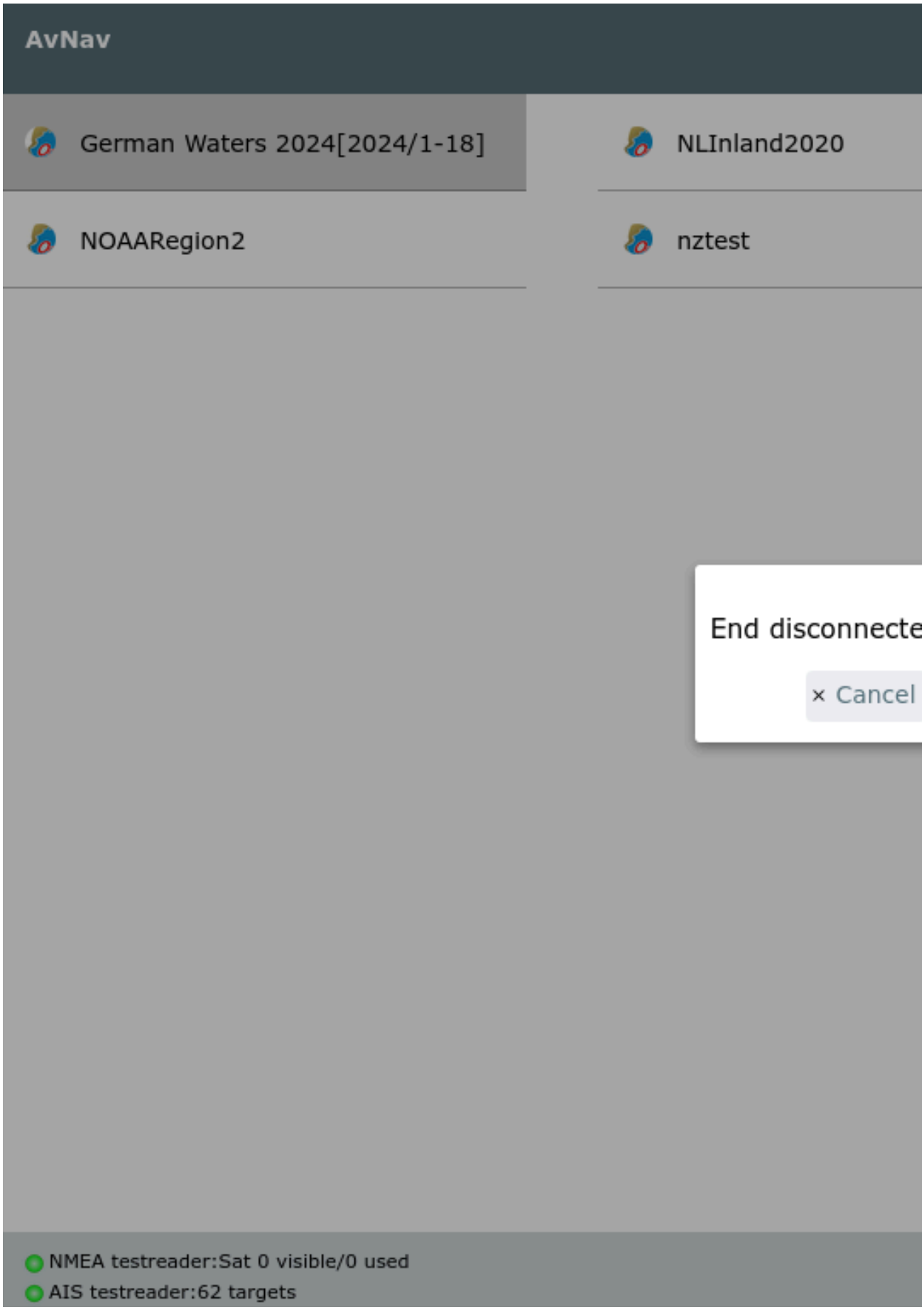
With the button you control how the route data is synced with the server. If it is active (green border) all changes on routing data (waypoints, start routing, stop routing, change route) will be synced with the server immediately. This way they will become visible on all connected displays.

If it is switched off, all routing changes will only be stored locally. In this mode you can e.g. try out a new route without disturbing the helms man or the auto pilot. If you connect later on, data from server will always take precedence. To keep your changes you need to store them e.g. in a new route with different name. If you are activating a route that only has been stored locally (and you are connected again) this route now will be synced to the server. At the files/download page the symbol next to the routes indicates if they are synced with the server.

The connected/not connected mode only affects routing data and waypoints. All other functions are always connected with the server (except MOB - this is a routing function and not available when disconnected).



Since 20240520 the title bar shows a small red icon if you are in disconnected mode. By clicking this icon you can bring up a dialog to end the disconnected mode.

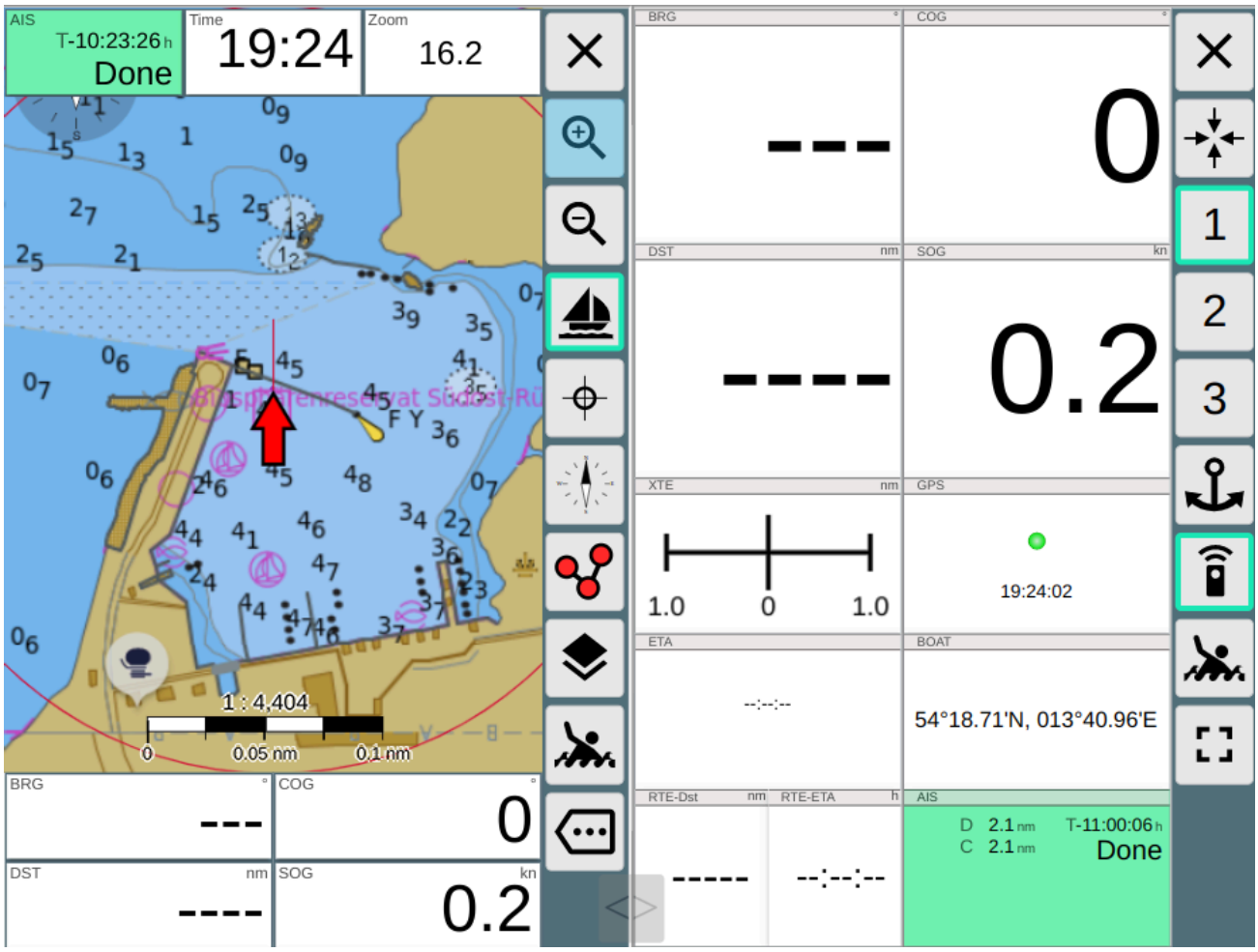


Anchor Watch

On the dashboard page you can activate an anchor watch.

Split Mode

Since version 20220819 AvNav can split it's main window. In this mode there are two instances of AvNav running in the same browser.



Both instances are mainly independent from each other and can be operated separately.
They share most of the AvNav settings.
Only a couple of settings will be handled separately for each of the instances (left/right). On the settings page they will be marked with a *.
If you change settings (except the ones with a *) the changes will become visible on the other instance immediately.

The following settings are specific per side:

Name	Meaning

connectedMode	The "plug button" - if it is switched of the instance cannot make any changes at the server. You can test a route e.g. in this mode without influencing the server.
layoutName	The layout to be used
remoteChannelName	The name for the remote control channel
remoteChannelRead	Read from the remote control channel
remoteChannelWrite	Write to the remote control channel

If you would like to have other/more settings to be specific per side you can create a file splitkeys.json in the user directory. The default file can be found at GitHub. The names for the keys must be taken from the source code in the properties section. All preoperties you list in the splitkeys.json will afterwards be handled separately per side.
The settings that are specific per side will not be kept for the normal display when leaving the split mode (but will be kept for the next split mode entering).

Alarms will only be handled by the right instance.

Using the Split Button on each instance you can leave the split mode.

You can drag and drop the slider to change the window split.

Since 20240616 you can force AvNav to start in the same mode (split/unsplit) like it was left by enabling Settings->Layout->"start with last split mode".

AvNav Chart Import Page

You can configure the settle times for files and directories and the scan interval. The default interval of "0" will let the importer only scan once every 24h - but on every upload a scan is triggered.

If you copied some files to the import directory by hand, just click the  button beside the scanner and trigger a rescan.

Copying Files(Experts)

If you run into trouble uploading large files you can copy the files (or even complete directories) to the importer directory by hand. You will find it at DATADIR/import. DATADIR will be $HOME/avnav or $HOME/avnav/data depending on your set up.
After you copied a file or direcftory you need to trigger a rescan by hand.

To avoid copying lare files or directory trees you can also create a file with the extension ".clk" (converter link file) in the importer directory.
Inside the file just put one line with the absolute path to the file or directory you would like to convert.
Don't forget to trigger a rescan after you saved the file.

Eingebaute Konverter

AvNav bringt die folgenden Konverter mit.

.kap (BSB)

For kap (BSB) charts you can pack multiple files into a zip archive and upload this archive.
The connversions has the following steps:

- Sorting the charts into layers (with possible conversion)
- Creation of the tiles
- Building the gemf file

Details (Experts only)

Here are some details about the conversion steps of the internal kap files converter. Normally you do care about them.
But if you are unhappy with the results of the conversion you still could run the conversion using the command line.
The read_charts.py is located at /usr/lib/avnav/chartconvert.

> *The first step is fast. All chart data will be read, resolution and range will be detected. If necessary charts would be converted/reprojected. As a result there will be a chartlist.xml in workdir/<name> . The command line would be:*
>
> ```
> read_charts.py -g -o name -m chartlist inputdir [inputdir...]
> ```
>
> *Afterwards you can check the chartlist.xml and you could e.g. move some charts into a different layer.*
>
> *The second step will take more time - now the tiles are created:*
>
> ```
> read_charts.py -g -o name -m generate inputdir [inputdir...]
> ```
>
> *The file chartlist.xml must already exist at workdir/<name>. The creation will run multi threaded.*
>
> *At the end you need to create the gemf file:*
>
> ```
> read_charts.py -g -o name -m gemf inputdir [inputdir...]
> ```

> *You could also combine all steps:*
>
> ```
> read_charts.py -g -m all [-o name] inputdir [inputdir...]
> ```

.mbtiles

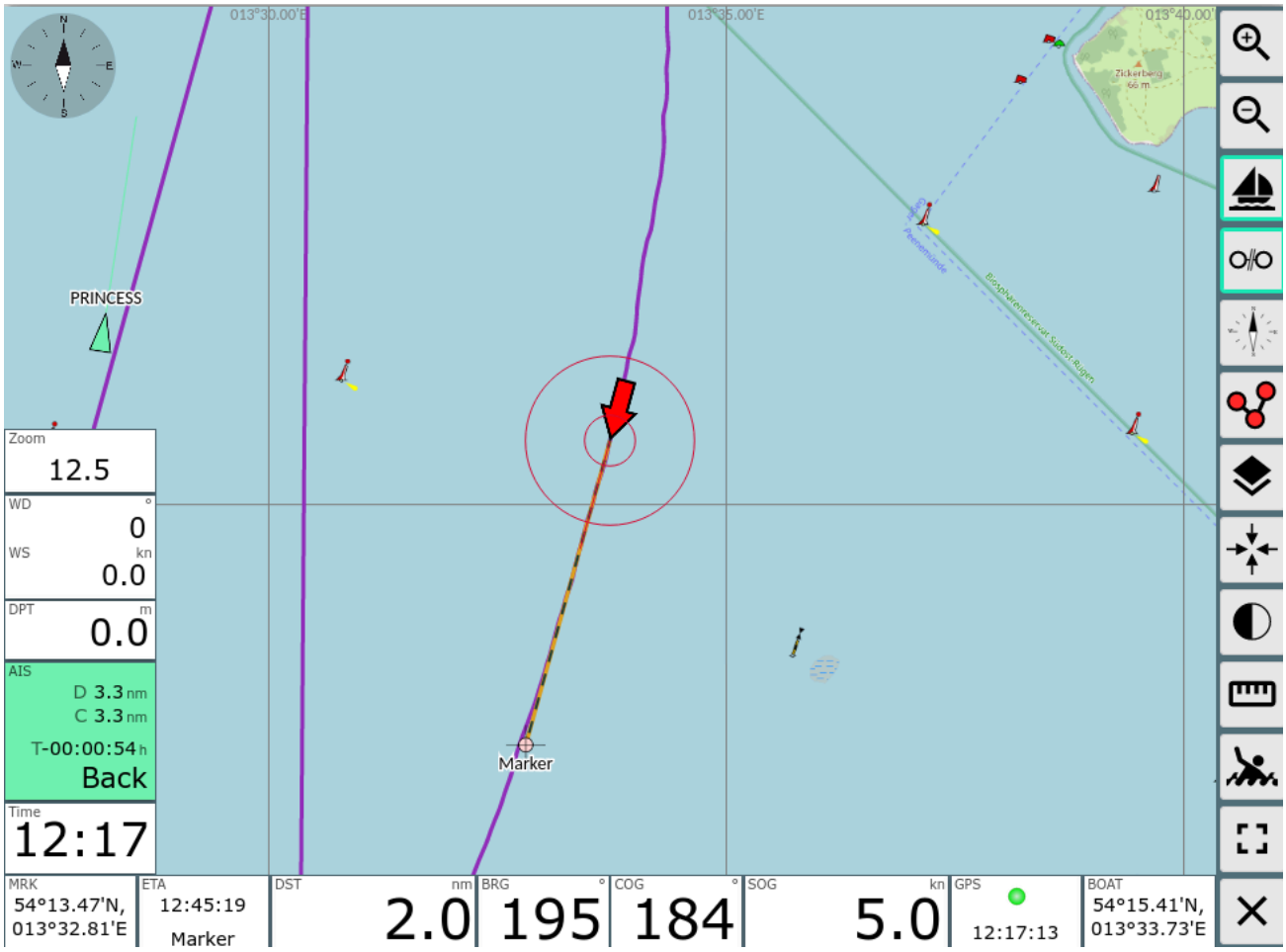
In principle AvNav can handle .mbtiles files directly. But ot could make sense to convert them to .gemf any way - e.g. for copying them to the external charts directory on Android.
Until now the converter can only handle mbtiles with the default xyz scheme.

.navipack

Conversion into gemf.

The Navigation Page

This page is the one normally used for navigation. In the screenshot an active waypoint is shown and the chart is locked to the boat position (i.e. will move with the boat).



Buttons

Icon	Name	Funktion
	ZoomIn	Zoom in

	ZoomOut	Zoom out
	LockPos	lock the chart center to the boat position can only be activated if there is a valid position
	StopNav	stop the navigation only visible if a waypoint or route is active
	LockMarker	start waypoint navigation. the center of the chart (cross) will become the target waypoint. only visible if there is no active route or waypoint
	CourseUp	rotate chart - course forward will be at top
	ShowRoutePanel	switch to route editor (also by clicking on the route display if there is any)
	MOB	man over board (see main page)
	FullScreen	Fullscreen on/off (supported browsers only)
	Measure	set a marker at the current map center and show a line to the center together with course and distance (also in the "center Display" widget) - measure tool
	Overflow	Display a second button list if the screen is too small to fit all buttons. Only visible if you did not select "2 button columns" at Settings/Layout.

	NavOverlays	Activate or Deactivate Overlays
	Night	Activate/Deactivate of the night mode (since 20210619, if Settings/Buttons/night mode on navpage is enabled)
	GpsCenter	Center map to boat position (since 20210619)
	Dim	Dim Mode. The screen will be dimmed and all buttons become inactive. Leave this state by clicking anywhere on the screen. This button is only visible in the Android app or when using the BonjourBrowser (version 1.5 and above). This button really dims the complete screen. This way you can limit the power consumption of your device if you do not need an instant display. It can also prevent overheating when running on high brightness and on high temperatures.
	RemoteChannel	selection of the remote control channel and mode (send/receive).
	Cancel	back to main page

This is the navigation view. In the middle there is the chart display with the boat position (red arrow). The yellow and green symbols are the AIS targets in vicinity (10nm, can be changed in settings) together with their current course as well as name or MMSI. The orange line indicates the course towards the current target waypoint. The dotted line show the original course from the start of navigation to the target waypoint. You can move and

zoom the chart with the normal gestures. For zooming you can also use the buttons +/- on the right side.

You can have up to 3 Navigation Circles being drawn around the boat to be able to estimate distances. You can change them via settings->navigation display. Standard values are 300m and 1000m.

(since 202011xx) The length of the line indicating your current course ("Course Vector") can be adapted in the settings (Navigation/Boat Course Vector Length). You will set the seconds and the length will be computed using your current boat speed (default: 10 minutes). The width of this line is controlled by the width for the Navigation Circles.

The same settings also apply to the course vectors of AIS targets.

For the boat direction and the course vectors you can select various modes (since 20220421) - you can choose at settings->navigation->boat direction (see the GitHub discussion):

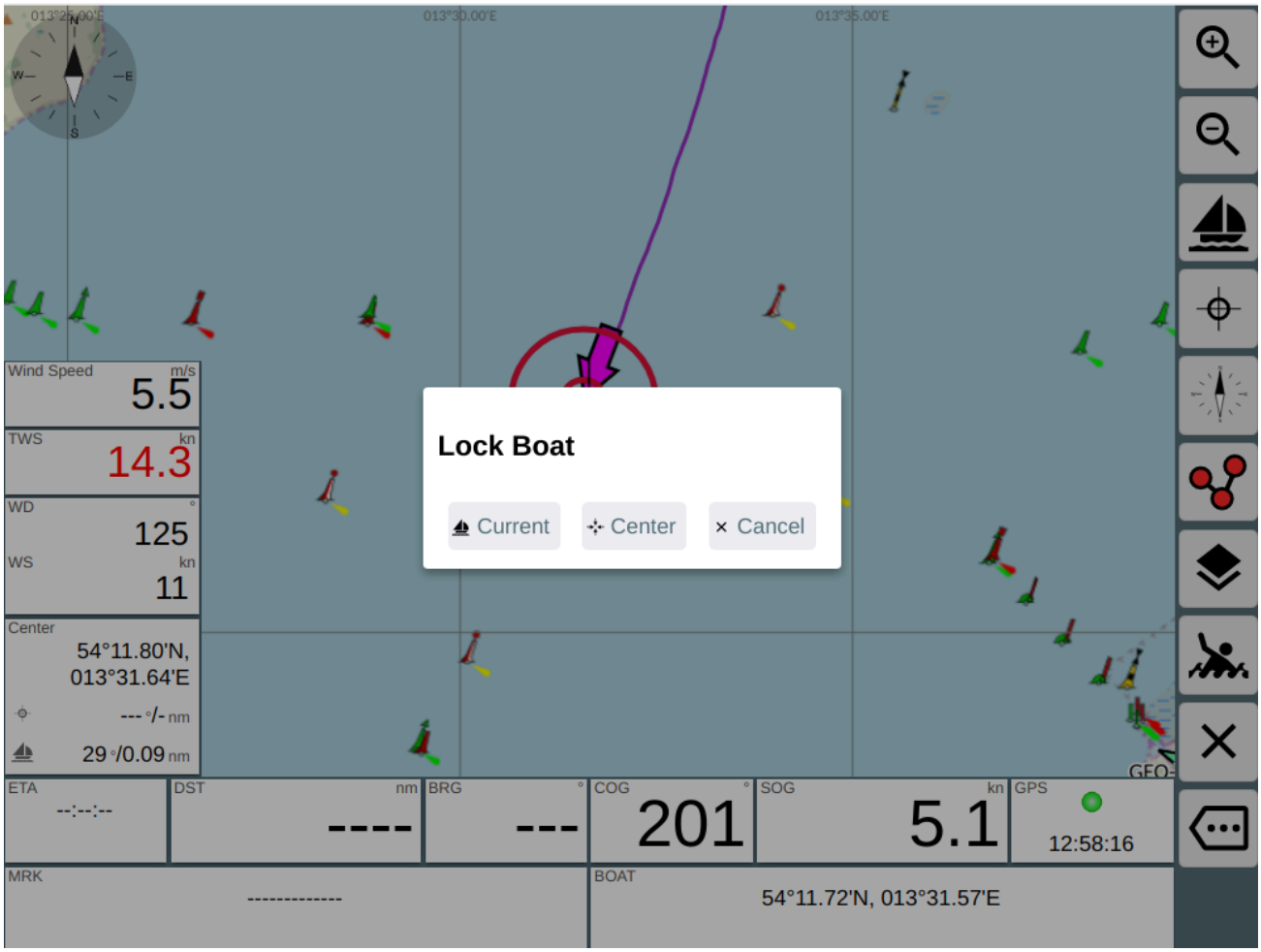
Setting	Meaning	Symbol-Name
cog	boat direction and course vector COG	boatImage (arrow)
hdt	boat direction based on Heading True (fallback to COG if not available), course vector based on COG. Optionally HDT as dotted line(settings->navigation->add dashed vector for hdt/hdm)	boatImageHdg (Boat)
hdm	boat direction based on Heading Magnetic (fallback to COG if not available), course vector based on COG. Optional HDM as dotted line(settings->navigation->add dashed vector for hdt/hdm)	boatImageHdg (Boat)

In the settings you can additionally activate a "no boat movement" detection. (settings->navigation->zero SOG detect). If this is activated and the boat speed (SOG) drops below 0.2 kn (settings->navigation->zero SOG detect below (kn)) the boat symbol will change to a red circle (boatImageSteady).

Lock Mode

If the chart is locked to the boat position (like in the picture) it will always keep the boat in the center of the screen and move the map accordingly.

Since version 20230614 you can allow (settings/Map "allow move when locked") to move the map when locked.
When you finish moving the map the boat will be held at the current position on the screen.
But it will be ensured that the boat at least is visible.
Since version 20210619 you can lock the boat position to any place on the screen even initially. You need to set (Settings/Map/boat lock mode) either "current" or "ask" .
With "current" you will keep the boat at the point at the screen where it is when you "lock".
With "ask" there will be a dialog at every "Lock" :



AIS Target Display

AvNav shows AIS Targets up to a configured distance to the current vessel posiion.
The display includes some information about the target (configure it at the settings page, AIS) and some motion vectors that show the movement of the AIS target.

AIS target symbols can be replaced by user defined symbols - optionally different ones depending on the AIS shiptype. For the own boat and optionally for each AIS target a course vector points to the position to be reached within a defined time interval (default: 10 minutes).

Since 20230614 AIS atons will also be displayed (you need to have settings/AIS "only show moving targets" switched off and "show other" switched on). For each AIS target you can display an estimated position

depending on the age of the received information, course and speed (settings/AIS "show estimated position").
You can select if an AIS target should be rotated by its HDG (if received - settings/AIS "use heading for direction") - otherwise COG will be used. The course vector of an AIS target will always be rotated by COG.
There are different symbols for AIS targets - for details refer to "user defined icons".
If you enable this in the setting the targets will also show an estimated position (shadow) based on the age of the AIS information.

By clicking an AIS target (or the "next AIS" display) you will be shown all information for this target and you can change to the list of all AIS targets.

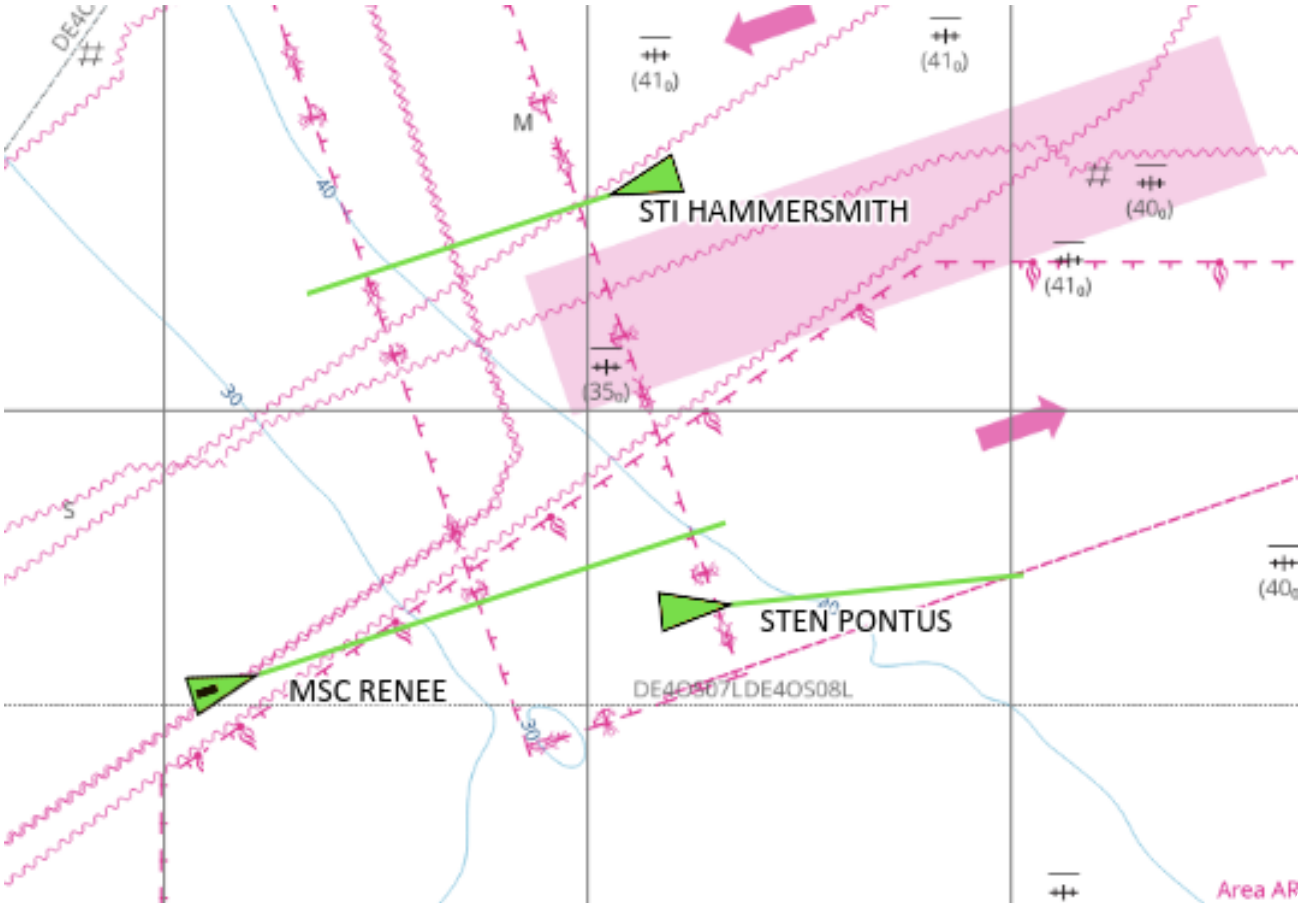
AIS Motion Vectors

A basic introduction to the topic of true and relative motion vectors and how they are used in navigation can be found at

- https://msi.nga.mil/Publications/RNMB (page 59)
- https://www.youtube.com/watch?v=8YUic4LdWFg

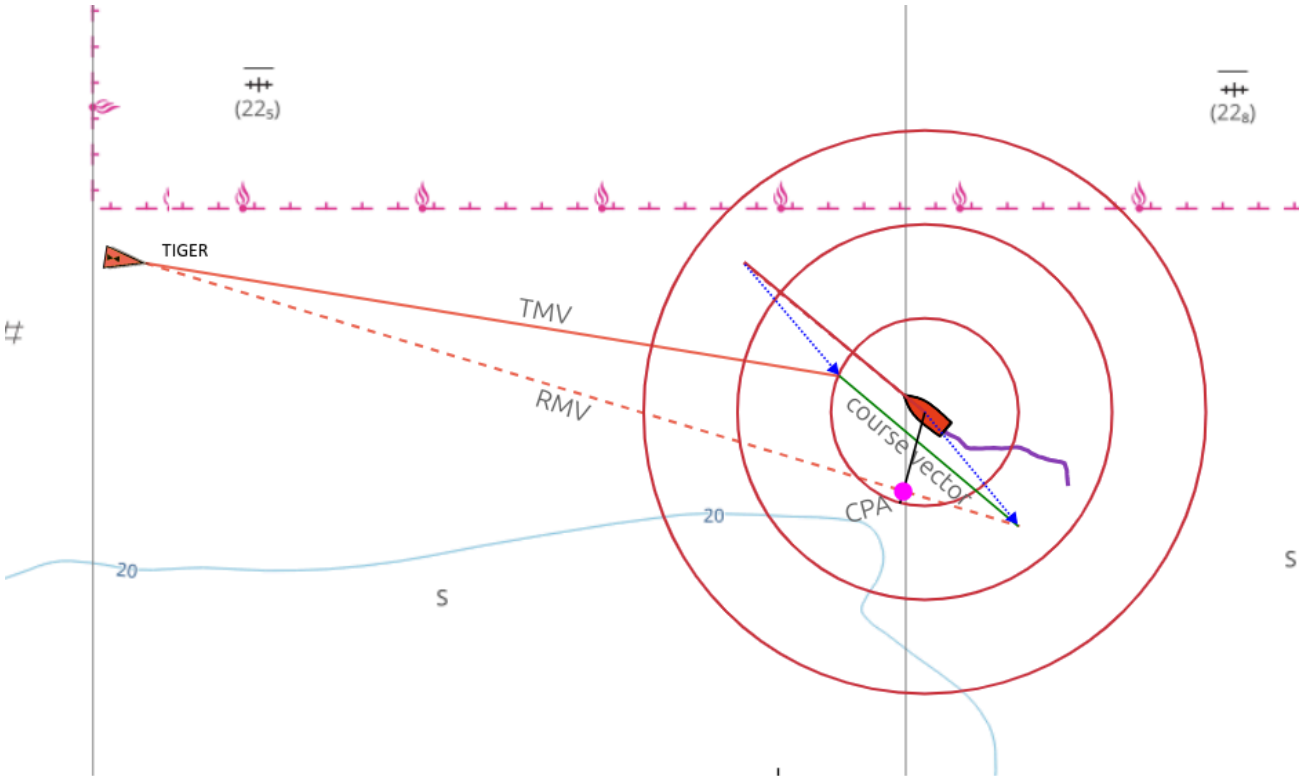
True motion vectors

AvNav displays the estimated track over ground for AIS targets if settings->AIS->use-course-vector is activated. A line is drawn from the last known position of the target in the direction of the course over ground (COG) of the target with the length of the course over ground (SOG) multiplied by boat-course-vector-length. This line is the so-called *true motion vector*, or TMV for short.



Relative motion vectors (since 20240520)

In addition, *relative motion vectors* can be displayed in AvNav. To enable this, set a value greater than zero in settings->AIS->relative-motion-vector-range, then RMVs are also displayed as dashed lines for targets that are within this distance.

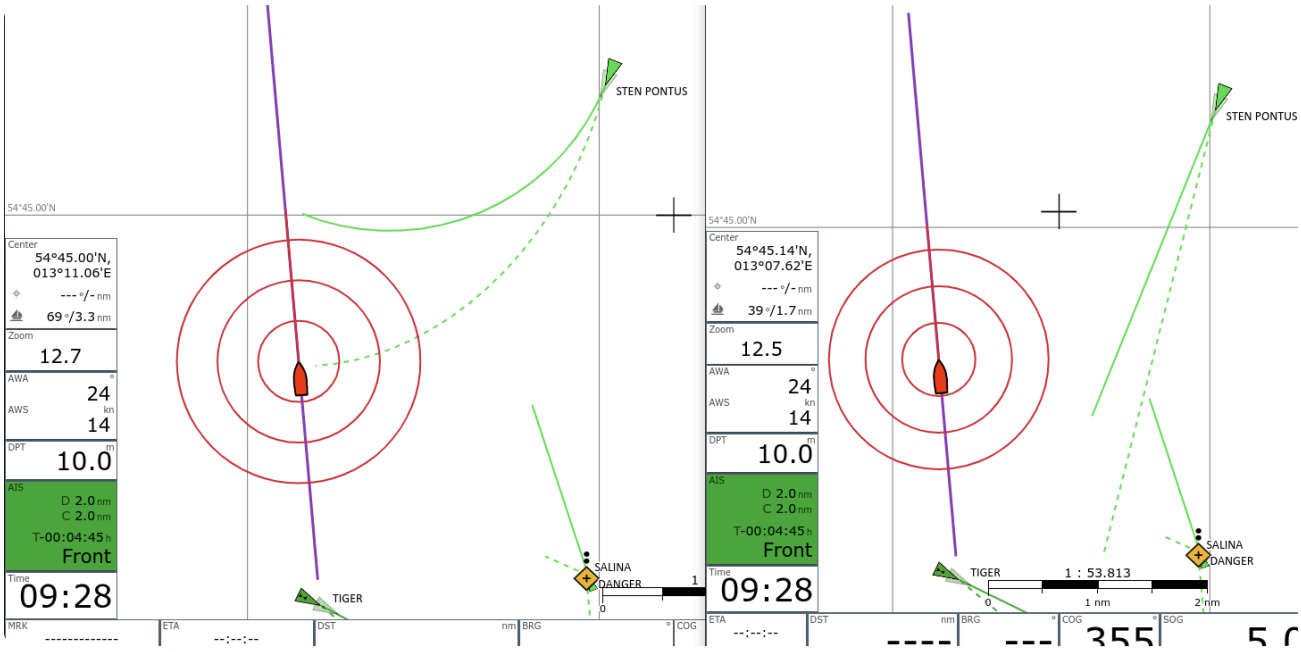


The RMV shows the movement of the target *relative* to the own ship, it results as the difference between TMV and the own course vector, so that TMV, RMV and the own course vector form a triangle. If the RMV of a target points directly at your own ship, there is a risk of collision. The position of the CPA can also be read directly from the RMV; the crossing of a line perpendicular to the RMV going through your own ship with the RMV.

The RMVs correspond to the tracks that the targets would leave on a radar screen.

Curved vectors (since 20240520)

If you activate settings->AIS->curved-vectors, a rate-of-turn (ROT) present in the AIS data is evaluated and the rotation of the target is taken into account when displaying the vectors. The TMVs and RMVs are then displayed as curved lines. The curved vectors may indicate a potential collision much earlier than the uncurved vectors.



Displays (Widgets)

At the left side you have (top down):

- current chart zoom, preferred zoom factor in brackets if auto zoom activated
- the closest AIS target (green), turning red on AIS warning or the selected AIS target (yellow)
- current time

The display of the closest AIS target will turn red if expected CPA is closer than 500m (can be changed in settings). If the display is yellow it does not show the nearest target but a different one selected on the ais info page. If you click on this display you move to the ais info page.

(since 202011xx) The icons used for your boat and the AIS targets can be changed using a file images.json. Additionally you can scale the AIS symbols (AIS/Icon Scale) and you can define a border(AIS/Border Width). You can also switch off AIS/AIS Use Course Vector, if your browser is too much slowed down by the computation and drawing of the AIS course vectors.

The most important instrument data are shown at the bottom of the navigation page. At the left, data for currently active waypoint (if any) are displayed

- Position
- ETA
- Course
- Distance (nm)

On the right the boat data:

- Course
- SOG (kn)
- Position
- local time from GPS
- GPS indicator: green - gps data ok, red: no gps data

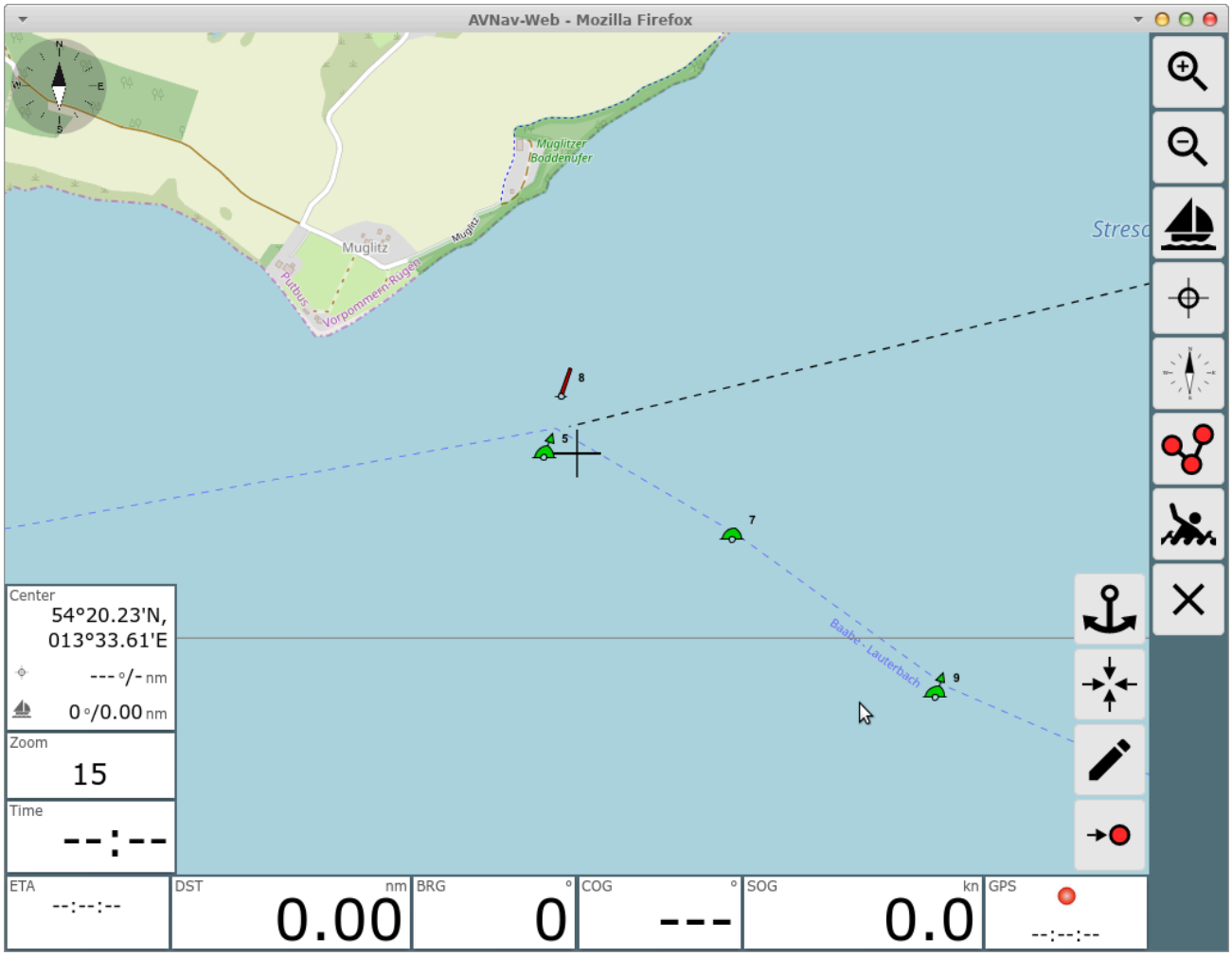
Depending on display width and font size setting (widget font size) you will have 2 rows of data (you can change this in settings at "2 widget rows"). Items that do not fit anymore will be hidden completely.

You can set up averaging (settings->navigation) for the boat data (position, course, speed). If turned on the captions at the displays will turn red.

All display can be adapted with the layout editor.

If you click on the lower right displays you will be taken to the dashboard.

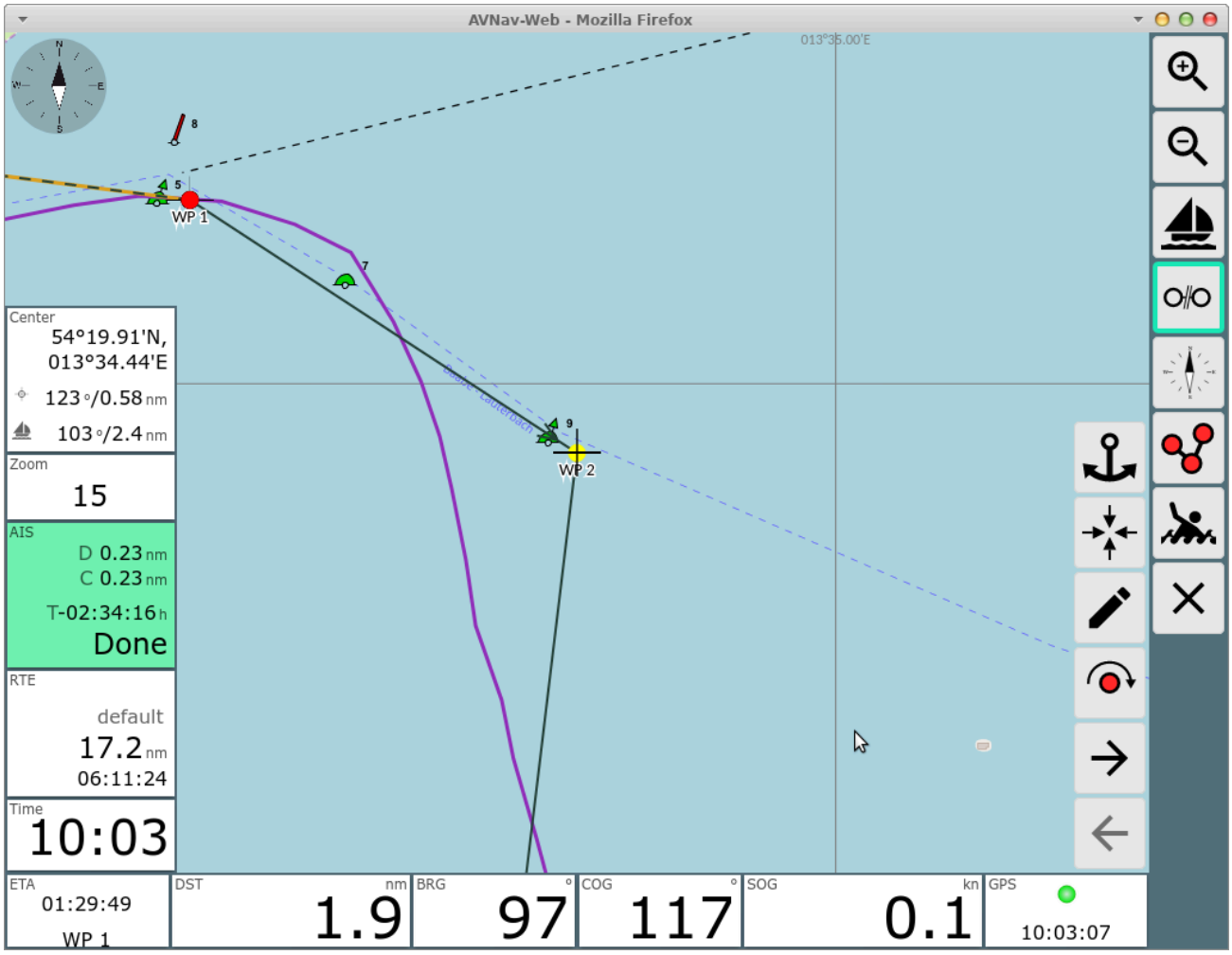
Clicking on the lower left displays will bring up a couple of additional waypoint buttons.



Icon	Name	Function
	AnchorWatch	switch on anchor watch (see dashboard)
	WpLocate	move chart center to waypoint
	WpEdit	edit waypoint you can change the waypoint's name and position in the dialog
	WpGoto	start navigation to this waypoint only visible if there is no active route
	NavRestart	restart navigation to the current waypoint. This especially sets a new course and restarts the XTE computation.

	Will only be visible if a navigation is active currently.

If a route is active, the waypoint buttons will change slightly.



The additional buttons

Icon	Name	Function
	NavNext	start navigation to the next waypoint of the current route
	WpNext	center chart to next waypoint
	WpPrevious	center chart to previous waypoint

	NavRestart	restart navigation to the current waypoint. This especially sets a new course and restarts the XTE computation. Will only be visible if a navigation is active currently.

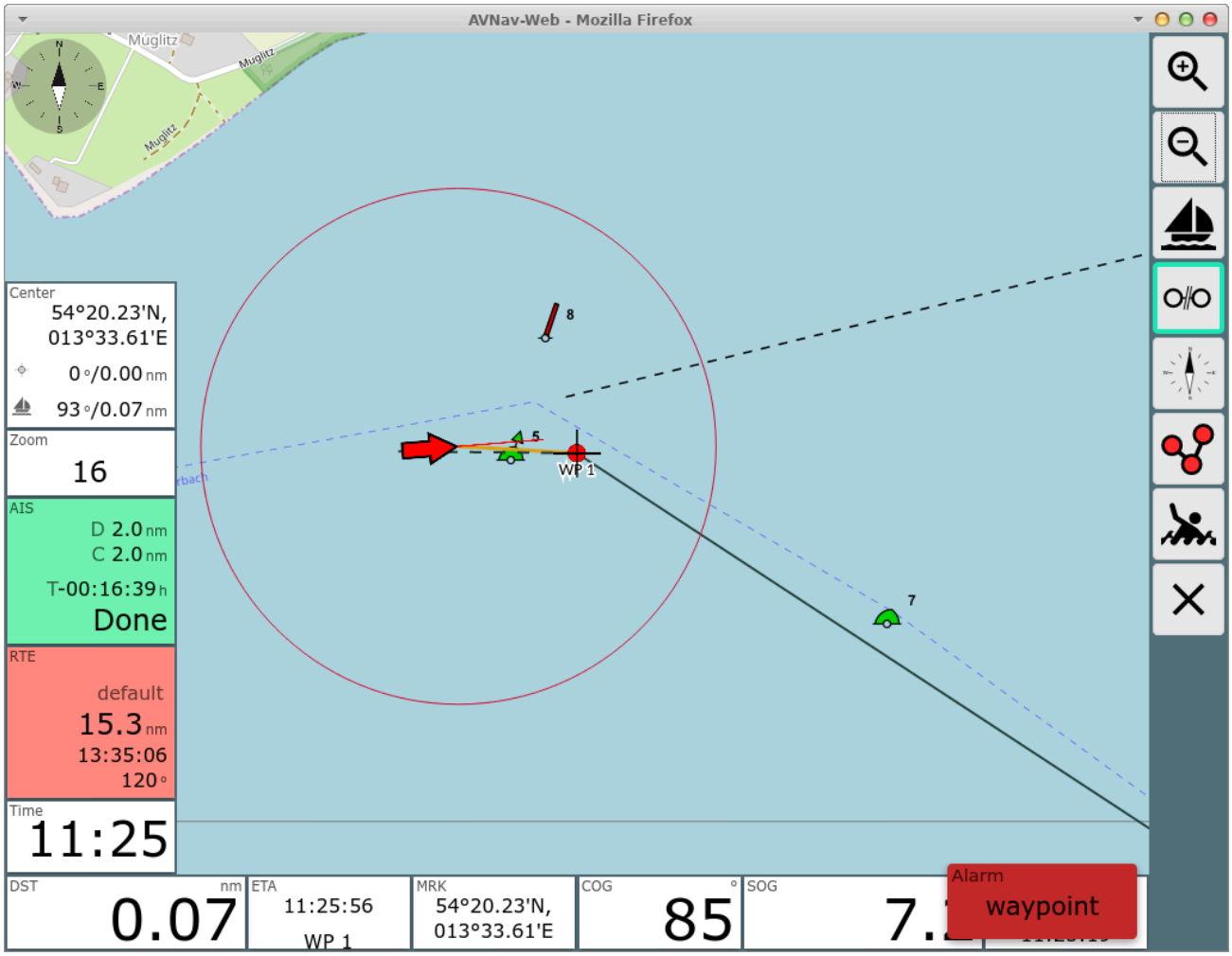
If a route is active there will be a display for the route data on the left side (name, remaining distance, ETA).

In routing mode the next target waypoint will be activated automatically if the following conditions are met:

- the boat is within the approach radius to the current target waypoint (default 200m - settings->route->approach)
- the second condition depending on the selected mode is met (early/90/late) - see Next Waypoint Handling

In approach to the waypoint the display of the route parameters will turn red and the course to the next waypoint will be shown additionally. Additionally a waypoint alarm is triggered.

If next waypoint is not activated automatically (e.g. because you are too far), you can click on the waypoint data and use the  button to set the next waypoint as target.



Special Functions

Simple Waypoint Navigation

Steps:

1. Unlock Chart (if on)
2. Stop Nav
3. Move chart until your target is below the center (cross), use zoom if necessary
4. LockMarker (start navigation)
5. Lock Chart

Bearings

If a waypoint is active and the chart is not locked to the boat position you have a "Center" display at the left side. This shows the center position, the

course and distance from the boat to the center and from the waypoint to the center. For a simple bearing just move the chart center (cross) to the bearing target and read the bearing from the "Center" display.

Feature Info

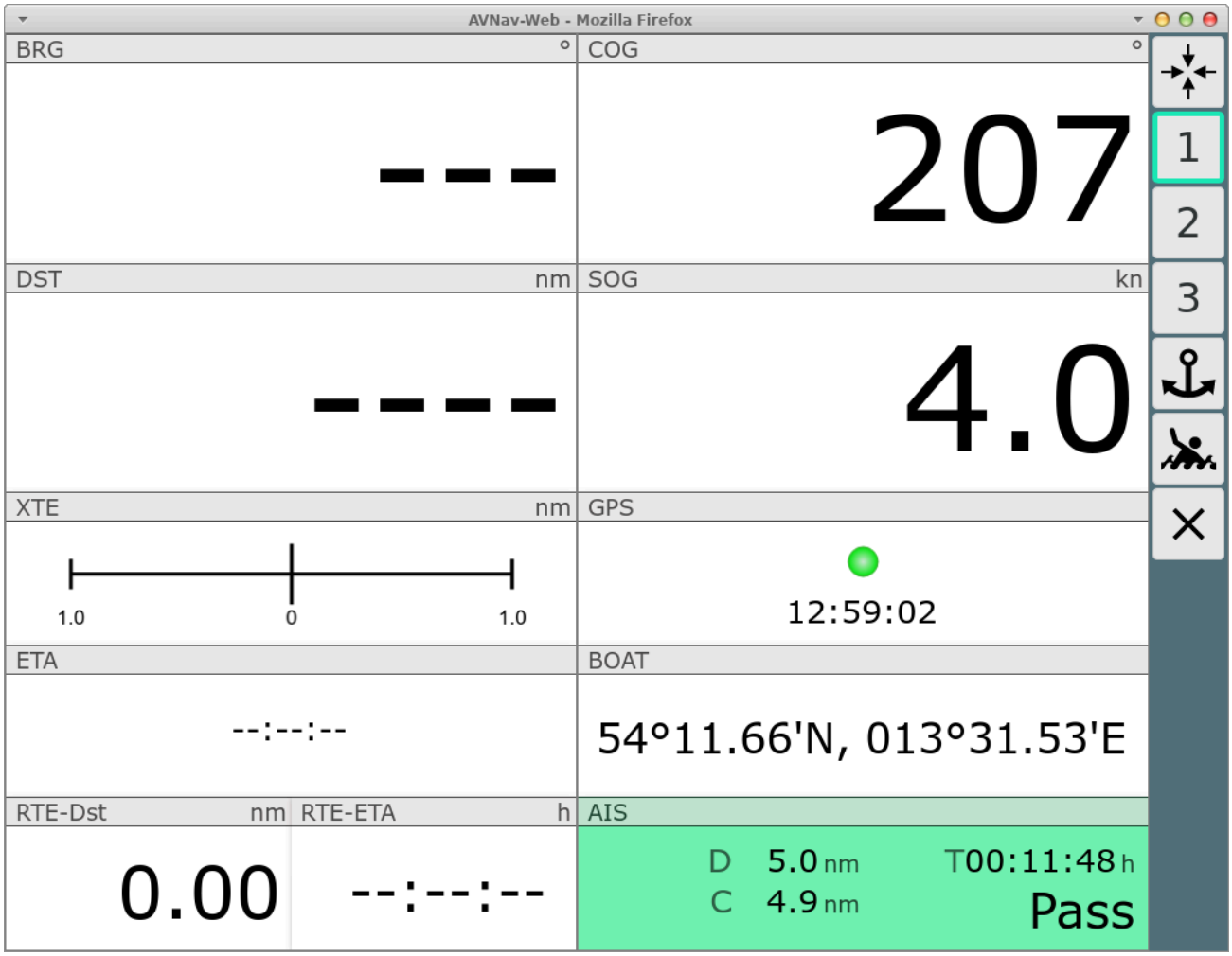
When clicking on the map a feature info dialog will pop up. Within this dialog you also have a button to immediately start a navigation to the clicked point.

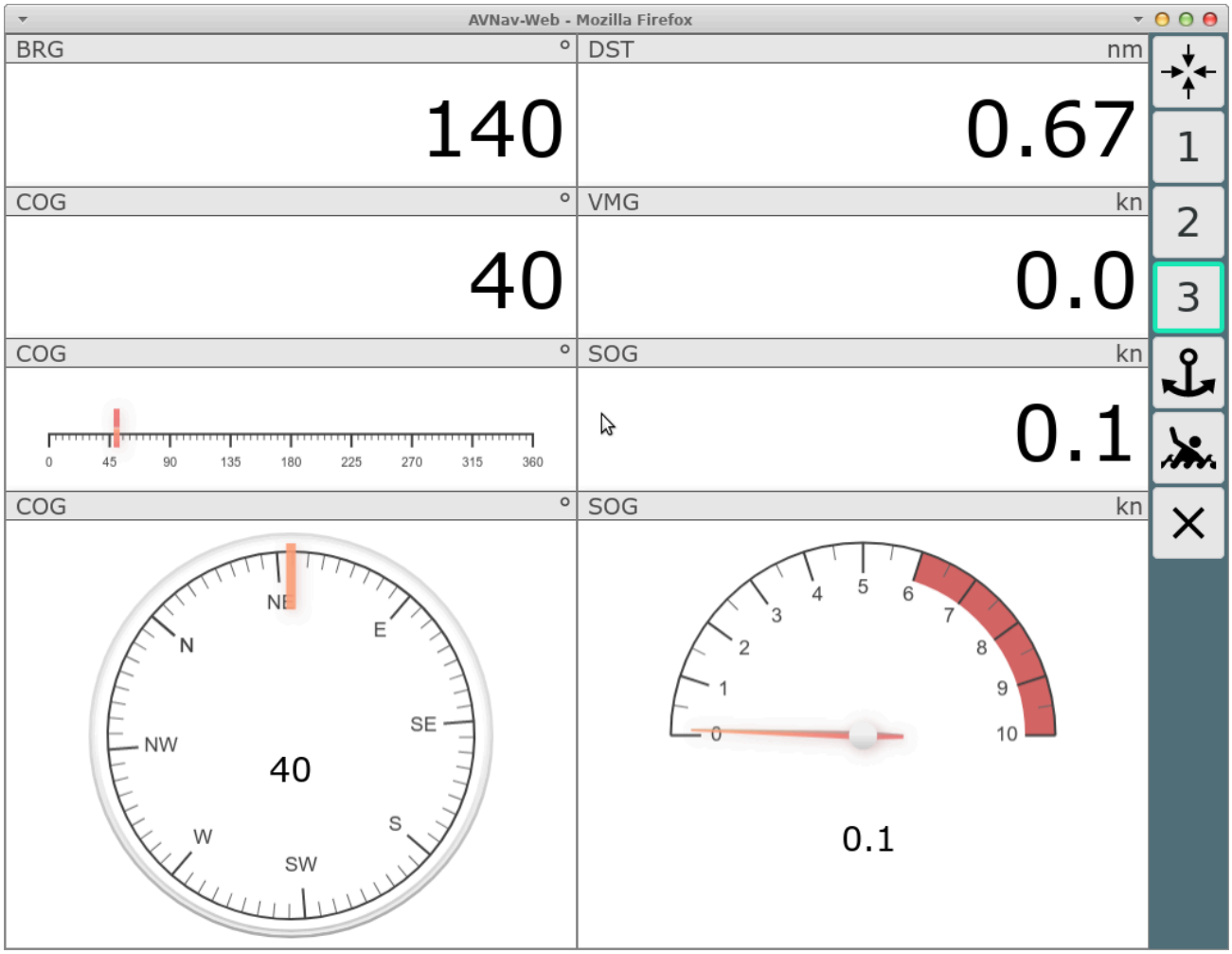
To start a route - change with  to the route editor.

The Dashboard Page

By pressing the button (000) on the mainpage or by clicking on the display's right bottom area on the navigation page you will see this page. No active chart is required to display this page.

You can configure up to 5 dashboard pages in the layout.





Buttons

Icon	Name	Function
	GpsCenter	center chart to waypoint and go back to previous page
1,2,...	Gps1, Gps2, ...	select the dashboard you would like to display. You can customize the display by selecting an available layout at settings->layout or you can adapt the layout using the layout editor.
	AnchorWatch	activate anchor watch, see below
	FullScreen	Fullscreen on/off (supported browsers only)

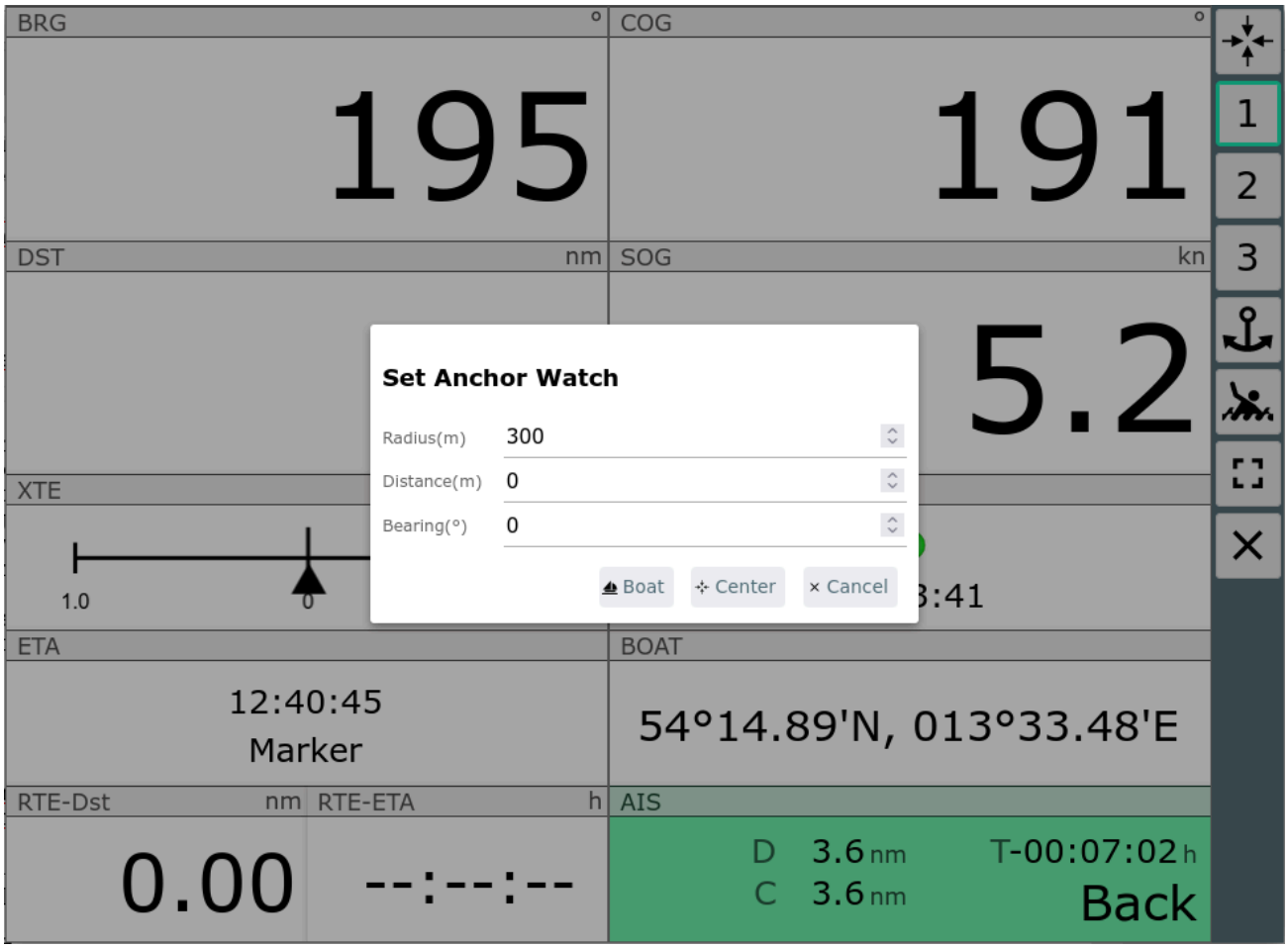
	MOB	Man over Board (see main page)
	Overflow	Display a second button list if the screen is too small to house all buttons. Only visible if you did not select "2 button columns" at Settings/Layout.
	Dim	Dim Mode. The screen will be dimmed and all buttons become inactive. Exit dimmed mode by clicking anywhere on the screen. This button is only visible in the Android app or when using the BonjourBrowser (version 1.5 or later). This button really dims the complete screen. This limits power consumption of your device if you do not need an instant display. It can also prevent overheating when running on high brightness and on high temperatures.
	Cancel	back to previous page

Clicking on the AIS target (lower right) will take you to the AIS page, any other click will return to previous page.

Special Functions

anchor watch

By clicking on the  button you activate the anchor watch.

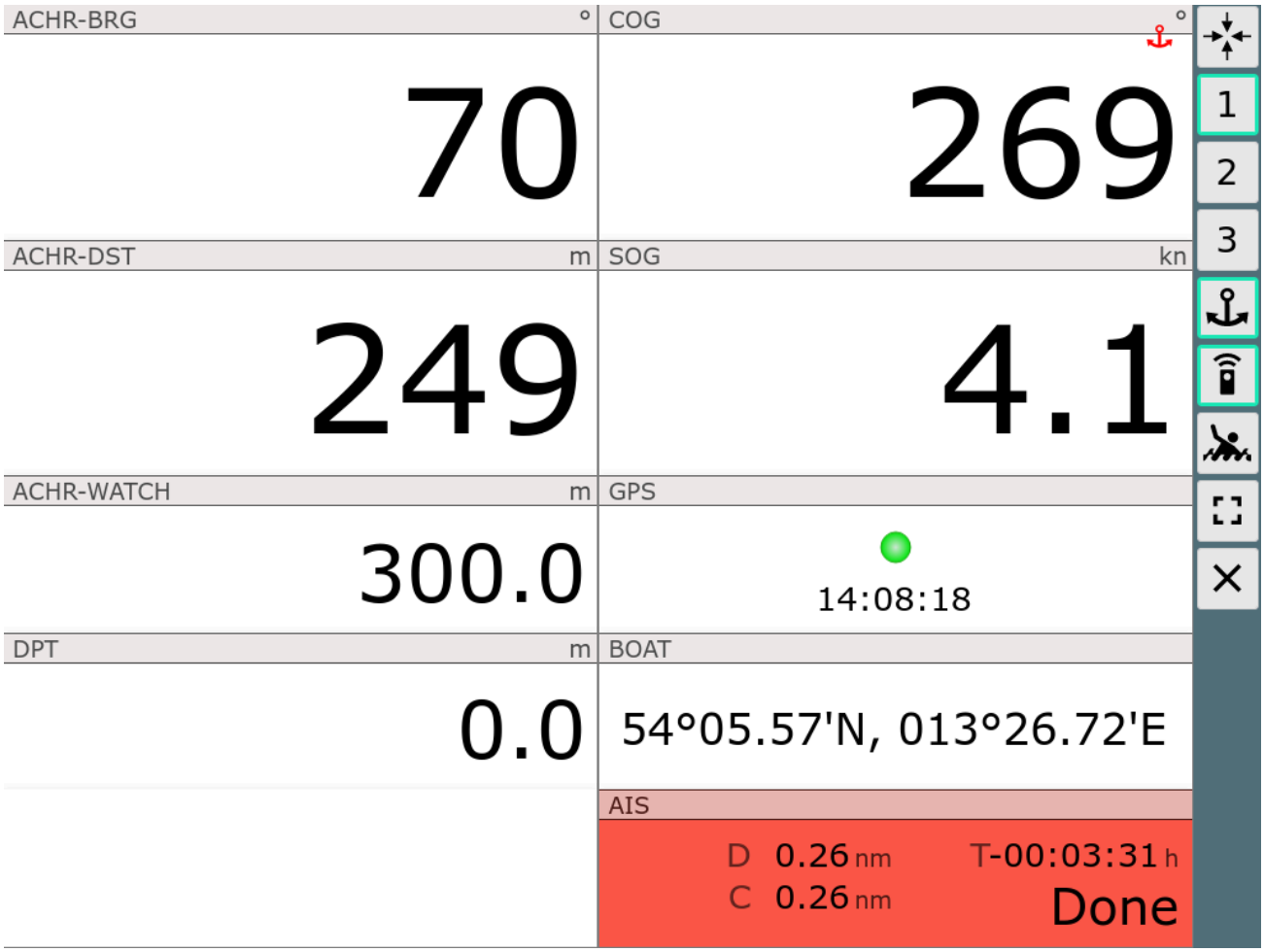


In the dialog you select whether the anchor should be set at the current boat position or at the center of the map. Additionally you can select the tolerated radius around the anchor position (the default can be changed at settings). By using distance and bearing you can set an offset to the selected position for the anchor (this way e.g. considering the length of your chain and the direction).

When activated the ⚓ button will display a green border and the monitoring will start. If your boat moves outside the defined radius an alarm will be triggered. Additionally, losing the GPS signal will also set off an alarm. The monitoring is done at the server side - so you may switch off any display. Then, of course, you need to have a sound device installed at your server.

The anchor watch can also be activated by clicking on the lower left displays on the navigation page.

When the anchor watch is active some dashboard pages will change their content.

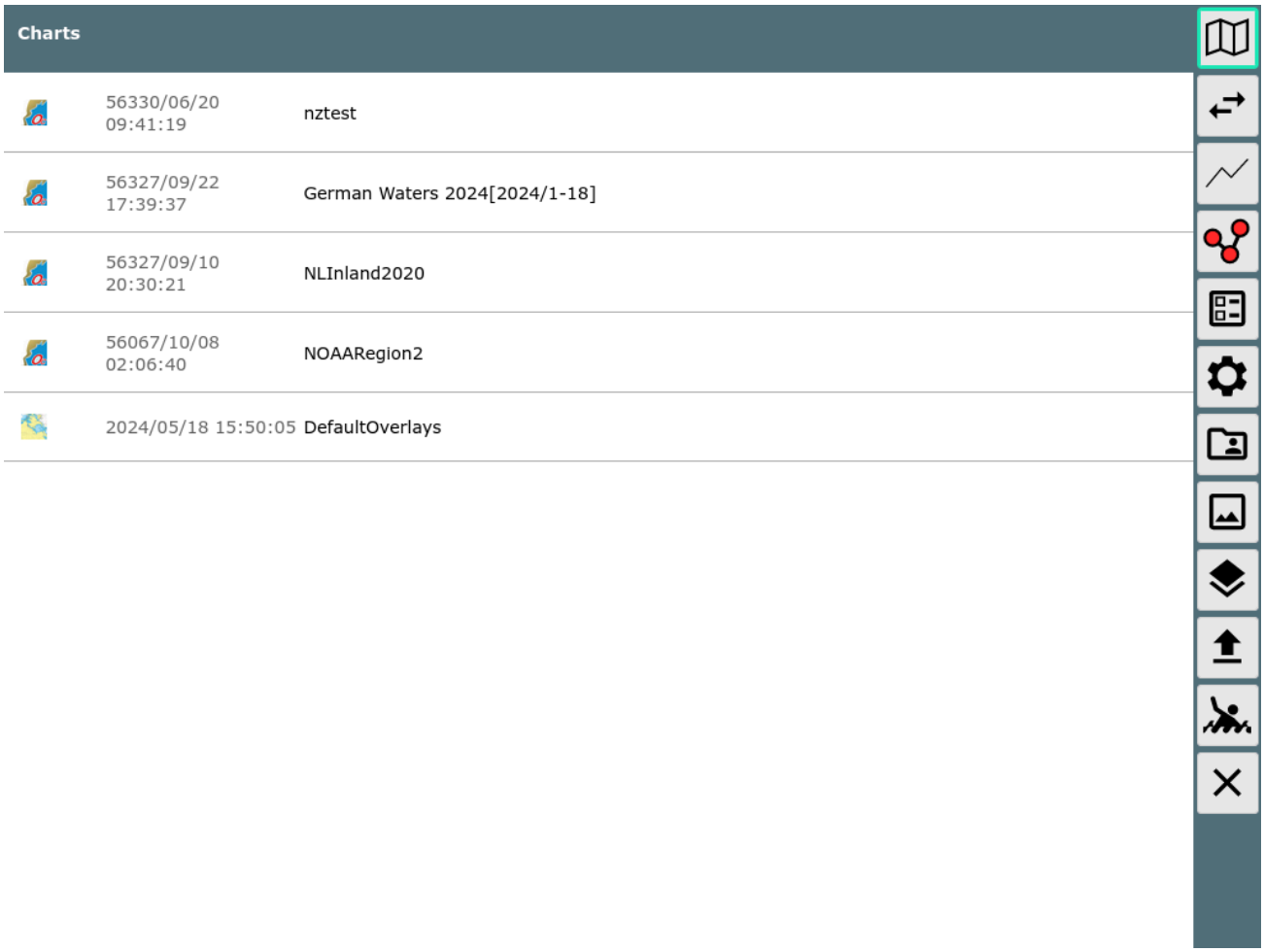


Ab 20240520: Wenn die Ankerwache aktiv ist, wird das kleine rote Ankersymbol auf allen Seiten angezeigt.
Mit einem Klick auf dieses Symbol oder auf den aktiven Anker Button wird ein Dialog zum Bearbeiten der Ankerwache angezeigt.

Man kann die Ankerwache verändern oder beenden. Mit Cancel wird sie unverändert fortgesetzt.

The Files/Download Page

This page is accessed by clicking the ⬇ button on the mainpage. Here you can download/upload/delete/edit/view charts,tracks,routes, user files and images.

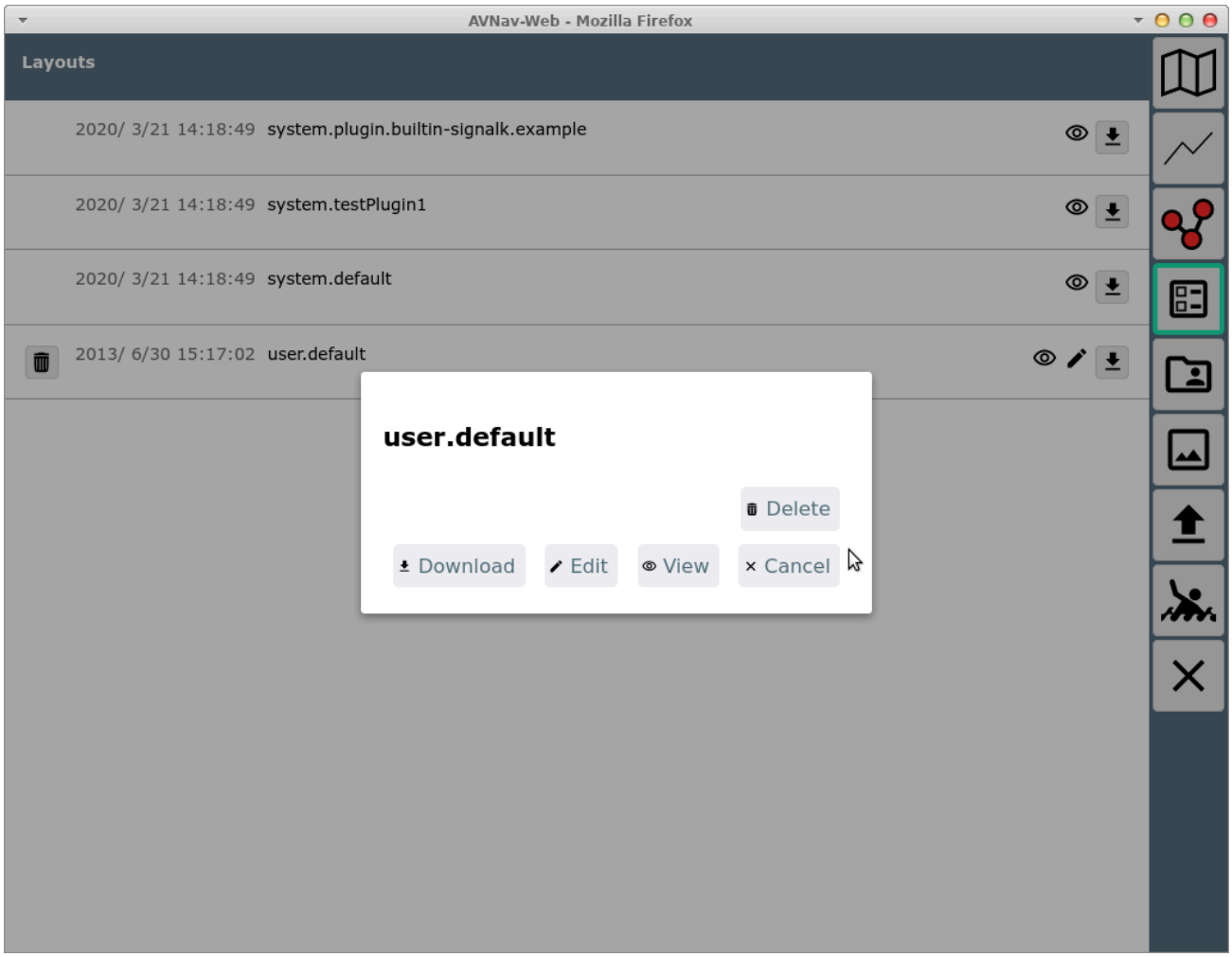


Buttons

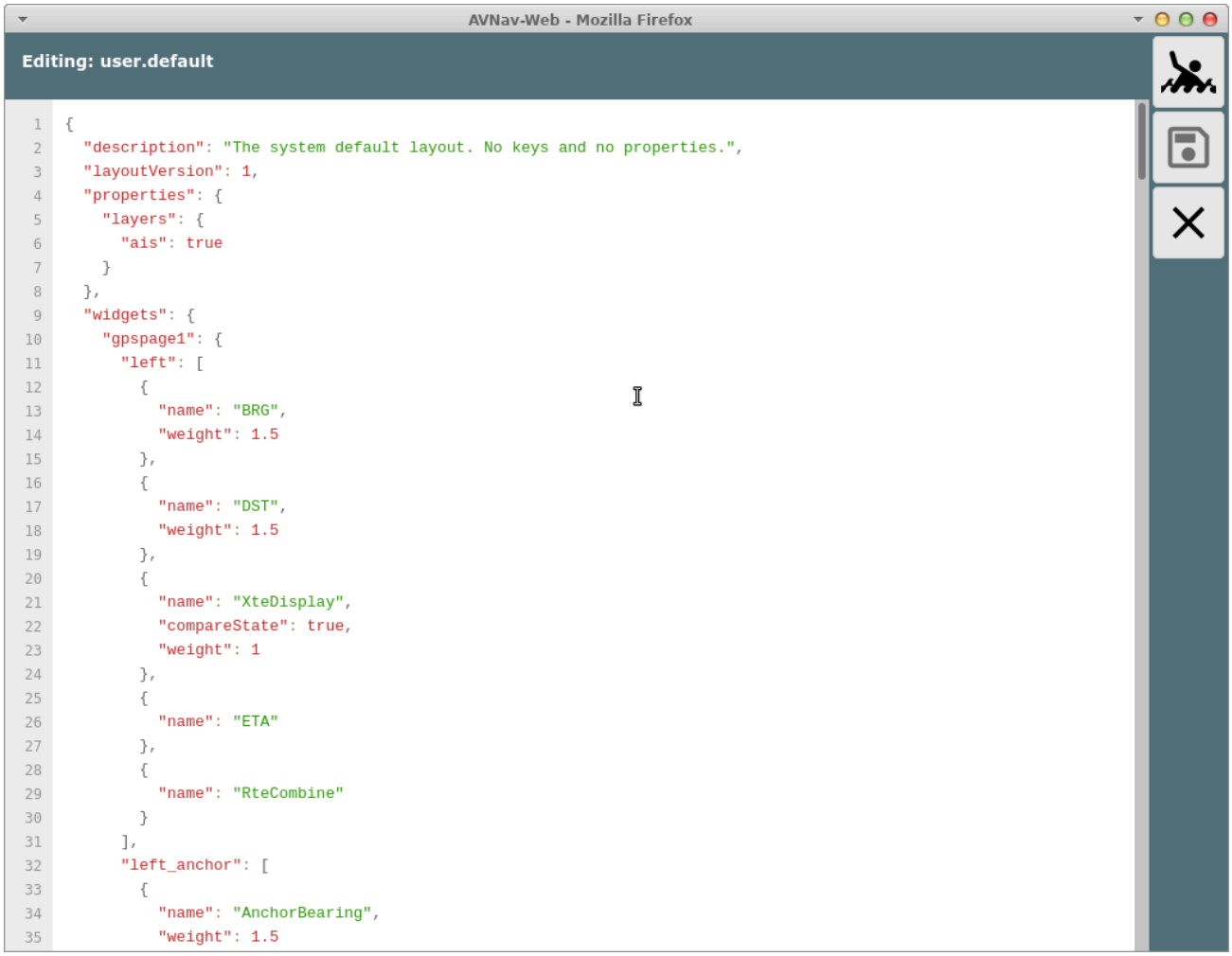
Icon	Name	Function
	DownloadPageCharts	show list of charts
	DownloadPageImporter	show the import page
	DownloadPageTracks	show list of tracks
	DownloadPageRoutes	show list of routes
	DownloadPageLayouts	show list of layouts
	DownloadPageUser	show list of user files

	DownloadPageImages	show list of user images
	DownloadPageOverlays	show list of overlay files
	DownloadPageUpload	upload a file to the currently shown category
	MOB	man over board (see main page)
	Cancel	back to previous page

For each entry some information is displayed along with a  delete and a  download button. They are only visible however if the operation is permitted for the particular element (e.g. no deletion of demo charts or system layouts).
Eventually more icons indicate operations valid for this element. By clicking the element a dialog will open displaying available actions for the element.



If available ("Edit" / "View") you can switch to an edit/show view. For routes "Edit" will bring up the Route-Editor.



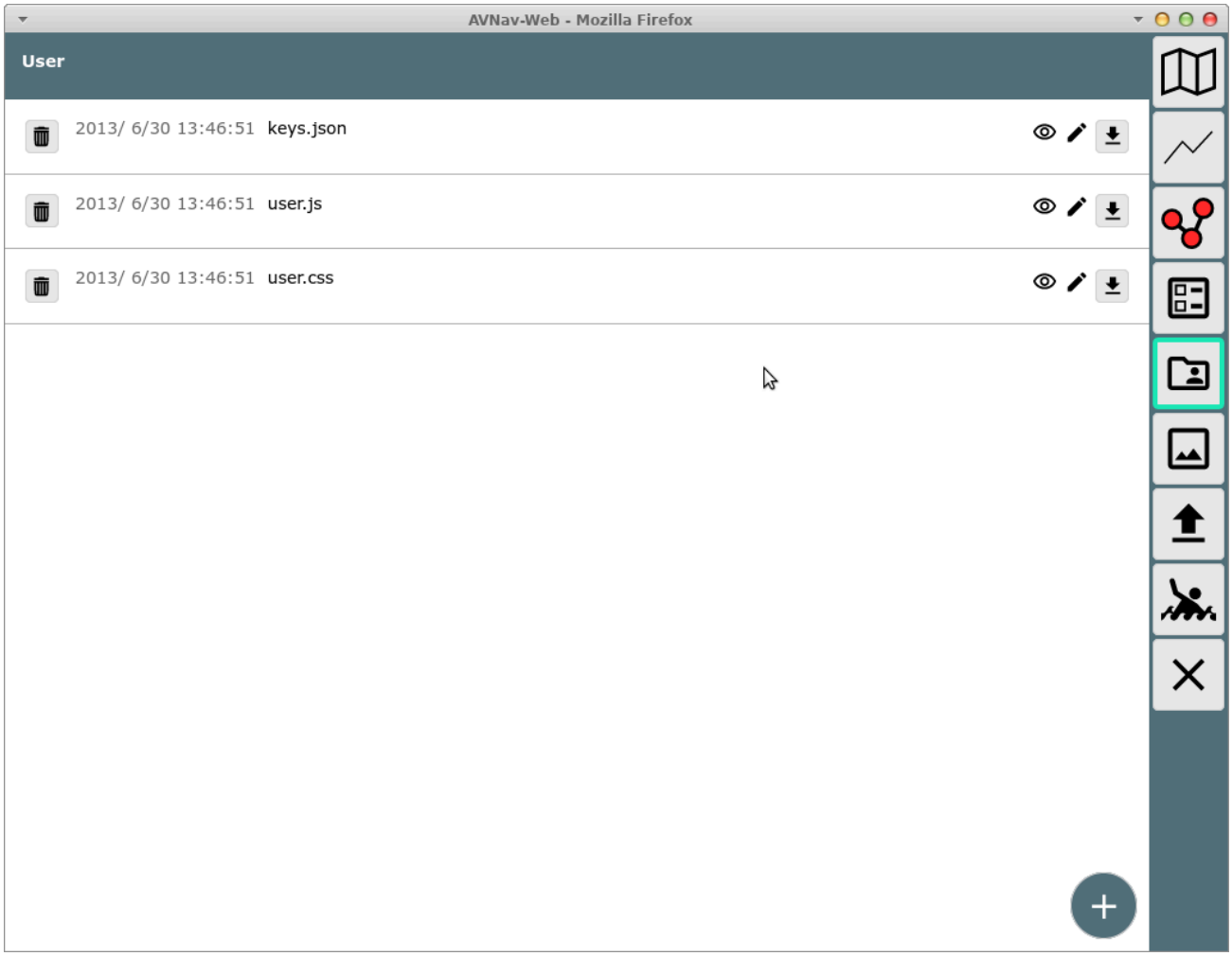
In this view you can make changes - (caption "Editing") or the content is dispayed. By clicking the changes will be stored.

Special Functions

User Files

Switching to user files permits to e.g. edit user.js and user.css. The file keys.json contains user defined keyboard shortcuts.

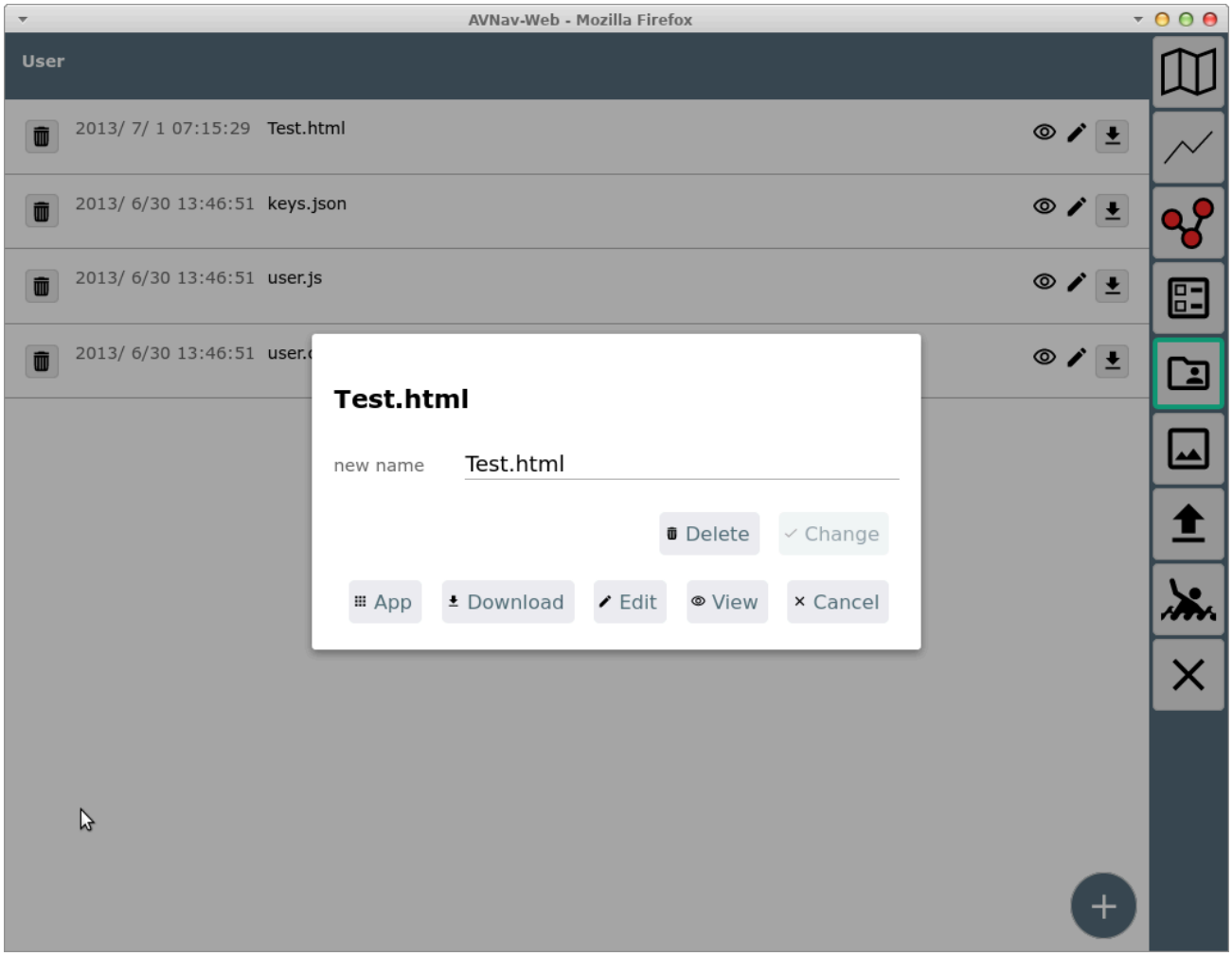
The file images.json can be used to adapt some of the symbols in AvNav.



Here you can also upload additional files needed to adapt AvNav (e.g. css files, java script files or HTML files). These files can be accessed via the url /user/viewer/<name>.
Images should be uploaded in the Images category. They will be accessible via /user/images/<name>.

Clicking the "+" button (lower right) you can create a new file. This could be a HTML file to create a User App .

For a HTML file the action dialog contains an " App" button that you can click to directly create a User App. You have to be sure to upload an Icon image file beforehand.



Charts - mbtiles Format

Using a mbtiles chart could lead to problems if the tiles internally are not ordered according to the default "xyz" format. Unless valid information is provided in meta data, AvNav assumes "xyz". If you experience display problems with a mbtiles chart you can try to set the format to "tms" within the action dialog. This change will be written to the chart file.

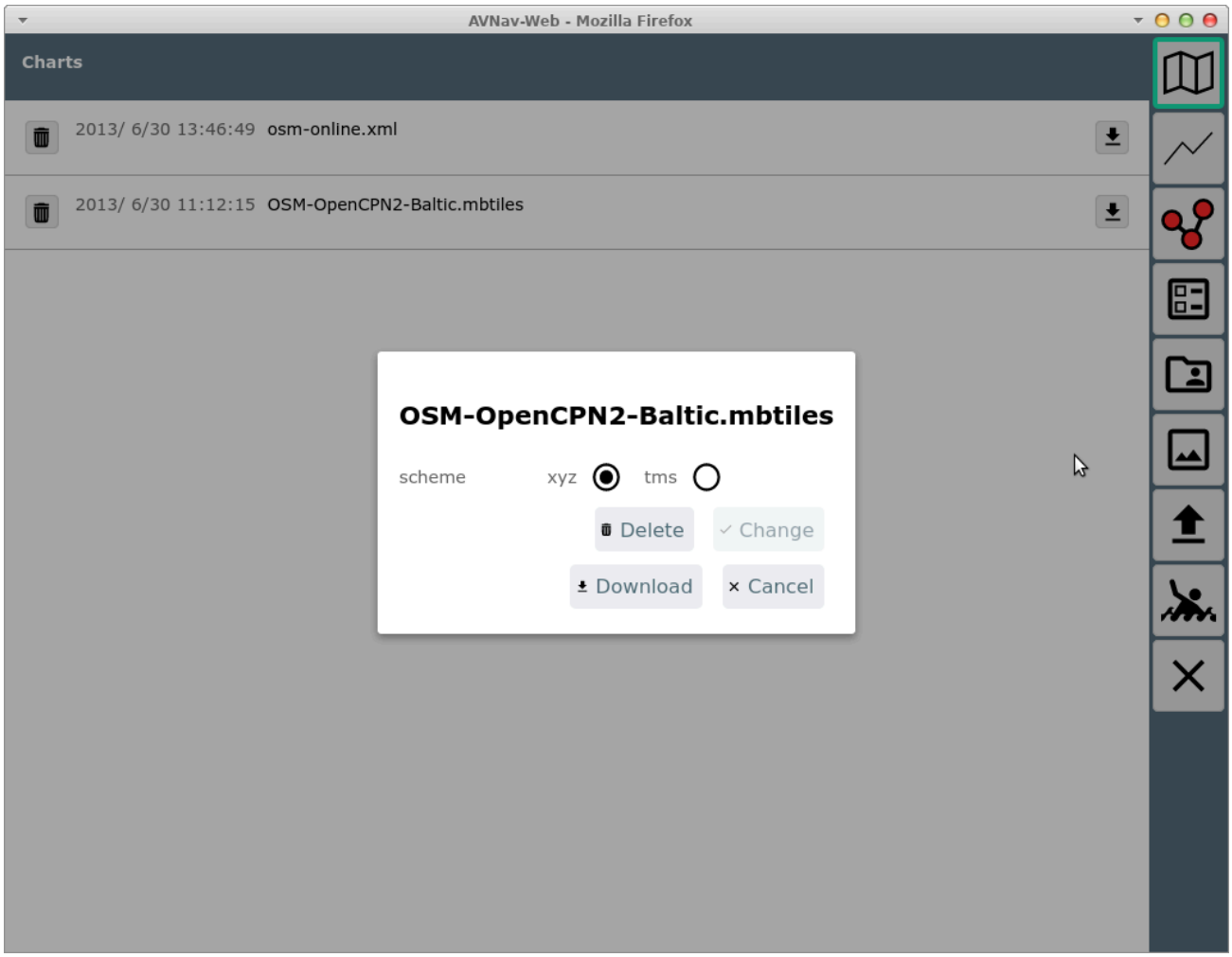
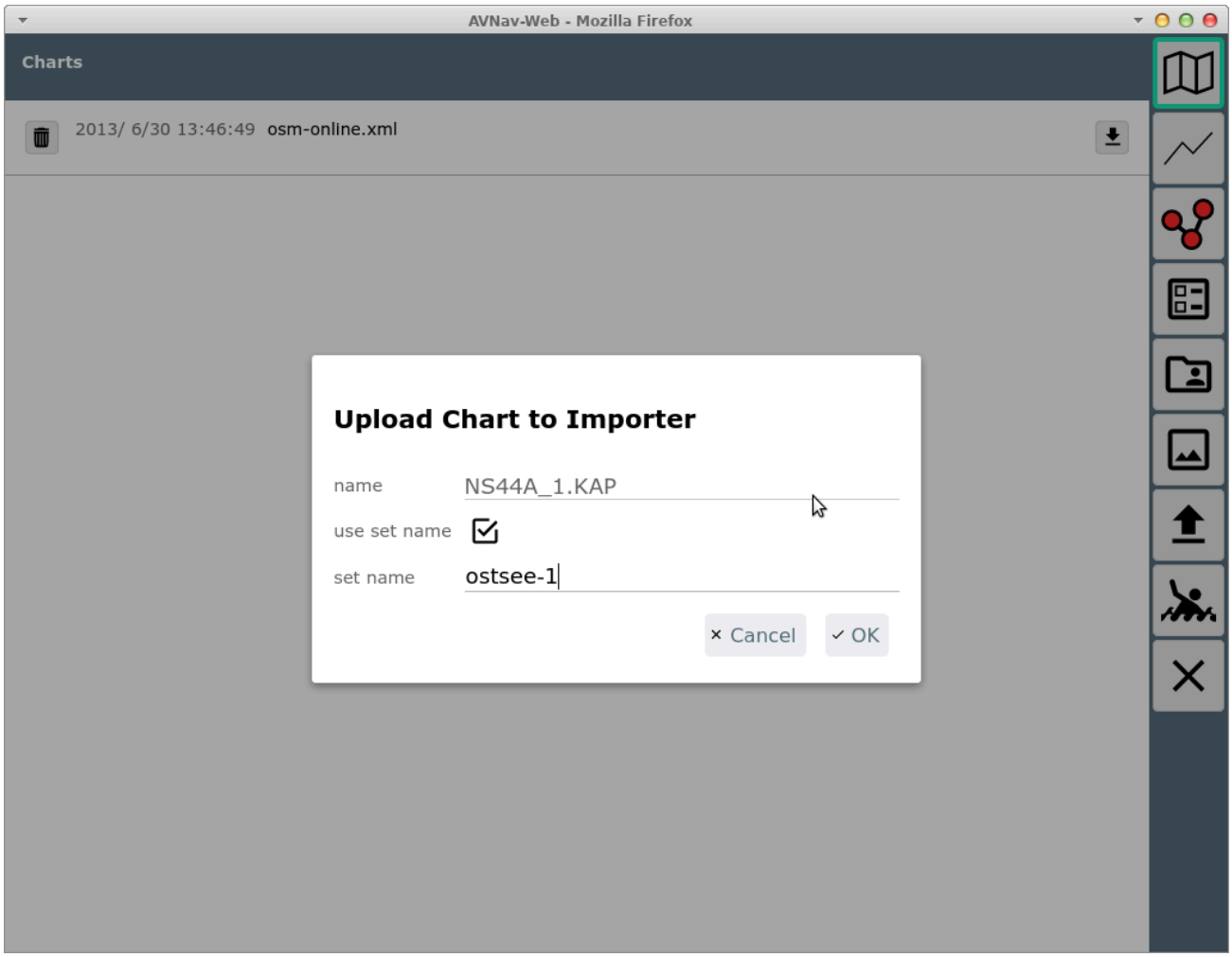


Chart Upload

At the category 🗺 you can directly upload chart files. Files that can be handled by AvNav will be uploaded directly into the charts directory and will be available immediately (gemf, mbtiles, xml files).

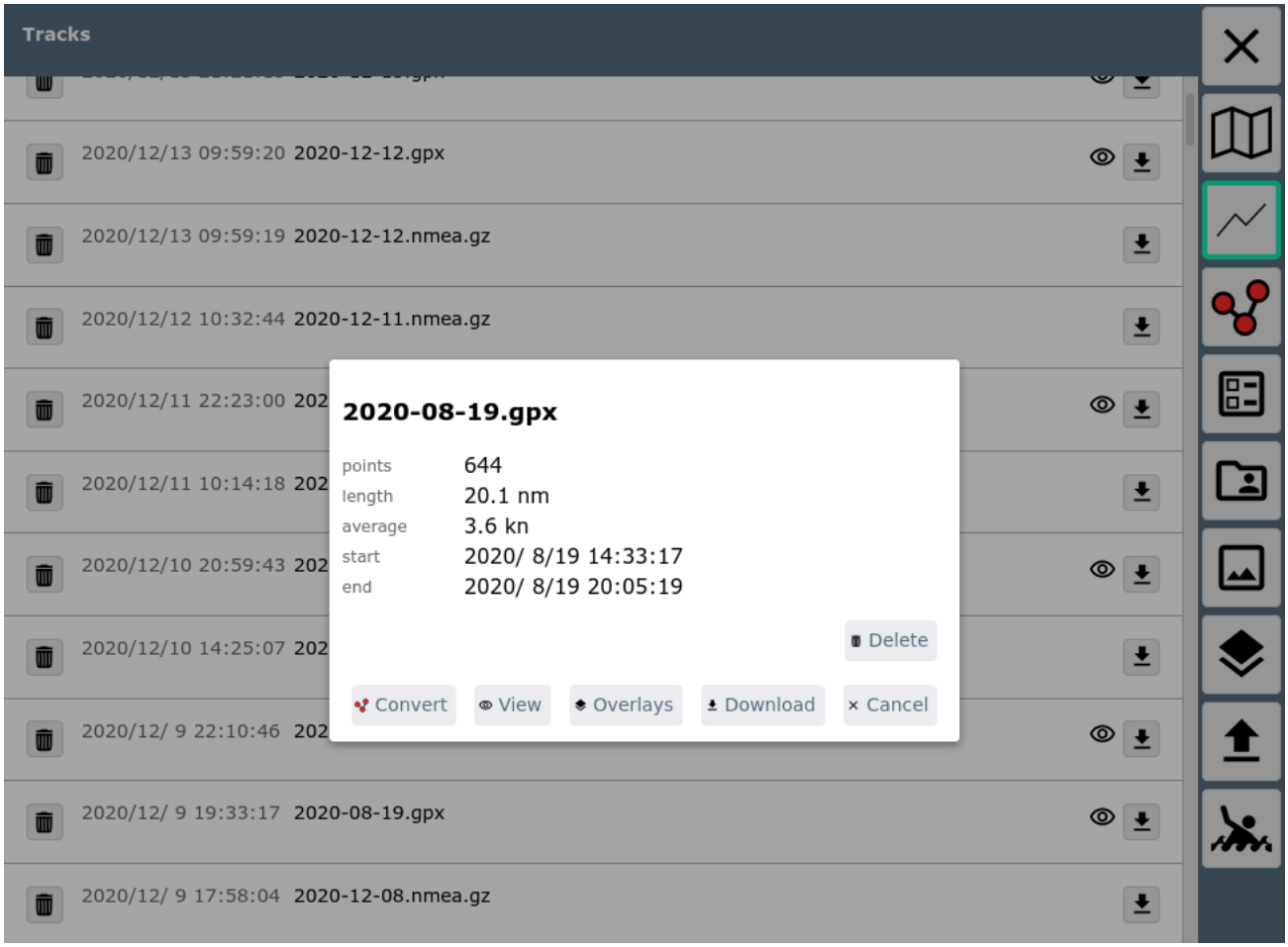
For charts requiring Conversion you will be asked if they should be put into the importer's input directory (not on android).



You can optionally select a "set" name - this determines name of the gemf file to be created. This way you can merge multiple chart files into a single gemf file. The state of conversion can be checked at the import page (it will be opened automatically).
After an upload the importer still waits some time allowing more files to be uploaded and included in the same set. They all will be handled with one conversion.

Tracks

The info dialog for tracks contains some additional information.

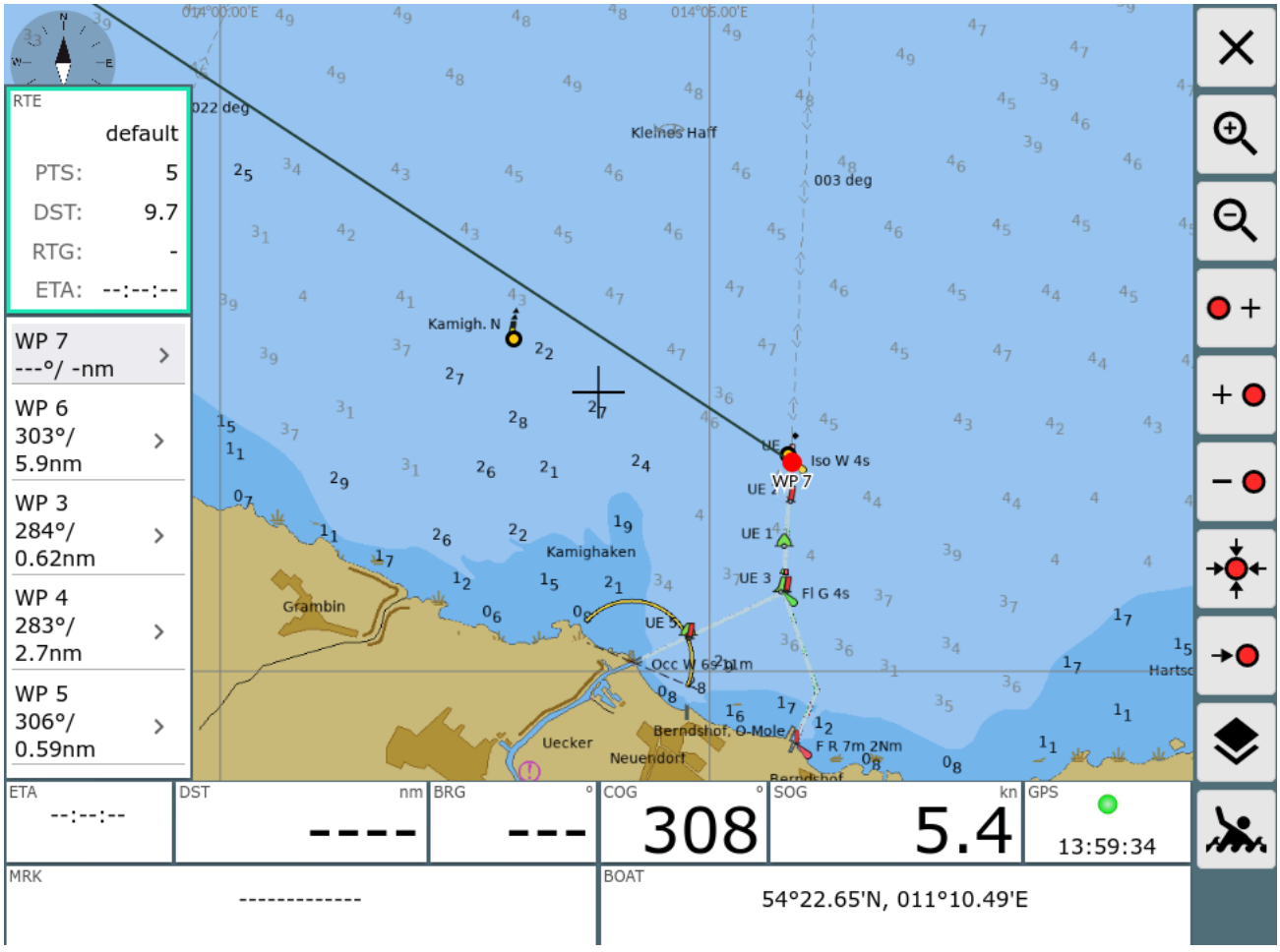


Additionally the conversion of the track to a route can be started with the Convert button. Clicking the overlay button you can add the track to overlays or remove it from them.

The Route Editor

Overview

You will enter the route editor by clicking the button on the navigation page . Additionally you can get to it via the Edit button in the route's info dialog on the Files/Download page.



Buttons

Icon	Name	Function
	ZoomIn	zoom in
	ZoomOut	zoom out
	NavAddAfter	add a new point to the route the chart center (cross) will be added after the point that is currently marked red
	NavAdd	add a new point to the route the chart center (cross) will be added before the point that is currently marked red
	NavDelete	delete the point that is currently marked red
	NavToCenter	move the currently marked red point to the center of the chart (cross)
	NavGoto	start navigation to the currently marked red point
	NavOverlays	show and hide Overlays
	MOB	man over board(see main page)
	Cancel	back to the navigation page

At this page you can create and edit a route.

There is always a currently selected point within the route. It is marked red on the chart and marked grey in the list on the left side. You can change the selected point by clicking a different point on the chart or in the list on the left side.

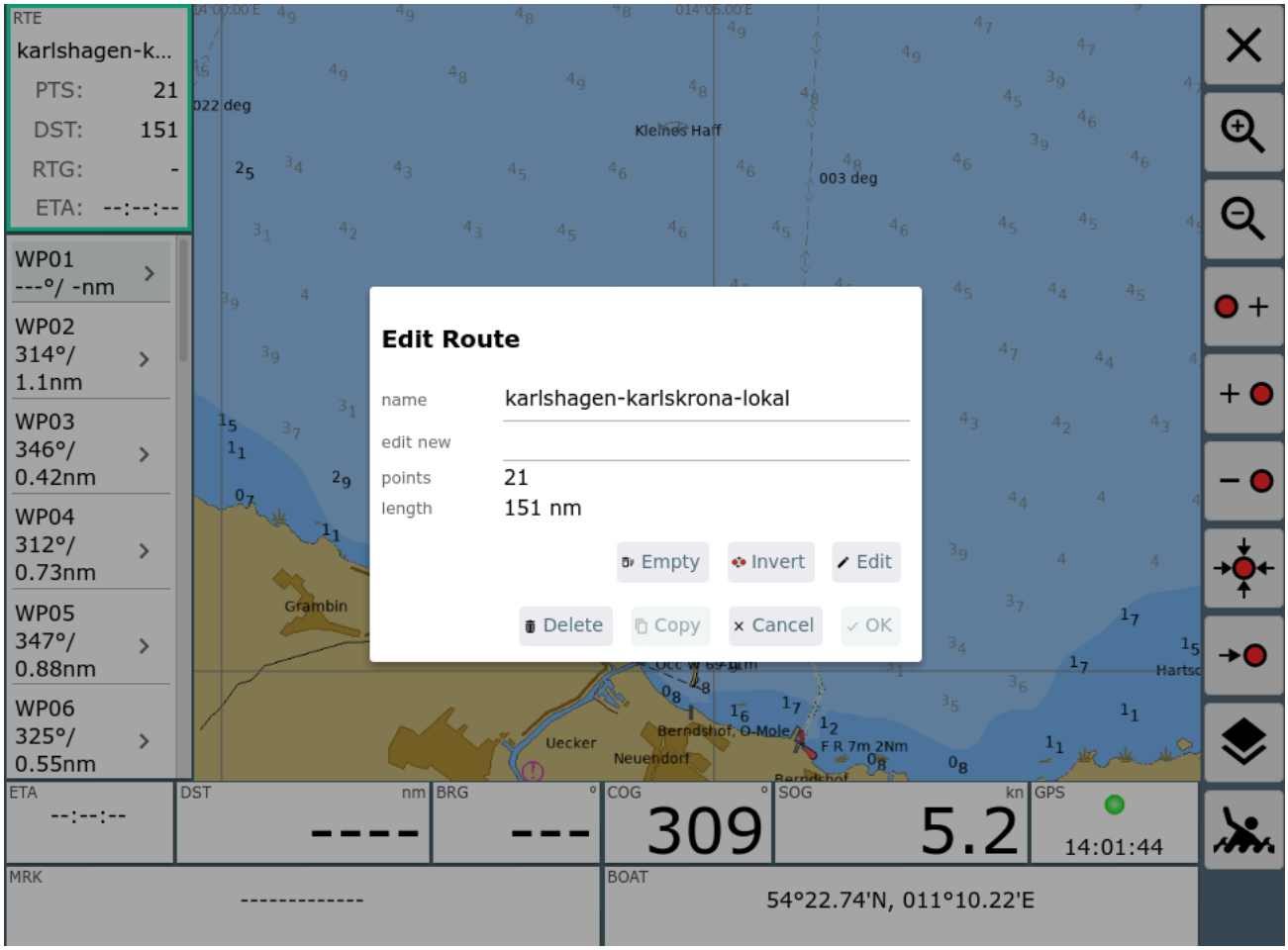
If you are editing the currently active route, the route info at the left side will display a red border (otherwise green).
Leaving the editor will take you back to the active route.

All changes become effective immediately - there is no undo.

At the left side you see a list of all the route's waypoints including courses and distances. On top of the list the number of waypoints, the overall route length and - if the route is active - the estimated ETA are displayed.
By clicking an already selected waypoint in the list you open an edit dialog to change its name or position.

Route Dialog

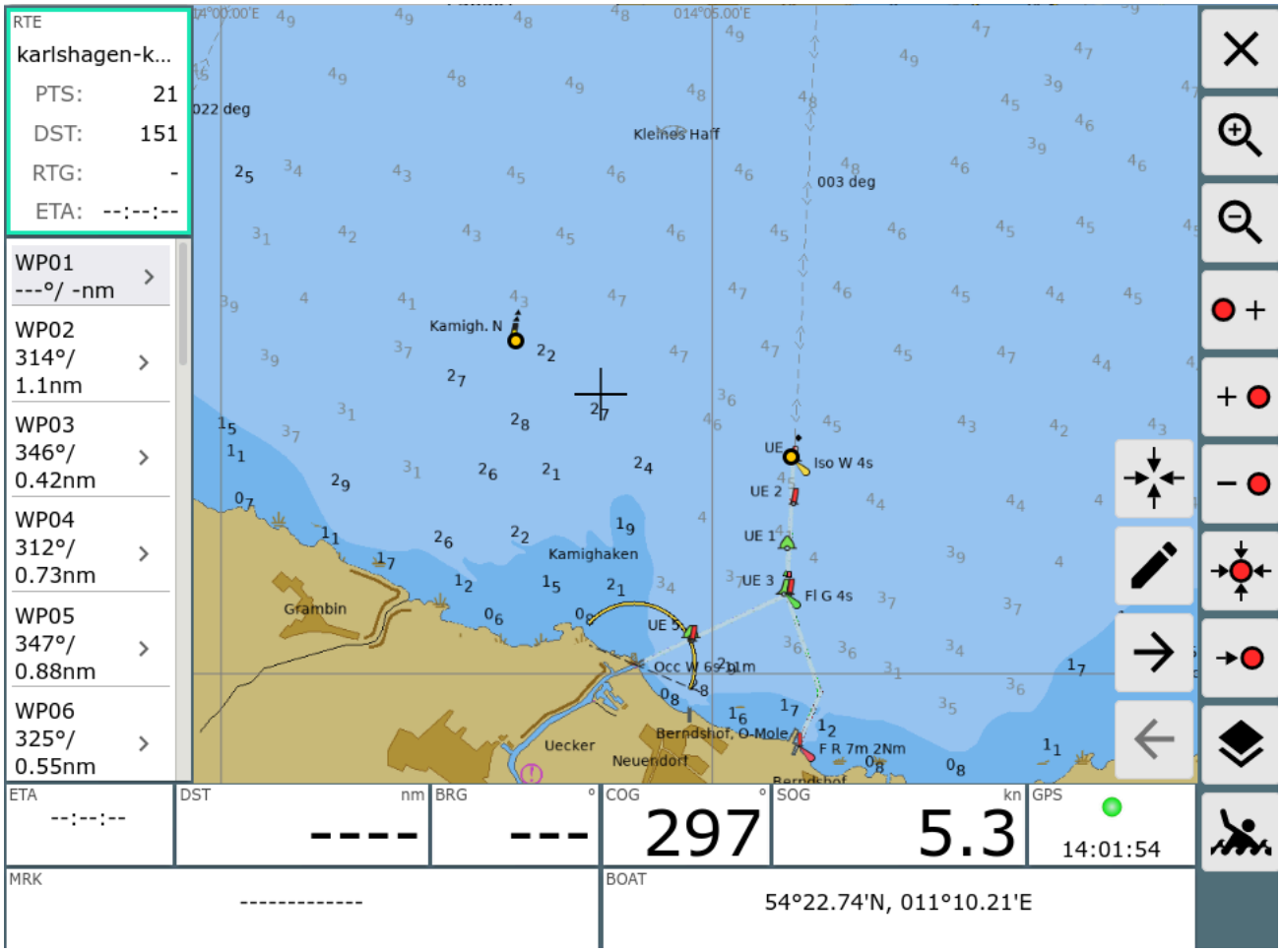
Clicking on the route info on the left side opens a dialog.



Within this dialog you can change the name of the route, select a new route to be edited, copy the route, delete it, empty or invert it.

Clicking Edit will take you to the route list. At this page you can delete and edit all points, change the name, invert or load a new route.

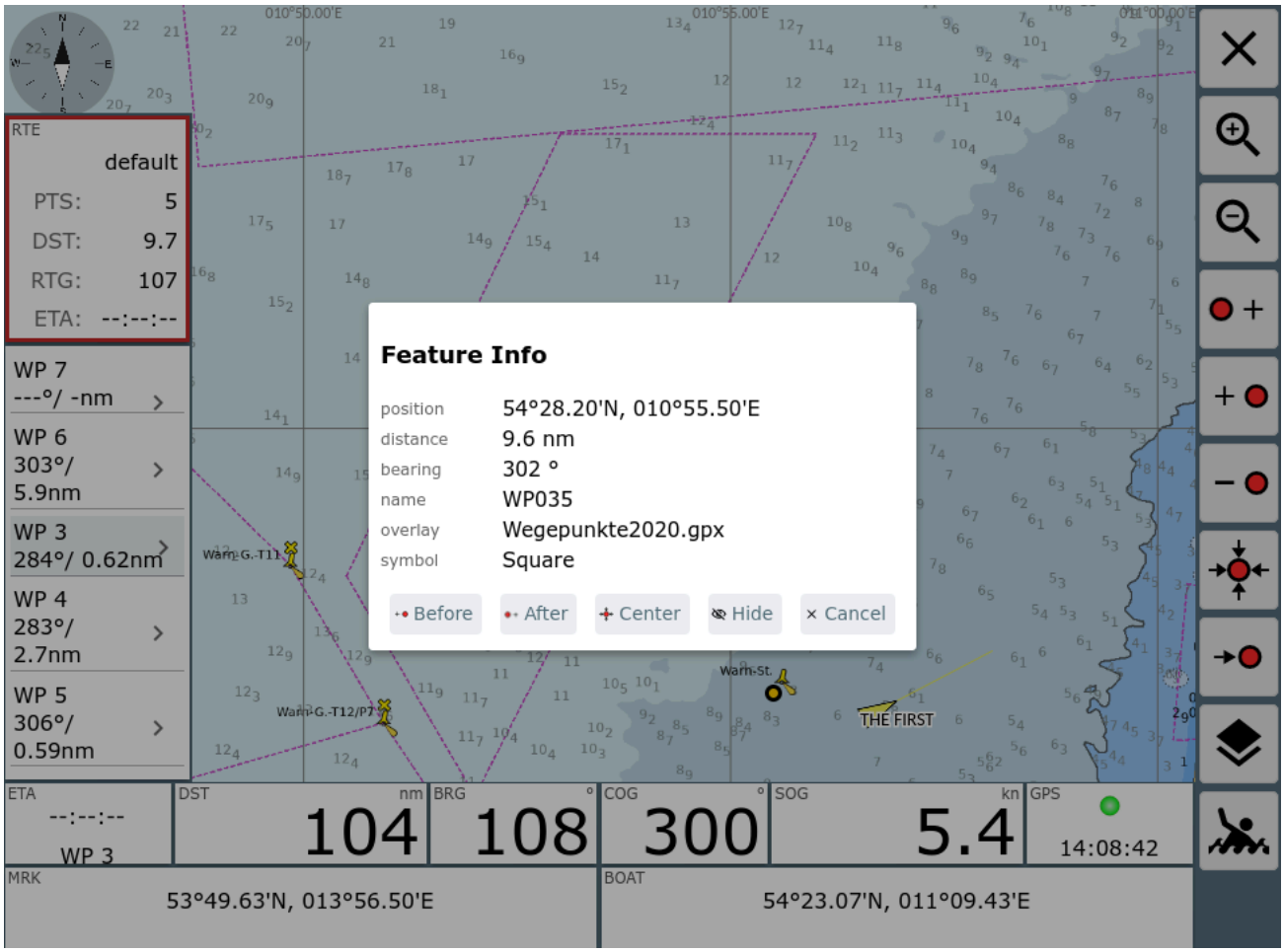
Clicking the bottom left panel activates some additional buttons.



Using those buttons you can move between the points in the route or you can edit the selected waypoint.
Clicking the bottom right panel centers the chart at the boat's position.

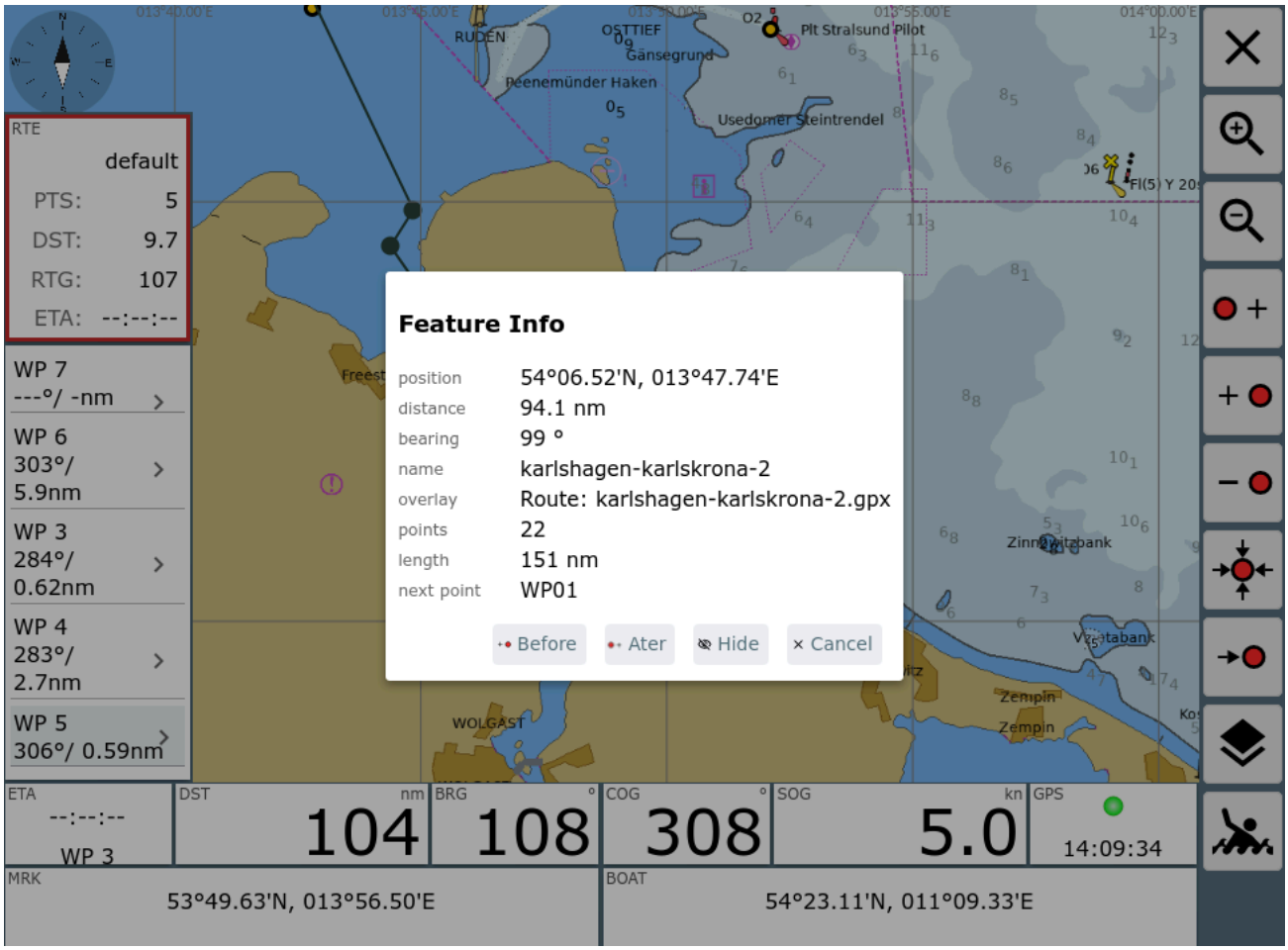
Using Overlays

If an overlay is visible (like in the picture) - or within ocharts you can open a dialog by clicking an object on the chart. This dialog provides some additional functions.



You can insert the object you clicked into the route (befor or after the current red point). This way you can easily e.g. use points from a waypoint file to create your routes.

If the overlay is a route you can insert the part after the clicked point into your current route.



The dialog shows the selected start point of the route and you can decide to insert before or after your current red point.

Converting Tracks to Routes

AvNav continously records tracks. A new track file (gpx) is generated every day.

Those tracks (or tracks uploaded via Files/Download page) can be converted to routes to be used in navigation.

Concept

This conversion typically requires to reduce the number of points in the track as they often contain more than 1000 points. Otherwise using such a route is unfeasible.

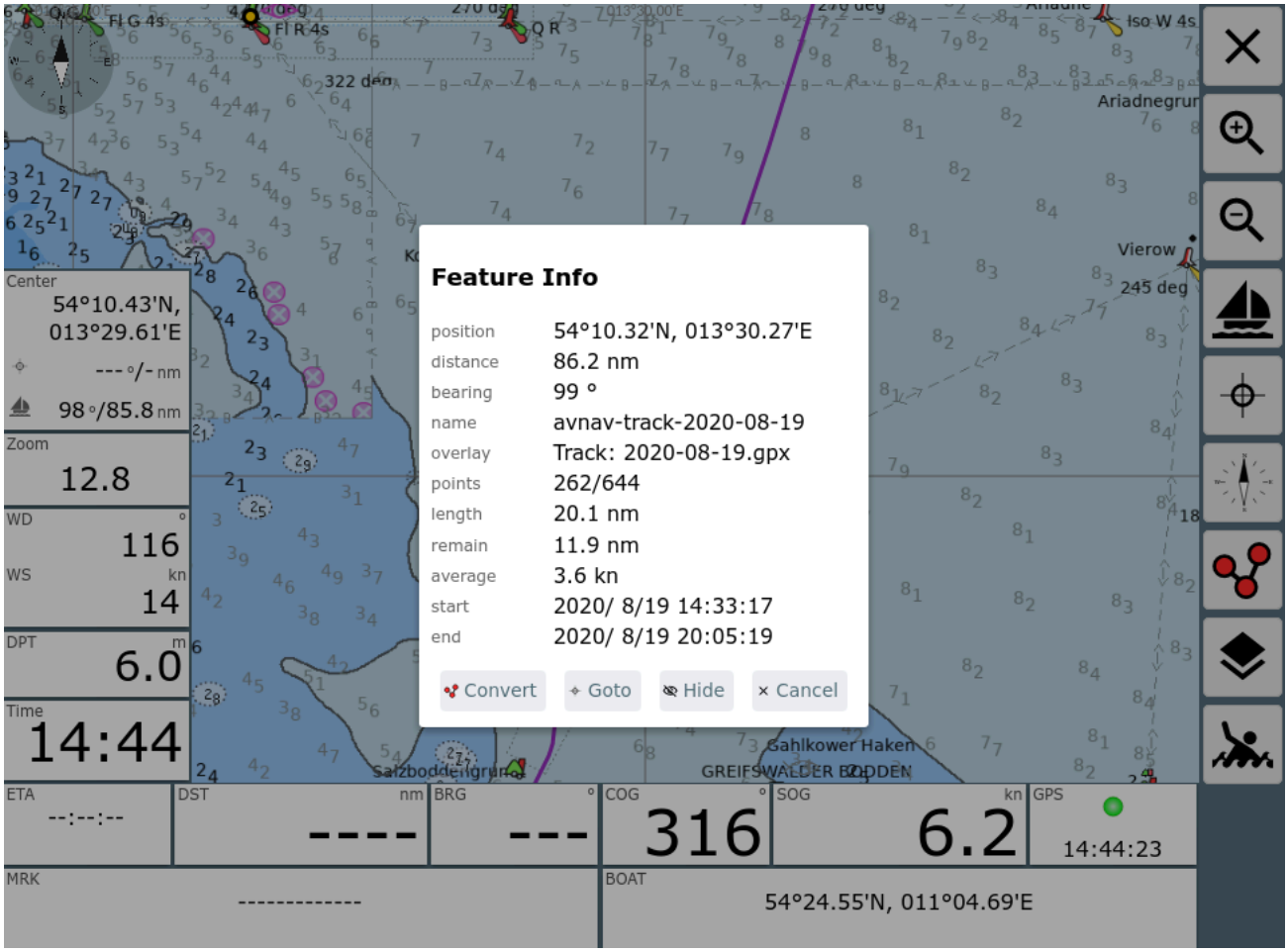
AvNav is using an algorithm similar to GPS Babel. One by one, points with smallest distance from the vector between the two neighboring points are eliminated.

You can provide a parameter (Xte) to define the maximum tolerable distance. The larger distance you allow, the fewer points the created route will contain. For usabilty reasons, you should try to keep your routes below 50 waypoints.

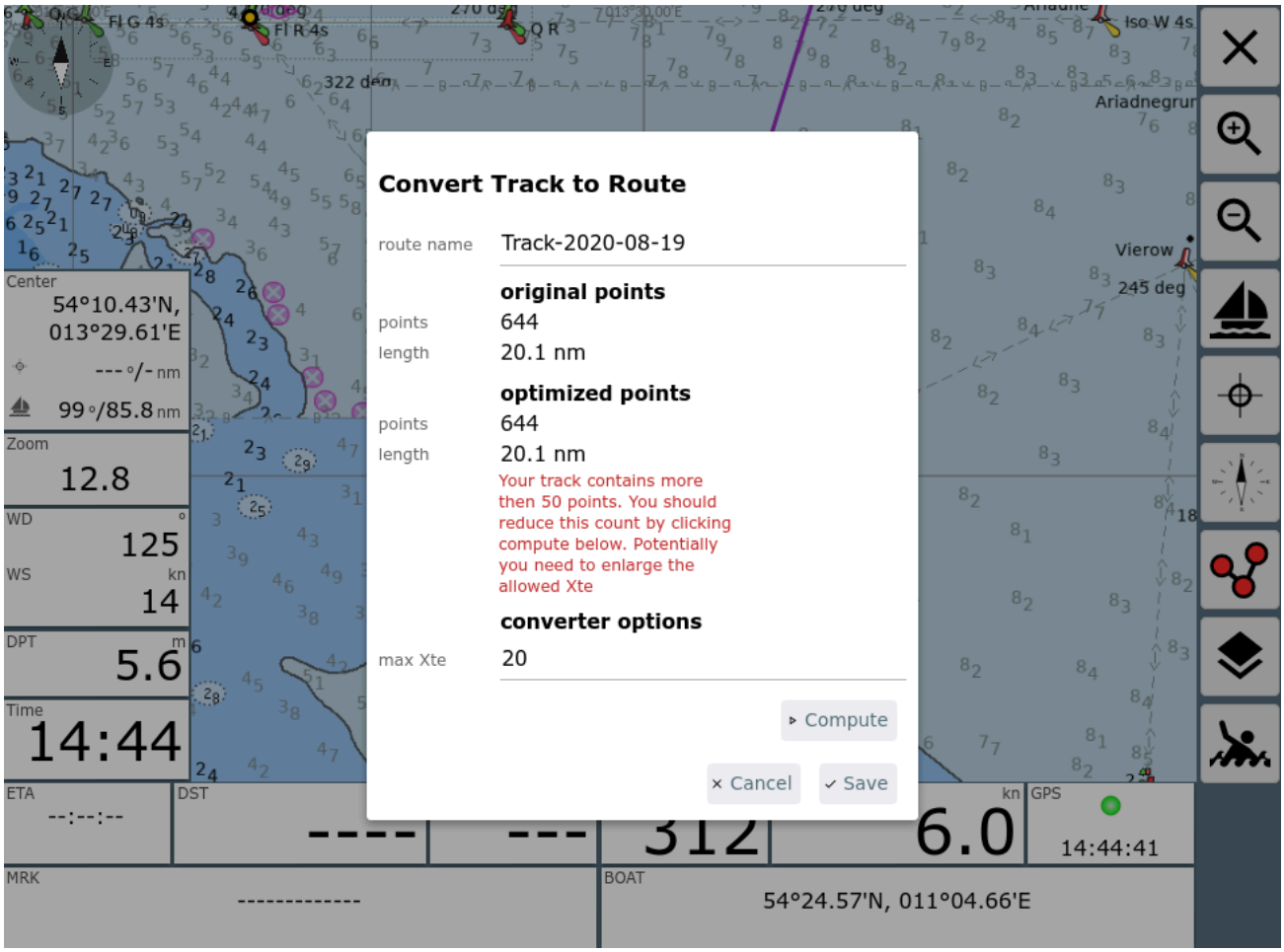
After the conversion you can fine tune the route in the Route-Editor.

Handling

You enter the conversion dialog either from the Files/Download page from within the Track Info Dialog (Convert Button) or from the info dialog if you have the track selected as an Overlay on the chart and click on a point within it. Again, you press the Convert Button - see picture.

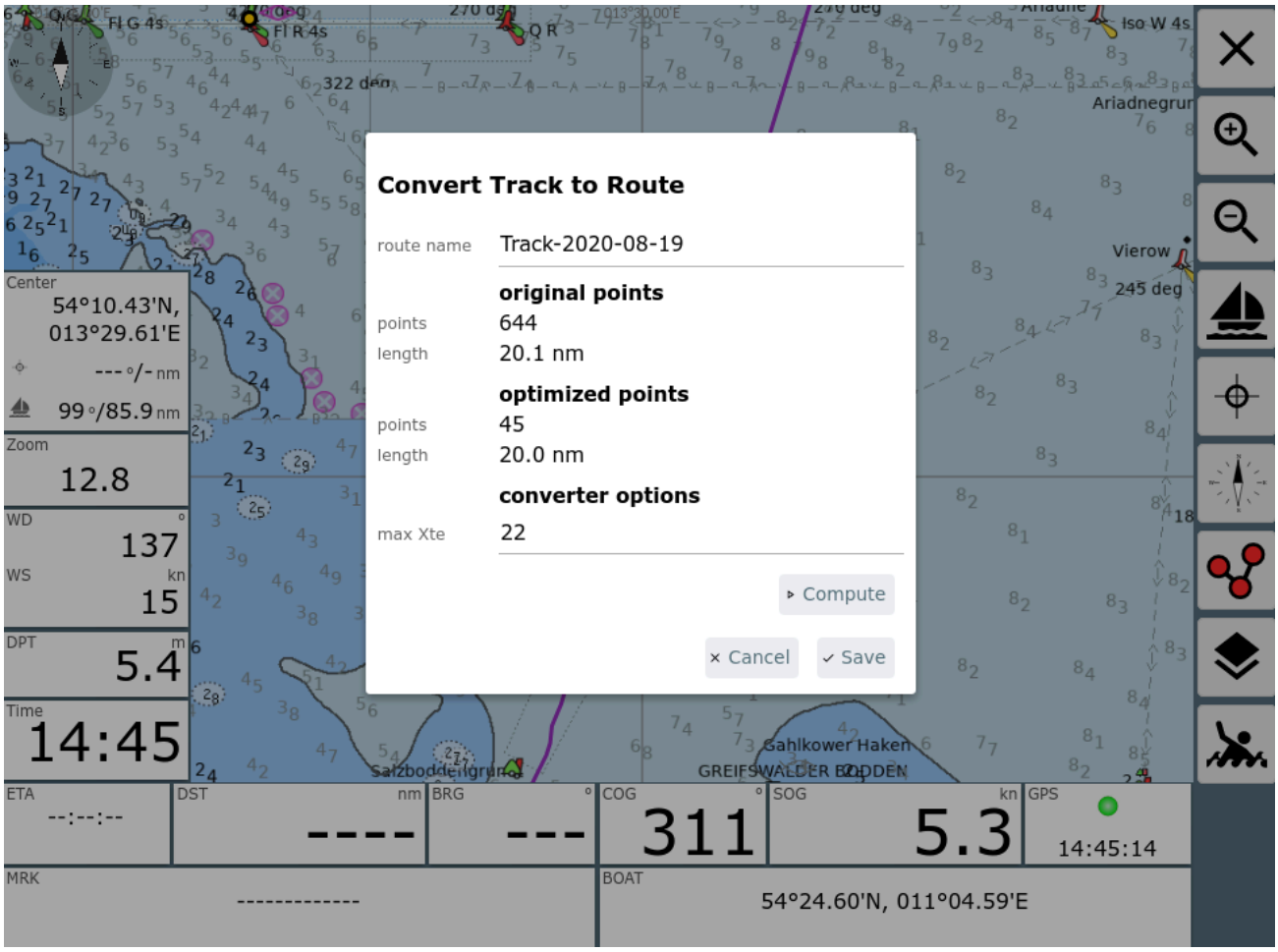


After clicking "Convert" the conversion dialog pops up.



Normally (as in the picture) you will see a warning about point count exceeding 50.

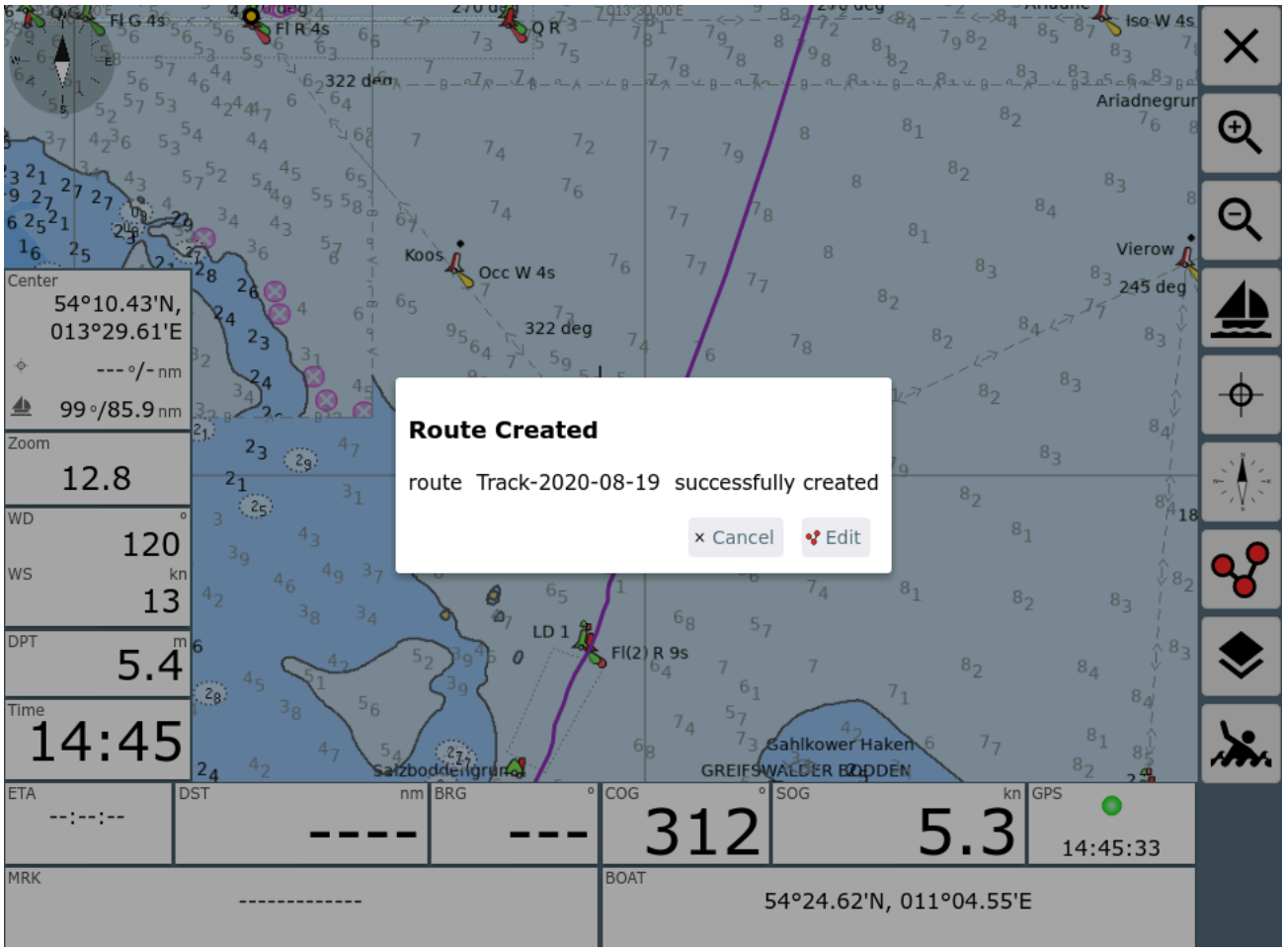
Using "max Xte" you configure the converter and start the conversion clicking "Compute". If the number of points still does not fit you can try again with a different Xte.



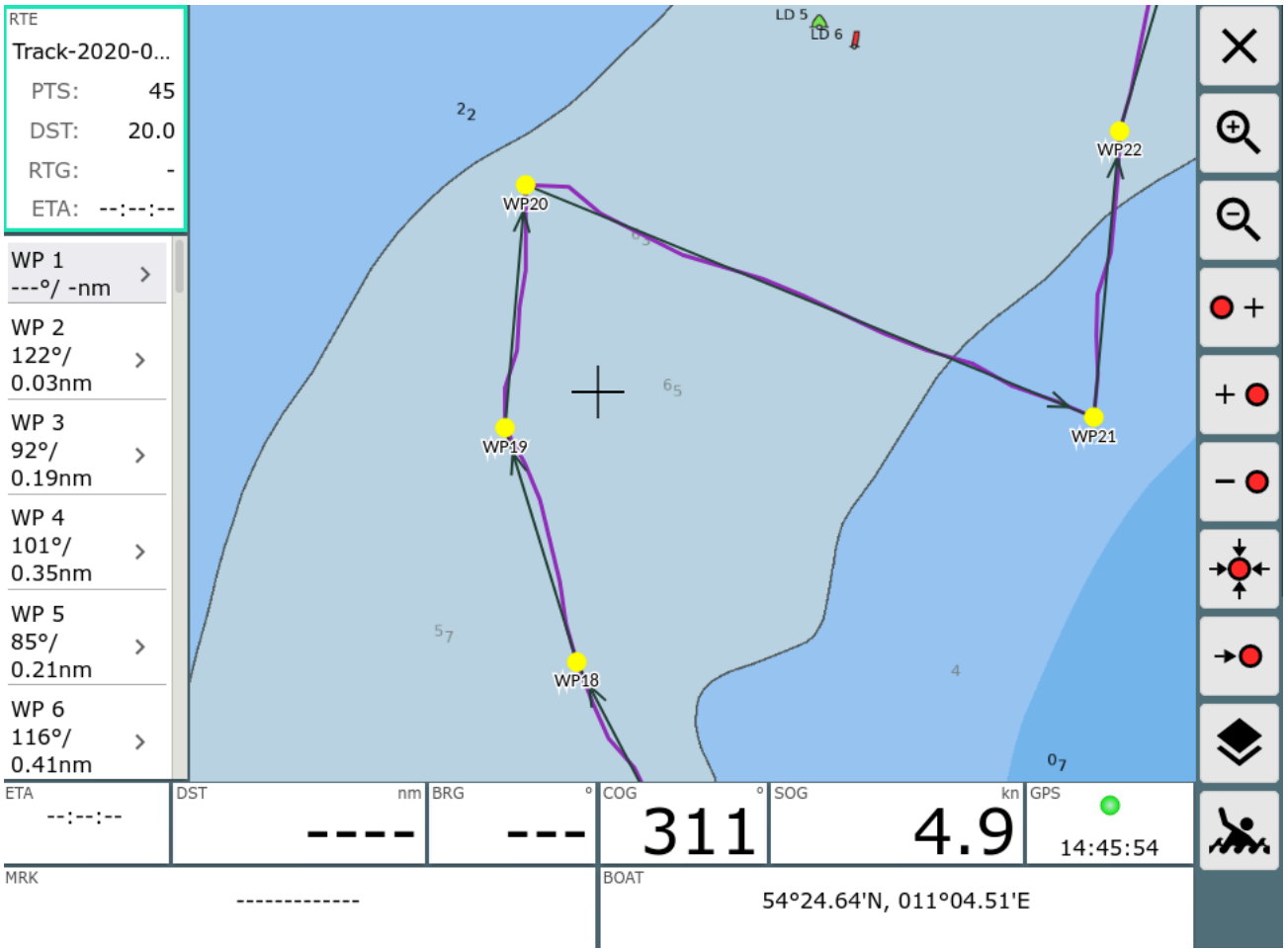
In above example a max Xte of 22m produced a good route.

Within the dialog you can change the name for the new route (and decide to potentially overwrite an existing one with the same name).

After saving you can directly start editing the route.



At this point you preferrably had the track displayed as an overlay to the chart, so you can easily check if all the important track waypoints are included in the route.

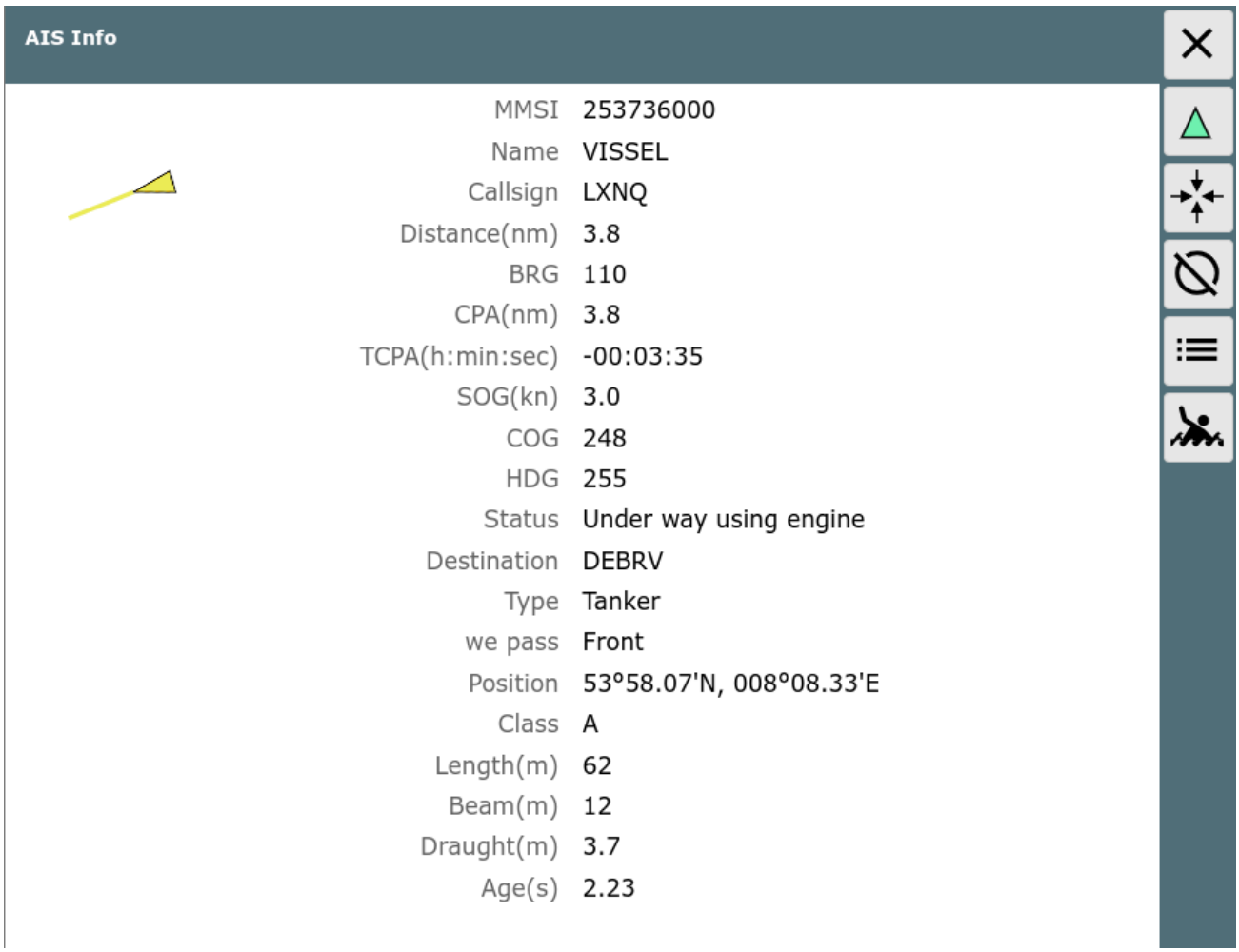

RTE
Track-2020-0...
PTS: 45
DST: 20.0
RTG: -
ETA: --:--:--
WP 1
---°/ -nm
WP 2
122°/
0.03nm
WP 3
92°/
0.19nm
WP 4
101°/
0.35nm
WP 5
85°/
0.21nm
WP 6
116°/
0.41nm
LD 5
LD 6
WP18
WP19
WP20
WP21
WP22
ETA
--:--:--
DST nm

BRG °

COG °
311
SOG kn
4.9
GPS
14:45:54
MRK

BOAT
54°24.64'N, 011°04.51'E

The AIS Info Page



This page will be displayed when clicking the AIS widget on the navigation page or a dashboard page - or by clicking an AIS target on the chart.

Details for the AIS target are shown.

Buttons

Icon	Name	Function
	AisNearest	center to the closest target and display it on the navigation page

	AisInfoLocate	center the chart to this target an display its data on the navigation page
	AisInfoHide	Hide the AIS target for some time (Settings/AIS "hide time") - default 15s
	AisInfoList	display the list of all AIS targets
	MOB	man over board (see main page)
	Cancel	back to previous page

The AIS List

From the Ais Info you get here by clicking the button.

Buttons

Icon	Name	Function
	AisNearest	back to "normal mode", the closest target will be displayed centered on the navigation page
	AisSort	change the sort order of the list

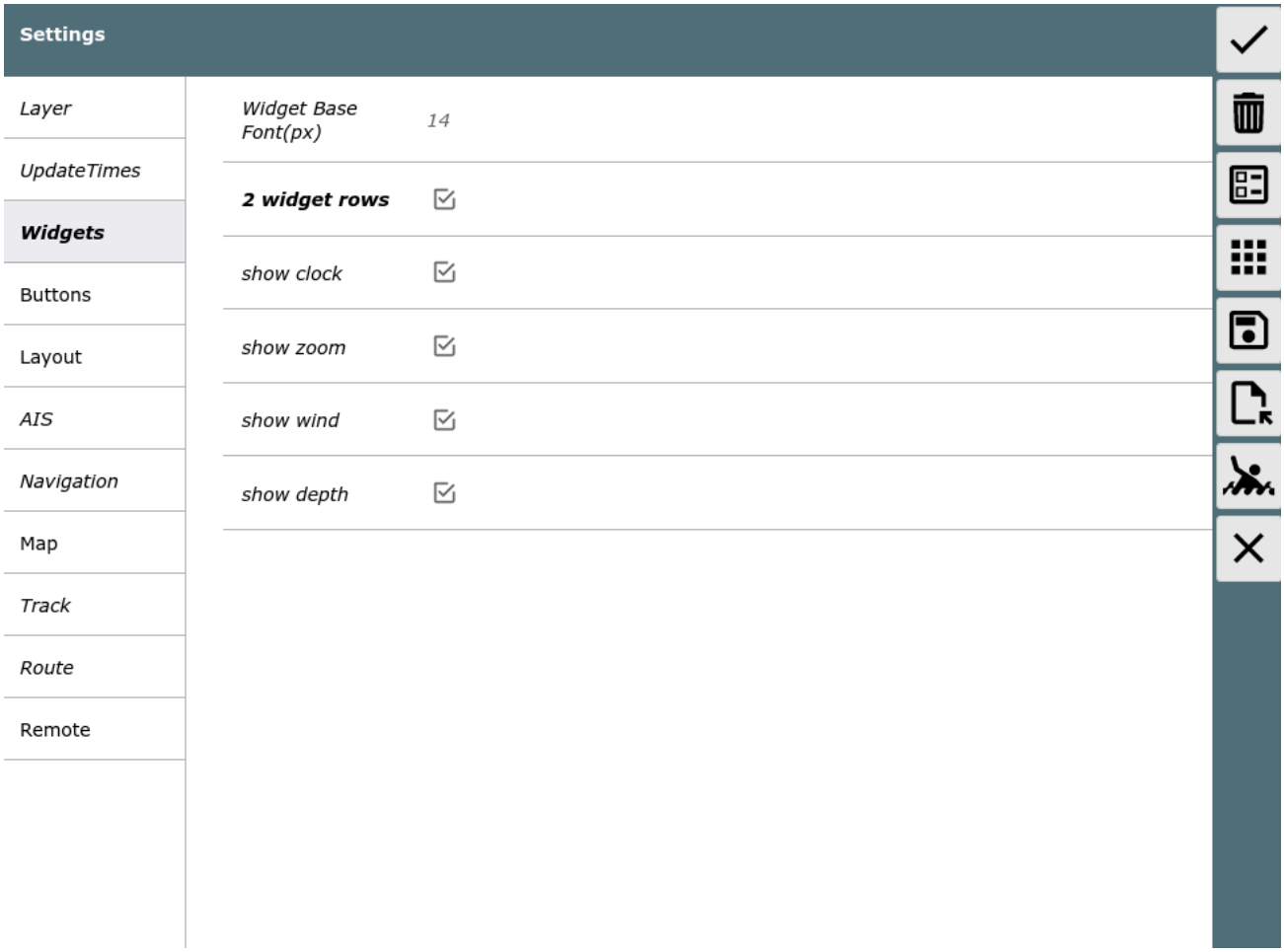
	AisLock	disable the automatic list update (toggle)
	AisSearch	filter the shown ais targets. A filter dialog will be shown to input a search item. This will be machted against: name,mmsi,callsign,shipname. Only matching items will be shown in the list. As the button is a toggle button the second click will disable the filter.
	MOB	man over board (see main page)
	Cancel	back to previous page

This page lists all targets within a 10nm range to the current position or to the center of the chart (sorted by CPA). By clicking a row you will return to the chart centered to this target.

By clicking either second column heading or the  button you can change the sort order.
Clicking an item in the list will bring you to the aisinfo page for this item.

The Settings Page

From the main page you get here using the button.



Buttons

Icon	Name	Function
✓	SettingsOK	accept changes and go back
🗑	SettingsDefaults	reset all settings to their default values
	SettingsLayout	start of the layout editor

	SettingsAddons	go to user app configuration
	SettingsSave (since 20220225)	Save the current settings to the server
	SettingsLoad (since 20220225)	load a settings file from the server (you will be able to select from a list)
	MOB	man over board (see main page)
	Cancel	discard changes and go back

You can set many parameters affecting your display. All those settings are stored in the current browser (associated with the URL you used to open AvNav). They are not stored on the server!

For most values there is a separate reset button. Globally you can reset to defaults using the  button. A confirmation dialog is displayed when leaving the page without saving.

Most of the settings should be self explanatory. Some more sophisticated ones are described below.

Category	Value	Description	Default
Buttons	auto hide buttons on NavPage	Hide the buttons on the navigation page after a timeout (can be configured). Clicking on the right border will bring the buttons back.	off

Buttons	auto hide buttons on Dashboard Pages	Hide the buttons on the navigation page after a timeout (can be configured). Clicking on the right border will bring the buttons back.	off
Buttons	time(s) to hide buttons on enabled pages	Time (in seconds) until the buttons are hidden on the pages where this is configured.	30
Buttons	show shade when buttons hidden	Show a shadow area on the right side when the buttons are invisible. You need to click on this are to bring the buttons back.	on
Layout	red icons in title	show small red icons in the page upper right corner for anchor watch and disconnected mode	on
Layout	title icons on dashboard page	also show the red icons for anchor watch and disconnected on dashboard pages	on
Layout	start with last split mode	if enabled the WebApp will start in the same mode (split/unsplit) like it was left (since 20240616)	off

Navigation	boat direction	Define which NMEA value will be used to draw the boat symbol orientation.: COG, HDT, HDM. If you configure HDT or HDM and this value is not being received it will fall back to COG. The course vector will be determined by COG in any case. Since 20220421: If HDT or HDM are used a boat symbol will be drawn instead of the default arrow being used for COG. Can be customized by user icons .	COG
Navigation	add dashed vector for hdt/hdm	Show an additional dashed course vector if you configured HDM or HDT for the boat orientation.	on
Navigation	Rotation Tolerance	If you configured "course up" for the map display it will not be rotated immediately for course changes below this value. This will make the map display more readable.	15
Navigation	zero SOG detect (20220421)	If the boat is rotated based on COG and there is no movement any more (SOG below limit) switch of course	

		vectors and map rotation. Show a circle for the boat.	
Navigation	zero SOG detect below (20220421)	Limit for the SOG (in kn) for the zero SOG detect feature.	
AIS	Class B rel size	class B AIS targets can be shown at reduced (increased) size	0.6
AIS	reduce details in AIS list	in the AIS list you can reduce the amount of information that is shown for every target. This can be helpful if you have a laggy display on slow devices.	off
AIS	First AIS label Second AIS label Third AIS label	select which information should be shown near the AIS target on the map	first: name/mmsi
AIS	only show moving AIS targets	if selected - do not show AIS targest that do not move	off
AIS	min speed (kn) for AIS target display	only if "only show moving AIS targets" - the maximum speed of an AIS target to be considered "not moving"	0.5kn

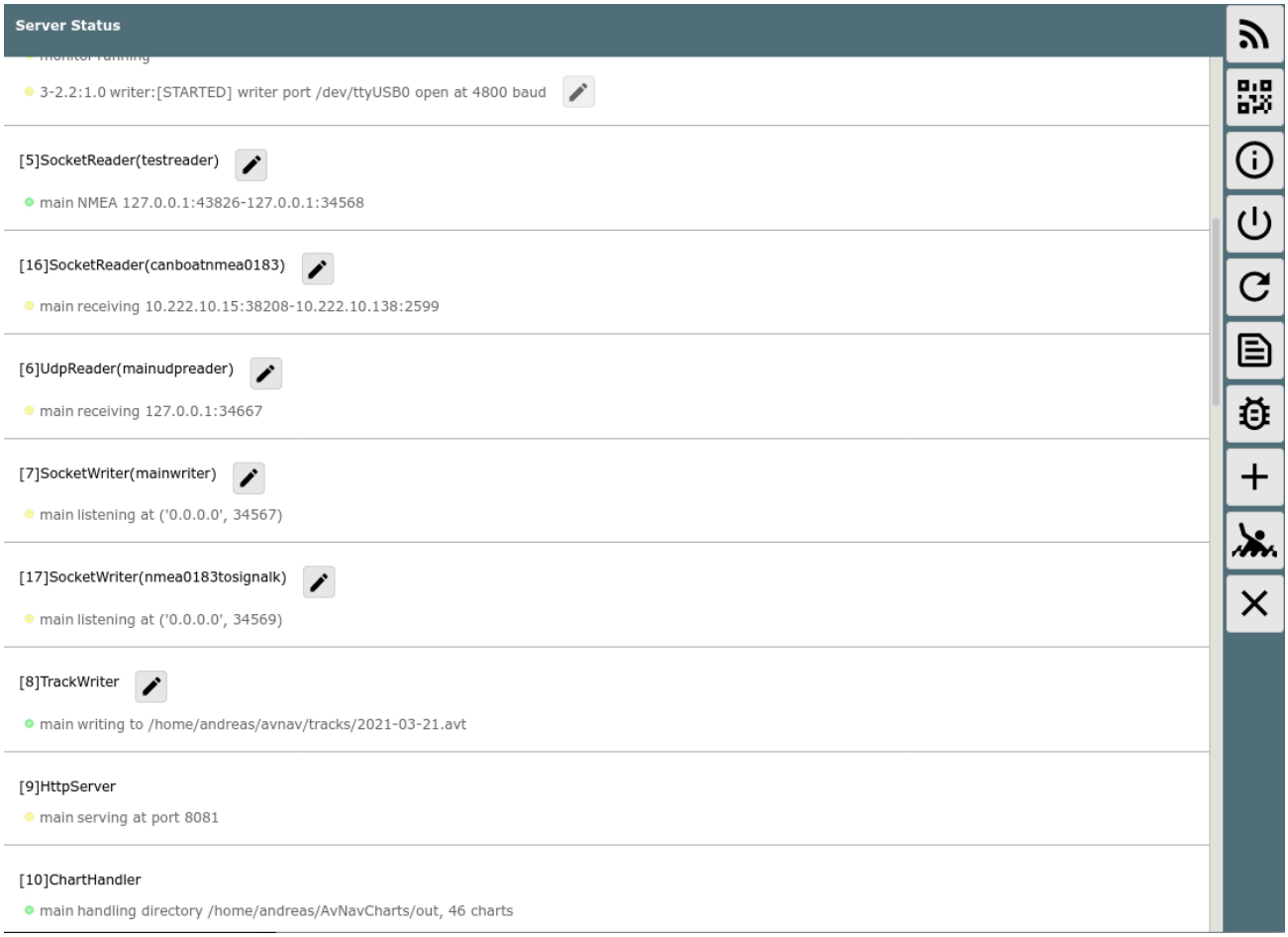
Map	start with last map	if activated the WebApp will directly open the navigation page with the last used map on start up (since 20240616).	off
Map	automatic zoom	If you have locked the map to the boat position the zoom level will be automatically adapted to the available tiles in the area of the current display. So if you e.g. move from an area with higher zoom levels available (like a port) to an area where only lower zoom levels ara available the map display will zoom out automatically (not for o-charts).	on
Map	float map behind buttons	If you activate this setting the buttons and widgets will "float" on top of the map without a fixed background.	off
Map	Increase Fonts on High Res	Increase the symbol sizes and text fonts on high resolution displays.	on
Map	scale the map display	Some raster charts have a display with very small fonts and symbol sizes. This sometimes is hardly readable. With this setting you can scale up (or down) the display of the chart tiles	1

		to adapt to your personal preferences.	
Map	zoom up lower layers	If there is no tile for the selected zoom level available just try to load tiles from lower levels and scale them up. As this requires addtional communication with the server it could impact the performance. The value defines how many lower levels will be tried to find a tile.	4
Map	zoom up lower layers for online sources	The same behavior like "zoom up lower layers" for charts that are directly loaded from the net (also for o-charts). Typically it does not make a lot of sense for such charts - so the default is off(0).	0
Map	Click Tolerance	The "catching area" in pixels for the click on chart objects. A large area would potentially select the wron object, smaller areas could make touch actions difficult.	60
Map	lock boat mode	you can modufy the behavior of the lock button  on the navigation page. You can determine at which	center

		screen position the boat will be locked. center: center of the screen current: current position of the boat on the screen ask: show a select dialog every time
Route	Approach	Destination to waypoint (in meters) for potential switching to next waypoint if: - the distance to the current target waypoint increases and - the distance to the next waypoint decreases
Remote	...	See the description

The Server/Status Page

From the main page you get here using the button. You will see internal status information from the server.



Buttons

Icon	Name	Function
	StatusWpa	got to wifi configuration to set up the connection to an external wifi (not on android,only visible if configured)

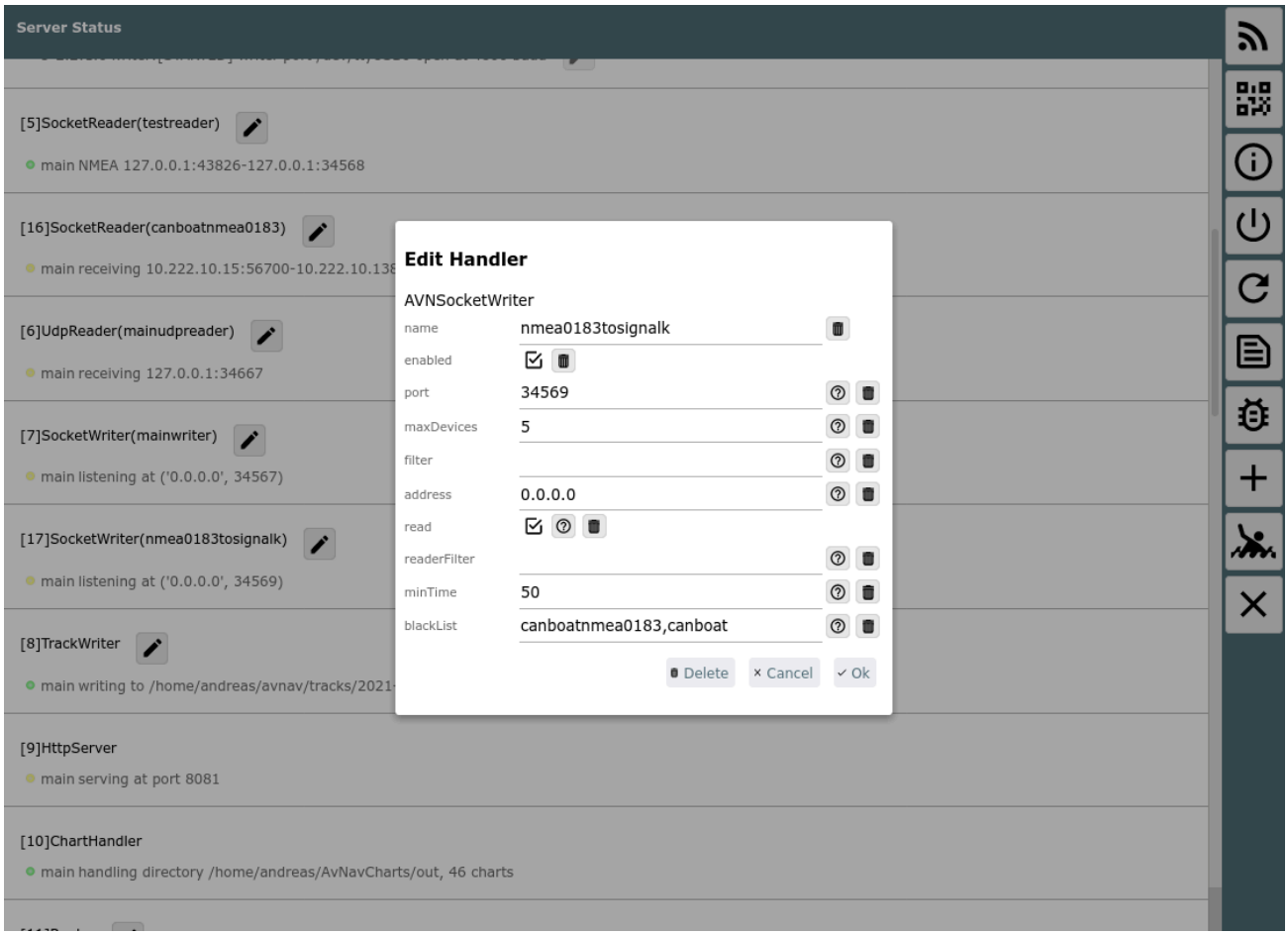
	StatusAddresses	go to the list of server addresses. QR codes will be shown that allow for scanning by a different device for easily connecting this device
	StatusAndroid	git to android settings (only android)
	StatusShutdown	start the shutdown of the server computer(not android). There will be a soft shutdown of the server. There is a safety dialog before the shutdown is initiated. After the shutdown has started you should wait for the headline to turn red.
	StatusRestart	restart the avnav server software (new 20210322)
	StatusLog	Show the current server log (app. 300000kByte from the end), also allow to download the log from here (new 20210322)
	StatusDebug	shows a dialog to enable debug mode on the server for a certain time. Additionally you can set a filter for the log output.
	MainInfo	license and privacy information
	StatusAdd	Add a new handler to the server configuration (like serial input/output, socket connection,...)
	MOB	man over board (see main page)
	Cancel	zurück zur vorigen Seite.

The status informations of the various server parts are shown. You are able to see which connections are there and if NMEA data is received. This page is important for trouble shooting.
The page has an auto update.

Server Configuration

New 20210322, Android from 20210424 - see Android Doc

The server has a lot of internal "handlers" that deal with different tasks. Most of them can be configured on this page using the ✎ beside the status information. This will bring up a dialog to configure the handler.



You can adapt various settings for the handler. For most of them there is a help button available giving you some short help. By using the 🗑 button you can reset a value to its default. If the handler supports deleting you can also remove it here. Most of the handlers also support enable and disable. So you can typically just disable a handler instead of deleting it. You can find

detailed information about the handlers and their parameters at the config documentation.

All changes that you will make here will be stored in the avnav_server.xml but will become active without a restart. The status page display will typically need a couple of seconds until your changes will become visible there.

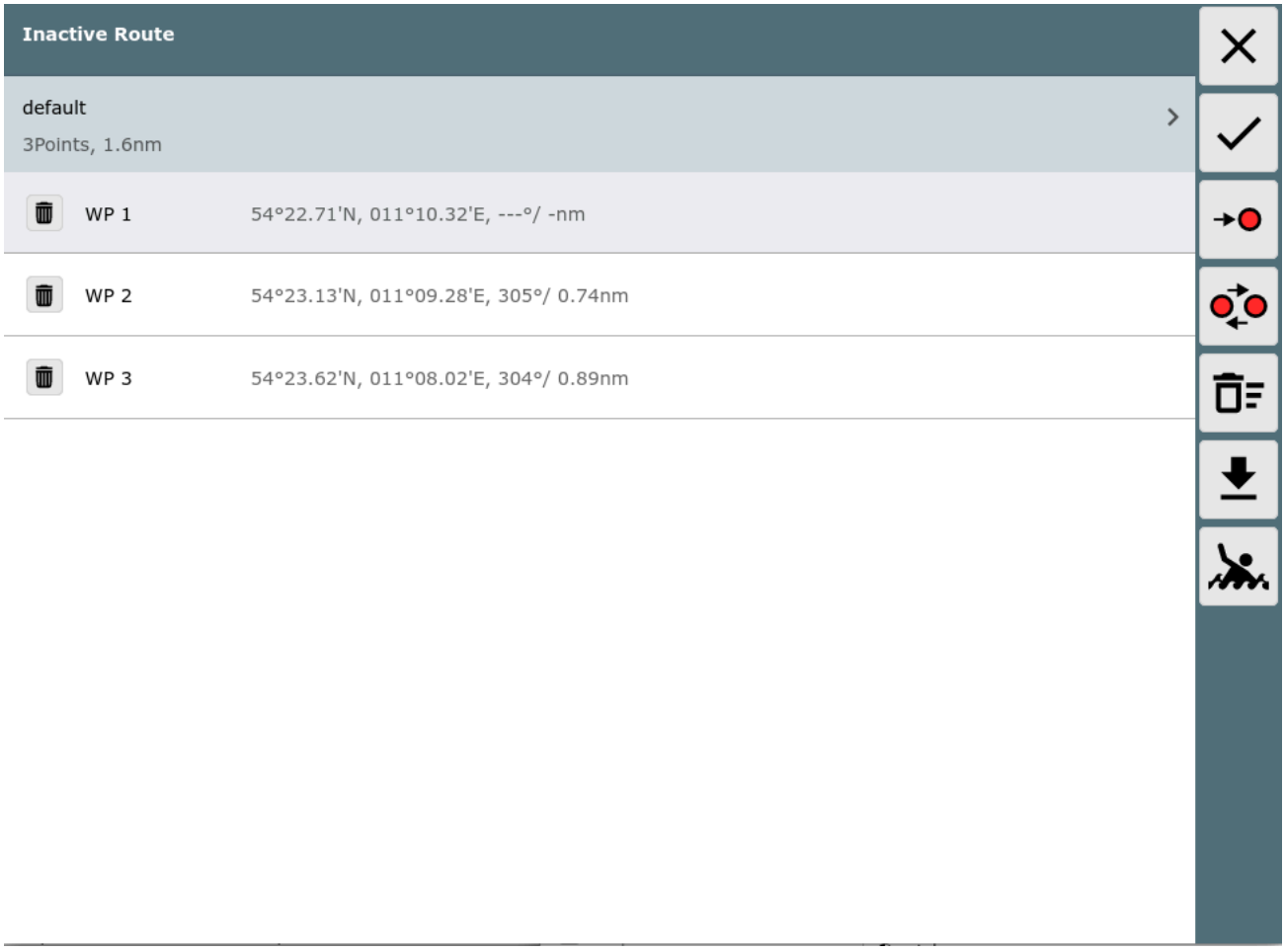
You can add a new handler using the + button.

Whenever you select a port for sockets or a serial interface AvNav will check internally to avoid using the same resource multiple times.
If for some reason AvNav will corrupt it's config file during the change process it will fallback on the next start to the last working version and will display the error at the Config Handler on this page.

Route List Page

Overview

You can reach this page by clicking "Edit" in the Route Dialog shown at the Route Editor.

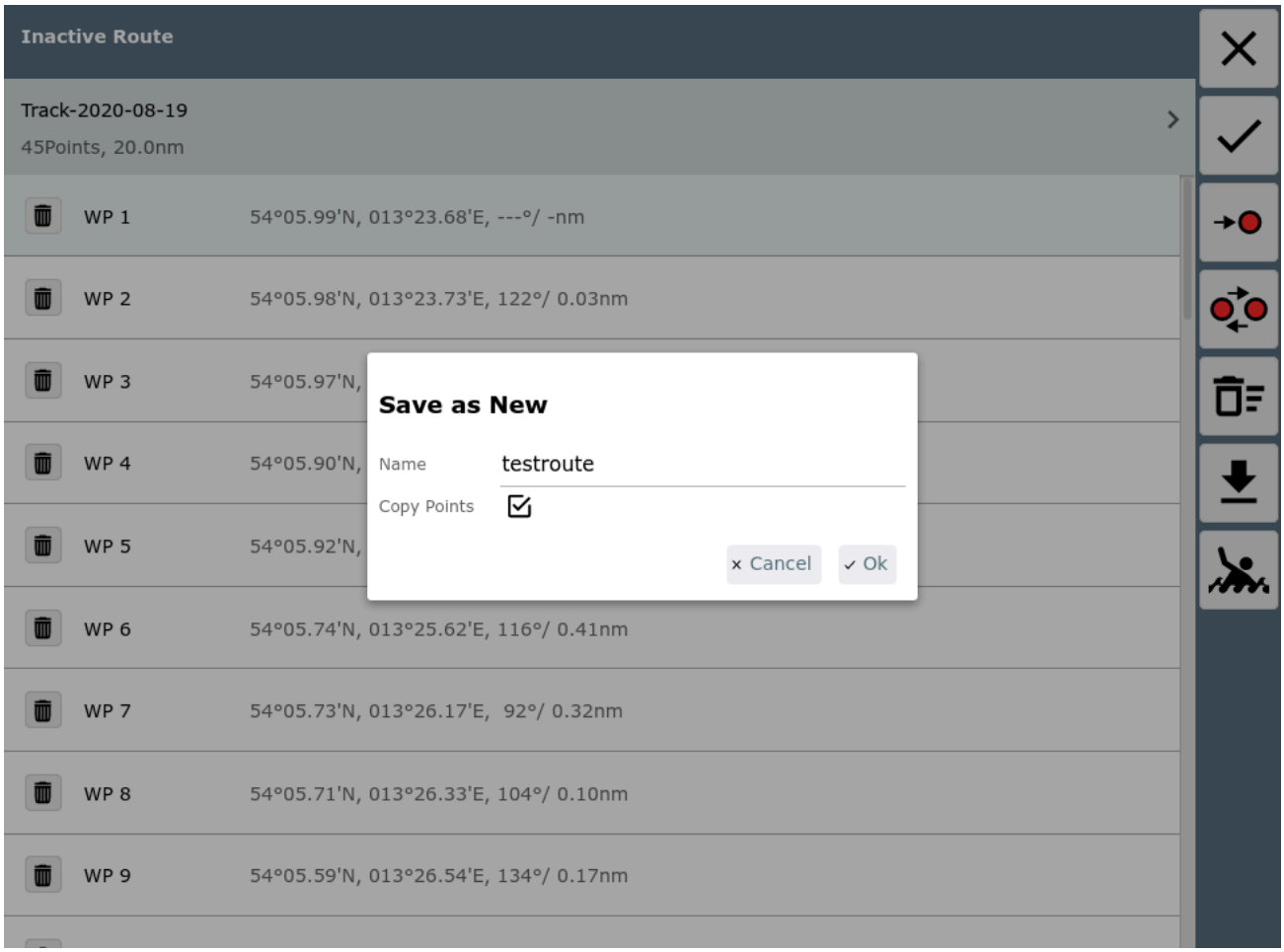


Buttons

Icon	Name	Function

	RoutePageOk	Save changes and return to Route Editor.
	NavGoto	Start navigation to the currently selected waypoint (marked by grey background).
	NavInvert	Invert the route.
	NavDeleteAll	Empty the route
	RoutePageDownload	Open the Files/Download Page routes section. You can select a route to be edited - or upload/download routes. After selecting a route there you will return to this page.
	MOB	Man over Board(see Mainpage)
	Cancel	Cancel (discard changes) and return to the Route Editor

You can edit all the points in the route or create a copy of the route. By clicking on the route name a dialog is displayed.

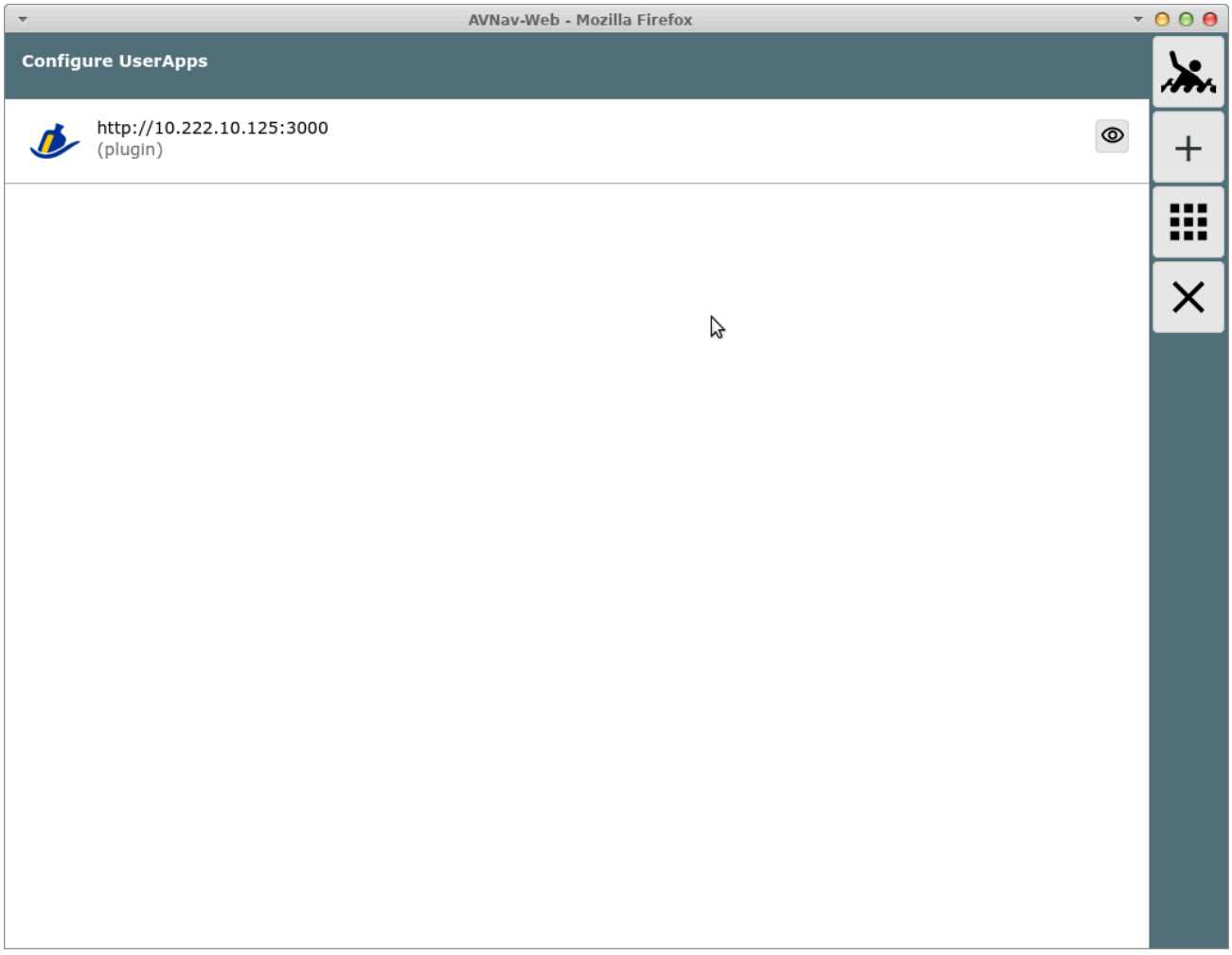


After clicking "Ok" you return to the Route Editor to edit your new route.

Configuration of User Apps

You will get here by clicking the button on the settings page

A user app is an external or internal HTML page to be shown at the user app page in an iframe.

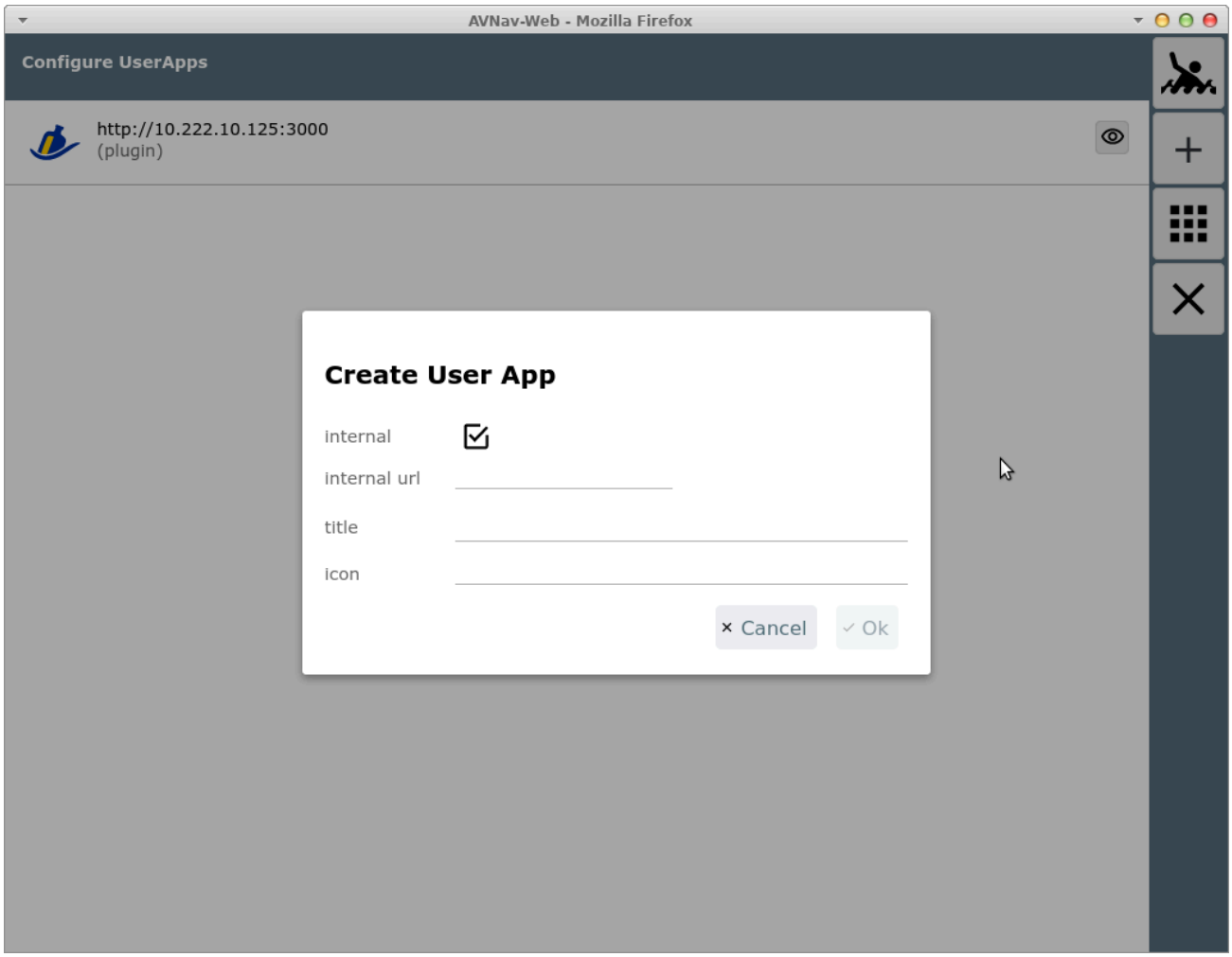


All configured user apps will be shown.

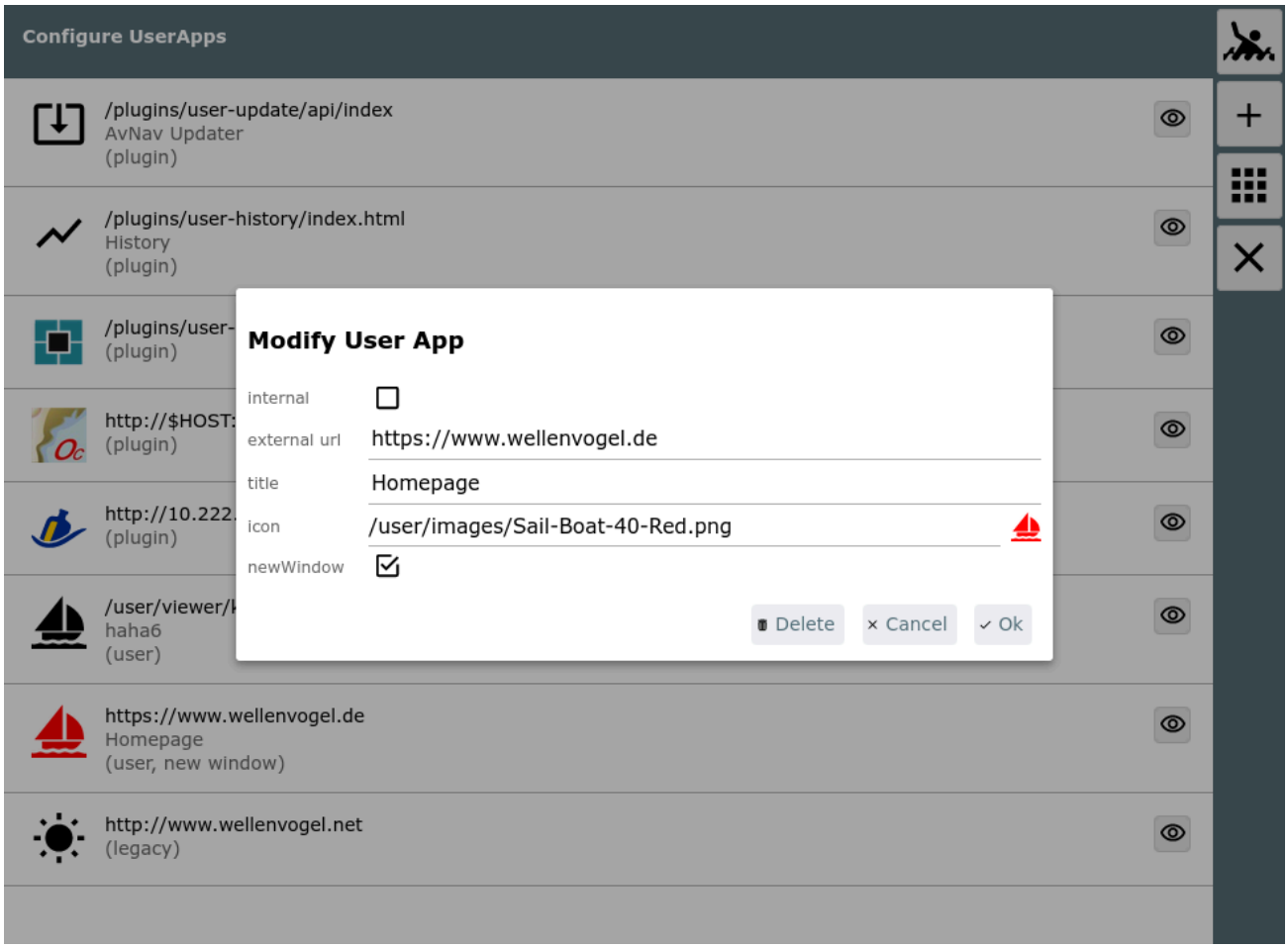
Buttons

Icon	Name	Function
	MOB	man over board (see main page)
+	AddonConfigPlus	add a new configuration
	AddonConfigAddOns	show the user app page
	Cancel	back to the previous page

For each configuration the url and an optional title are displayed. Additionally the origin of this configuration is indicated (plugin in the screenshot). For defunct configurations (e.g. missing icon) "invalid" will be shown. By clicking the configuration you can modify it. Clicking on  will take you to the user app page and the selected configuration will be displayed there.
When adding or editing a configuration a dialog will open.



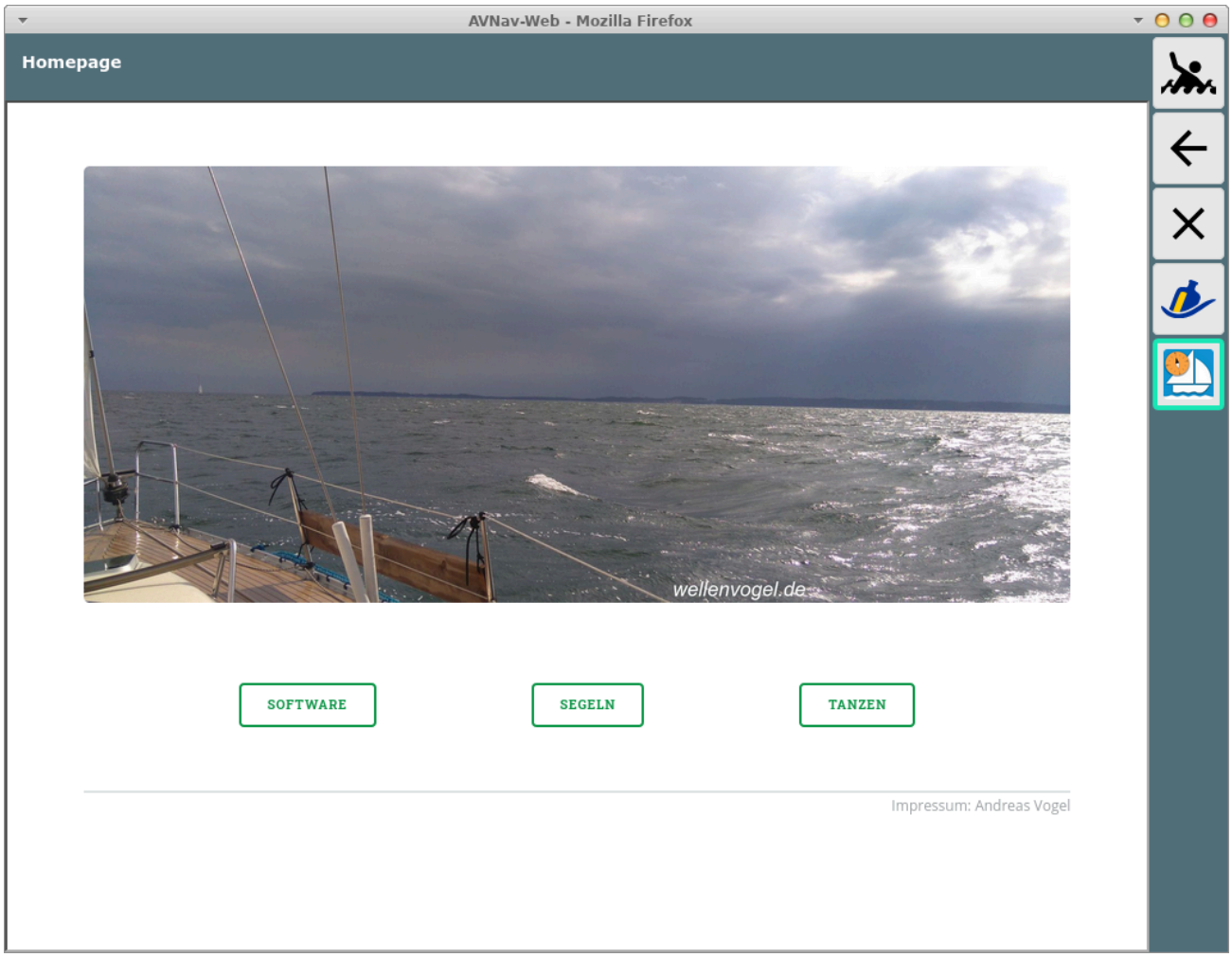
You can select to use either an internal or external HTML page (check box). In the screenshot an internal page is selected. If you select an external page you need to provide the complete url as http(s)://.... If the page lives on the same server as AvNav (like the signalk Web UI) you should replace the hostname with $HOST in your url. AvNav will automatically use the correct ip address in this case.



Internal HTML pages must be uploaded to the user file directory prior to defining them as user app.

For each user app an icon file (svg, png, jpeg) is required. This also has to be uploaded via the Files/Download page, either to the user files or to the images category.
After selecting the url and the icon you can provide a title. An empty title empty will result in no caption at the page.
After saving you can directly test it using the button.

If "newWindow (ab Version 20220225) is unchecked the web page will be displayed in an iframe - otherwise in a new browser window / browser tab.

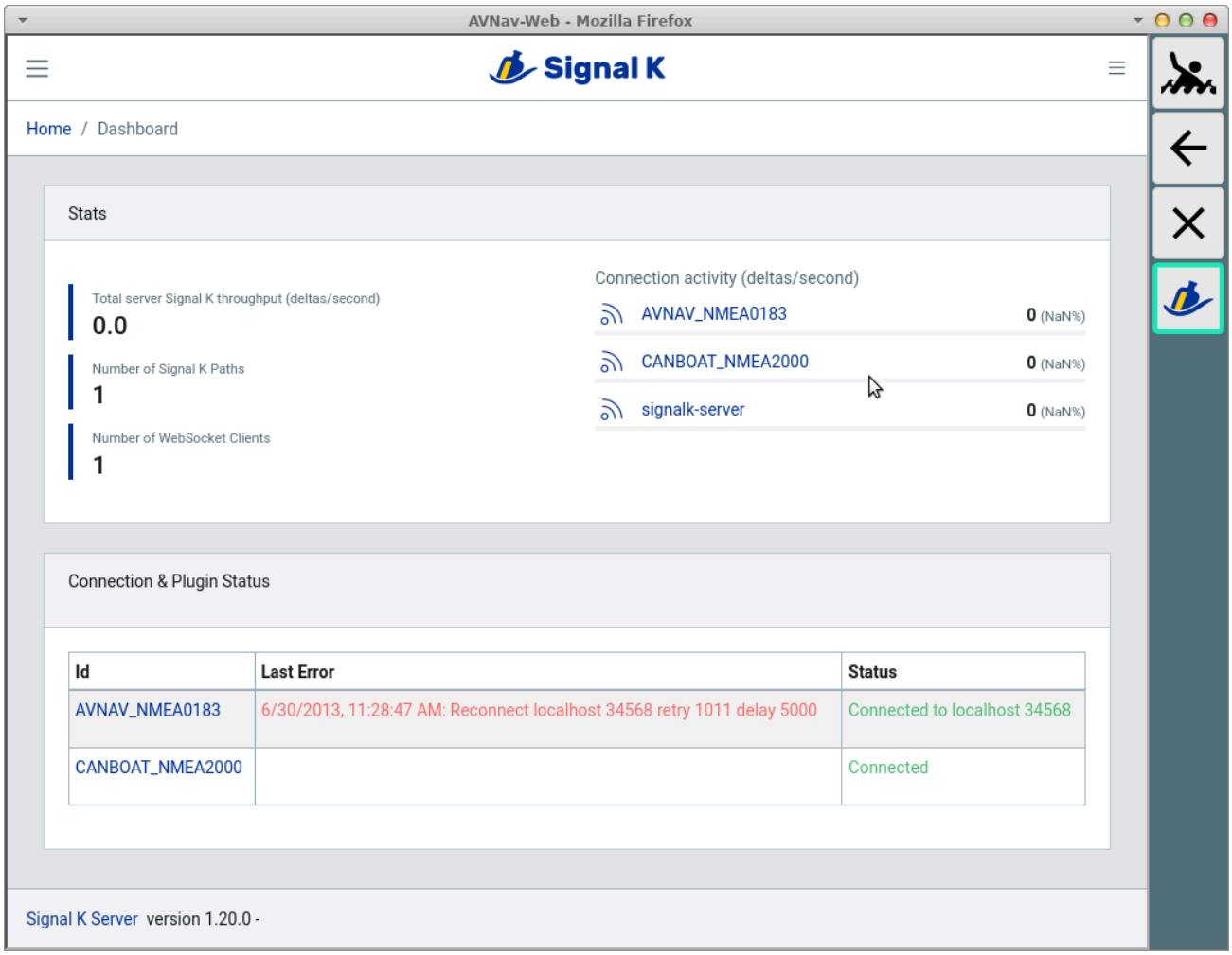


Deleting a HTML file on the files/download page will also remove any user app configuration holding an internal URL referring to it.
If a icon file is deleted, the user app configuration referencing it will remain, however it will turn invalid. Invalid configurations will not be displayed at the user app page.
But you still can modify the configuration on this page to make it valid again.

Whenever a user app configuration is stored, the server updates its avnav_server.xml file. To avoid situations of AvNav not starting next time (because something went wrong during this writing back), an avnav_server.xml.ok file is written on every successful start of AvNav.
If AvNav is unable to start at a later point (due to a corrupted avnav_server.xml) this fallback file will be read instead.

The User App Page

You arrive here using the button on the main page . The button will onyl be visible if such user apps have been configured or have been created by plugins like signalk.



Buttons

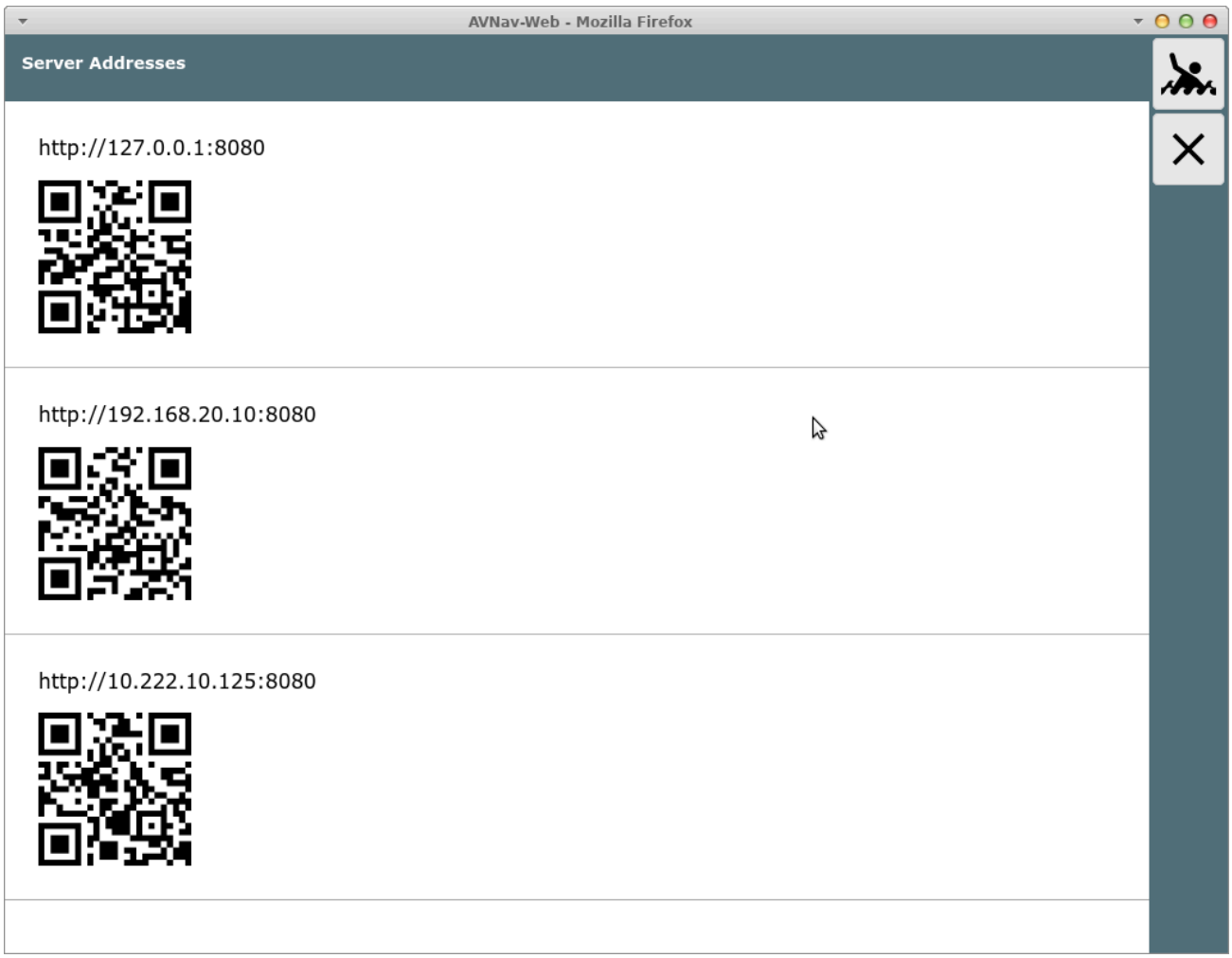
Icon	Name	Function

	MOB	man over board (see main page)
	Back	back button of the browser. Can be used for navigation inside the shown pages. Will not return to previous page within AvNav.
	Cancel	back to mainpage
andere Icons	---	select the user app to be shown

On this page external or internal HTML pages configured as user apps will be displayed in an iframe. Some external pages may prevent this, so you have to try out.

In the screenshot above you see the Web GUI of SignalK as configured by the signalk plugin.

The Address Page



All addresses that can be used to connect to AvNav are shown here. In normal setups (i.e. AvNav is working as an access point and you are using the avnav wifi) the relevant addresses start with 192.168....
If you additionally configured an external wifi connection this network's address will be displayed as well.

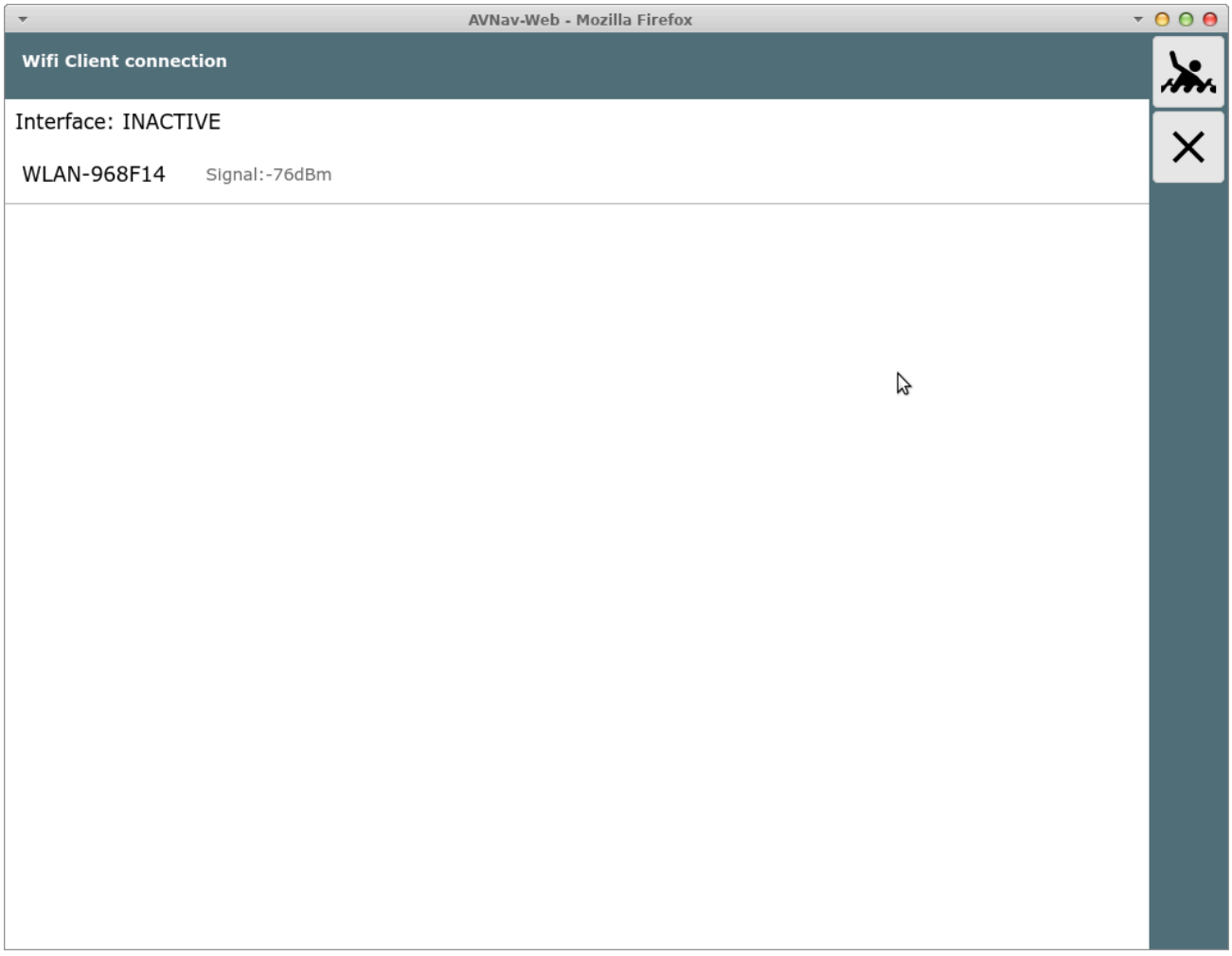
By scanning the QR code with another devices you can easily connect to the AvNav server. You only have to choose the correct address for you - depending on the network you are connected to.

This way to connect is an alternative to the variants described in the Introduction.

The Wifi Configuration Page

- not on android -

This page is accessible by clicking the button on the status page.



This page will only be visible if wifi client handling is configured and a wifi adapter is plugged into the correct USB port.

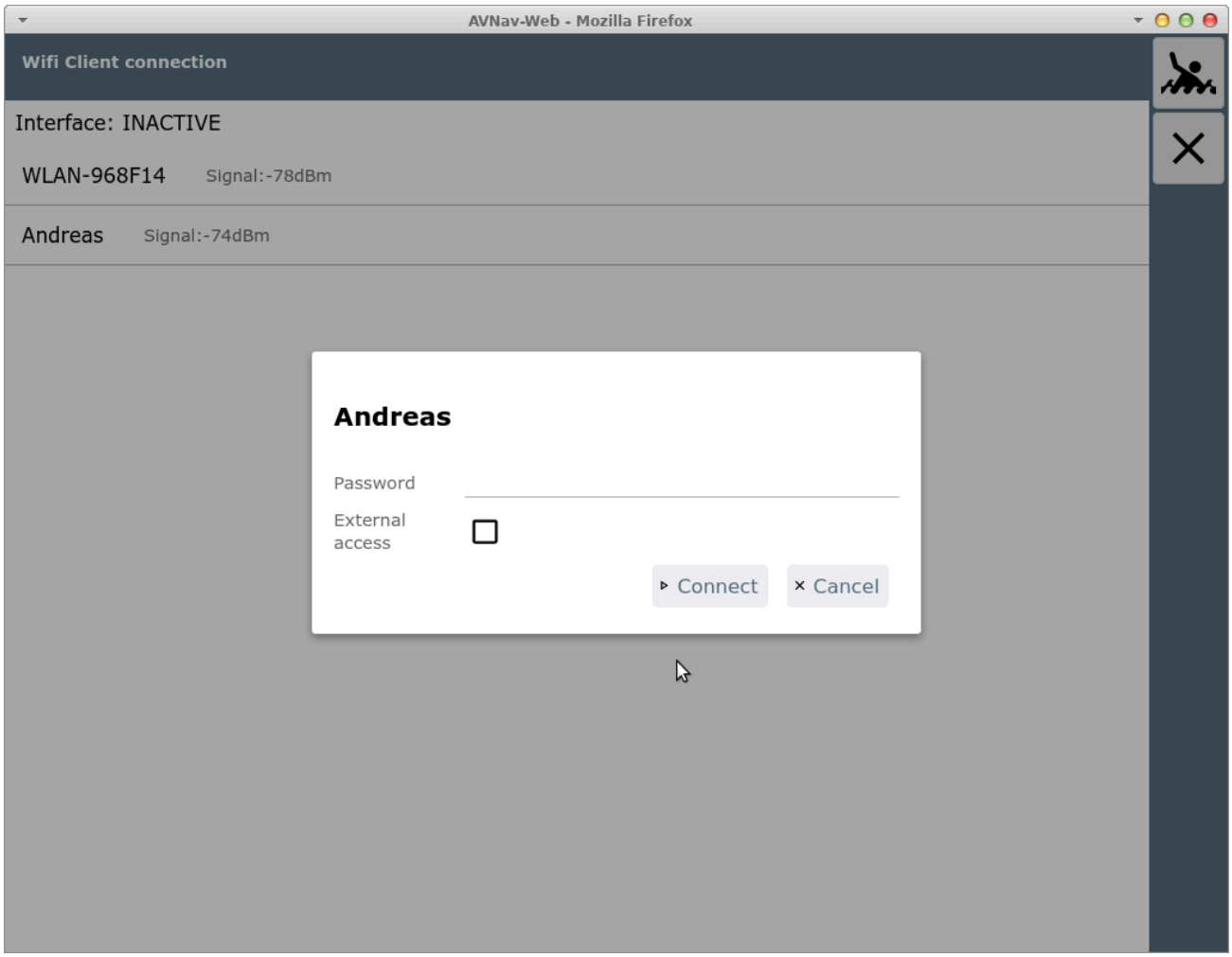
In the picture a Raspberry Pi 3 is shown. Newer ones are equipped with blue USB3.0 sockets. You must connect to the blue socket that is close to the mainboard.

On the Pi's OS level a network interface "wlan-av1" must be visible. In the configuration (avnav_server.xml) an entry

```
<AVNWpaHandler wpaSocket="/var/run/wpa_supplicant/wlan-av1">
</AVNWpaHandler>
```

must exist.

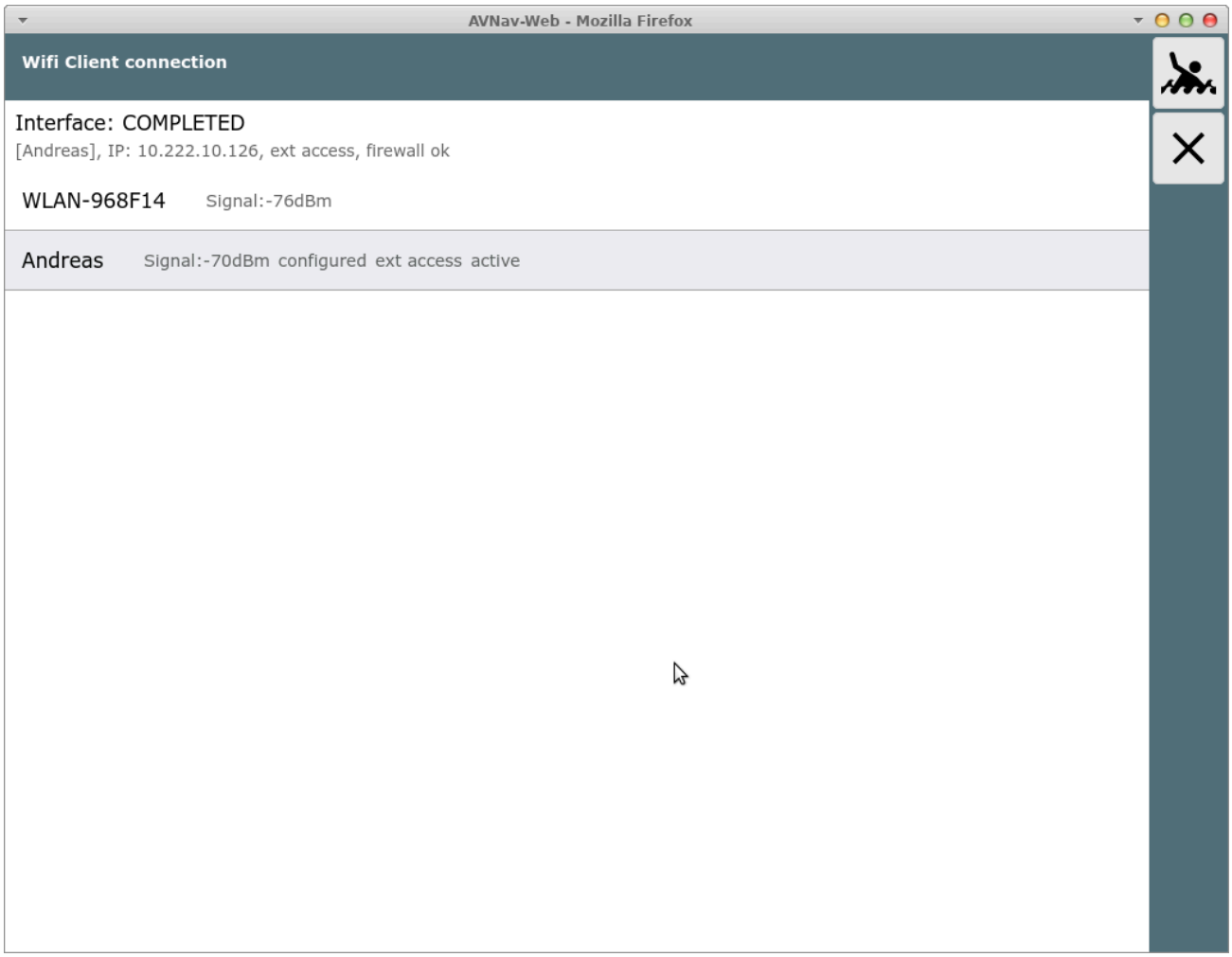
The page displays all wifi networks in range and all configured networks. By clicking an entry you can connect to this network.



Check "external access" to allow external access from this network to your Pi.

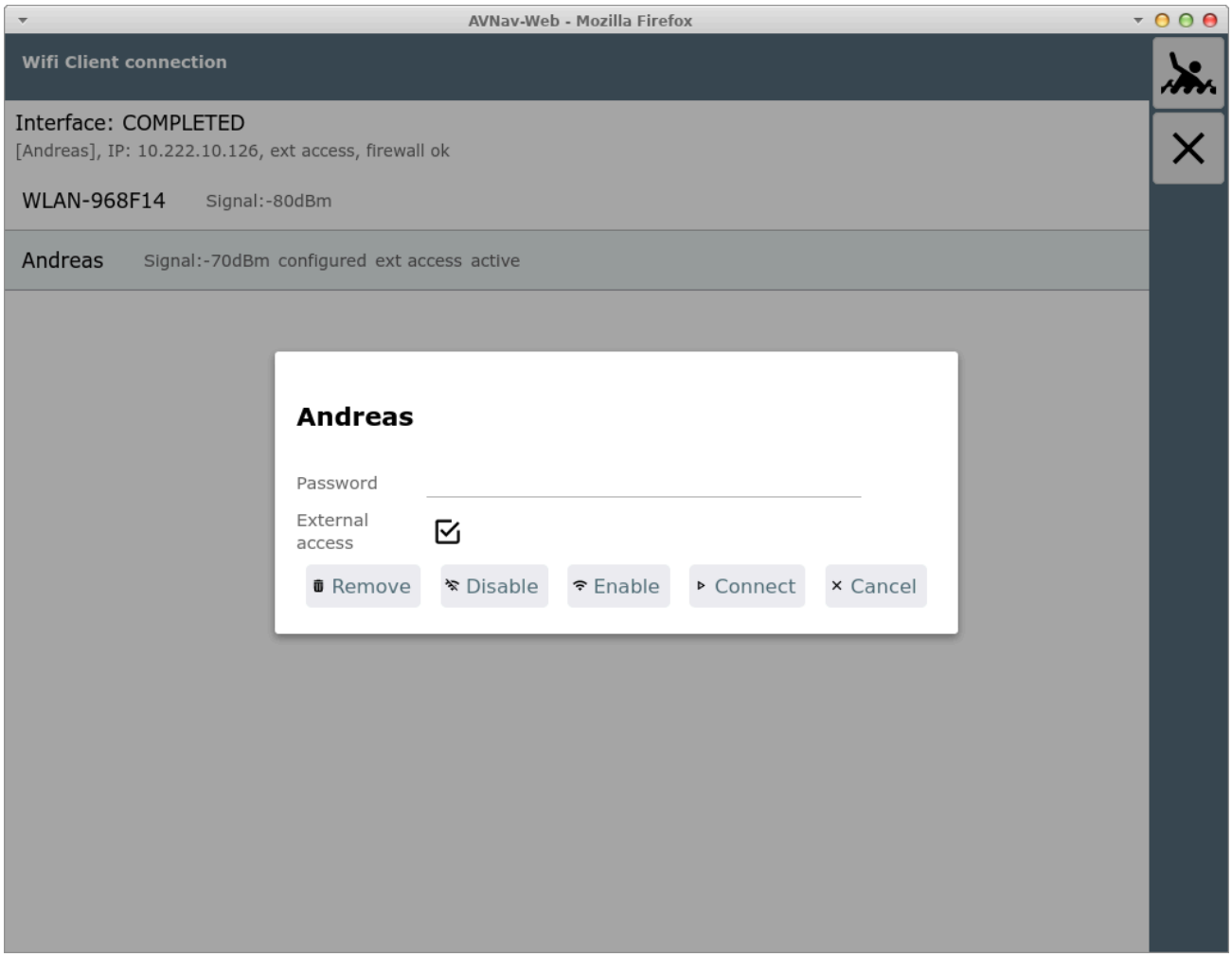
Warning: Never enable external access in a public wifi network! There is no protection in AvNav and anybody within the same network could access your system.
However you may use it if you e.g. run your own LTE router and would like to allow your connected devices to access AvNav.

An established connection will be indicated and the active network is marked by a grey background in the list.



While connecting some status information will be visible close to "Interface".

To disconnect you can click the network again. You can disable it or completely remove it from the configuration.



All the wifi information is stored in the file /etc/wpa_supplicant/wpa_supplicant.conf.

Remote Control

Since version 20210619 AvNav has the ability to control the display on one device from another one - or from the server.
You could e.g.utilize this function if you have a display in your cockpit - but you cannot directly reach it with your hands. So you could have another device (like a smart phone or a real remote control) in reach and use this one to change the display.

Functions

You can switch pages via remite control and you can trigger some other actions on a page (like selecting the chart or changing to another dahsboard page).

In the navigation view the moving and centering of the map or the zoom in / zoom out will be transferred.

Configuration

AvNav has 5 remote control channels. For every display device you can select which channel it should use and whether it should send or receive on this channel (Settings/Remote). For a remote connection to become active you need to assign at least 2 devices to one channel and have one of them enabled sending and one of them receiving.

You can also enable both sending and receiving on one device. Such a device will send out local actions and receive remote ones (after a "dead time" after each local action).

At the server you need to enable the remote channel handler /default: active).

In the server version (not on Android) remote control commands can also be received via UDP or by a plugin.

There is a plugin for the Open Boat Projects IR Remote from chrhartz.

Via UDP or by a plugin various remote control commands can be received (see at Details).

Details

Internally each browser will connect to the configured remote control channel on the server /using WebSockets).On this channel it will send and/or receive commands (depending on it's configuration).

Remote control commands are either key presses or some more complex commands like switching a page.

The commands are expected to be strings, at the UDP API terminated by a NewLine.
They consist of a type and parameters.

Key-Commands

A key command consists of a "K" a space and the key code.

```
K Ctrl-
K a
```

The keys will trigger actions according to their configuration(see Keyboard-Control).

Complex Commands

Those commands directly trigger various actions within AvNav. Parameters are often encoded in JSON.
They are inteded to be used for controlling one display from another - and there format can easily change in the future.
Basically the look like:

```
CP navpage
```

In this example: Switch to page "navpage". A current list of commands can be found in the source code at remotechannel.js.

AvNav Android

After having the AvNav web GUI for the Raspberry Pi already in place for some years, there is now also a genuine android version. It requires android 4.4 (KitKat) or later.

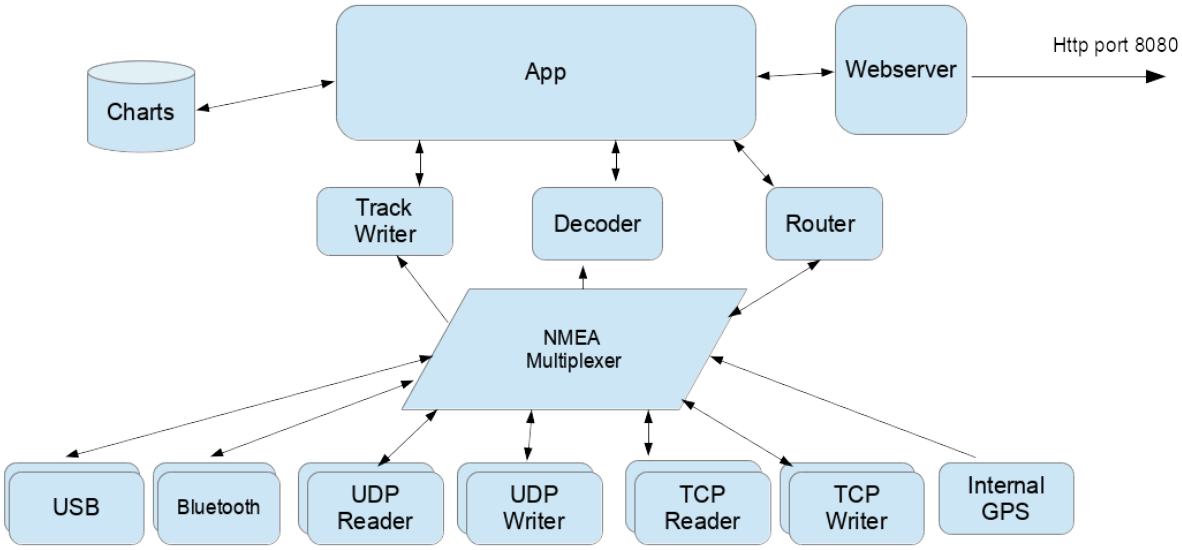
The latest version is available at downloads. Alternatively in the Play Store. You can find previous versions or daily builds via the installation instructions.

Basically the app is packaging the WebApp (see User documentation) into an android app. For Charts check the hints at creating/converting charts.

Functions

(new since 20210424)

Internally the app consists of a couple of functional entities.



Anav Android

The NMEA multiplexer handles NMEA0183 data from various sources. Beside the internal GPS data can be read from sources like TCP connections, UDP ports, USB devices, Bluetooth... Most sources support sending and receiving of NMEA data at the same time. You can configure multiple sources of each type (within the app configuration there is a "handler" for each data source).

name within the app	description
InternalGPS	The location data from the internal GPS will be sent to the multiplexer as NMEA data.
TcpReader	A TCP connection to an external system. AvNav acts as TCP client and will open up the connection. You can either use a plain IP address or a hostname as destination (mdns names like avnav.local are supported).

TcpWriter	AvNav listens for connections from other apps or systems and sends out the NMEA data from the multiplexer. AvNav acts as TCP server.
UdpReader	AvNav receives UDP data at the configured port.
UdpWriter	AvNav sends out UDP data to the configured address and port.
UsbConnection	You can receive (and send) data via a connected USB serial converter device (requires USB OTG functionality of your Android device).
Bluetooth	Connection to a bluetooth device. You need to pair the device (outside AvNav) before you can use it.
NMEA0183 service	A connection to a system the provides its NMEA data as a TCP service via mdns (Bonjour/Avahi) - like e.g. SignalK. AvNav establishes a TCP connection to this device.

You can configure the NMEA multiplexer in a very flexible manner. For each connection you can define input and output filters or blacklists.

Internally the multiplexer hands over its data to the other modules of the app. The decoder prepares the NMEA data for further usage inside the app.

The app itself with the chart display and the dashboards can be used as a normal android app. Additionally you can activate the integrated web server. This allows access to all app functions from a browsers on the same or a different device (similar to the AvNav server variant).

The display part of the app can be terminated while the multiplexer continues to run in background. So AvNav can act as a NMEA data provider

for other navigation apps. You would configure a TcpWriter within AvNav and connect from the other app to localhost (127.0.0.1) and the configured port.

The app can use the device's internal GPS. Additionally you can receive NMEA data via TCP/IP, bluetooth or by an USB-serial adapter if your device supports USB OTG. As a sender for NMEA data you can e.g. use the raspberry version - or any other NMEA0183-wifi gateway. You can also use a plain bluetooth GPS mouse if your device does not have an internal GPS.

Charts and stored Data

Charts are primarily expected in gemf format. Since version 20200325 you can also use mbtiles or xml files for online chart sources. Charts may be located in 2 directories:

- in the 'charts' folder within the app's working directory (choosen at start up). As this must be writable it will typically not be possible to put this on an SD card in newer android versions.
- in a freely selectable, different directory (also on an SD card) "additional charts dir" in the settings
  In the external charts dir you cannot directly use mbtiles. They must be copied into the working directory using the app .

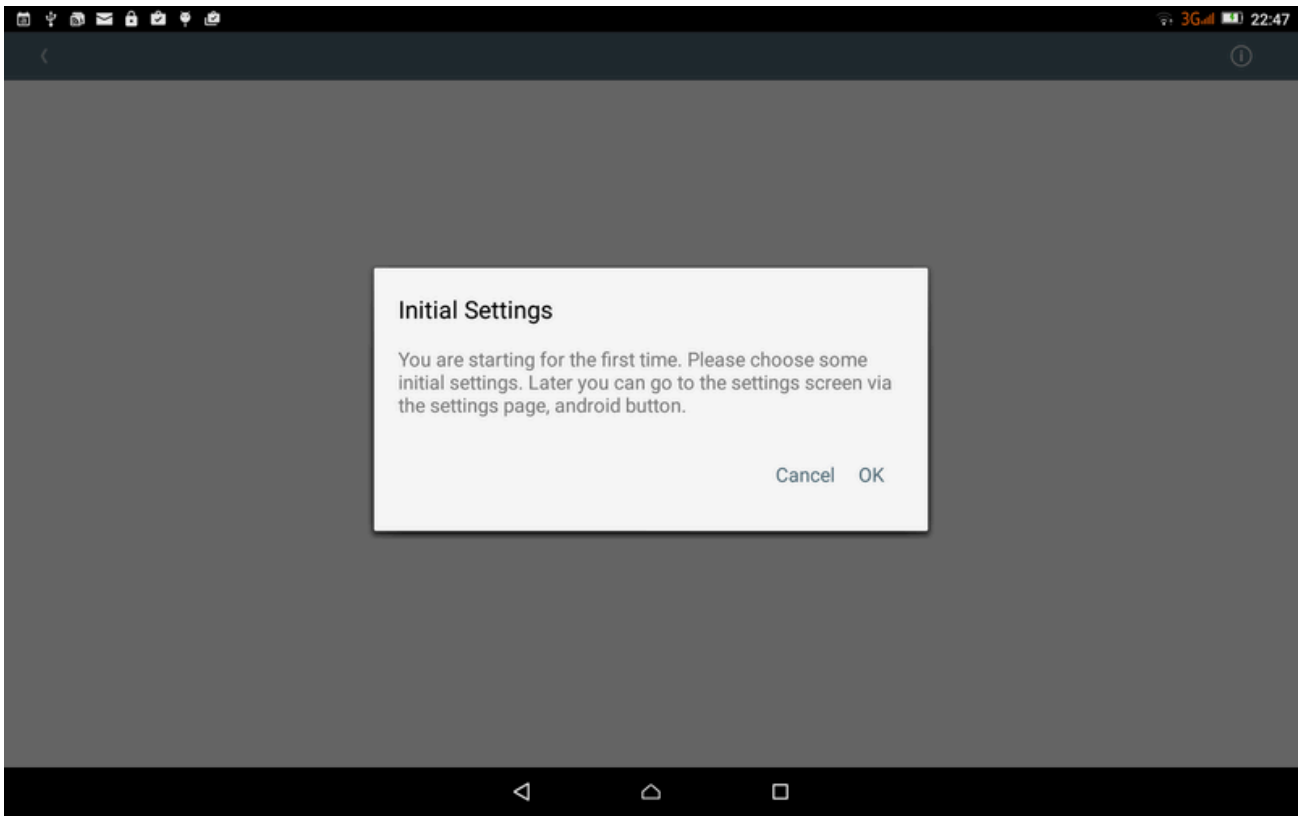
Charts have to be copied to either directory or uploaded by the app. Initially there are some demo charts available, however they require an online internet connection.
If the chosen working directory is not yet present it will be created on first start up (default: internal-sd-card/avnav).

Track and log data will be written to the track directory within the working directory.

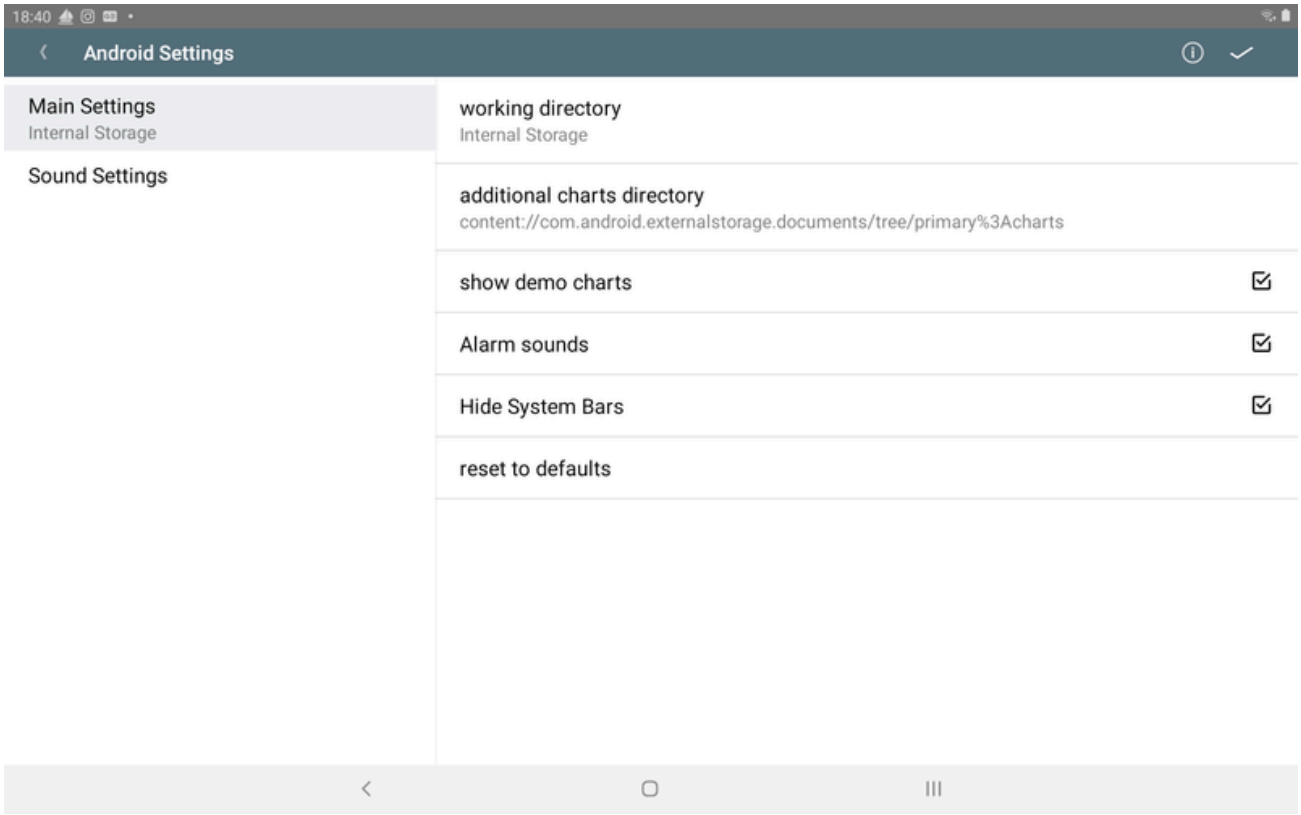
In the routes sub folder, routes will be stored in gpx format. Routes and tracks can be made available to other apps using the download function. To save them you need to have a file manager installed.

Usage

After first start up you will see an introduction page:



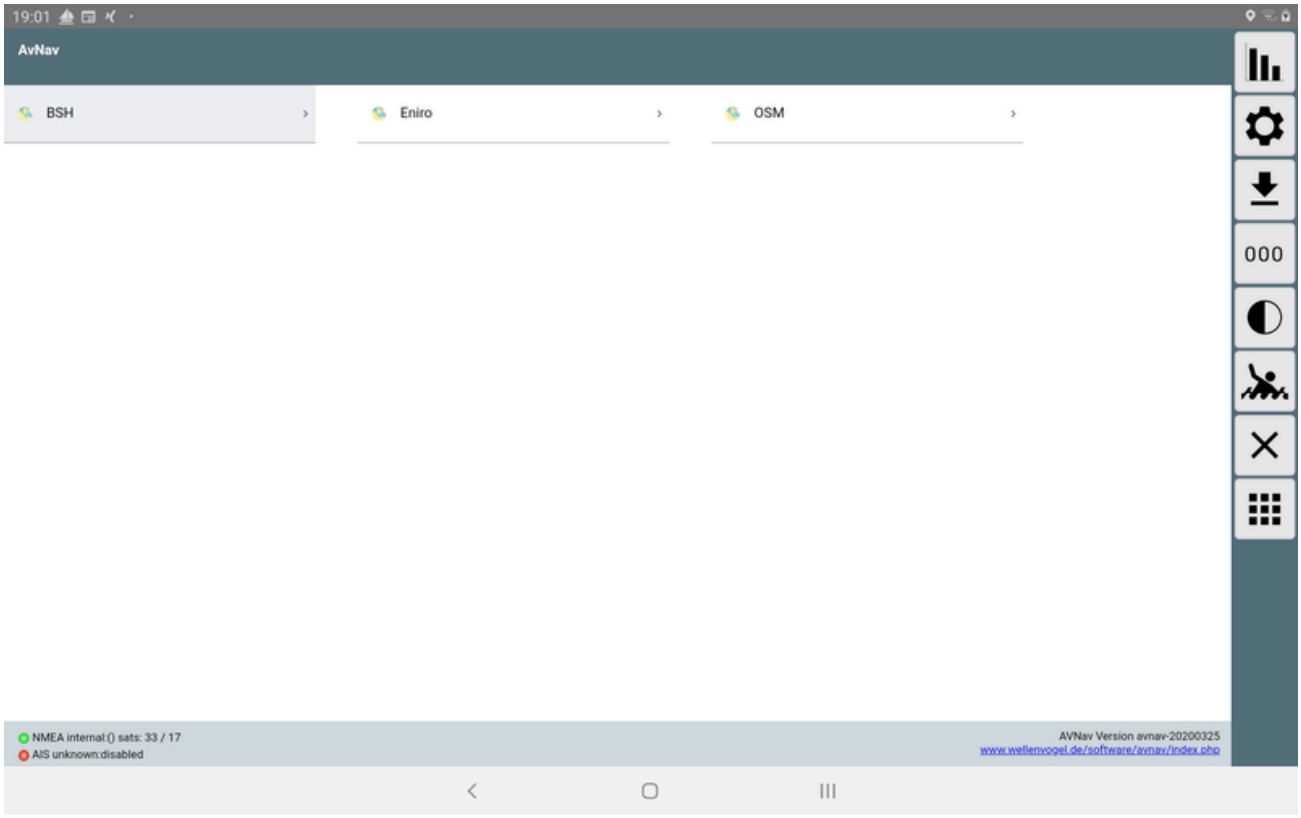
After clicking ok you will see the main settings page.



Amongst other settings, you can select the working directory and the external charts directory.

A complete list of settings can be found here.

You can leave the settings page with the upper right button or by pressing the back button. You will get to the main page of the app.



In the screenshot a valid internal position is shown (green bubble), but currently no AIS data (red bubble).
To return to the android setting move to the settings page within the app and click the android button there.
On further starts, the app will directly present the main page.

External Access

(new since 20210424).

AvNav supports being accessed from other devices using a web browser. To provide this function you need to enable the WebServer in the settings of the app (Status/Server Page).
You need to switch on "externalAccess" (only do this in trusted networks).

With 'mdnsEnabled' a bonjour enabled app (like BonjourBrowser) can directly connect to the server.

In previous versions AvNav was using different modes for that.

- Normal
- External Browser

In normal mode the browser is integrated and it behaves like a standard android app.
In external browser mode the app starts up a web server at a user definable port (default 34567).

Background

The NMEA multiplexer (and the Web Server) of AvNav can run in background. You would e.g. use this if your display is on a different device.

To enter the background mode open the close dialog on the main page ✕ and select "BACKGROUND".

Via the android notification (in the Android notification bar) you can bring back the app to foreground - or close it completely.

Settings

The settings are split into two separate categories:

- Specific Android settings
- Settings for the multiplexer and other main parts

The android settings can be accessed via the button on the settings page or on the server/status page.

Android Main Settings

Name	Meaning	Default
working directory	working directory for your data (sub folders charts,tracks, routes, user, layout)	/storage/sdcard/avnav
additional charts directory	external chart directory (not for mbtiles), SD card recommended	---
show demo charts	demo charts (only with internet connected)	on
Alarm sounds	switch off alarm sounds. You may need to switch them off in your browser as well.	on
Hide System Bars	hide the top and bottom android bars	off
reset to defaults	reset the multiplexer settings to defaults	

Android Sound Settings

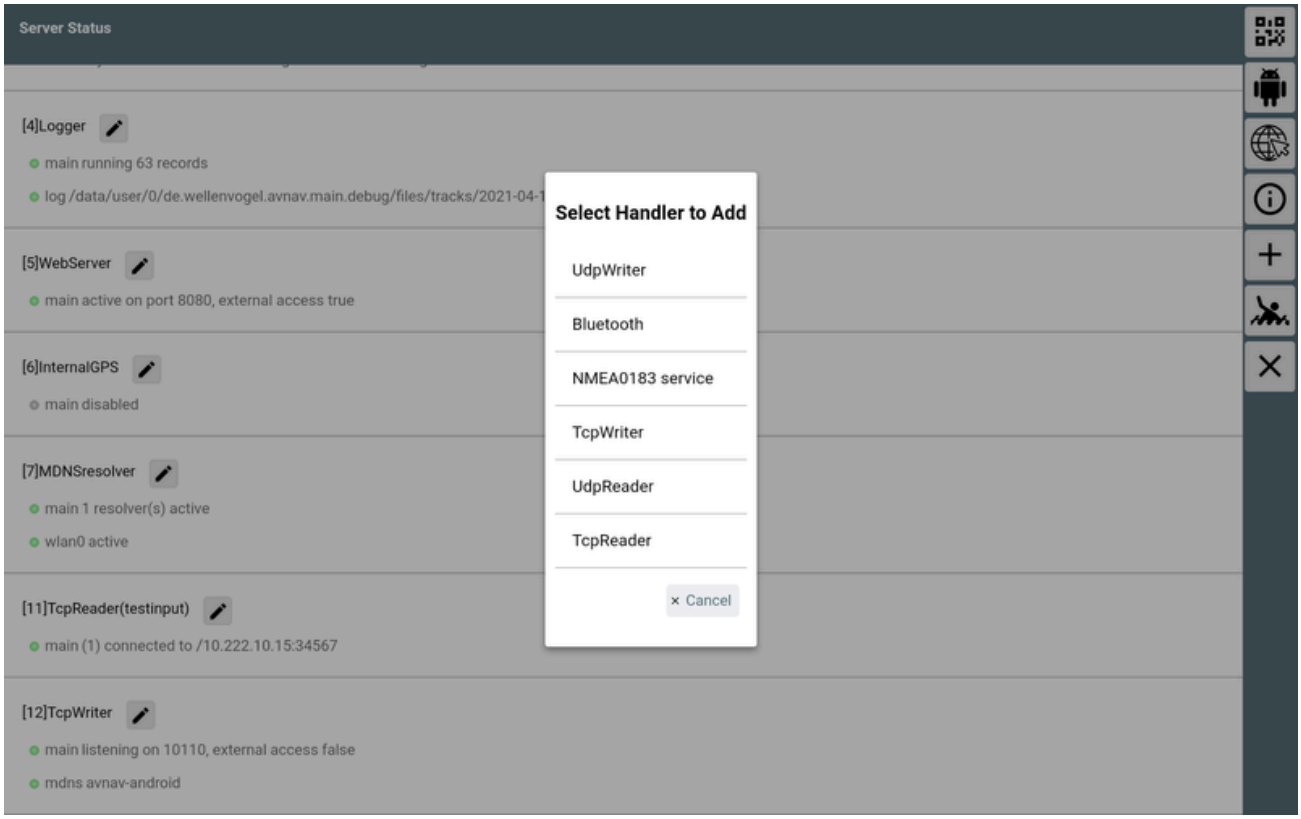
Name	Meaning	Default
Sound for XXX alarm	select the sounds for the alarms	built in
reset to defaults	reset all sound settings to defaults	

Multiplexer Settings

(new since 20210424)

The multiplexer settings are available on the Status/Server Page . For each function (e.g. for each data source of the multiplexer) a so called 'handler' is available. You can check the status of the handlers and change their configuration.

New data sources (or targets) can be added using the + button. A select dialog will provide all handlers that can be currently added (you will e.g. see an USBConnection only if there is an USB device actually connected).



Next to handlers in the status list a button permits to to edit the configuration of the handler.
Most of the parameters shown in the edit dialog have a button providing a short help for the parameter. Using the button you can reset the particular parameter to it's default value.

Some of the parameters are common for multiple handlers:

Name	Description	default
enabled	Activate/deactivate the handler.	depends on handler
name	Name of the handler. Can be used in black lists.	empty
port	TCP or UDP port	
filter/readerFilter/ sendFilter	A NMEA filter. You can define the NMEA sentences to pass. Multiple filters have to be separated by comma. For records starting with a $ the next 2 characters will be ignored (the talker id). A filter for all $XXRMC records will look like: $RMC For AIS only (no other NMEA data): ! All RMC and RMB records: $RMC,$RMB If you want to invert the filter, prefix the expression with ^. ^$RMB,^$APB	empty
blacklist	A comma separated list of names. NMEA data from sources with those names will not be sent out.	empty

The "Handlers" and their parameters:

Decoder

Parameter	Description	default
ownMMSI	Own MMSI, will be suppressed in AIS displays	empty
posAge	Allowed age (in seconds) for the GPS position. After that time the position will be dropped if no new position has been received.	10
nmeaAge	Allowed age (in seconds) for NMEA data (except position)	600
aisAge	Allowed age for AIS Data	1200
readTimeout	Timeout in seconds for the display whether valid data is available or not	10

Route

Parameter	Beschreibung	default
computeRMB	if switched on RMB records will be created if the routing is active	on

Track

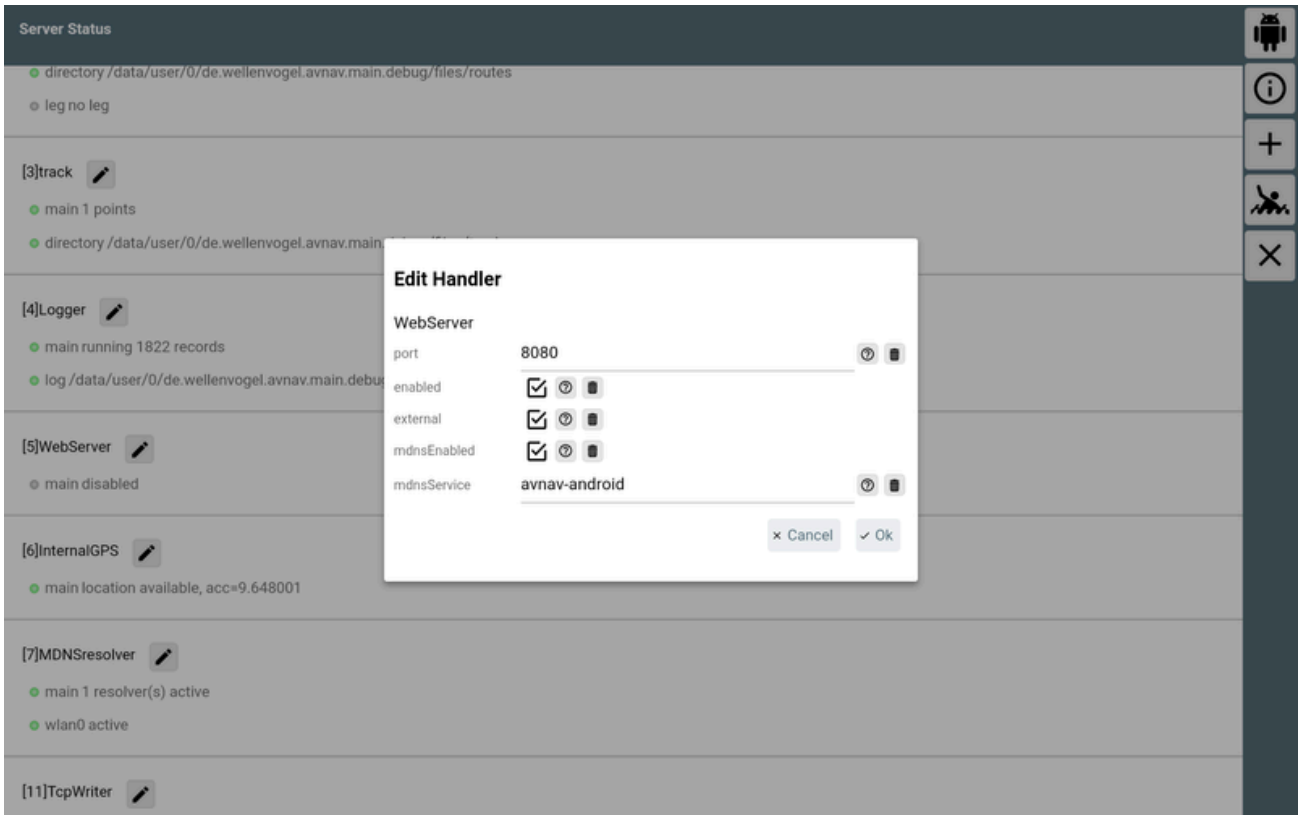
Parameter	Beschreibung	default
interval	Interval in seconds for writing the track as gpx file.	300

distance	Minimal distance between two trackpints in meters	25
minTime	Minimal time in seconds before a new track point is written.	10
length	Length of the shown track (in hours)	25

Logger

NMEA logger

WebServer



Parameter	Description	default
port	The TCP Port the server is listening on	8080
external	If active external devices can connect (otherwise: only local apps)	off

	Hint: Be careful to only use this in a trusted network. There is no further protection against unauthorized access inside the app.	
mdnsEnabled	Announce the service via mnds (will allo Bonjour apps to find it)	on
mdnsService	The name you will see in mdns.	avnav-android

InternalGPS

The internal GPS of the device.

MDNSResolver

The handler for resolving and announcing mdns names.

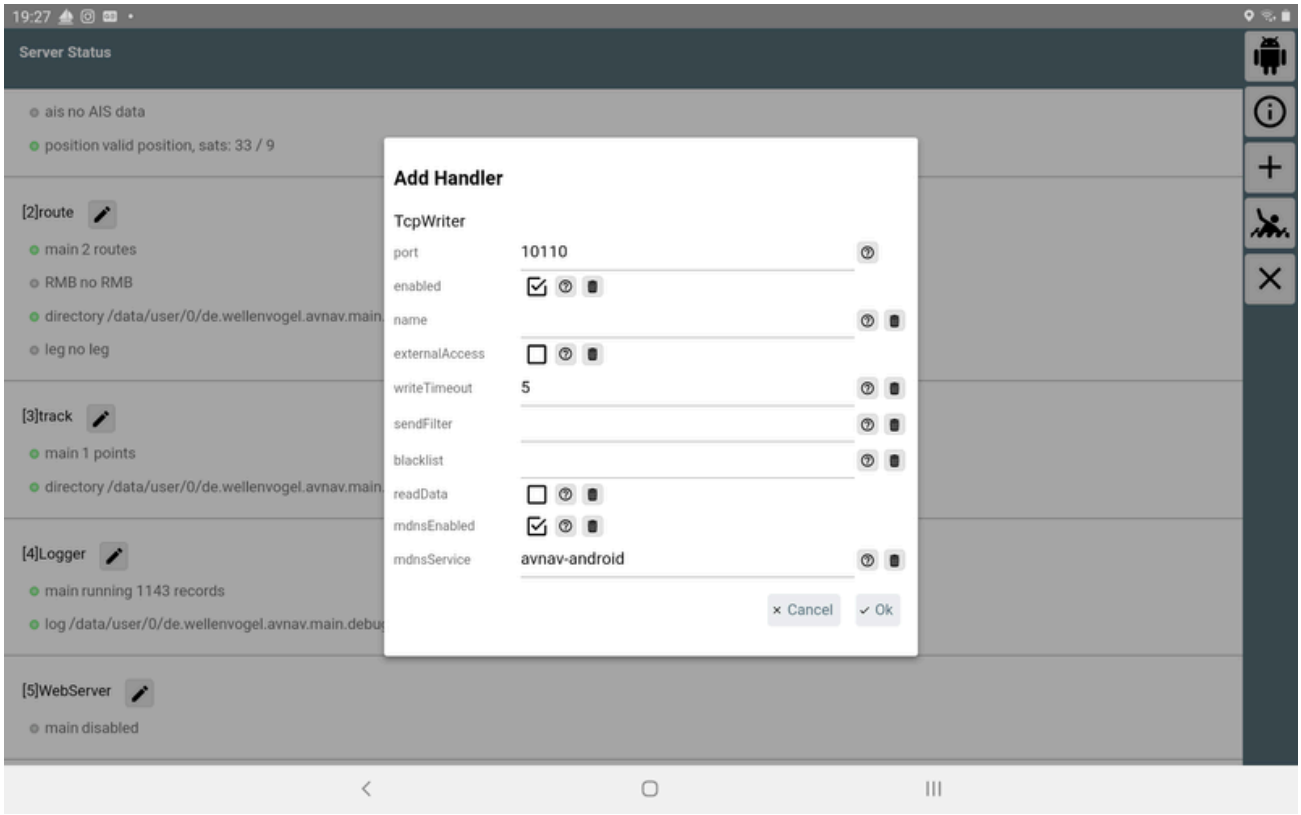
TcpReader

Parameter	Description	default
ipAddress	The ip address or the hostname of the server you want to connect to. This can also be a MDNS name like avnav.local .	---
port	The Ip port you want to connect to.	---
sendOut	If active NMEA data will be sent out (otherwise received only) on this connection.	off
readTimeout	Mark the connection as inactive if no NMEA data are received for this time (seconds).	10

writeTimeout	Write timeout for one NMEA record (in seconds). Close and reopen the connection. Use 0 to disable writeTimeout.	5
connectTimeout	Timeout for connecting (seconds, 0 - system-default).	0
closeOnTimeout	Close (and reopen) the connection if no NMEA data has been received within readTimeout.	on

TcpWriter

A TCPWriter provides NMEA data for other apps.



Parameter	Description	default
port	The IP port that the server will listen on	---

externalAccess	If active other devices can connect. Otherwise only apps from the same device. Hint: Be careful to only use this in a trusted network. There is no further protection against unauthorized access inside the app.	aus
writeTimeout	Write timeout for a NMEA record (in seconds). Close and reopen the connection, choose 0 to disable writeTimeout.	5
readData	If active also read data from an established connection (otherwise write only)	off
mdnsEnabled	Announce this service via mdns (type: _nmea-0183._tcp)	on
mdnsService	The name of this service on mdns	

UdpReader

An UDP Reader receives data from an UDP port.

Parameter	Description	default
port	The UDP port the data will be received on.	---
externalAccess	If active external devices can send data. Otherwise only apps from the same device. Hint: Be careful to only use this in a trusted network. There is no further protection against unauthorized access inside the app.	off
readTimeout	Show the connection to be inactive if no NMEA data have been received for this time	10

	(seconds).

UdpWriter

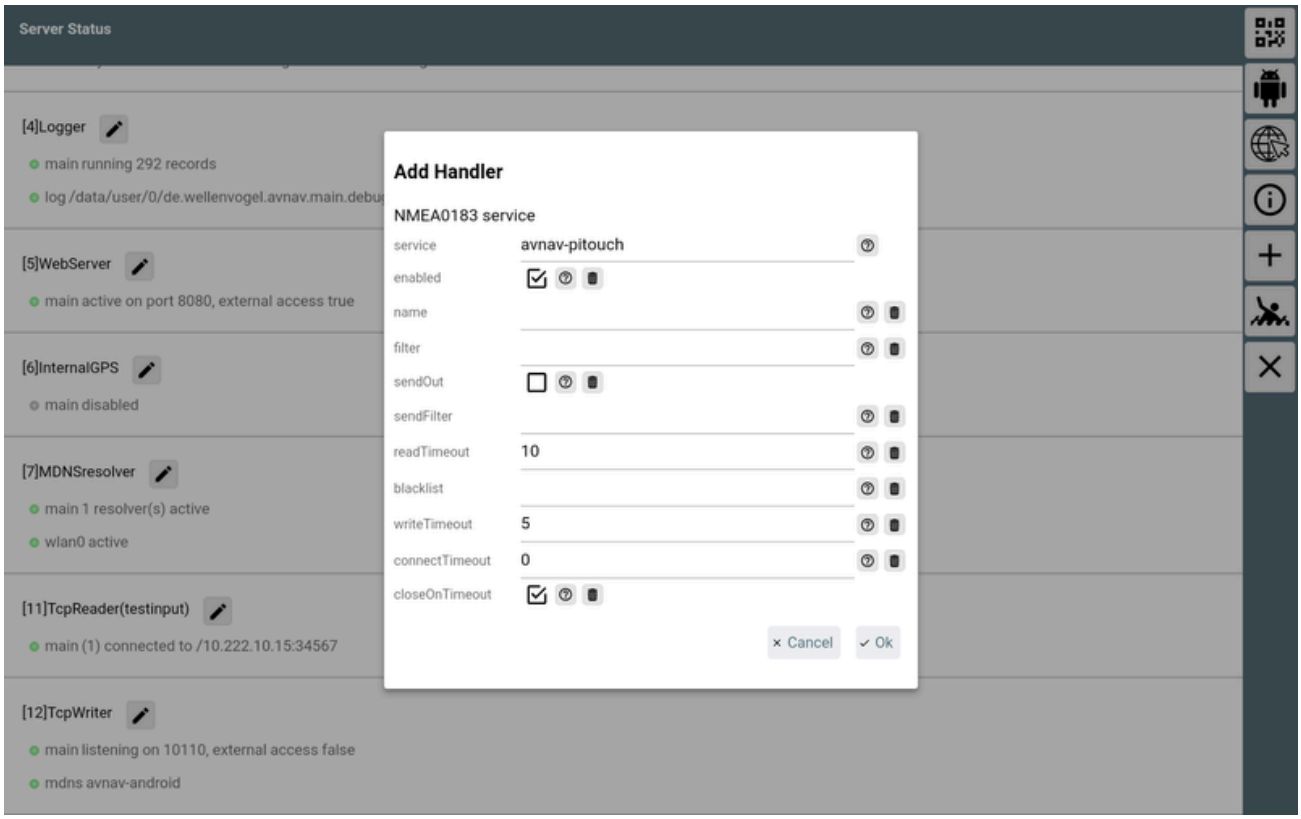
An UdpWriter will send out NMEA data via UDP to another app.

Parameter	Description	default
ipaddress	The IP address or the hostname for the server you want to connect to. This can also be a MDNS name like avnav.local .	---
port	The destination port	---
broadcast	send the data as broadcast (requires the ipaddress to be a valid broadcast address)	off

NMEA0183 Service

A NMEA0183 service mainly works like a TcpReader, without the need to define IP address and port of the origin. Instead you select the name of a MDNS service providing NMEA data from a list of detected services (type: _nmea-0183._tcp). If there is e.g. a SignalK server or an AvNav server (> 20210415) in your network - they will announce their services this way (if configured to do so).

Advantage of using a service: it will continue to work if you change your network topology.

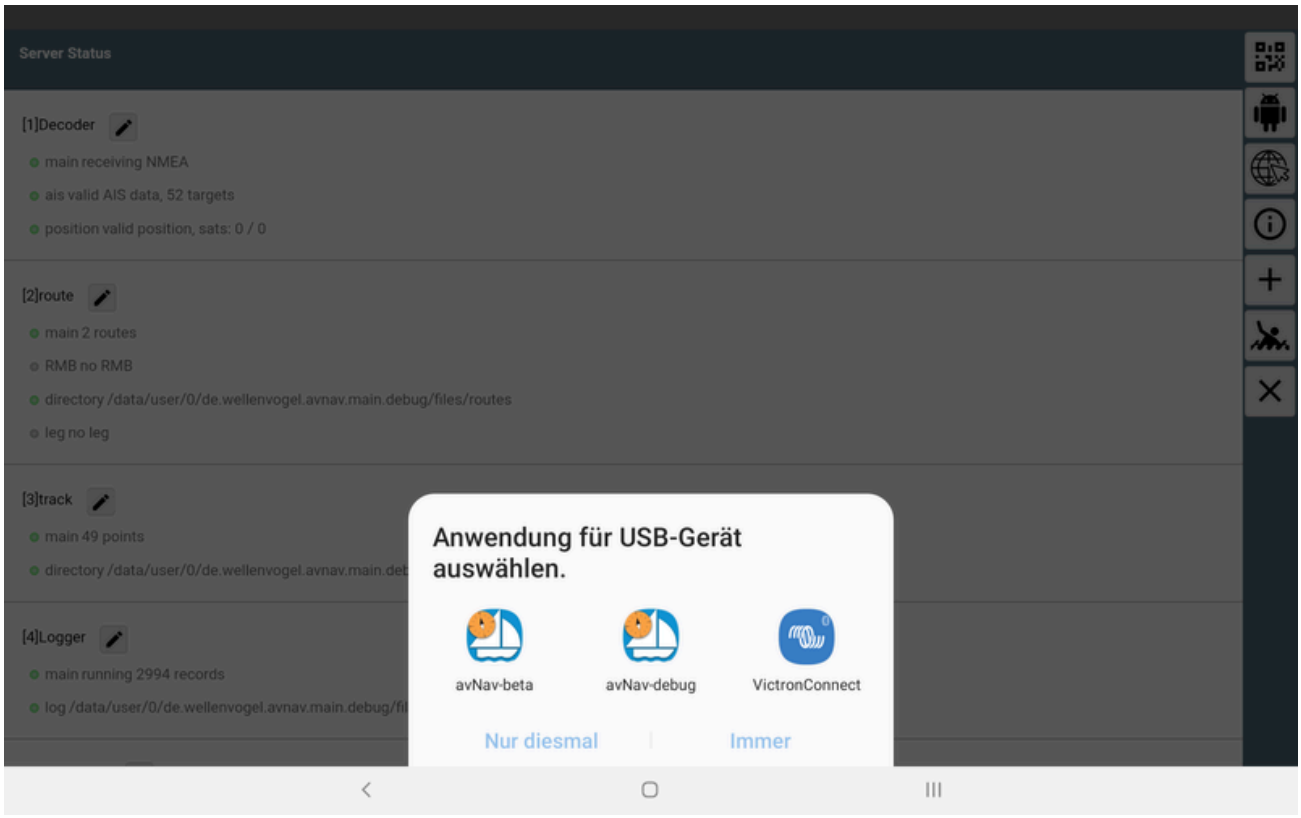


Parameter	Description	default
service	The name of the service (select from a list of found services)	---
sendOut	also send out NMEA data (otherwise read only)	off
readTimeout	Mark the connection as inactive if no NMEA data were received for this time (seconds).	10
writeTimeout	Write timeout for a NMEA record (in seconds). Close and reopen the connection, choose 0 to disable writeTimeout.	5
connectTimeout	Timeout for connecting (seconds, 0 - system-default).	0
closeOnTimeout	Close (and reopen) the connection if no NMEA data were received within	on

	readTimeout.

UsbConnection

AvNav will get activated when an USB device is connected. Typically it makes sense to start AvNav first and afterwards connect the USB device. This way you can allow permanent access to the USB device for AvNav. AvNav will immediately start up the configuration dialog for the newly connected device.



Parameter	Description	default
device	The USB device (more exact: the port it is connected to). Select from a list.	---
baud rate	The serial baud rate.	9600
flowControl	none|xon/xoff|rts/cts - flow control if supported by the adapter	none

sendOut	also send out NMEA data (otherwise read only)	off
readTimeout	Mark connection as inactive if no NMEA data were received for this time (seconds).	10

Bluetooth

Before you can use a bluetooth device you must pair it (outside of AvNav).

Parameter	Description	default
device	The bluetooth device. When creating the handler you can select from a list of all paired devices. Only after saving the settings AvNav will actually try to connect to the device.	---
sendOut	also send out NMEA data (otherwise read only)	off
readTimeout	Mark the connection as inactive if no NMEA data were received for this time (seconds).	10

Configuration and Adaptation

There are various ways to adapt AvNav to your personal needs:

- Adapt various aspects of the WebApp - Settings Page.
- Adapt value displays (widgets) on the navigation page and on the dashboard pages: adapting the layout.
- Adapt appearance via css. As the app itself is browser based, you can adapt the look and feel via css stylesheets. This is described at adapting via css
- Adapt icons (boat, AIS,...)
- Adapt keyboard mappings
- Adapt server configuration (avnav_server.xml) - not on android
- Control the server's wifi connections - Wifi Page - not on Android and OpenPlotter

Layout Adaptation

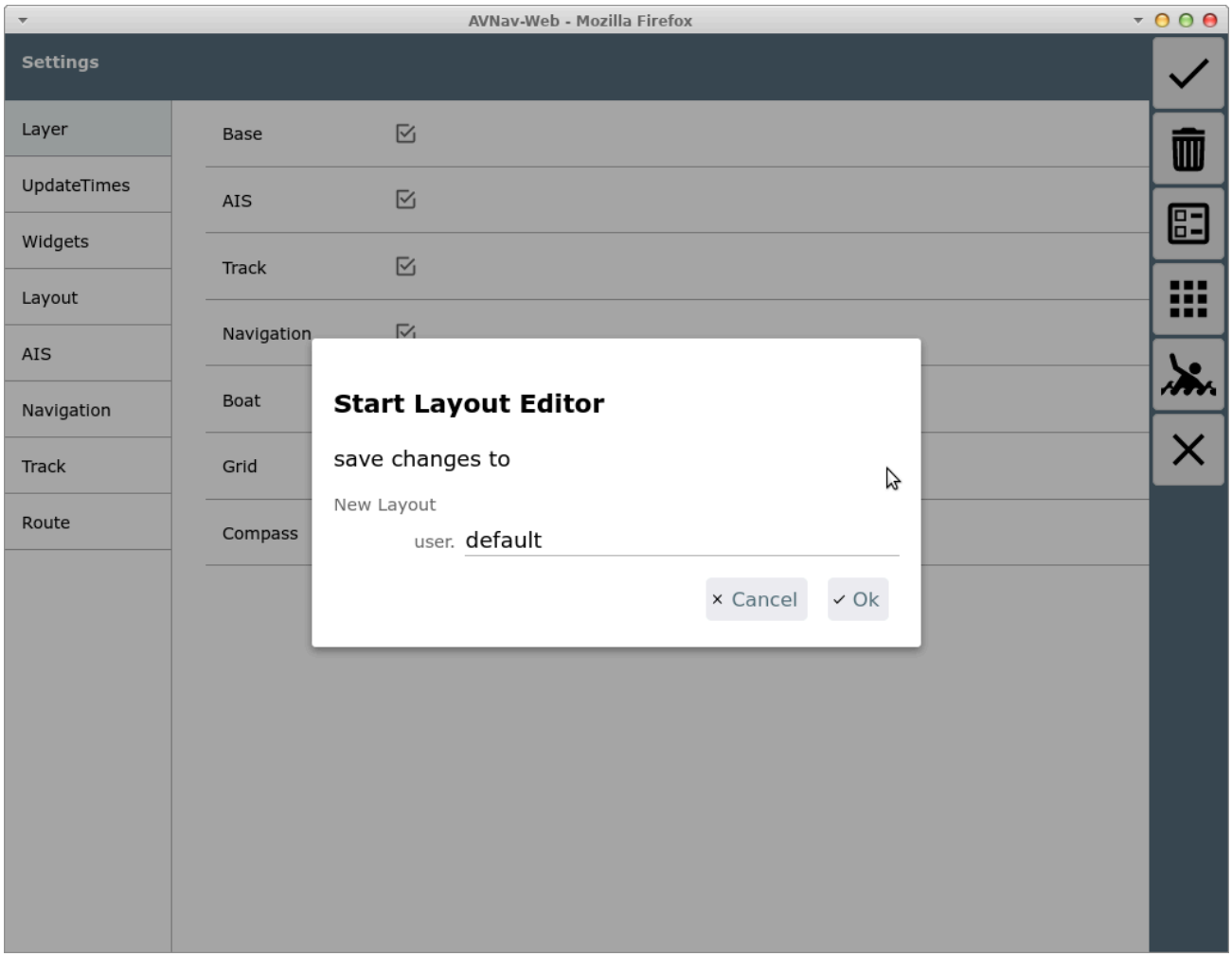
The displays (widgets) on navigation and dashboard pages are controlled by a configuration json file ("layout"). A couple of predefined layouts are included with AvNav. They all have the "system." name prefix. As a user you can define your own layouts and store them (they will have the name prefix "user."). Those user layouts will be stored at a folder "layout" below the BASEDIR (/home/pi/avnav/data on the raspberry). You can directly edit those files, download and upload them - or you can modify the layout using the built in layout editor.

The preferred way of changing a layout should be the layout editor as this is the least error prone way.

The active layout can be set via Settings /Layout.

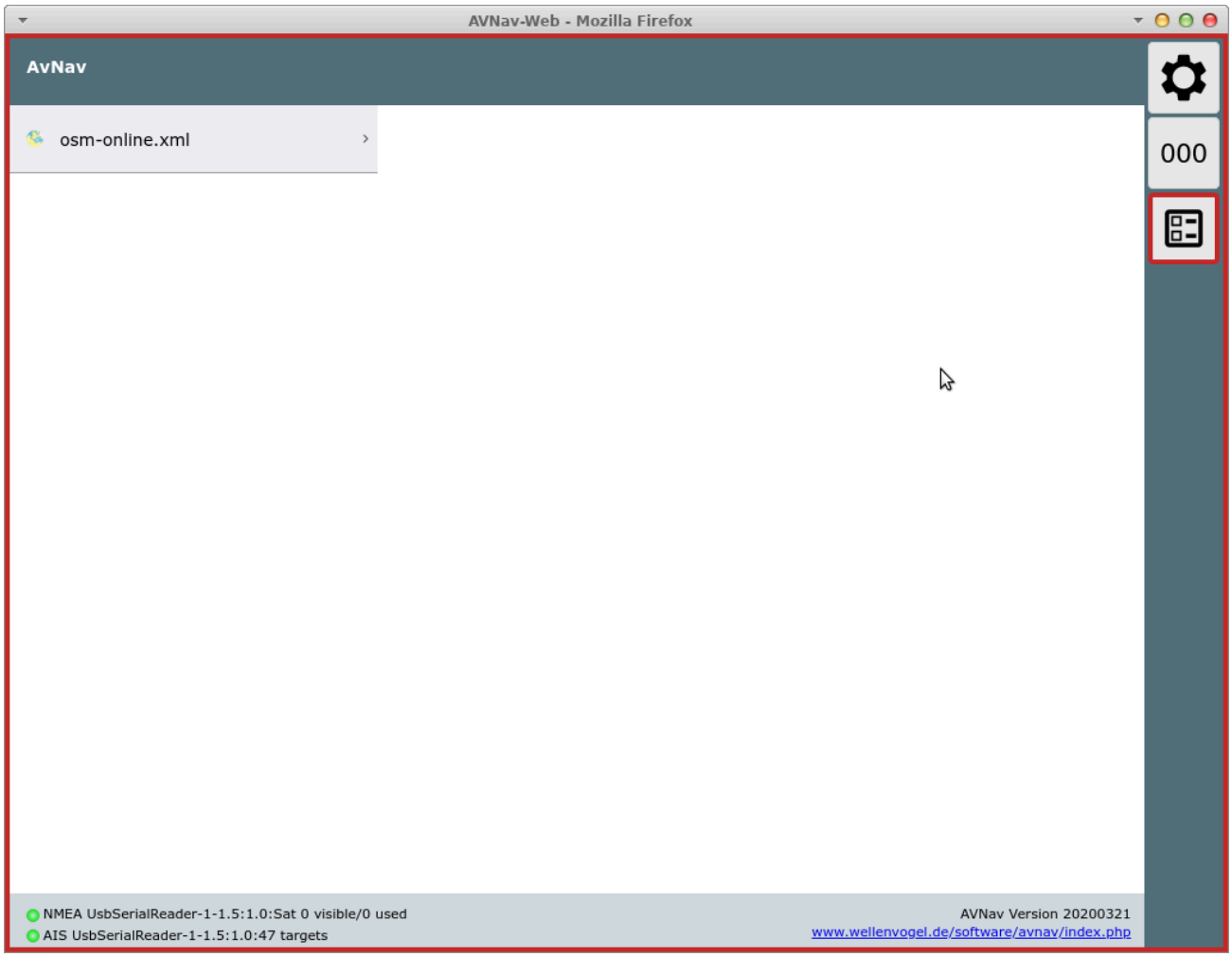
Layout Editor

You can invoke the layout editor via the Settings page , Layouts .

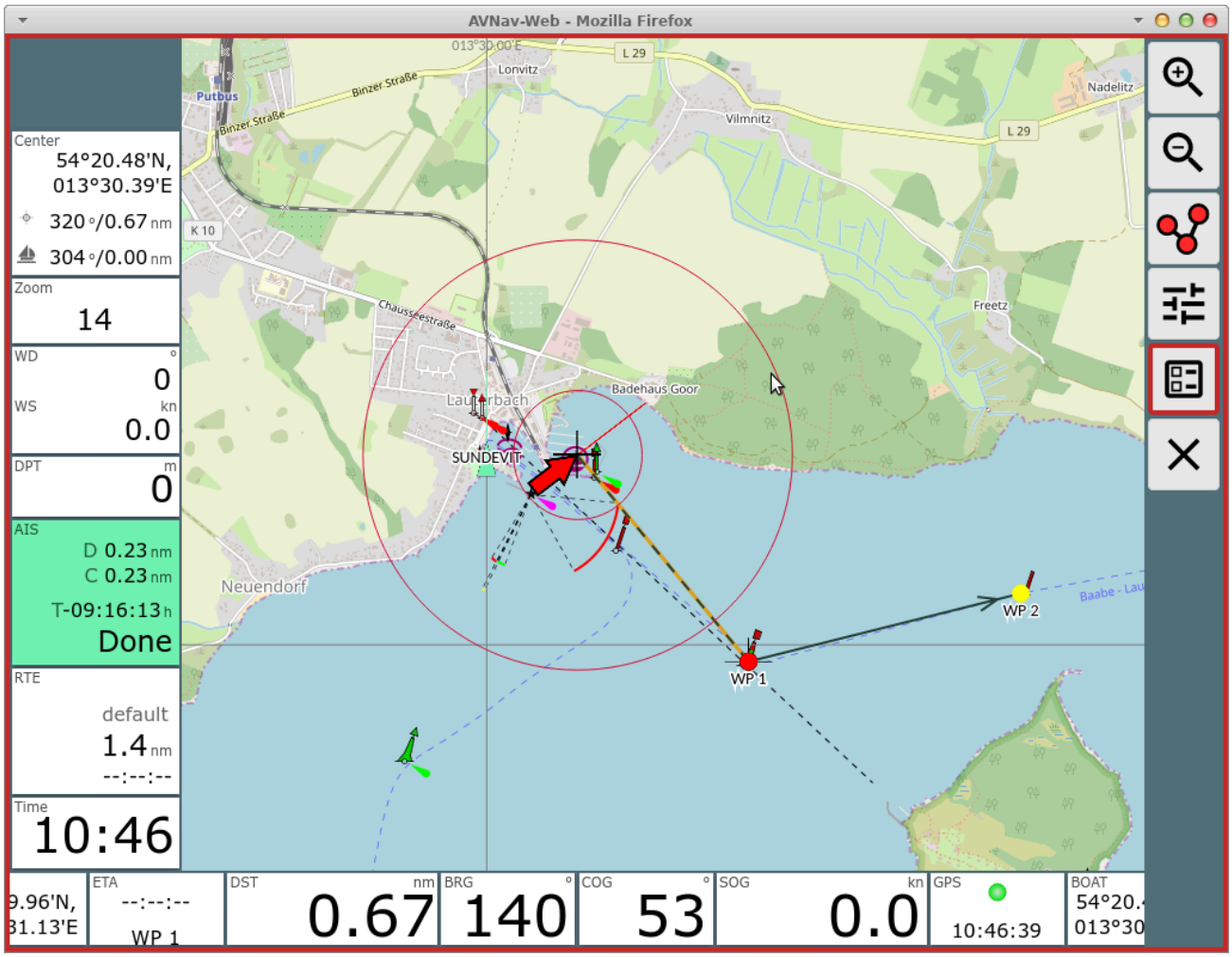


This dialog asks for a file name for the layout to be modified. If a system layout is active (that you cannot change) a new 'user' layout with the same name will be created and activated. If the layout already exists you can decide to overwrite it - or just choose a new name. You can still decide at the end of the editing process to discard your changes.

After starting the layout editor a red frame will be displayed - only pages where you can adapt the layout are visible.



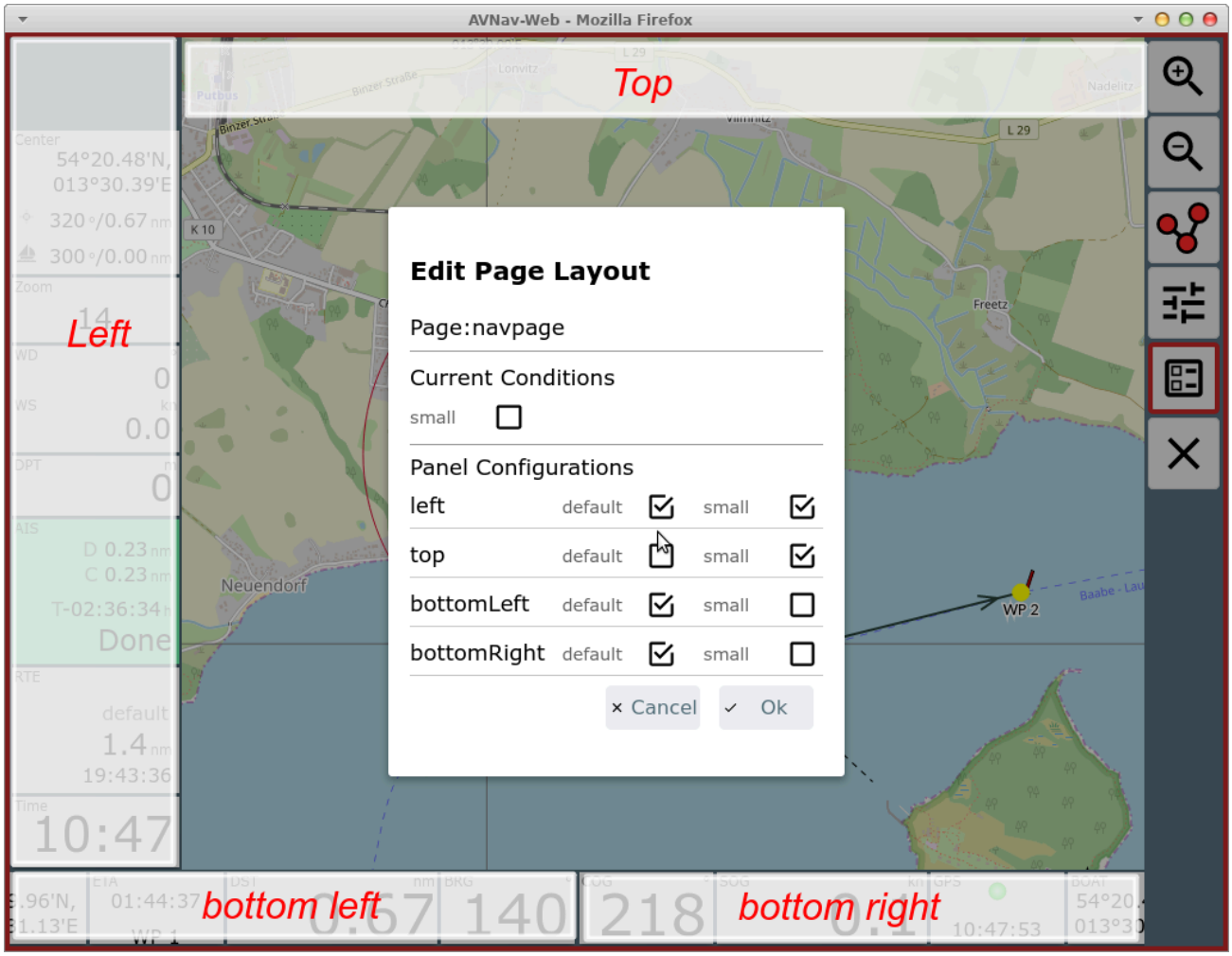
By selecting a chart you can open the navigation page as usual.



All widgets are located in so called "panels". Within a panel you can drag and drop the widgets as long as editor mode is active (but you cannot move them from one panel to another one this way).

On the navigation page you can assign different widgets to the panels depending on your display width. On a small display (value can be changed at Settings/small display) you can have different panels than on a large one.

Via you open a dialog to configure the panels on a page.

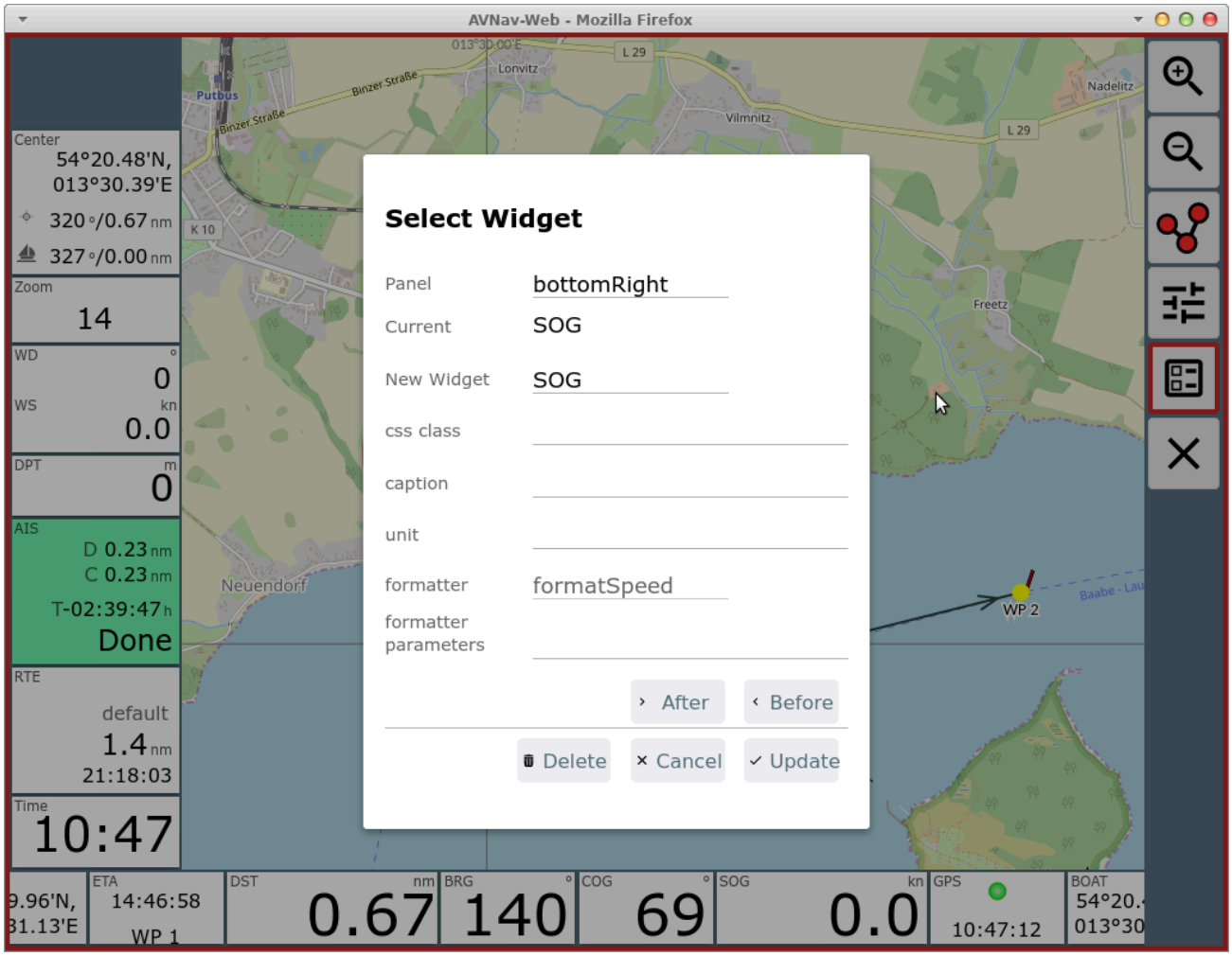


The available panels are marked red in the screenshot. You can select which panels should be visible - depending on the display mode (small/default). If the "small" checkbox is checked you define a different set up for the small mode. In the example above the top panel is not visible in normal mode, the left panel has a different content in normal and small mode, whereas the bottom panels have the same content both in normal and small mode. At "current conditions" you can select if you are editing the panels for normal or small mode.

Both the bottom panels may have 2 rows of widgets (Setting: "2 widget rows"). Widgets that do not fit any more will be hidden completely.

Widget Dialog

After completing the panel dialog you can move the widgets within the panels or click on a widget or a panel's free space to open the widget dialog.



Depending on the kind of widget the dialog will contain different settings.

If you would like to move the widget to another panel you can select the target panel in the top entry of the dialog.

"Update" will store your changes.

By clicking "New Widget" you can select a new widget from a list of available widgets.

Each widget requires information about the data to display. All displayable data are internally available in a store and each item is addressed by a key.

Depending on the type of widget, the key(s) it uses to fetch the data from the store is either fixed (i.e. built in) or has to be chosen in the dialog. Caption, unit and the css class can be set for most widgets. Assigning a css class is useful if you want to adapt the appearance via css.

Formatter

Most of the widgets need a formatter to convert data from internal format into the display format. In most cases the formatter used by a widget is fixed. Some, however, allow parameters (comma separated list) to be assigned in this dialog - e.g. you can change formatSpeed to convert to m/s instead of kn).

The list of formatters (and their parameters):

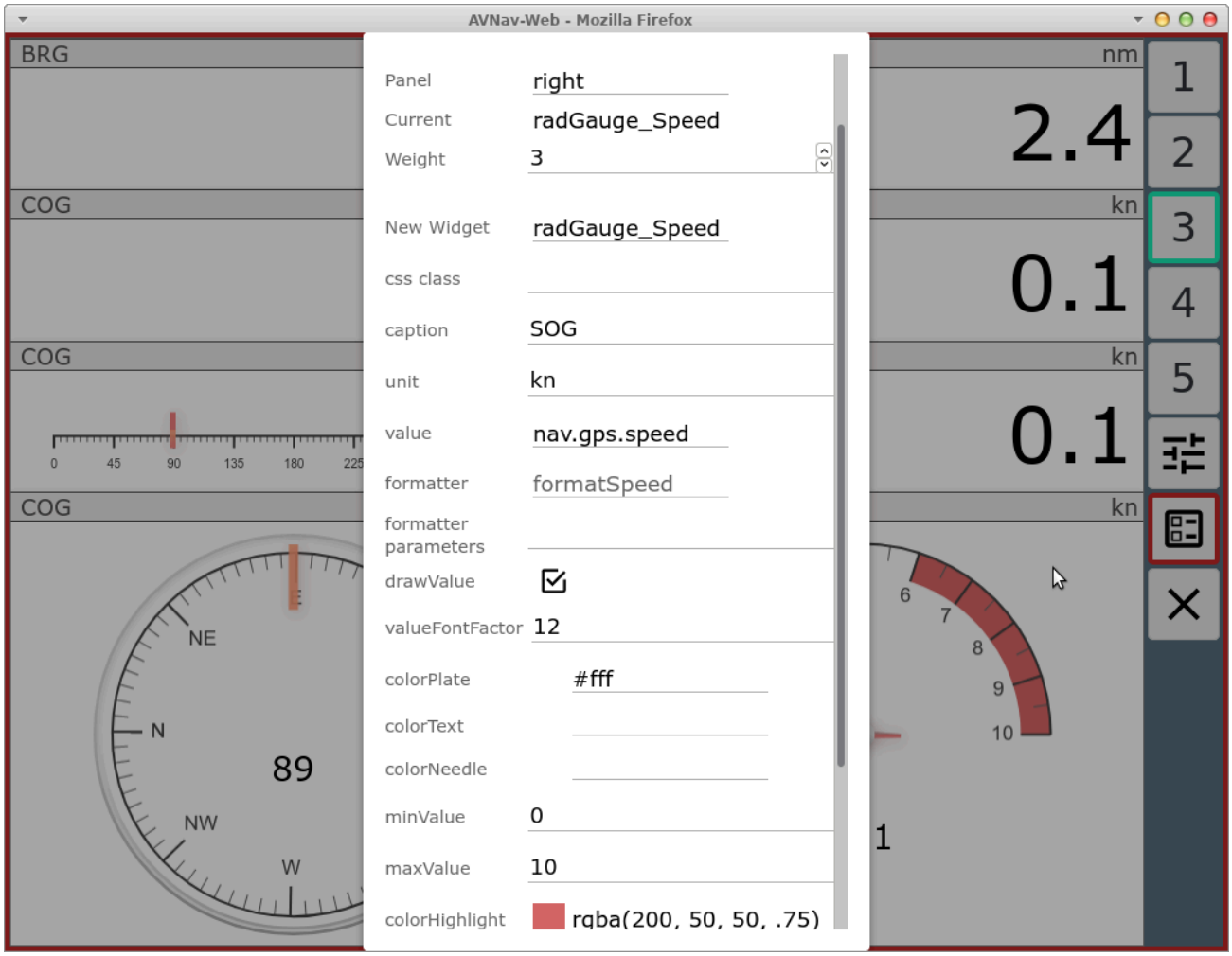
Name	Description	Parameter
formatDecimal	format a decimal value	fix, fract, addSpace e.g.: 3,1
formatDistance	distance in nm	one parameter (unit): nm - distance in nm m- distance in m instead of nm km - distance in km instead of nm
formatSpeed	speed in kn	one parameter (unit): ms - m/s instead of kn kmh - km/h instead of kn
formatDirection	formatting degrees	one parameter: inputRadian - input in rad instead of degrees
formatTime	Time value (internal value must be Date) (hh:mm:ss)	

formatClock	Time value (internal value must be Date) (hh:mm)	
formatDateTime	Date and Time (internal value must be Date)	
formatDate	Date (internal value must be Date)	
formatString	pass through	
formatTemperature	format temperature (since 20210106), input in kelvin	one Parameter (unit): celsius, kelvin
formatPressure	format a presssure (since 20210106), input in pa	one Parameter (unit): pa, hpa, bar

Plugins or custom extensions can potentially add more formatters.

If you do not want to change the existing widget but add a new one - just select "Before" or "After" to insert it.

If the widget supports further parameters they will be displayed in the dialog.



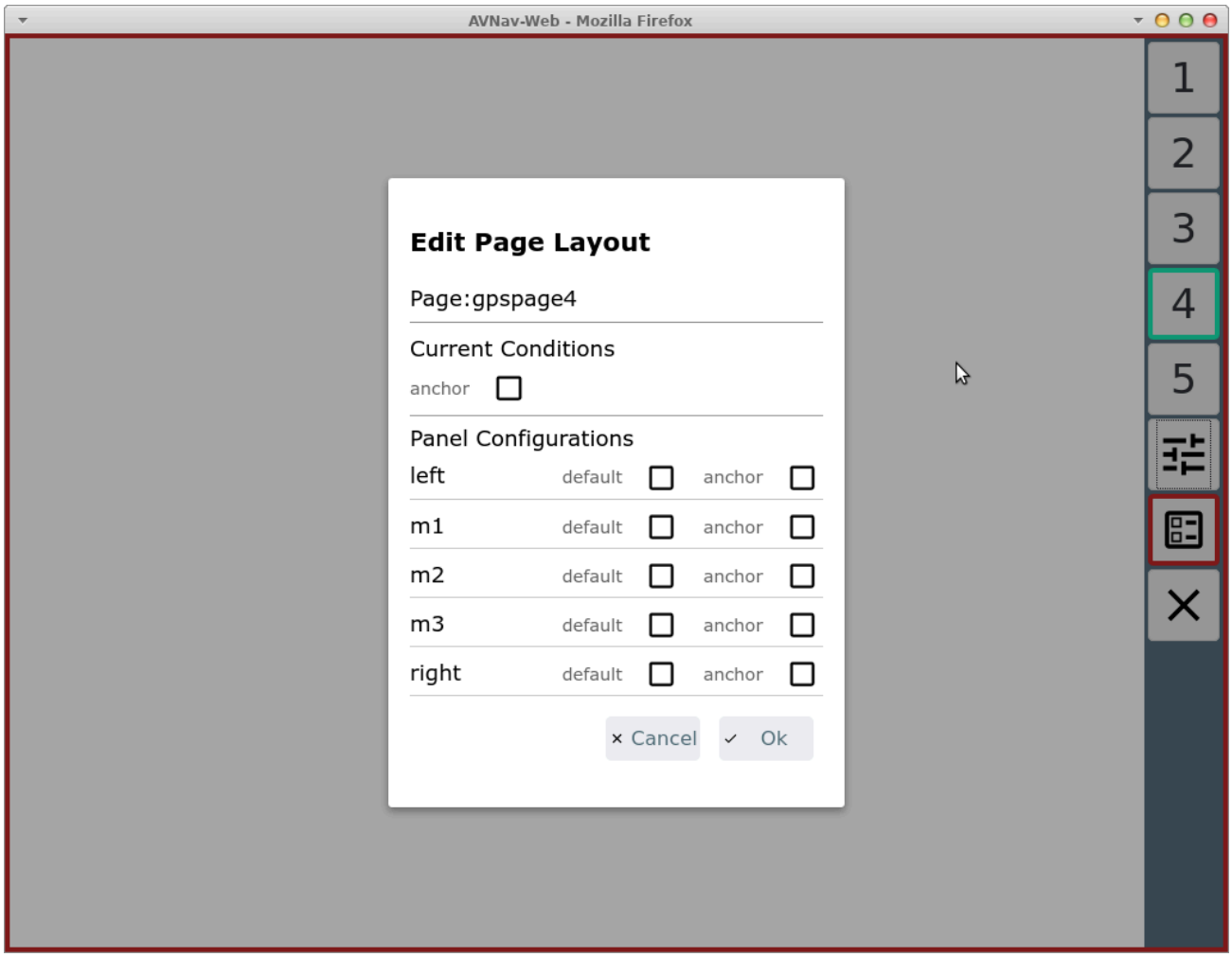
In the example a configuration for a graphics display (internally canvas-gauges is used).

From the navigation page you can switch to the dashboard pages via the main page.

Dashboard Pages

You can configure up to 5 dashboard pages. Each of them can hold up to 5 panels.

On an empty dashboard page you have to start with the panel configuration - .

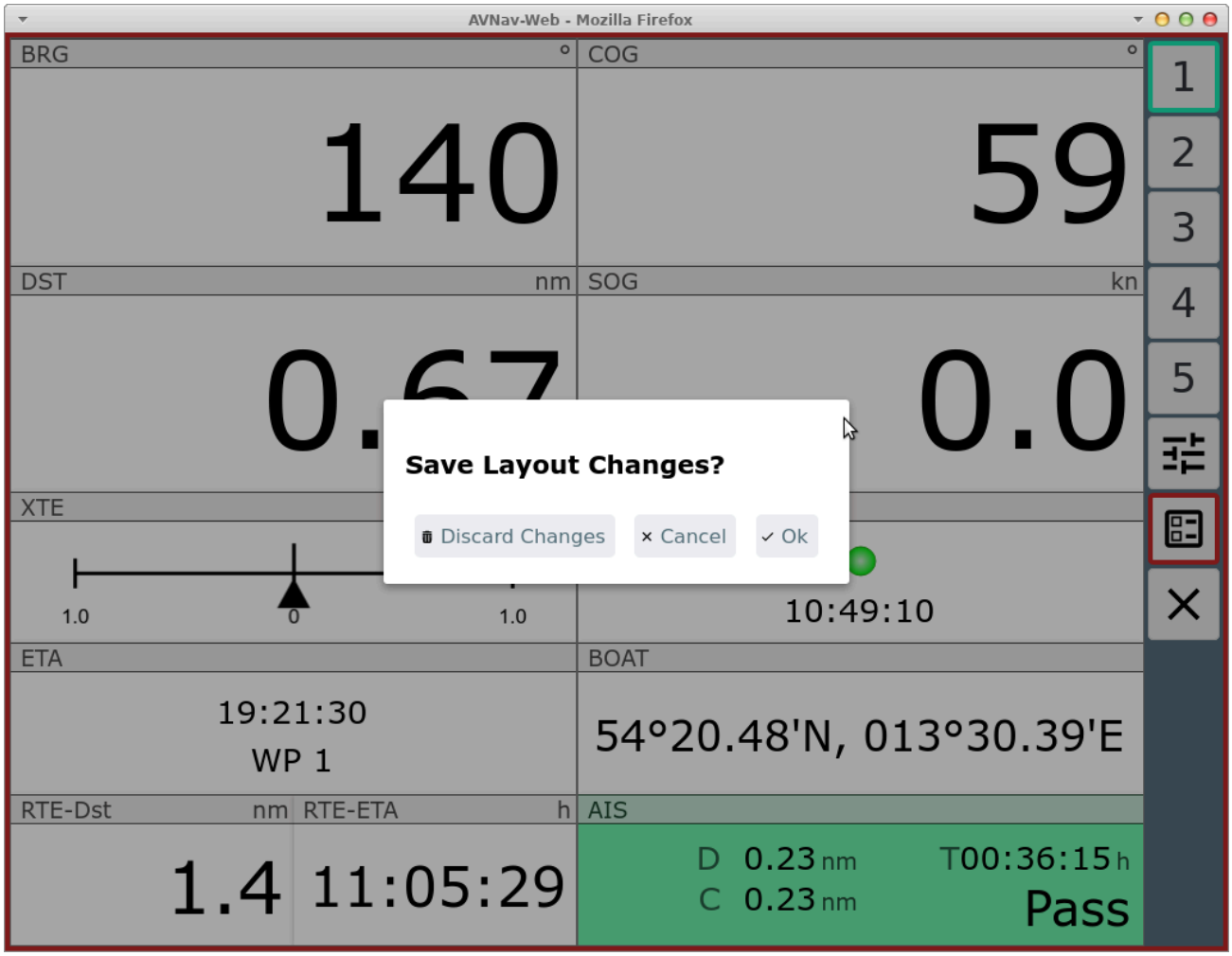


You can define a different layout for anchor watch state (this feature is used in dashboard page 1 of default layout).

Dashboard pages without panels will not appear in the live display.

Principally you can also modify the the route editing display (from the navigation page) however the currently edited route and the list of waypoints must remain visible there at all times.

When you are done with the layout editing you still need to save your layout using the button (red border) on each page.



At this point you can still decide to discard all changes.

The saved layout will immediately become active in your current browser (and will be stored at the server). Other browsers will only use the new layout after reloading the app or changing their active layout.

Layout Download/Upload

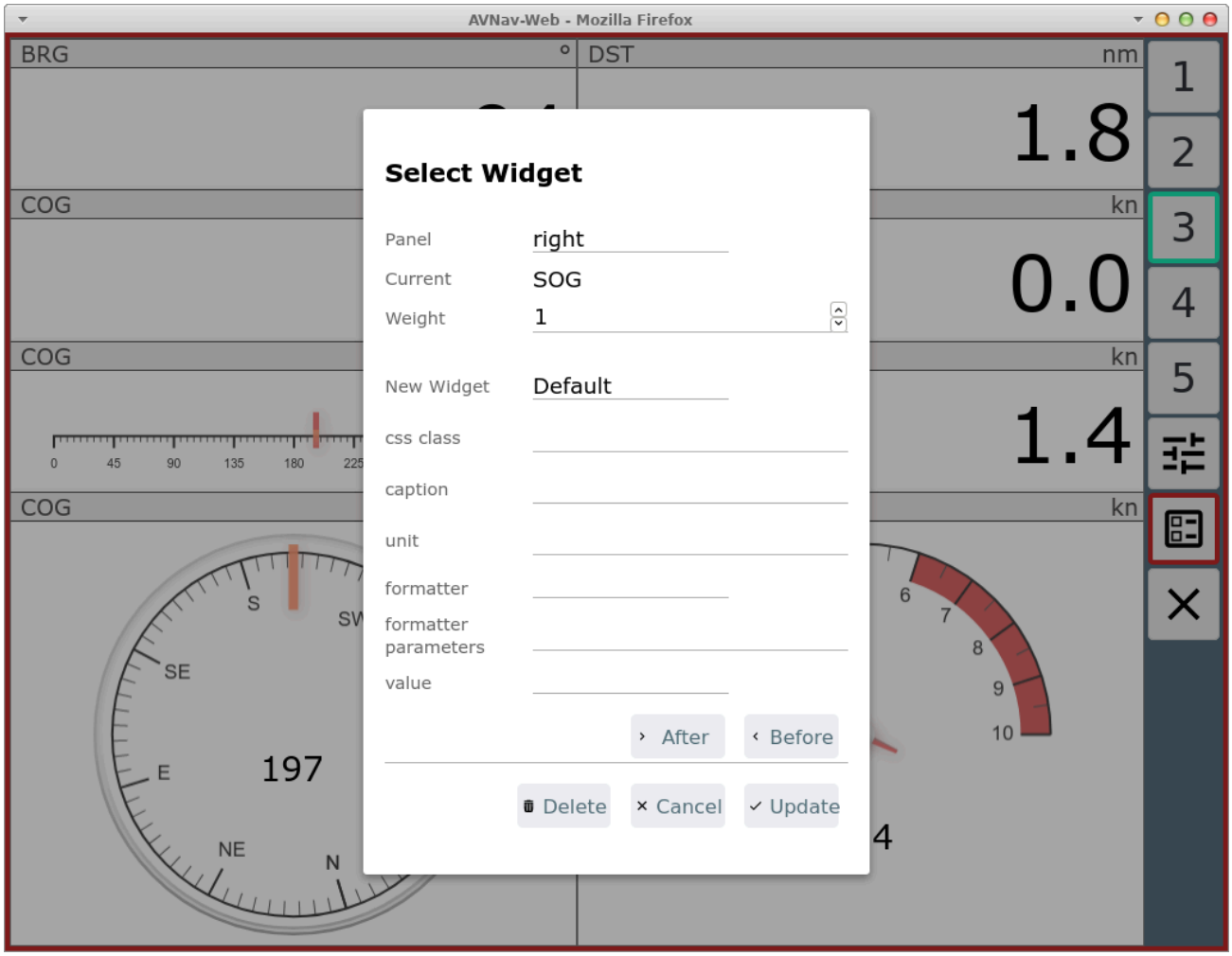
Via the Files/Download page , subpage you can download/upload/edit and delete the layout files. The currently active layout and system layouts can only be downloaded.

If you delete a layout currently loaded by another browser, this browser will write the layout back to the server when it reloads the app - so the layout will reappear.

Special Widgets

In addition to a long list of predefined widgets (like SOG, COG, BRG, AisTargetWidget, ... where you can only change the caption, unit, formatterParameters and css class) there are a couple of special widgets.

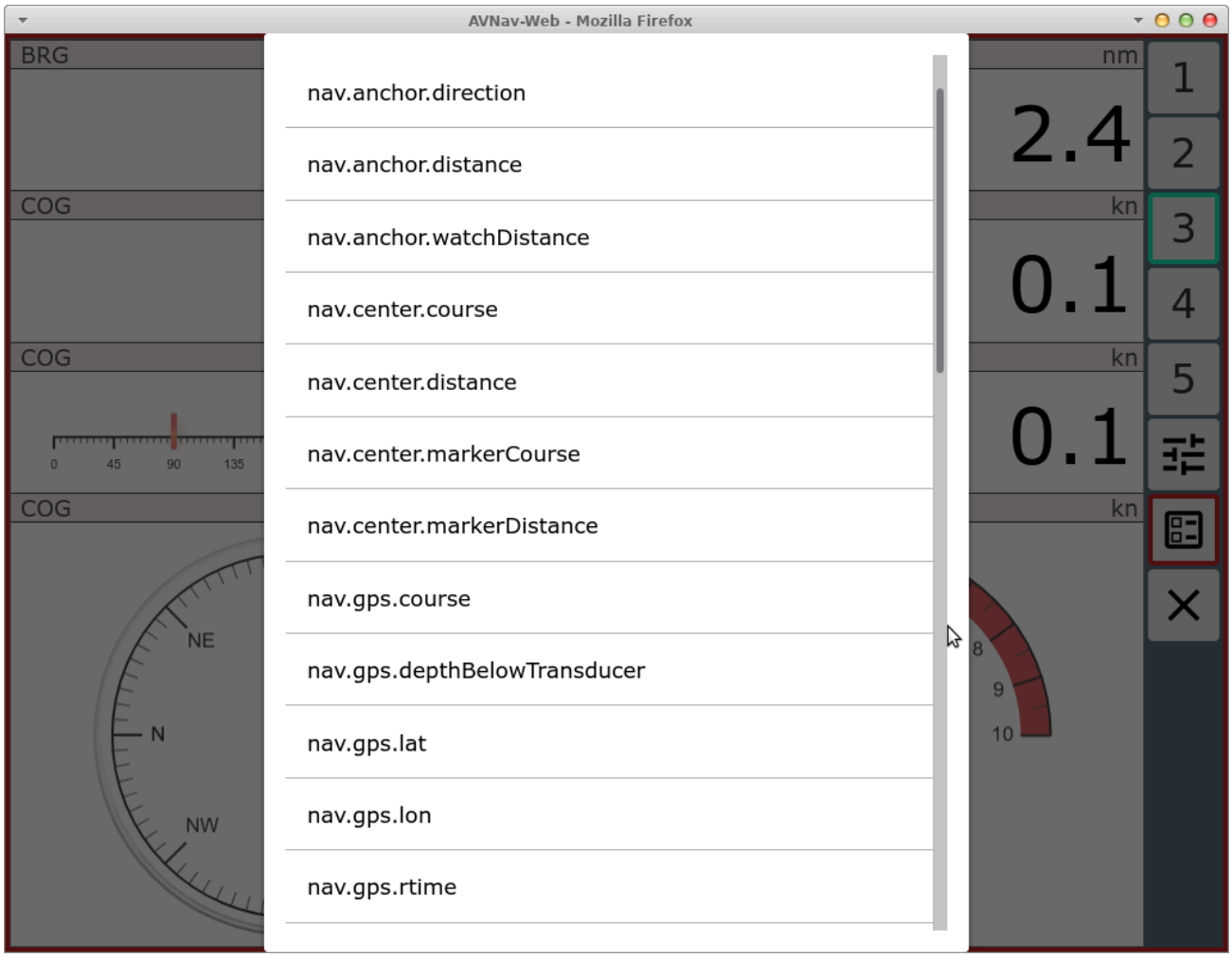
Default



This widget internally serves as basis for most simple display widgets. But you can also use in the layout editor it to display custom values.

In any case you need to select a value (the key for the data store) and a formatter.

When clicking "value" you will be prompted to choose from a list of available data.



First of all those are the values predefined internally in AvNav. If you have enabled the signalk integration you will also see all received values from signalk (below vessels/self) with a prefix nav.gps.signalk.

You have to take care to select a formatter that fits the value you selected.

Gauges

As already mentioned AvNav integrates canvas-gauges. AvNav preconfigures a couple of widgets based on this library. Their parameters can be set in the layout editor.

There are linear gauges (linGauge...) or radial gauges (radGauge...).
Via the layout editor only a couple of the possible parameters can be set directly.

If you want more control or utilize different features of the canvas gauges library you should define own widgets as described in adapting with user

specific java script code .

SignalK

The signalk plugin also provides a couple of widgets - they especially contain formatters for values that are otherwise not included in AvNav like signalKCelsius to display temperatures in °C.

They only make sense if you have appropriate signalk data, of course.

Adapting the Look and Feel of a Widget

If you would like to adapt the look and feel of a widget you should add a css class to it. Afterwards you can assign rules to it in your user css.

Own Widgets

With user defined Java Script Code you can easily create your own widgets - both with simple HTML and with canvas based graphics.

User Symbols

Since version 20201030.

A couple of the symbols used in AvNav can be adapted to your preferences. You can resize the existing symbols, set a couple of properties or replace them by own symbols.

If you are going to use your own symbols you need to upload them as .png files into the images directory - see description for user files.

The symbols you are going to change need to be described in a json file (images.json) in the user directory.

An example of this file may look like:

```
{
  "boatImage": {
    "anchor": [20,0],
    "size":[40,71]
  },
  "markerImage":{
    "src": "/user/images/Marker.png",
    "anchor": [15,15],
    "size":[30,30]
  },
  "aisNormalImage-Sail":{
    "src": "/user/images/Sail-Boat-40.png",
    "anchor": [15,15],
    "size":[30,30],
```

```
    "courseVectorColor": "#ff00ff",
    "rotate":false
  },
  "aisNormalImage-Military":{
    "anchor": [32,0],
    "size":[64,120]
  }
}
```

Since 20230614 there is a base configuration that is used by the system. In the user images.json you can overwrite entries from the base config.

For each symbol you want to replace there must be an entry with the correct name.

Common Images

boatImage	boat symbol on navigation page
boatImageHdg (20220421)	boat symbol if hdt or hdm are used
boatImageSteady (20220421)	boat symbol if zero SOG detect is active and not boat movement
markerImage	symbol for active waypoint
anchorImage	symbol for the anchor if anchor watch is activated
measureImage	symbol for the starting point of the current measure

AIS Images

For AIS icons you have a couple of parameters that you can consider. Each AIS target will be in a particular state - shown by a different color.

State	Meaning
Normal	AIS target
Warning	the closest AIS target with a CPA lower then the threshold
Tracking	the AIS target you have selected at the AIS info page
Nearest	the closest AIS target

Beside the state AIS icons can be defined based on the kind (Normal/Aton) and a couple of parameters (ship type, navigation status, aid type).

Basically you could define an own icon for each combination of parameters and state. But from 20230614 you can leave the handling of the state (color) to AvNav. To make this possible you need to create the icons in a way that there is one color that can be replaced by the state color. You need to define this color in the parameter "replaceColor" (see the defaults for examples).

Basically you can define the following icons (examples):

Key	Meaning
aisImage	default Ais icon
aisImage-status1	Icon for AIS targets with navigationl status 1(At Anchor), see the source code for other values.
aisImage-typeFishing	Icon for AIS targets with ship type 30(Fishing), see the code for values

aisatonImage	default AIS icon for atons
aisatonImage-type9	AIS icon for atons of aid_type 9 (Beacon, Cardinal N), see the code for values

If you do not want to use "replaceColor" you can define different icons depending on the state:

aisWarningImage, aisNormalImage, aisWarningImage-status1, ...

Icon Parameters

For each of the symbols in the file you can specify the following parameters:

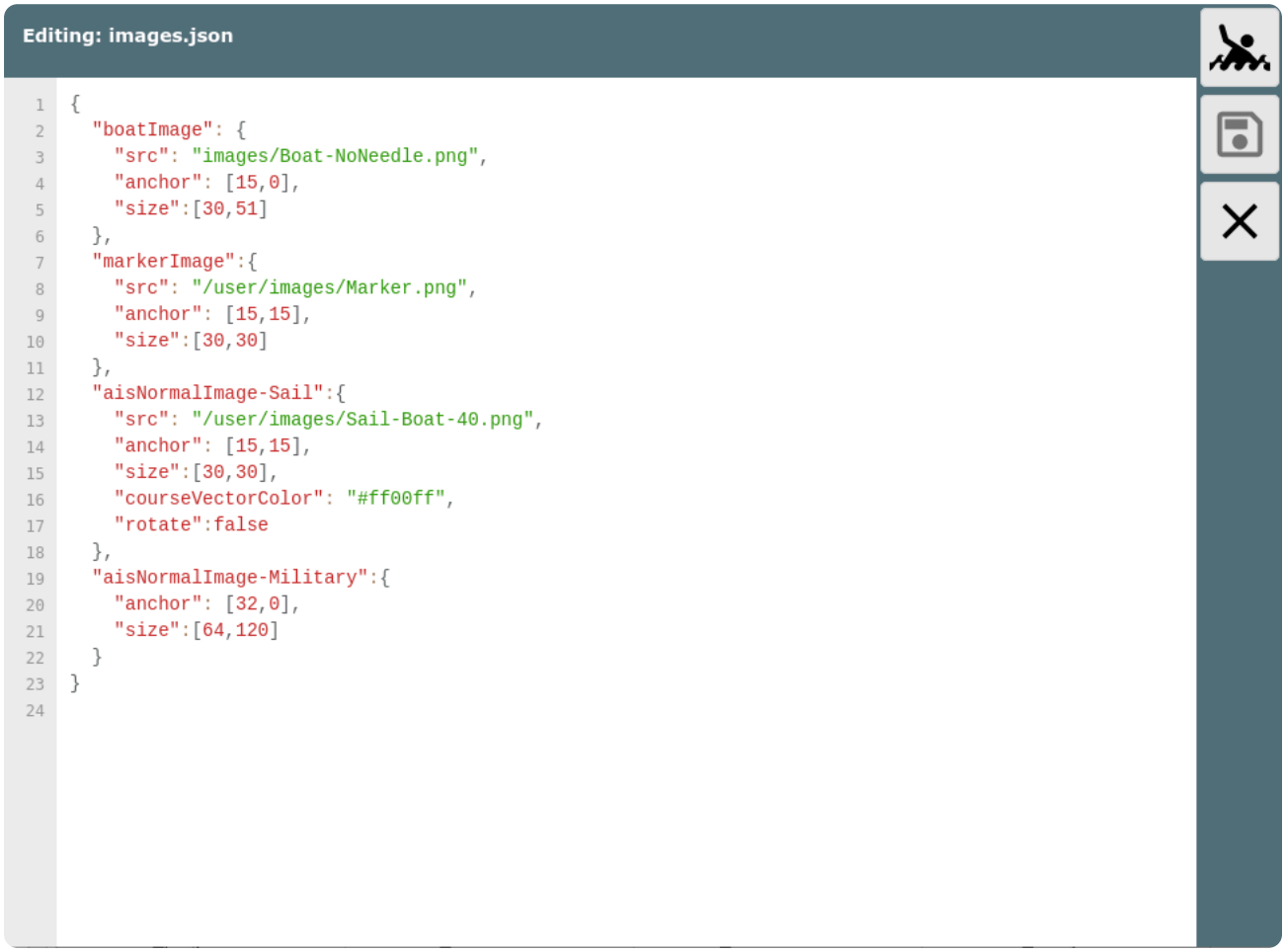
src	The source url for the image. Normally /user/images/XYZ.png for a file that has been uploaded to the images folder. If you do not specify this parameter the system will use the built in symbol. You still can e.g. alter the size. Your image files should have a reasonable size (e.g. 2x the specified size) - but not too large as otherwise the performance could suffer. If you have vector graphics (like svg) you can use a program like inkscape to create png's from them.
size	[width,height] - must be specified as an array - see the example. This defines the size the image will be

	scaled to. If you did not specify the src parameter, size applies to the built in image.	
anchor	[x,y] - the point (related to the image's width, height coordinates) to be anchored at the item's position on the map.	
rotate	true or false - if you set this to false the symbol will not be rotated according to its course.	not for markerImage
courseVector	true or false - if you set this to false, no course vector for this symbol is displayed.	not for markerImage
courseVectorColor	the color for the course vector. This allows you to adjust the course vector color to the image colors.	not for markerImage
replaceColor (since 20230614)	The color to be replaced by the state color.	only ais...Image
textOffset (since 20230614)	An array [x,y] for the base text offset. The system will add some additional offset based on the course (mainly y). The x value must be related to the size of your icon.	only ais...lImage

Parameters not being provided will be replaced by defaults. You can also only provide some parameters without an src attribute - this way you can e.g.

change the size of AIS symbols based on there type. Normally you also have to adapt the anchor parameter if you change the size.

When you edit the images.json you have to be careful to create valid json. When using the AvNav edit function at the Files/Download page (sub page) syntax will be automatically checked when saving the file.



After you saved the file, you need to reload AvNav.

Keyboard Support

AvNav supports keyboard shortcuts for important functions.
You can freely assign keys to functions inside AvNav.

Concept

The assignment is defined in 3 levels (columns):

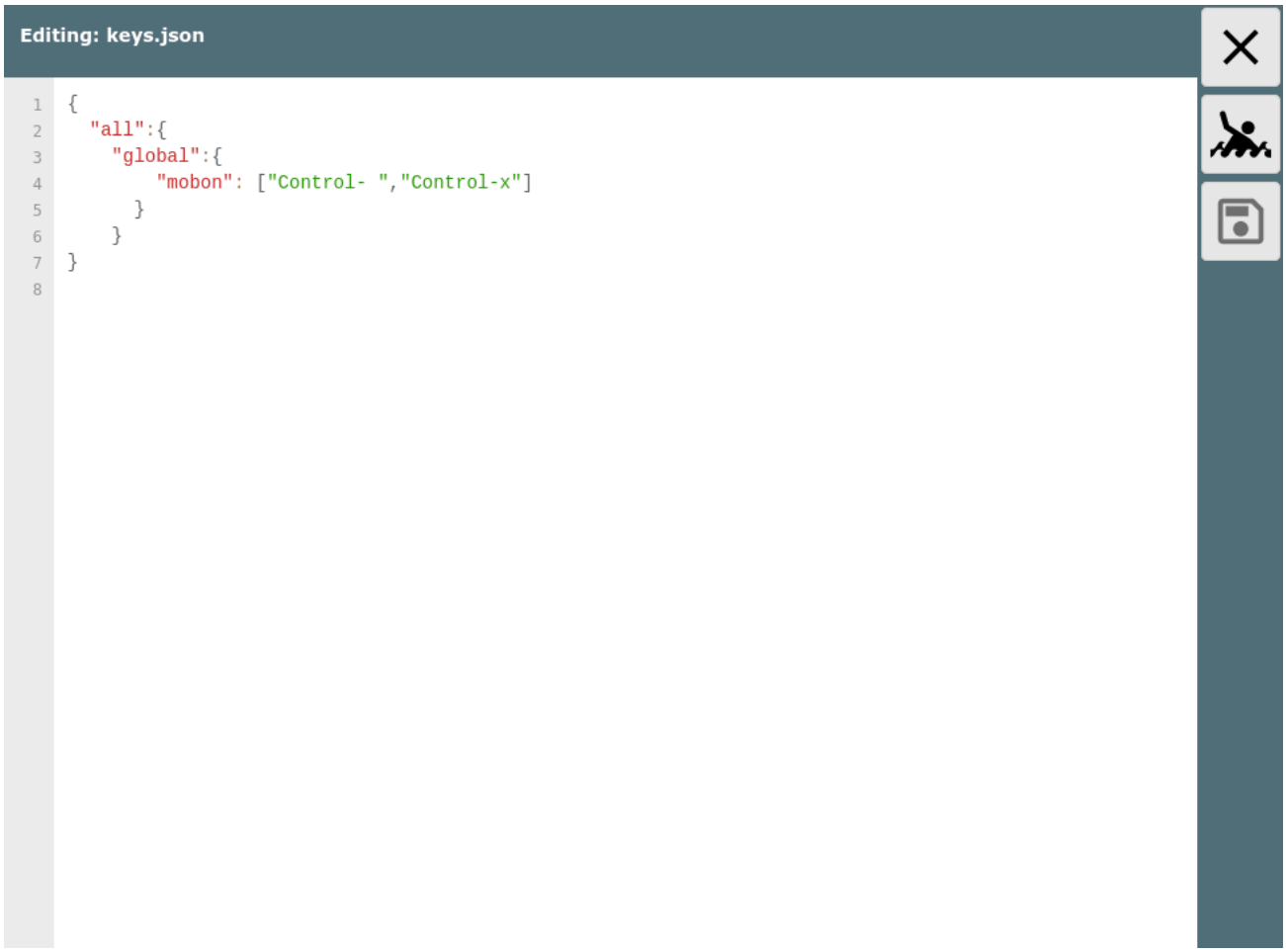
1. Page
   This is the page shown in AvNav (see user documentation). By using the keyword "all" you can assign the function to all pages.
2. Group
   This just groups the functionality.
3. Function
   The particular function to be triggered (like the click on a button).

You can assign one or multiple keys to each of those triples (page, group, function). The more specific configuration will take effect (e.g. an assigment for navpage will win against an assignment for all).

Configuration

The assignment is handled via a file keys.json in the user directory. This file can be edited directly there. Additionally there is a builtin keys.json file holding the defaults.

The keys.json in the user directory overwrites the defaults.



In this example I created a keyboard assignment for "man over board" to the keys Ctrl-Space and Ctrl-x.

If you just assign a single key the square brackets are not necessary. After you saved your changes you must reload the AvNav page.

Pages Groups and Functions

The list of pages, groups and functions below will reflect the current state when creating this documentation. Step by step new ones will be added.

For the names to be used for the keys refer to the documentation. If the Control key is pressed additionally the string "Control-" will be prepended.

The function names within the group "button" are the names you find beside the buttons in the user documentation.

Clicking a widget can be achieved by using the group "widgets" and entering the name of the widget as the function (the name as displayed in the layout editor). A SOG widget could be triggered with

```
"all":{
   "widgets": {
      "SOG": "s"
   }
}
```

using the key "s" on all pages.

The dialog buttons can be assigned using the group "dialogButton" and the name of the button as function. To determine the name you can use the developer tools of your browser (e.g. "inspect element"). You should be careful to only assign special keys to dialog buttons - otherwise you might not be able anymore to make normal inputs.

In the table below I listed the groups and functions that either have a default key assignment - or are neither buttons nor widgets. The texts in brackets are just hints about the functionality.

Assignments

Seite	Group	Function	Default Keys
all	button	Cancel	"Escape"
	map	zoomIn	["+","PageUp"]
		zoomOut	["-","PageDown"]
		up	"ArrowUp"

		down	"ArrowDown"
		left	"ArrowLeft"
		right	"ArrowRight"
		lockGps (keep map center at boat position)	"l"
		unlockGps	"u"
		toggleGps	["t","Control-a"]
		toggleCourseUp	"b"
		centerToGps (one time center map to boat)	
	alarm	stop	"a"
	global	mobon	["Control- "]
		moboff	
		mobtoggle	
		anchoron (start anchor watch at current position, since 20220421)	"i"
		anchoroff	"Control-i"

		(20220421)	
	addon	0 (first addon)	"Control-0"
		1	"Control-1"
		2	"Control-2"
		3	"Control-3"
		4	"Control-4"
		5	"Control-5"
		6	"Control-6"
		7	"Control-7"
gpspage (dashboard)	button	Cancel	["d","Escape"]
navpage (navigation page)	widget	AisTarget	"a" (goes to the Ais Info page)
		COG	"d" (switches to the Dashboard, with the key d you can so toggle between navigation page and dashboard page)
	button	LockMarker (start navigation to current map center)	"g"

		StopNav	"s"
		ShowRoutePanel (open the Route Editor)	["Control-r","r"]
	map	centerToGps (one time map center to boat position)	"c"
	page	centerToTarget (move the chart to have the current target waypoint at center)	"w"
		navNext (navigate to the next point from the route)	["n","Control-n"]
		toggleNav (navigation on/off)	["Control-g"]
mainpage (Hauptseite)	page	selectChart (goto the navigation page using the selected chart)	"Enter"
		nextChart	["Tab","ArrowDown"]

		previousChart	"ArrowUp"
	button	ShowSettings	"Control-+"
		ShowStatus	"Control-s"
		ShowGps	"d"
		Night	["c","Control-c","Control-g"]
infopage (license)			
addonpage (user apps)			
addresspage (QR codes)			
statuspage (server status)			
wpapage (wifi control)			
routepage (route list)			
downloadpage (files/download)			
settingspage			
editroutepage			

(route editor)
addonconfigpage (configuration of user apps)
viewpage

AvNav Server Configuration

== not for Android ==

Introduction

On start up the AvNav server reads its configuration from an xml file - avnav_server.xml.
This file normally is located at /home/pi/avnav/data on the raspberry, otherwise at $HOME/avnav - see installation.

If this file still does not exist on start up it will be created from a template - fitting either the raspberry set up or other systems.

This template will be /etc/avnav_server.xml (from 20230426) on a raspberry pi (with the package avnav-raspi installed). If this one does not exist, AvNav falls back to a template installed with the package.
If you start AvNav from the command line using "avnav" you can define the template to be used with the -t parameter.
On package updates of AvNav this file normally will not be changed. Potentially it could be necessary to add some entries to enable new functions. In this case you will find some hints in the release notes.

With every successful start AvNav (from version 20200325) creates a copy of this file with the extension .ok. If during the next start up parsing of the config file fails, AvNav will fall back to this .ok file. With this fallback handling it will be ensured that AvNav can start up, even if a change of the original config file (what AvNav is doing in some situations) leaves the file in a defect state. If you don't want this fallback (for testing) add the -e command line switch.

If AvNav completely fails to start, you can try to remove the avnav_server.xml completely. This way AvNav will start up with a clean template.

Starting from 20210322 **it should not be necessary anymore to modify the config file "by hand"**. Instead use AvNav itself (Server/Status page) to edit most parts of the configuration. This also eliminates the need of a manual restart when changing parameters.
In the following descriptions the column "online" will give you a hint whether this value can be changed directly at the server/status page.

If you need to change sections not editable there, you still can use the avnav-update-plugin to edit it directly from within your browser.
So the following text simply serves as reference.

If you change the avnav_server.xml you need to restart AvNav. If it is running as a system service you can do this with the command

```
sudo systemctl restart avnav
```

I recommend to do a start up of AvNav from the command line whenever you changed the config file to watch for serious errors. The commands would be:

```
sudo systemctl stop avnav
avnav -e
^C
sudo systemctl start avnav
```

Option -e, since version 20200325. It prevents the fallback to avnav_server.xml.ok (what would hide your errors). ^C interrupts the running AvNav.

Content

Within avnav_server.xml there are entries for the various parts of AvNav. You will find a lot of commented examples in the templates.

Basically there are 3 categories of AvNav parts:

1. parts that have to occur exactly once.
   example: AVNConfig, AVNHttpServer,...
2. parts that normally don't need to be part of avnav_server.xml. You only insert them if you want to change some of the defaults.
   example: AVNAlarmHandler, AVNChartHandler,...
3. parts that can be contained once or multiple times. This includes all the input and output channels. If no entry is present the functionality will not be available.

Some properties will be available at various parts - as described here:

Name	Description	Example
enabled	Many handlers can be switched on or off with this parameter at the server/status page.	on
name	name of an input or output channel. This name will be shown at the status page and it can be used in a blackList to avoid output of data from this channel. There is always an internal default for the name.	nmea0183tosignalk
filter	Filtering of NMEA date. You can provide filters separated by comma. Only matching NMEA data will pass this filter. To make them independent of talker ids, the 2	$RMC,^$RMB,!AIVDM

	characters following a $ will not be considered. To e.g. only pass $GPRMC sentences (and any other $xxRMC) the filter would be: $RMC. If you prefix the filter with a ^ it will be negated, i.e. ^$RMC means. No $xxRMC records. For AIS data you can use "!" or "!AIVDM" .	
readFilter	If the channel is a combined reader/writer this is the filter for the input path	
blackList	List of channel names (separated by comma). Data originating from those channels will not be sent out. Consider the camel case - i.e. capital L.	nmea0183tosignalk
priority (since 20220421)	All NMEA input channels have a priority field. This controls which value will win if the same item will be decoded from multiple channels. The default priority is 50 you can adapt upwards and downwards. The SignalK integration has priority 40 by default.	50

The following tables list the main parts together with their parameters. If some properties in a template are not listed here, just leave them as they are.

AVNConfig

Base configuration and system time handling, category 1 (1x, mandatory)

Name	online	Description	default/template
settimecmd		command to set the system time. It will receive the time to be set as UTC timestamp like to be used for date -u	only set with the avnav-raspi package
settime (from 20220421)	X	if set setting the system time is enabled (requires settimecmd to be set additionally)	on
maxtimeback	X	maximal time (in seconds) the system time will be set backwards before all internal data will be cleared.	5
systimediff	X	maximum delta (in seconds) between system time and gps time before the system time is set.	5
settimeperiod		min time in seconds between 2 attempts	3600

		to set the system time	
ntphost (from 20220421)	X	NTP server to be queried if there is no valid GPS time	pool.ntp.org
switchtime (from 20220421)	X	time (in s) after setting the time before it is changed between valid GPS time and ntp time (and back). This time will also be used to wait for a valid GPS time when starting up (before NTP is used)	60
ownMMSI	X	MMSI of own vessel, will be filtered from the received AIS data	
expiryTime	X	time (in s) that received NMEA data will remain valid	30
aisExpiryTime	X	time (in s) that received AIS data will remain valid	1200

AVNFeeder

The internal NMEA list and decoder unit. category 2 (once, optional since version 20200325).

This part has no parameters anymore. In previous versions you could configure gpsd here, so potentially there still will be some parameters in your configuration. When you previously had an old version installed this entity will be named AVNGpsdFeeder.

AVNHttpServer

The internal HTTP server. category 1 (once, mandatory).

Apart from parameters for the AVNHttpServer there are some child entries that could be repeated multiple times. Normally there is no need for any change at those entries (Directory,MimeType).

Beside the httpPort you would normally not change anything.

Parameters for AVNHttpServer

Name	Description	default/template
httpPort	listener port for the HTTP server	8080
navurl	REST interface, no change	/viewer/avnav_navi.php
index	start page, no change	/viewer/avnav_viewer.html
httpHost	bind address for the server	0.0.0.0
numThreads	number of threads being used by the HTTP server	5

Parameters for Directory

Normally those values will be replaced by the -u command line parameter. You should not change them.

Name	Description	default/template
urlpath	the URL (without /)	
path	real path on the system	

Parameters for MimeType

You may configure (additional) mime types for the server. Potentially your own application requires some extension here if you have special files.

Name	Description	default/template
extension	extension(e.g. .avt)	
type	mime type (e.g. text/plain)	

AVNBlueToothReader

Reading of bluetooth devices that have a serial profile. category 3 (once, optional)
Only possible if your server has a bluetooth device.

Name	online	Description	default/template
maxDevices	X	max number of concurrently connected bluetooth devices	5

deviceList	X	Comma separated list of bluetooth device ids. If provided only those devices will be connected.
filter	X	filter for NMEA data
name	X	
priority	X	
enabled	X	

AVNSerialReader

Reading of serial devices. category 3 (multiple times, optional).
This reader only should be configured for devices that are directly connected to a built in UART. For USB devices use the AVNUsbSerialReader.

Name	online	Description	default/template
name	X	channel name	intern gebildeter Name
port	X	device name, e.g. /dev/ttyAMA0	
baud	X	Baudrate. If minBaud is also set baudRate is the maximal baud rate for automatic baud rate detection.	4800

minbaud	X	Minimal baud rate for auto detection. If not set or 0 - no automatic detection.	
timeout	X	Timeout in s, if no data received within this time the device will be closed and reopened.	1
bytesize	X	serial byte size	8
parity	X	parity	N
stopbits	X	number of stopbits	1
xoxoff	X	xon/xoff protocol (0: off)	0
rtscts	X	RTS/CTS protocol (0: off)	0
numerrors	X	After that number of errors the device will be closed and re opened.	20
autobaudtime	X	Time in seconds to recognize a newline during automatic baud rate detection.	5
filter	X	NMEA filter - see filter	
enabled	X		
priority	X		

AVNSerialWriter

Output via a serial device. Could also read from this device. category 3 (optional)
Only use this for direct serial devices, for USB devices use AVNUsbSerialReader

Name	online	Description	default/template
name	X	channel name	
combined	X	if "true", also read from this channel	false
readFilter	X	filter for the input path	
blackList	X	blackList, comma separated list of channel names from which no data should be written	
....		all parameters from AVNSerialReader	

AVNUsbSerialReader

Handles serial devices connected via USB. category 3 (once, optional).
This worker process scans all connected USB devices. It tries to open those with a serial profile, detect their baud rate and read NMEA data. This way AvNav normally auto detects all such devices.
You can define separate rules for individual devices. As device identification we use the USB port id where the device is plugged into. To find out this id simply watch the status page when you connect the device.

The parameters consist of 2 parts:

- Attributes for the entry itself
- Child entries of type UsbDevice

Example

```
<AVNUsbSerialReader maxDevices="5" allowUnknown="true"
baud="38400" minbaud="4800">
        <UsbDevice usbid="1-1.2.1:1.0" baud="38400"
minbaud="4800" filter="$RMC"/>
</AVNUsbSerialReader>
```

Parameters for AVNUsbSerialReader

Name	online	Description	default/template
maxDevices	X	maximal number of simultaneously connected USB devices	5
allowUnknown	X	only if this attribute is true, unknown devices (i.e. lacking an UsbDevice entry) will be handled	true
...		all parameters from AVNSerialReader except port. Those parameters will be used for unknown devices.	

Parameters for UsbDevice

Name	online	Description	default/template

usbid	X	USB Port identification e.g.. "1-1.2.1:1.0", mandatory	
type	X	type of the device reader, writer, combined, ignore, use ignore if you don't want the device to be used	reader
...		all parameters from AVNSerialReader if type = "reader" (except port, this will be set internally)	
...		all parameters from AVNSerialWriter if the type is combined or writer (except port, this will be set internally)	

AVNUdpReader

Opens a UDP port and reads data from it. category 3(optional, multiple times).

Name	online	Description	default/template
name	X	channel name for blackList and for display	
port	X	UDP port	
host	X	Bind address	0.0.0.0
minTime	X	if set: time in s before the next record is read	0

filter	X	filter for NMEA data	
enabled	X		
priority	X		
stripLeading	X	Remove anything before $ or ! in received lines.	off

AVNUdpWriter

Sends out NMEA Daten via UDP. category 3 (optional, multiple times)

Name	online	Description	default/template
name	X	channel name	
port	X	UDP destination port	
host	X	UDP destination address	
filter	X	filter NMEA data	
broadcast	X	set to true if you want to send broadcast	false
blackList	X	blackList for channel names	
enabled	X		

AVNSocketWriter

Output that opens a listener port, waits for connections and sends out NMEA data (TCP server). category 3 (multiple times, optional).

Name	online	Description	default/template
name	X	channel name	
port	X	listener port	
address	X	if set - bind to this address	0.0.0.0
filter	X	filter for NMEA data	
read	X	if true, data is also read from connections	false
priority	X	only if read is enabled	
readFilter	X	if read is true: NMEA filter for input direction	
blackList	X	blackList comma separated list of channel names (no data sent out coming from this channels)	
minTime	X	minimal time in s between 2 consecutive NMEA messages	0
avahiEnabled	X	if activated the server will announce this service via MDNS (Bonjour/Avahi). The service tpye is _nmea-0183._tcp	false

avahiName	X	the name that this service will be visible with in MDNS	avnav-server

AVNSocketReader

Input. Connects to TCP server reading data from it (TCP client). category 3 (multiple times, optional)

Name	online	Description	default/template
name	X	channel name	
port	X	TCP destination port	
host	X	TCP destination address (ip or hostname)	
timeout	X	connect timeout in seconds	10
minTime	X	min time between 2 received messages	0
filter	X	filter for NMEA data	
writeOut	X	send out NMEA data on this connection	false
writeFilter	X	filter for the sent NMEA data	empty
blackList	X	, separated list of source names that will not be sent out	empty
enabled	X		

priority	X		
stripLeading	X	Remove anything before $ or ! in received lines.	false

AVNNmea0183ServiceReader

This reader is very similar to the AVNSocketReader. But instead of configuring host and port you will configure the serviceName. AvNav will search for services of type _nmea-0183._tcp and will provide you a list of available services. Using this reader will give you connectivity even if the network topology potentially will change.

Name	online	Description	default/template
serviceName	X	The name of the service (AvNav will provide a list of found services)	--
timeout	X	connect timeout in seconds	10
minTime	X	min time between 2 received messages	0
filter	X	filter for NMEA data	empty
writeOut	X	send out NMEA data on this connection	false
writeFilter	X	filter for the sent NMEA data	empty
blackList	X	, separated list of source names that will not be sent	empty

		out
name	X	
enabled	X	
priority	X	

AVNBME280Reader

Reader for BME280 via I2C. category 3 (optional)
Writes MDA and XDR records.
Will only be available if python3-smbus is installed

Name	online	Description	default/template
name	X	channel name	
addr	X	I2C address of the sensor	0x77
interval	X	time between 2 NMEA records in s	5
writeXdr	X	write XDR if true	true
writeMda	X	write MDA if true	true
namePress	X	XDR transducer name for pressure	Barometer
nameHumid	X	XDR transducer name for humidity	Humidity

nameTemp	X	XDR transducer name for temperature	TempAir
enabled	X		
priority	X		

AVNBMB180Reader

Reader for BMP180 via I2C. category 3 (optional)
Writes MDA and XDR records.
Only available if python3-smbus is installed.

Name	online	Description	default/template
name	X	channel name	
addr	X	I2C address of the sensor	0x77
interval	X	time between 2 NMEA records in s	5
writeXdr	X	write XDR if true	true
writeMda	X	write MDA if true	true
namePress	X	XDR transducer name for pressure	Barometer
nameTemp	X	XDR transducer name for temperature	TempAir
enabled	X		

priority	X		

AVNSenseHatReader

Reader for SenseHat I2C. category 3 (optional)
Writes MDA and XDR records.
Requires python3-sense-hat to be installed

Name	online	Description	default/template
name	X	channel name	
interval	X	time between 2 NMEA records in s	5
writeXdr	X	write XDR if true	true
writeMda	X	write MDA if true	true
namePress	X	XDR transducer name for pressure	Barometer
nameTemp	X	XDR transducer name for temperature	TempAir
nameHumid	X	XDR transducer name for humidity	Humidity
nameRoll (from 20220421)	X	XDR transducer name for roll	Roll

namePitch (from 20220421)	X	XDR transducer name for pitch	Pitch
enabled	X		
priority	X		

AVNTrackWriter

Writes tracks in gpx format and in a simple ASCII format. category 3 (once, optional)

Name	online	Description	default/template
interval	X	minimal time between 2 track points (seconds) before written	10
mindistance	X	minimal distance between 2 track points (meters)	50
trackdir	X	track directory	<datadir>/tracks
cleanup	X	maximum age (in hours) for track data sent to the WebApp. Still files are stored to hold older data but these are not available to the WebApp.	25
writeFile	X	Write to file. If off, only have the track in memory.	on

AVNRouter

Handling of routing data (waypoints, routes, anchor alarm). Computing of RMB, APB. category 1 (once, mandatory).

Name	online	Description	default/template
name	X	channel name (for generated records)	
routesdir		directory for routes	<datadir>/routes
interval	X	time (in s) between RMB/APB records	5
computeRMB	X	compute RMB if a waypoint is active	true
computeAPB	X	compute APB if a waypoint is active	false
useRhumbLine (since 20220819)	X	use the rhumb line mode for routes	false
nextWpMode (since 20220819)	X	select the switching mode for the next waypoint in a route (late, 90, early)	late
nextWpTime (since 20220819)	X	The time to wait after the waypoint alarm (in seconds) before switching to the next waypoint	10

		(only nextWpMode = early).	

AVNNmeaLogger

Writes NMEA logs into the track directory. category 3 (once, optional).

Name	online	Description	default/template
maxfiles	X	number of files (1 per day) that are kept.	100
filter	X	filter for NMEA data	"$RMC,$DBT,$DBP"
interval	X	minimal time (in seconds) before a next record of the same type is written	5
enabled	X		

AVNImporter

Import Charts that still need conversion. category 2 (once, optional)

Name	online	Description	default/template
importDir		directory to read charts for import	<datadir>/import
workDir		working directory for the converter	<datadir>/work

waittime	X	converter waittime in seconds to start conversion after a new/changed file has been detected	30
knownExtensions	X	list of extensions the WebApp offers to upload to the importer.	kap,map,geo
keepInfoTime		time in seconds an information on import is kept	30
enabled	X		

AVNWpaHandler

Configuration for external wifi connections. category3 (once, optional)

Name	Description	default/template
wpaSocket	the control connection to wpa_supplicant	/var/run/wpa_supplicant/wlan-av1
ownSsid	own SSIDs - those will be filtered from the list	avnav,avnav1,avnav2
firewallCommand	if configured, you can enable	sudo -n $BASEDIR/../raspberry/iptables-

	access to the pi from the connected wifi network	ext.sh wlan-av1

AVNCommandHandler

Execute commands (e.g. for alarms). category2 (once, optional).

AVNCommandHandler itself has no parameters. There can be child configurations for various commands. The default configuration is:

```
<AVNCommandHandler>
     <Command name="sound" command="mpg123 -q"
repeat="1"/>
</AVNCommandHandler>
```

Parameters for Command

Name	Description	default/template
name	name for the command	
command	system command to be executed	
repeat	number of repetitions	1

AVNAlarmHandler

Management of Alarms. category 2 (once, optional).

Changed from 20220421.

The default configuration:

```
<AVNAlarmHandler>
        <Alarm name="waypoint" category="info"
repeat="1"/>
        <Alarm name="connectionLost" category="info"
repeat="1"/>
        <Alarm name="anchor" category="critical"
repeat="20000"/>
        <Alarm name="gps"  category="critical"
repeat="20000"/>
        <Alarm name="mob" category="critical"
repeat="2"/>
</AVNAlarmHandler>
```

Since version 20220421 you should remove existing alarm entries from your avnav_server.xml if you do not have special needs.
When using the defaults you can choose the sound directly at the server status page and you can e.g. use an own sound file that has been uploaded to the user directory.

If you would like to trigger special commands you need to configure this in avnav_server.xml .

```
<AVNAlarmHandler>
        <!-- legacy way of configuring alarms - still
supported but not recommended, use category at least and
optionally parameter -->
        <Alarm name="gps" category="critical"
command="gpsAlarm"
parameter="$BASEDIR/../sounds/anchorAlarm.mp3"
repeat="20000"/>
        <!-- with the next line we configuer a special
command that will be called when we receive a "sinking"
notification from SignalK
             the sound is determined by the category -
```

```
and this is also the parameter that the command will
receive -->
        <Alarm name="sk:sinking" command="sinkingAlarm"
category="critical" repeat="2"/>
</AVNAlarmHandler>
```

Parameter for Alarm

Name	Description	default
name	Name of the alarm (mandatory)	empty
category	Categorie (info,critical)	empty
command	Command to be executed (must be configured at AVNCommandHandler). If not configured the default command will be used.	empty
autoclean	Switch off the alarm when the command has finished.	off
sound	The name of a sound file. If provided this will be used for the sound in the browser and as parameter for the command. Relative pathes are based on the internal sound dir or on the user dir. If not set the sound will be dtermined by the category - or as last resort by 'parameter'.	empty
repeat	Number of command and sound repetitions	1
parameter	If provided this is the parameter for the command. If neither sound nor category are set	

	this is treated as the path to a sound file and alos used for the browser.

Parameter for AlarmHandler

Name	online	Description	default/template
infoSound	X	Name of a mp3 file for the sound in category info. Can be selected form a list of sound files from the user dir (upload your sounds via the Download page) or from the internal sound dir.	waypointAlarm.mp
criticalSound	X	Name of a mp3 file for the sound in category critical. Can be selected form a list of sound files from the user dir (upload your sounds via the Download page) or from the internal sound dir.	anchorAlarm.mp3
defaultCommand	X	Command for alarms without an own command definition.	sound

		This command has to be configured at AVNCommandHandler.	
stopAlarmPin	X	Only on Raspberry Pi. If set (board numbering), a low at this pin will switch off all currently active alarms.	leer

AVNPluginHandler

Management of plugins. category 2 (once, optional).
AVNPluginHandler is handling plugins. They can be installed in different directories:

- builtin: /usr/lib/avnav/server/plugins
- system: /user/lib/avnav/plugins
- user: /home/pi/avnav/data/plugins

Beside the parameters for the plugin handler itself, each plugin could expect parameters. The name of the plugin consists of the category and the plugin directory name. Example:

```
<AVNPluginHandler>
        <builtin-signalk enabled="true"/>
        <builtin-canboat enabled="true"
allowKeyOverwrite="true" autoSendRMC="30"
sourceName="canboatgen"/>
</AVNPluginHandler>
```

Parameters for AVNPluginHandler

Name	Description	default/template
builtinDir	directory for built in plugins, no change	/usr/lib/avnav/server/plugins
systemDir	directory for plugins installed with packages	/usr/lib/avnav/plugins
userDir	directory for user plugins	/home/pi/avnav/data/plugins

Parameters for builtin-canboat

Name	online	Description	default/template
enabled	X	only if true the plugin will become active	false
allowKeyOverride	X	set to true to allow the plugin to set date and time in the internal store	false
port	X	canboat json port	2598
host	X	host for n2kd	localhost
autoSendRMC	X	if no RMC record is seen in the NMEA data within this period (in seconds) but valid position and date/time data were received from n2k - generate a RMC record (important	30

		to have date and time on NMEA0183)	
sourceName	X	channel name for RMC	plugin-Name
timeInterval	X	minimal time (in seconds) before we read a new time value from NMEA2000	0.5
timePGNs	X	PGNs for setting the time	126992,129029

AVNChartHandler

Management of charts. category 2 (once, optional)

Name	Description	default/template
period	time between 2 reads of the chart directory	30
upzoom	number of zoom levels above the highest available	2

AVNUserHandler

Management of the user files. category 2 (once, optional)

Name	Description	default/template
interval	time (in s) between 2 reads of the directory	5

AVNImagesHandler

Management of user images. category 2 (once, optional)

Name	Description	default/template
interval	time (in s) between 2 reads of the directory	5

AVNUserAppHandler

Management of configured User Apps. category 2 (once, optional)

This is somehow a special handler. Normally initially there should be no configuration. Within the App you can configure the user apps. Manual change of the configuration is not recommended.

AVNAvahiHandler

Controls how AvNav registers at Avahi(MDNS).

Name	online	Description	default/template
serviceName	X	The name that will be visible in tools that are able to browse MDNS information. This is not the host name that you would use in e.g. avnav.local!	avnav
maxRetries	X	How many retries AvNav would make if the name is already used by someone	20

		else. Retries would add a "-nn" suffix to the name.	
timout	X	timeout when connecting to the avahi daemon (s)	10
enabled	X		

AVNSignalKHandler

New with 20220421.
For a description refer to the SignalK Documentation.

Extensions and AddOns

You can extend the functionality in different ways. Some extensions are already installed together with AvNav or are available as seperate packages.

Adding Other Web Pages

== not on Android ==

You can add other web pages (either externally or on the same server like AvNav) as so called "User Apps".
There is a dialog based configuration available at the User App Config Page.
The Pages will become visible at the User App Page.

Extending the Functionality of the Web App

With some lines of java script code you can add functionality to the web app. Especially you can define own display (text based or graphics based), you can add new formatters for display values - or even add buttons that will trigger various actions.
Your java script code will go to a user.js file and AvNav provides an integrated editor for this file on the Files/Download page.
Beside the java script code you typically will also need some css code to tune the display of your new entities - this css will go into a user.css file.

Plugins

== not on Android ==

Plugins allow to extend the functionality of AvNav both on the server side and in the display (client). AvNav already provides some built in plugins, others can be installed as packages. And you can develop your own.

Plugins provided with AvNav:

- SignalK
  Connect to a SignalK Server. See the detailed description.
- Canboat
  This implements NMEA2000 support using canboat. See the detailed description.

For the development of own plugins and a list of plugins that you can install as packages refer to the plugin description.

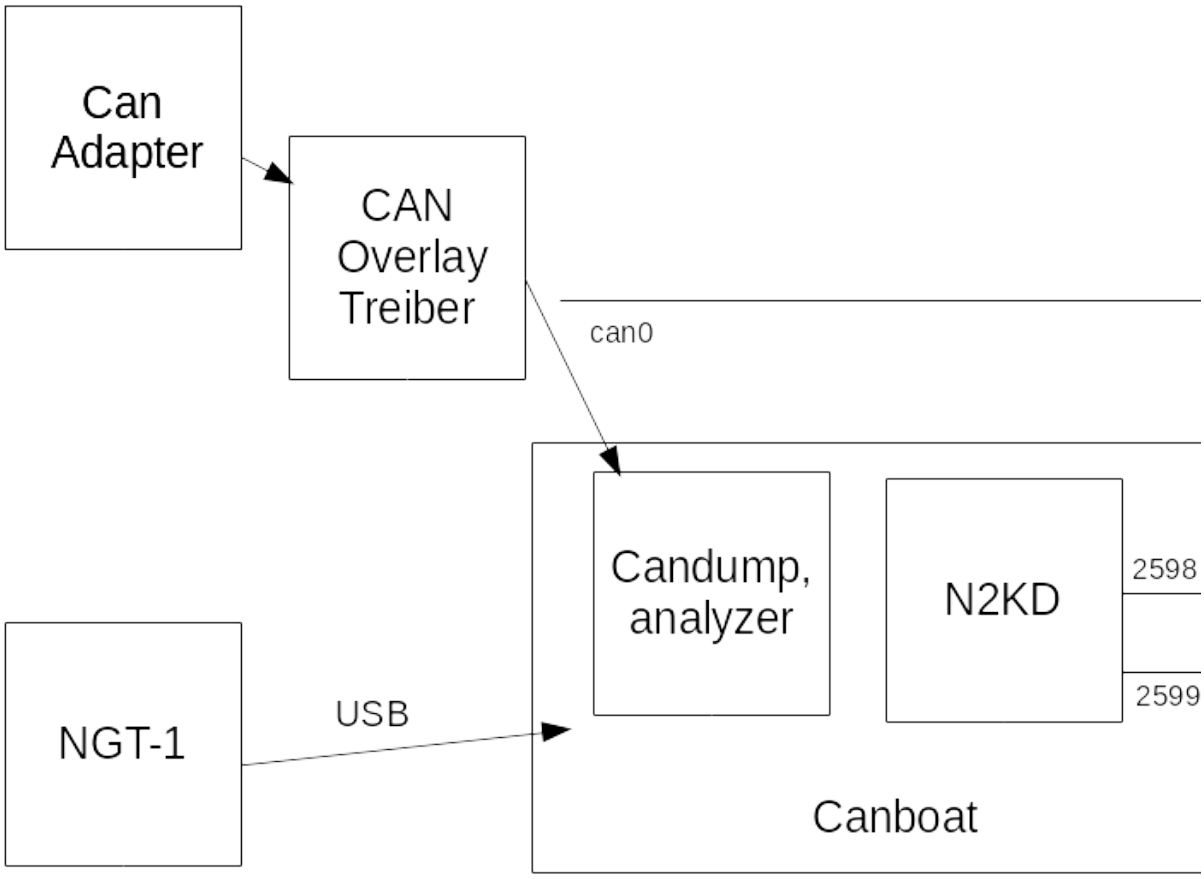
Interworking with Canboat and SignalK

Starting with release 20200204 AvNav can interact with canboat (NMEA2000) and signalk.

Important Hint: Since version 20220421 the handling for SignalK changed.

Canboat (NMEA2000)

Canboat supports interfacing to NMEA 2000 bus systems using either a CAN adapter (e.g. MCP2515 or a Waveshare RS485 CAN-HAT) or USB attached devices like the Actisene NGT-1. For simple adapters you need to ensure that they have 2 voltage supplies (3.3V and 5V) to be usable for the raspberry pi. Many of the simple ones do not have that!



In the diagram the basic setup is shown as we provide it with the headless images.

If you connect a CAN adapter via SPI typically you need to enable an overlay in /boot/config.txt. Appropriate entries are prepared to use the MCP2515 and only need to be uncommented. Potentially you have to adjust the clock frequency and the GPIO pin used for the interrupt. Such can adapters are listed as network interface (probably they require configuration, but they are preconfigured in the images).
The interface should be visible with

```
ifconfig can0
```

If you are planning to use an Actisense NGT-1 connected via USB refer to the documentation at canboat.

AvNav is communicating with n2kd. This daemon converts incoming NMEA2000 data to NMEA0183 (although not completely). The configuration for n2kd is located at

```
/etc/default/n2kd
```

Within our images this file predefines a connection to can0. If you are going to use an USB based adapter you have to modify this configuration.
In this case, please also add an entry in avnav_server.xml to exclude this adapter from auto detection (watch the AvNav status page while connecting the adapter - note its USB id and use it for the entry):

```
<AVNUsbSerialReader .....>
<UsbDevice usbid="x:y.z" type="ignore"/>
....
```

When traffic is present on the NMEA 2000 bus and if everything is setup correctly, you should see NMEA data / json data at ports 2599 and 2598. You can check this with

```
nc localhost 2599
```

If you are unable to see data you can check the canboat status with

```
sudo systemctl status canboat
```

There should be 2 connections between AvNav and n2kd (predefined in the images). AvNav receives NMEA0183 data on one connection (port 2599) and some of the json data on the other connection (port 2598). The latter is necessary as n2kd does not provide any NMEA0183 record containing a

complete time stamp (like RMC). To get the data from NMEA2000 AvNav directly decodes the pgns 126992 and 129029 to internally set date and time. It can also generate a RMC record internally (if valid position data is received via NMEA0183).

To setup the connections following configuration settings are necessary within avnav_server.xml. Our images have them preconfigured, otherwise they can be copied & pasted from the template at /usr/lib/avnav/raspberry/avnav_server.xml.

```
<AVNSocketWriter port="34568" maxDevices="5"
   filter=""read="true" minTime="50"
   name="nmea0183tosignalk"
   blackList="canboatnmea0183,canboatgen"/>
<AVNSocketReader port="2599" host="localhost" filter=""
name="canboatnmea0183"/>
<AVNPluginHandler>
  <builtin-canboat enabled="true"
allowKeyOverwrite="true" autoSendRMC="30"
sourceName="canboatgen"/>
</AVNPluginHandler>
```

The first entry defines an additional output port for AvNav. This port provides all received/created NMEA 0183 data - excluding data received from canboat. This port is used for SignalK integration. Canboat data are omitted as signalk directly reads data from the NMEA 2000 bus.

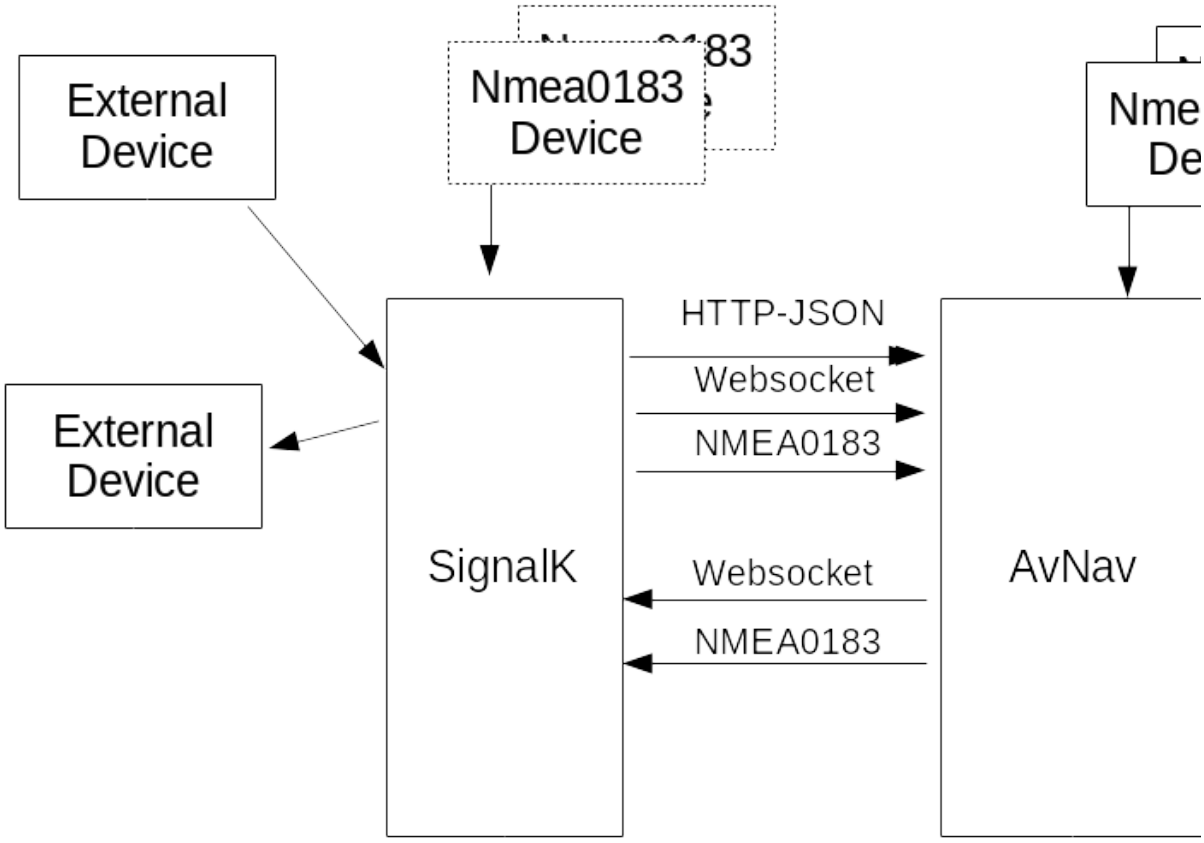
The SocketReader at localhost:2599 reads NMEA0183 data from n2kd.

Direct reading of NMEA2000 data from n2kd is handled by a plugin. Therefore an entry has to be created in AVNPluginHandler. The settings as in above example activate the plugin. allowKeyOverwrite permits the plugin to set date and time. autoSendRMC=30 will advise AvNav to send out a RMC record every one second (if no other RMC records are seen within 30 seconds). For further explanation of the plugin parameters refer to the source code.

AvNav is currently not prepared to output data to NMEA2000. If this is required, you can handle this via signalk.

SignalK

With version 20220421 the integration of AvNav with SignalK has been extended.



As a first step when integrating AvNav with SignalK you need to decide about the basic flow of data.

There are 2 basic options:

1. NMEA data will be received directly by AvNav and will be forwarded to SignalK. This flow is used in the AvNav headless images. SignalK data are sent back (via HTTP-Json and websocket) to AvNav and can be displayed there.
   The data that is used by AvNav for navigation (including AIS data) will be decoded from the NMEA data directly by AvNav.
   Exception: NMEA2000 data that are received via canboat will have to be sent in parallel to SignalK.
2. NMEA data will be received by SignalK first and will be sent to via HTTP-Json and websocket to AvNav.
   This flow is used by OpenPlotter e.g..

For both variants AvNav can also send own data to SignalK. Currently those are routing data to the next way point (either as RMB/APB NMEA0183 data or as SignalK delta update - see below).

Additionally Notifications (Alarms) can be fetched and sent from/to SignalK.

With the version 20220421 the SignalK integration is not any longer handled by a plugin but by a dedicated "handler" - AVNSignalKHandler. You need to modify the Configuration at this handler now.

1. NMEA to AvNav and from there to SignalK

This flow is preconfigured in the AvNav headless images.

A AVNSocketWriter is created in AvNav (default: port 34568). This writer will forward received NMEA data. Forwarding of data from canboat will be prevented by blacklist entries.

```
<AVNSocketWriter port="34568" maxDevices="5" filter=""
read="true" minTime="50" name="nmea0183tosignalk"
blackList="canboatnmea0183,canboatgen"/>
```

On the SignalK side you need to configure a matching data connection for NMEA0183, TCP client.

By default the AVNSignalKHandler is configured to reach SignalK via localhost:3000 and to read all data below vessels/self. Those values will be stored in AvNav below gps.signalk,... and this way can be used for Displays. For fetching the data a mixture of polling with HTTP-Json and delta updates via a websocket connection is used. The polling will ensure proper sync of data, the websocket stream will give fast updates.

On the configured SocketWriter connection AvNav will also send its routing data as RMB / APB Sätze towards SignalK.
Additionally you can enable the sending and receiving of SignalK notifications at the AVNSignalKHandler.

Data that is arriving directly at SignalK via other pathes will be available in AvNav (as described below gps.signalk...) - **but they cannot be used for AvNav's navigation functions**.

If you would like to route your data to SignalK first you should have a look at the other data flow.

An advantage of flow 1 (NMEA to AvNav first) is the fact that this would still work even is SignalK is not available or not working for some reason.

2. NMEA to SignalK first

With this flow (that is preconfigured at OpenPlotter from AvNavInstaller version xxxx) NMEA will be received and stored by SignalK. You need to configure the necessary data connections in SignalK.

In AvNav you configure the AVNSignalKHandler to fetch the data from SignalK in a combination of HTTP-Json and a websocket delta update connection.
The flags "decodeData" and "fetchAis" will be set (see Configuration). This way received data will be stored in AvNav for it's navigation functions.
Additionally the data will be stored (a second time) as described at (1) below

gps.signalk.... to ensure compatibility for the display usage.
For the mapping of the received data refer to [Mapping].

Additionally the sending of data ("sendData") is activated at the AVNSignalKhandler. Routing data for the current way point and alarms will be sent to SignalK.
Notifications from SignalK will be received.
To be able to write data to SignalK you have to configure a user with write access at SignalK (see Configuration).

The NMEA connections to and from SignalK (port 10110) are not necessary any longer. You also do not need any plugin at SignalK to create NMEA data for sending it to AvNav.

If you update from an older version you can just simply disable the NMEA connections and adapt the settings of AVNSignalKHandler.

Selection of the Data Flow

For a decision whether to use flow 1 (NMEA to AvNav first) or flow 2 (NMEA to SignalK first) you can start from the default of your installation.

You should only change it if there are good reasons for. By using the configuration at the  status/server page you can configure your flow.
You can even create mixtures of them.

You only have to be careful to avoid loops - e.g. sending data from AvNav via NMEA0183 to SignalK and getting it back from SignalK again via NMEA0183. AvNav tries to address such mixed configurations with a "sourcePriority" at every connection. All NMEA connections have a default priority of 50 compared to a default of 40 for The AVNSignalKHandler. This way data that is decoded by AvNav itself will always win against data received from SignalK (until you change the priorities).

Konfiguration

The AVNSignalKHandler has the following configuration.

Name	Description	D
name	A name for the handler. Will be shown at the status page and used in logs.	e
enabled	If switched off the SignalK integration is disabled. Configured NMEA connections are not affected.	o
decodeData	If switched on the received data will be used for AvNav's navigation functions. See Mapping.	o O
fetchAis	If switched on AIS data (/vessels/*) is fetched from SignalK every 10s (aisQueryPeriod) and stored in AvNav as AIS data.	o O
priority	The priority of the decoded SignalK data	4
port	The SignalK HTTP port.	3
host	The SignalK server hostname/the IP address.	lc
aisQueryPeriod	Intervall (in s) for fetching AIS data	1
period	Period in ms for HTTP-Json queries to SignalK. if the python websocket libraries are available (the default) this interval will be enlarged close to the expiry time of navigation data in AvNav.	1
fetchCharts	Fetch informations about charts installed at SignalK. It requires the SignalK-chart-provider plugin to be installed.	o
chartQueryPeriod	Interval (in s) for querying the chart	1

	information.	
chartProxyMode	If SignalK is running on a different computer then AvNav it could happen that the Browser that displays AvNav cannot directly reach this computer. Therefore AvNav can proxy the chart requests to SignalK. This creates some additional load on AvNav and can be switched off. Normally you can leave the default. *sameHost*: only proxy if SignalK is running on a different computer then AvNav *never*: no proxy at all (you can use this if the browser can directly reach SignalK at the address configured here) *always*: alsways proxy. Can be used if e.g. the SignalK port cannot directly be reached from outside.	sa
ignoreTimestamp	Normally AvNav will consider the timestamp of SignalK data and will ignore data that is to old (expiryPeriod in AVNConfig). A potential time difference between SignalK and AvNav is considered. Sometimes SignalK does not use it's own local time for those time stamps but instead uses time stamps from the received data. Especially when using simulation data those times can be in the past and AvNav would ignore such data. By setting this flag AvNav will ignore those time stamps and will use it's own local time instead. It will update the timestamp whenever a value changes. This is not as accurate as using the original time	o

	stamps from SignalK but can help in making older simulation data working.	
sendData	Send data towards SignalK. You can still decide separately whether waypoint data and/or alarms should be sent. This will **not control the sending of NMEA data** to SignalK!	o O
userName	The name of a SignalK user with write access at the SignalK server. This user must exist with the correct rights at SignalK. Unfortunately SignalK has no interface to find out whether a particular user is able to write certain pathes. Only for localhost AvNav will check if the user has (in principle) write access.	a
password	The password for the configured user. For SignalK running at the same computer like AvNav this normally can be left empty (as long as the SignalK configuration is located at the default path $HOME/.signalk/security.json). If you encounter an error in the status display at "authentication" it could become necessary to set this password also for local access (e.g. the config being located at a different location). Remark: The password will be stored in clear text in avnav_server.xml.	<
sendWp	Send way point data to SignalK. See Mapping.	o
sendNotifications	Send AvNav alarms as notifications to SignalK - See Mapping.	o
receiveNotifications	Receive notifications from SignalK as alarms.	o

notifyWhiteList	A comma separated list of SignalK notifications, that should be received. The pathes are without the leading "notifications." e.g.: navigation.arrivalCircleEntered,mob,fire,sinking. If the list is empty all notifications will be received but still the notifyBlacklist is considered.	<(
notifyBlackList	A comma separated list of SignalK notifications that should not be received.	se
webSocketRetry	Interval (in s) for recreating a new websocket connection if the old one has been closed.	2(

Some of the parameters will only become visible if the "parent" parameter is set.

Mapping

SignalK pathes are mapped to AvNav data pathes:

/vessels/self/... => gps.signalk....

Those pathes are not used in AvNav internally but you can use them for display.

If "decodeData" is activatedthe following mappings will become active (also see in the code).

Remark: Courses/angles are stored in degrees inside AvNav whereas SignalK used rad. The mapping will convert between them. If there are multiple SignalK pathes in a mapping the first available will be used.

SignalK below /vessels/self/	AvNav
navigation.headingMagnetic	gps.headingMag
navigation.headingTrue	gps.headingTrue
environment.water.temperature	gps.waterTemp
navigation.speedThroughWater	gps.waterSpeed
environment.wind.speedTrue	gps.trueWindSpeed
environment.wind.speedApparent	gps.windSpeed
environment.wind.angleApparent	gps.windAngle
environment.wind.angleTrueWater (since 20240520)	gps.trueWindAngle
navigation.position.latitude	gps.lat
navigation.position.longitude	gps.lon
navigation.courseOverGroundTrue	gps.track
navigation.speedOverGround	gps.speed
environment.depth.belowTransducer	gps.depthBelowTransducer
environment.depth.belowSurface	gps.depthBelowWaterline
environment.depth.belowKeel	gps.depthBelowKeel
navigation.datetime	gps.time

navigation.gnss.satellitesInView.count	gps.satInview
navigation.gnss.satellites	gps.satUsed
navigation.magneticDeviation (since 20240520)	gps.magDeviation
navigation.magneticVariation (since 20240520)	gps.magVariation
environment.current.setTrue (since 20240520)	gps.currentSet
environment.current.drift (since 20240520)	gps.currentDrift

AIS are mapped like ("fetchAis" on):

SignalK vessels/*/	AvNav ais	Remark
mmsi	mmsi	only if mmsi is available the data is stored
name	shipname	
navigation.speedOverGround	speed	
navigation.courseOverGroundTrue	course	
communication.callsignVhf	callsign	
design.aisShipType	shiptype	
navigation.position.longitude	lon	

navigation.position.latitude	lat	
navigation.destination	destination	
sensors.ais.class	type	class A -> type 1 class B -> type 18 other -> type other
design.beam	beam	
design.length	length	
design.draft	draught	
navigation.state	status	
navigation.headingTrue	heading	
atonType	aid_type	

If "sendWp" is active the following mapping is used:

AvNav	SignalK vessels/self/
currentLeg.to.lon	navigation.courseGreatCircle.nextPoint.pos and navigation.courseGreatCircle.nextPoint.lon
currentLeg.to.lat	navigation.courseGreatCircle.nextPoint.pos and navigation.courseGreatCircle.nextPoint.lati
currentLeg.from.lon	navigation.courseGreatCircle.previousPoin

currentLeg.from.lon	navigation.courseGreatCircle.previousPoin
currentLeg.distance	navigation.courseGreatCircle.nextPoint.dis
currentleg.dstBearing	navigation.courseGreatCircle.nextPoint.bea and navigation.courseGreatCircle.bearingToDes
currentLeg.xte	navigation.courseGreatCircle.crossTrackErr
currentLeg.approachDistance	navigation.courseGreatCircle.nextPoint.arr
currentLeg.bearing	navigation.courseGreatCircle.bearingTrack and navigation.courseGreatCircle.bearingOrigin

If the AvNav routing mode is rhumbline all "courseGreatCircle" are replaced with "courseRhumbline". The doubled sending of some values with different pathes works around some issues on the signalK side (like https://github.com/SignalK/signalk-to-nmea2000/issues/94).

For notifications there only a few mappings, other will be mapped to "sk:"+name in AvNav. Example: notifications.sinking will become sk:sinking.

AvNav	SignalK vessels/self/notifications	Value
mob	mob	'state':'emergency', 'method':['visual','sound'], 'message':'man overboard'
waypoint	arrivalCircleEntered	'state': 'normal', 'method': ['visual','sound'],

		'message': 'arrival circle entered'
anchor	navigation.anchor	'state':'emergency', 'method': ['visual','sound'], 'message': 'anchor drags'

SignalK notifications without a mapping will be assigned to a category based on their state. With the catgeory you can define in AVNCommandHandler which command should be executed and which sound should be used.

emergency -> critical
normal -> normal

SignalK - Charts

Since version 202011xx an integration of the SignalK chart provider is available. Charts provided by SignalK can be selected on the main page. You have to enable the chart provider plugin within SignalK and configure/upload charts.

Normally AvNav will just provide the information about these charts, the browser will directly access SignalK to load the chart tiles.

If this is blocked (e.g. by firewall rules) you can proxy all chart tile requests via AvNav.

User Spezific Java Script Code

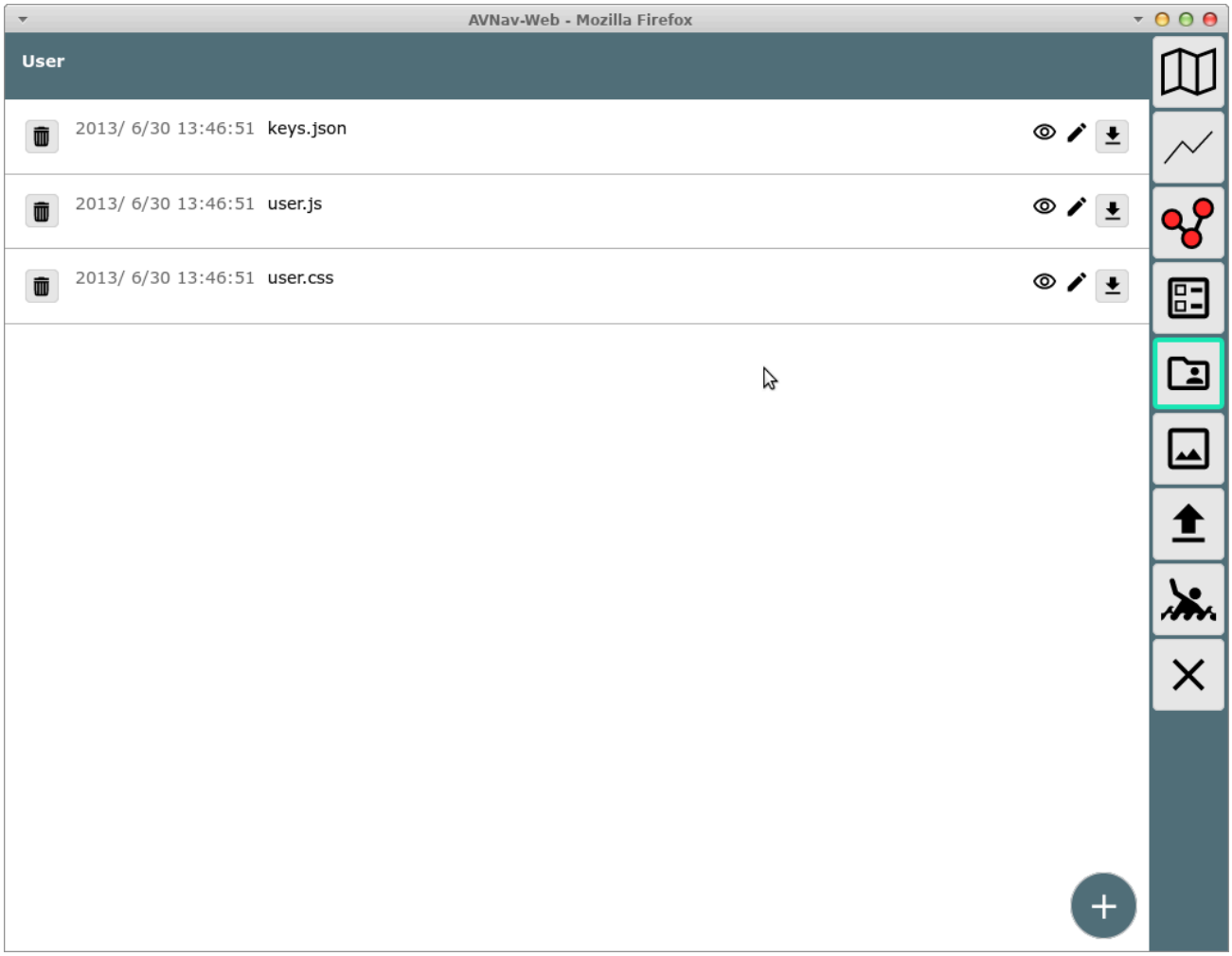
To adapt AvNav to your needs you can extend it with some java script code.

It is possible to define new data displays (widgets) to be placed using the layout editor. In principle you can run any java script code but you have to take care not to disturb the AvNav main functions.

The java script code has to be located at user.js in the directory BASEDIR/user/viewer.
(e.g. on the pi BASEDIR is /home/pi/avnav/data).

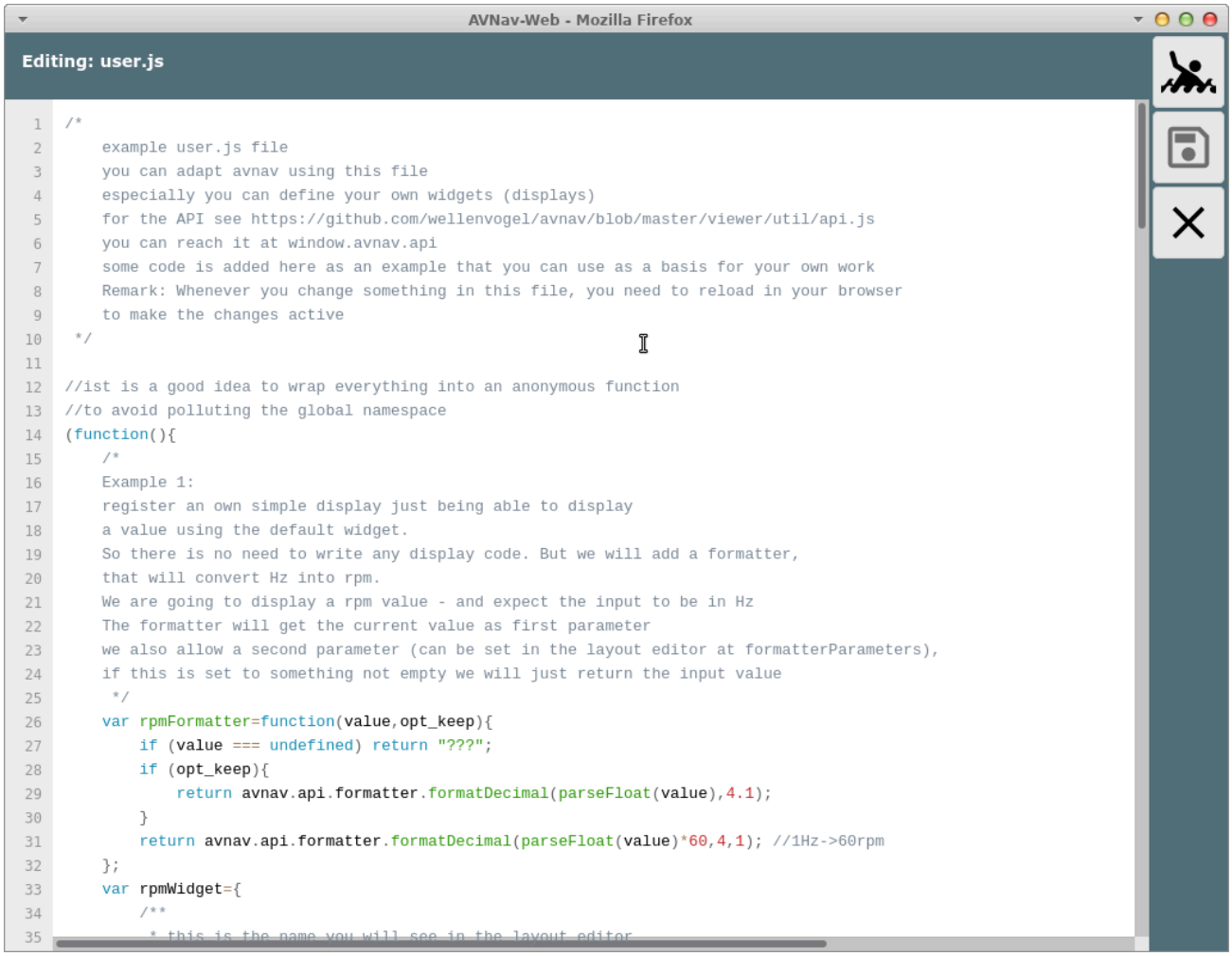
Editing

To simplify working on the code you can directly access the files in this directory via Files/Download page , subpage .



In the screenshot you see a file user.js - initally created from a template on the first start of the server.

By clicking the file and selecting "Edit" from the dialog you can immediately start working on the file.



There are a couple of examples already included in the file. They demonstrate some variants of new widgets. After editing use 💾 to store the file and reload AvNav to watch your changes in action.

I would recommend to download and backup the file regularily after editing since there is no version control included in AvNav.

The current file template you can also find on github.

Widgets

Basically you can add the following types of widgets:

- widgets with an own formatter (and potentially fixed values retrieved from the store) based on the default widget (example 1- user.js: rpmWidget, testPlugin: testPluing_simpleWidget)

- adaptations and extensions of the graphics widgets (canvas gauges) (example 2 - user.js: rpmGauge)
  This way you can access canvas widget parameters that are currently not directly accessible.
- widgets with own HTML code (example 3 - user.js: userSpecialRpm, TestPlugin: testPlugin_courseWidget)
- widgets with canvas graphics (example within the TestPlugin: testPlugin_courseWidget)
- widgets with own HTML that are communicating with the server part of a plugin (TestPlugin: testPlugin_serverWidget
- widgets that will draw graphics on the map (type: map) - since 20220819 e.g. SailInstrument

The interface to communicate with AvNav is available at github and in the example code.

For map widgets you can access the underlying libraries for geographical computations via the API (functions LatLon and Dms).

Canvas Gauges

For canvas gauge widgets you can set some parameters (see canvas gauges description) either to fixed values (in this case they must become part of the widget definition - see the values in the example starting from line 134) or you can make them settable by the user within the layout editor (put them into the editable widget parameters - example starting from line 156).

Additionally you can define an own formatter and set it as default for the widget.

If you would like to hide some predefined parameters in the layout editor you need to set them to "false" in the editable parameters.

```
var rpmGaugeUserParameter = {
      ...
      formatter: false,
```

```
        formatterParameters: false
        };
```

For every gauge widget you need to provide the parameter "type" - either "radialGauge" or "linearGauge".
Additionally they have the parameter

```
  drawValue (boolean)
```

This parameter controls whether the value is displayed as numeric (additionally) or not. The original parameter "valueBox" from canvas gauges is ignored!

Beside the parameters you can also define a translateFunction. This function receives an object with all the current values and can modify this before it is set at canvas gauges(example from line 104). This function needs to be "stateless". That means the output must only depend form the input or any other fixed values. Otherwise some changes potentially will not been drawn.

Own Widgets

For an own widget the following functions/properties can be implemented:

Name	Type	Usable for type	Description
name	String	all	the name of the widget a in the layout editor
type	string (optional)	alle	defines which type of wi be created Values: radialGauge, line map If you don't set the type default widget will be us renderHtml and no rend

			provided) or a special us be used.
renderHtml	function (optional)	userWidget	This method must retur containing valid HTML. It injected into the widget. event handlers to your e have to register them (se initFunction). In the HTM assign them with: `<button onclick="myHandle</button>` Note that is is not exactl as you only provide the r event handler - no java s The "this" inside renderH the widget context (an o specific for the particula If the event handler is ca "this" will also point to th The parameter of render contains all parameters and the values defined a The function will be calle the values change.
renderCanvas	function (optional)	userWidget, map	With this function you ca the provided canvas obj The second parameter o renderCanvas contains a

			parameters of the widge values defined at storeK The function will be calle the values change. The "this" inside renderC to the widget context (ar is specific for the particu For map widgets this car overlay on the map. At tl context you have functic between coordinates an pixels. it is important to correct canvas with save/restore widgets share the same
storeKeys	object	all	You have to provide the read from the internal st as parameters for the re function.
caption	string (optional)	all	A default caption.
unit	string (optional)	alle	A default unit
formatter	function (optional)	defaultWidget, radialGauge, linearGauge	A formatter for the value defaultWidget this functi mandatory.
translateFunction	function (optional)	alle	This function is called wi current values as param must return an object cc derived values.

			This may be used to trar before rendering if no ov is implemented - see exa
initFunction	function (optional)	userWidget, map	If defined, this function v once after creating the w any renderXXX function. context is provided as a and as the "this" variable The widget context has a eventHandler Property. have to define all event h be used in your HTML cc With a triggerRedraw fur also available at the cont force a new rendering of causing the renderXXX fu be called again. Starting with version 202 init function will receive parameter that has the p the widget (including all that you defined as edita parameters).
finalizeFunktion	function (optional)	userWidget, map	If defined, this function v before the widget is rem "this" refers to the widge Additionally the context provided as the first par in the initFunction). Starting with version 202 init function will receive parameter that has the p the widget (including all

	that you defined as edita parameters).

The following global variables are set for the java script code:

Name	plugin.js/user.js	Decsription
AVNAV_BASE_URL	both	The URL to the directory from where the java script code has been loaded. This can be used to load other elements from there. From user.js you can access files from the images directory with AVNAV_BASE_URL+"../images". AVNAV_BASE_URL+"/api" will give you the base URL for plugins to maintain communication with the python side.
AVNAV_PLUGIN_NAME	plugin.js	The name of the plugin.

After defining a widget you need to register it at AvNav (avnav.registerWidget).

Widget Context

User widgets and map widgets will receive a widget context. This will be created for every instance of a widget and will be provided to the following functions:

- initFunction (this and first parameter)

- finalizeFunction (this and first parameter)
- renderHtml (this)
- renderCanvas (this)

To make the access to "this" inside the functions working they need to be defined the classic way with "function" - not as arrow functions.

Correct:

```
let userWidget={
        renderHtml: function(context,props){
                return "<p>Hello</p>";
        }
}
```

Inside the widget context you can store userc data that will be needed in consecutive function calls.
Additionally it contains a couple of functions you can use in your widget code.

Name	Widget	Parameter	Description
eventHandler	userWidget	---	eventHandler is not a functio object. If you have event han your rendered HTML you nee the handler function there. E.g.: renderHtml returns <button onclick="clickHandler"/>) - yo register a function "clickHanc this.eventHandler.clickHandl {...} See TestPlugin.

triggerRedraw	userWidget	---	This function needs to be call widget would like itself to be after communicating with a s See TestPlugin.
lonLatToPixel	map	lon,lat	Converts longitude, latitude i coordinate for the renderCar Returns an array [x,y].
pixelToLonLat	map	x,y	Computes longitude and latit canvas coordinates x and y. Returns an array [lon,lat].
getScale	map	---	Returns the scale factor for th High resolution displays norn scale factor > 1. You should a dimension of your drawings (texts) depending on this scal
getRotation	map	---	Returns the current map rota radians!)
getContext	map	---	Returns the renderingContex canvas (only active inside the function)
getDimensions	map	---	Returns the width and height ([width,height]).
triggerRender	map	---	Same functionality like trigge userWidget.

Widget Parameters

As a second parameter you can provide an object containing parameters to be displayed in the layout editor.

Examples can be found in the user.js template. Values selected by the user in the Layout editor will become part of the properties provided to the renderHtml and renderCanvas functions (except for parameters of type KEY: the values read from the store will be provided).
For each of the parameters you can provide the following properties:

Name	Type	Description
	key	The name of the parameter as to be displayed in the layout editor and as to be available for the renderXXX functions.
type	string	STRING, NUMBER, KEY, SELECT, ARRAY, BOOLEAN, COLOR The type of the parameter. Depending on the type a different user dialog will be shown: for COLOR this will be a color selector, for SELECT a select list and for KEY the list of known items in the global store. For an array you can provide a list of values, separated by comma.
default	depending on type	The default value. For COLOR a color css property - like "rgba(200, 50, 50, .75)"
list	Array (only for type SELECT)	An array of strings or objects {name:'xxx',value:'yyy'} - they will be displayed in the select list.

There are some predefined parameters for the layout editor. For those no describing object with properties should be provided, just true or false (this defines whether or not they will be prompted in the layout editor).

Those are:

- caption (STRING)
- unit (STRING)
- formatter (SELECT)
- formatterParameters (ARRAY)
- value (KEY)
- className (STRING)

An example definition:

```
var exampleUserParameters = {
    //formatterParameters is already well known to
avnav, so no need for any definition
    //just tell avnav that the user should be able to
set this
    formatterParameters: true,
    //we would like to get a value from the internal
data store
    //if we name it "value" avnav already knows how to
ask the user about it
    value: true,
    //we allow the user to define a minValue and a
maxValue
    minValue: {type: 'NUMBER', default: 0},
    maxValue: {type: 'NUMBER', default: 4000},
   };
```

Formatter

Beside the widgets you can implement your own formatters preparing values for display.
Many formatters already are available in the system - see Layout Editor.

Since version 20210106 you can register your own formatters in AvNav and, by this, make them available to all other widgets. Basically a formatter is a function accepting the value to be formatted as first parameter and returning a string result.
The length of the string should be constant and independent from the current value (use space padding if necessary). This is to avoid interfering with automatic sizing on dashboard pages.

A formatter can accept additional parameters to control the output. Those parameters can be set with the widget property "formatterParameters" - typically in the Layout Editor.

Example:

```
const formatTemperature=function(data,opt_unit){
    try{
        if (! opt_unit ||
opt_unit.toLowerCase().match(/^k/)){
            return
avnav.api.formatter.formatDecimal(data,3,1);
        }
        if (opt_unit.toLowerCase().match(/^c/)){
            return
avnav.api.formatter.formatDecimal(parseFloat(data)-273.15,
        }
    }catch(e){
        return "-----"
    }
}
formatTemperature.parameters=[
    {name:'unit',type:'SELECT',list:
```

```
['celsius','kelvin'],default:'celsius'}
]

avnav.api.registerFormatter("mySpecialTemperature",formatT
```

registerFormatter will throw an exception if a formatter with the same name already exists.

Each formatter function should carry a "parameters" property. This property describes the values presented to the user in the layout editor as formatterParameters. The values in this definition follow the same syntax as for editable widget parameters.

Libraries and Images

Images and libaries uploaded to the same directory can be accessed by your java script code. Images additionally can be accessed in the images directory.

Embedding of libraries can be done like this:

```
var fileref=document.createElement('script');
fileref.setAttribute("type","text/javascript");
fileref.setAttribute("src", "my_nice_lib.js");
document.getElementsByTagName("head")
[0].appendChild(fileref)
```

I recommend to assign css classes to your own widgets so to provide easy means of adapting their look and feel later on by user defined CSS. You should not use HTML ids in your code as the widgets might be instantiated multiple times on one page.

If you need to download data from the server I recommend using fetch. All files in the user directory (or the plugin directory for plugin.js) can be accessed with AVNAV_BASE_URL+"/"+name.

If you need to create an additional file in the user directory (e.g. text or HTML) you can directly do this using the "+" button (lower right) - afterwards you can directly edit the file.

Feature Formatter

Since version 20210114 you can register functions to convert and format data from overlay files for the "Feature Info" dialog.
You can implement them in the user.js or by a plugin.

With

```
avnav.api.registerFeatureFormatter('myHtmlInfo',myHtmlInfo
```

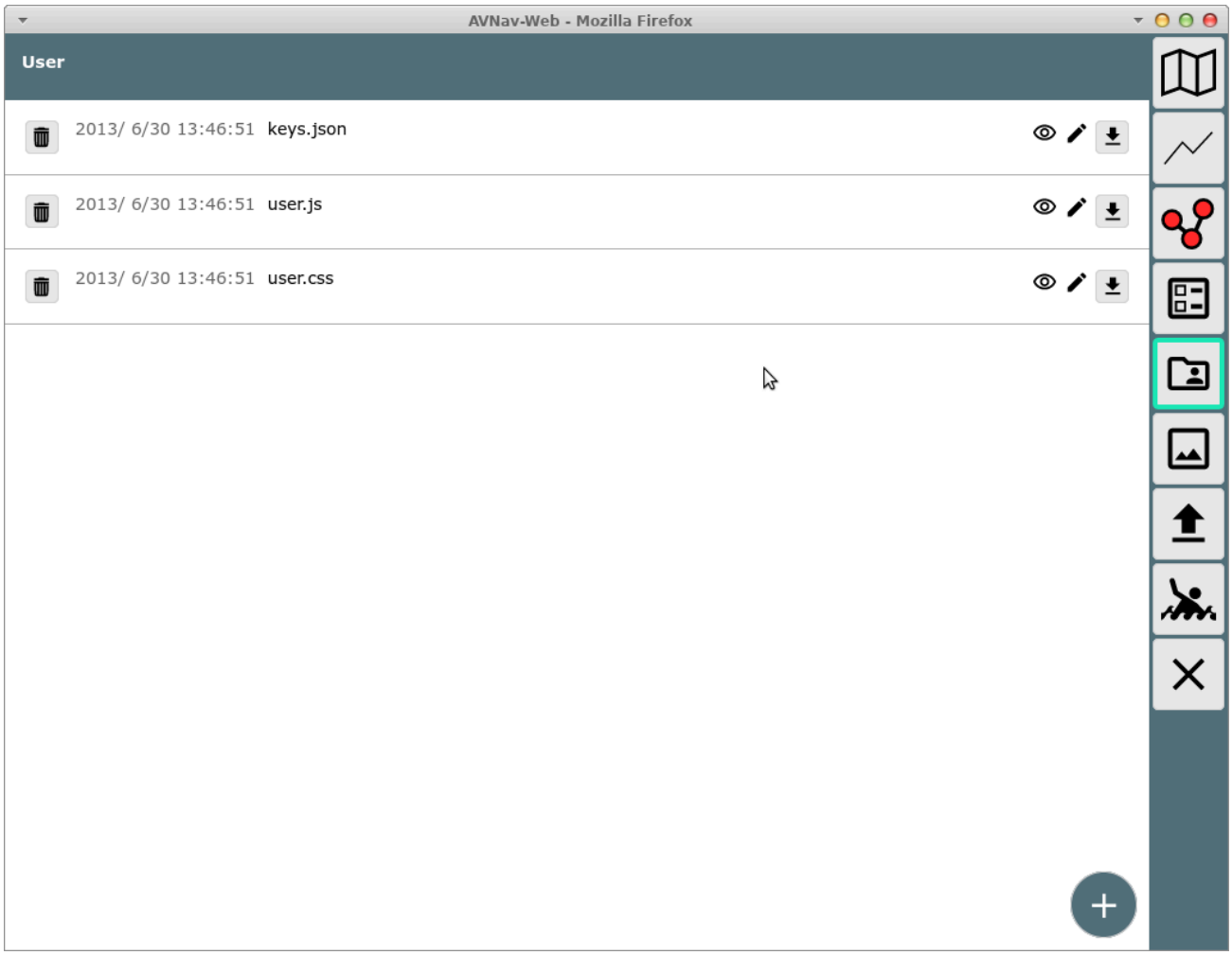
you register such a function. For details refer to Overlays.

User defined css

As the AvNav frontend is completely browser based (single page app with reactjs) you can adapt its appearance with css.

There is a file user.css intended to be used for your css rules. This file is located next to user.js in BASEDIR/user/viewer. (BASEDIR is /home/pi/avnav/data on a pi).

Via the Files/Download page and the subpage you can select the file to edit it.



By clicking the file and selecting "Edit" in the dialog you can start the editor. During the first start a template has been created that already contains an example.

Since Version 20210619 the name of the current layout is assigned as a CSS class to the application. All special characters will be replaced by "_". The layout user.default will become the CSS class user_default. Using this feature you can define different CSS for different layouts (e.g. for different devices).

```
.user_default .widget .COG {
  color: green;
}
.user_special .widget .COG {
  color: red;
}
```

A proven workflow is the usage of your browser's developer tools (Chrome, Firefox,...), selecting the elements you would like to adapt and testing your changes.

Afterwards you can directly copy the values to user.css and (after saving) test them by reloading the AvNav page.

Avnav Plugins

=== not for android ===

Plugins extend the functionality of AvNav. They can extend server functions (using python code) as well as the WebApp using Java Script or CSS.

Each plugin has to reside in a separate directory. The directory name defines the plugin name. 2 root directories are scanned by AvNav in search for plugins:

- "systemdir" - a directory to hold plugins installed for all users of a system (e.g. via a package). This is /usr/lib/avnav/plugins.
- "userdir" - a directory for plugins specific for a particular user. This is a sub directory of the user's "data dir" - /home/pi/avnav/data/plugins on the pi, $HOME/avnav/plugins on other Linux systems.

Beside those two an additional directory holds internal plugins (builtin).

In principle a plugin's server part can read and write data from/to AvNav at various interfaces. The WebApp parts would normally display such values or simply add new display functions to AvNav or adapt its look and feel.

Up to 3 files within a plugin directory are processed by AvNav. Additional files in the directory will be ignored by AvNav - however they can be required by the plugin itself.

Those files are:

- plugin.py - the server parts, optional
- plugin.js - the Java Script parts, optional
- plugin.css - the CSS parts, optional

There is a complete example for a plugin on GitHub.

Installation

To create an own plugin you can either provide it as a zip file or a debian package.
If providing a zip file it should at top level contain one subdirectory with the name of the plugin (do not include "plugin" or "avnav" in this name).
A user of the plugin would need to unpack the zip file in it's AvNav data dir, sub dir plugins (e.g. /home/pi/avnav/data/plugins on a raspberry pi).
This way the plugin would become a "user plugin"

If you are providing a debian package it should be named like avnav-xxx-plugin. The content should be unpacked to /usr/lib/avnav/plugins/<pluginName>.
This way the plugin will be come a "system plugin".
Plugin packages should contain the name of the plugin (i.e. the directory name) as a meta data field "avnav-plugin".
Example:

```
avnav-plugin: system-obp-plotterv3
```

This will help the avnav-updater to correctly identify whether a plugin package should be shown or not.
When you set the meta data field "avnav-hidden" to true, the package will only be shown from the avnav-updater if explicitely enabled.

List of Plugins

- ocharts - charts from o-charts
- ochartsng - new implementation for o-charts, S57 charts
- Seatalk Remote - in combination with the seatalk remote control from AK-Homberger
- History - Data history and display
- Update - update AvNav (and related packages) eliminating the need for command line access
  Also integrates a config file editor and a log viewer for AvNav
- MapProxy - integrate MapProxy to access and download from various online chart sources
- Obp-RC-Remote - plugin using the IR Remote from Christian
- Sail-Instrument-Plugin - decoding and computation of course and wind data, Sail Instrument
- rudder-angle - show the rudder angle (from SignalK)
- Obp-PlotterV3 - plugin for all special functions for the Open Boat Projects 10 inch plotter (V3)

plugin.js

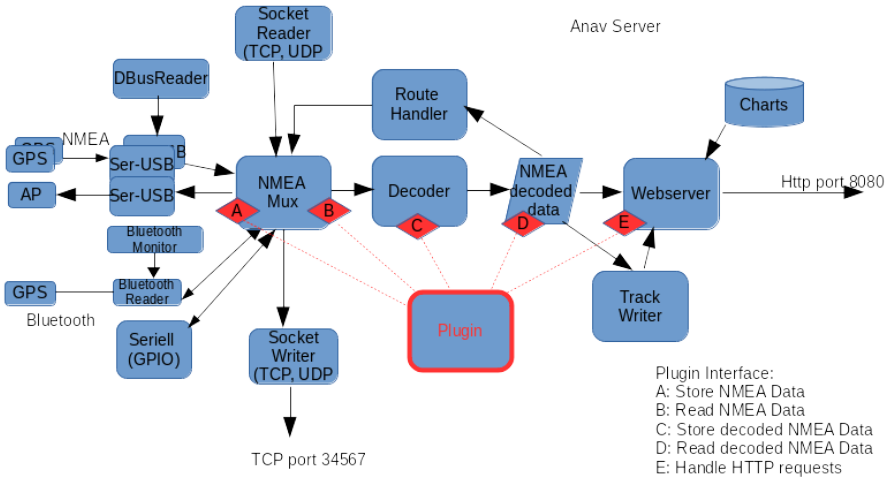
Java Script code file offering same functions as described in user specific Java Script code .

plugin.css

CSS code file offering same options as described in user specific CSS.

plugin.py

Overview



The diagram provides a rough overview of the AvNav server's internal structure together with the points (interfaces) where plugins can interact.

Point	Function	Example
A	Feed NMEA data to the internal list. These become available at all outputs. Hint: a decoder is required for the WebApp to access them	Read from a sensor and generate the correct NMEA0183 sentence.
B	Reading of inbound NMEA data. Here you can access (potentially using a filter) all NMEA data passing through AvNav.	Together with point "C": decoding of NMEA sentences.

C	Feed data into the internal store of AvNav. Data in the internal store is available in a tree like structure. Each leaf has a key in the form of "a.b.c....". Example: "gps.lat". All keys starting with "gps...." are sent to the WebApp automatically and become available there as "nav.gps...." - see layout editor and user specific Java Script. Keys need to be registered by the plugin before using them. It is not possible to override keys defined by AvNav itself. Exception: "allowKeyOverride" set to true by user.	Write the data from a sensor like gps.temperature.outside or write decoded NMEA data
D	Reading of data from the internal store	Computing of derived data (and writing them back at "C") - or just send values somewhere externally.
E	Handling of HTTP requests	The java script parts may send a HTTP request to be handled in the python code. Normally a dictionary will be returned (as json).

A plugin.py example is available at GitHub.

Basic Structure

To be recognized by AvNav the plugin has to provide:

1. A class within plugin.py (the name should be Plugin)
2. The class must contain a static method (@classmethod) with the name pluginInfo. It must return a dictionary.

   ```
   * description (mandatory)
   * data: list of keys to be stored (optional)
        * path - the key - see AVNApi.addData, all
   pathes starting with "gps." will be sent to the GUI
        * description
   ```

   An example could look like:

   ```
   @classmethod
   def pluginInfo(cls):
     return {
       'description': 'a test plugins',
       'data': [
           {
           'path': 'gps.test',
           'description': 'output of testdecoder',
           }
         ]
       }
   ```

3. The constructor must expect one parameter.
   When calling the constructor AvNav will provide an instance of the API as the parameter.
4. The class must implement a run method (without parameters).
   After initialization this method will be called in a separate thread.
   Normally you would have an endless loop here to provide the plugin functionality.

You can provide parameters for the plugin in avnav_server.xml. They can be read at the API using getConfigValue.

Plugin API

The API provides the following functions:

Function	Description
log,debug,error	Logging functions. Lines will be written to the AvNav log file. Avoid writing too many log and error entries. This would flood the log and obfuscate important entries (example: do not write an error every second)
getConfigValue	get a config value from avnav_server.xml.
fetchFromQueue	Interface B: read data from the internal NMEA list. There is an example in the API code. The filter option is working the same way as in avnav_server.xml.
addNMEA	Interface A: feed a NMEA record to the internal list. You can leave the check sum to be computed by AvNav. You can also block decoding within AvNav. The source parameter is the channel name you would use in blackList parameters .
addData	Interface C: feed data into the internal store. You can only write data with keys that have been announced by the return values of the pluginInfo method.

getSingleValue	Interface D: read data from the internal store. To combine multiple reads there is the method getDataByPrefix
setStatus	provide the current state of the plugin. This will be shown at the status page.
registerUserApp	You can register a user app as a plugin. You need an URL and an icon file. The icon file should be located in the plugin directory. You can use $HOST inside the URL. It will be replaced by the AvNav server's address. Example in the signalk plugin.
registerLayout	If the plugin provides its own widgets it could make sense to provide a layout for the user to select. Just save the layout in the plugin directory after creating it with the layout editor. Example within the signalk plugin.

registerSettingsFile (since 20220225)	Register an own settings file (that has been previously saved from the settings page). The file name (second parameter) is relative to the plugin dir. The name (first parameter) is shown to the user. Within this file you can use $prefix$ in the layout name if you want to refer to a layout that you register from the same plugin.
	`...`
	`"layoutName": "$prefix$.main"`
	`....`

	This will refer to a layout that you registered with the name "main".
getDataDir	The data directory of AvNav.
registerChartProvider	If the plugin provides charts you need to register a callback here to return a list of charts.
registerRequestHandler	If the plugin handles HTTP requests (interface E) you need to register a callback here. The URL triggering the callback is: <pluginBase>/api pluginBase is the value returned by getBaseUrl (/plugins/<name>). The java script parts can compute the API url using the variable AVNAV_BASE_URL: AVNAV_BASE_URL+"/api" In the simple case your callback should return a dictionary that is sent back as json.
getBaseUrl	returns the base URL for the plugin
registerUsbHandler (since 20201227)	registers a callback for an USB device. With this registration in place AvNav will not further care about this particular device. The provided callback will be called with the device path as a parameter as soon as the USB device has been detected. You can easily figure out the USB id by watching the status page when connecting the device. The USB id is bound to the USB socket - see AVNUsbSerialReader. Using this api a plugin can easily handle a serial

	device by its own. An example you can find on GitHub.
getAvNavVersion (since 20210115)	return the current version of AvNav (integer value). Can be used to test if certain functions can be expected to be available
saveConfigValues (since 20210322)	Store the plugin's config values in avnav_server.xml. The parameter must by a dictionary holding the values to be changed. The plugin must ensure that it can restart with the parameters provided here.
registerEditableParameters (since 20210322)	This registers a list of config values which can be changed during runtime. The first parameter must be a list of dictionary objects describing the parameters, the second parameter provides a callback to be called with the changed parameters (will typically call saveConfigValues). For the parameter descriptions refer to the source code.
registerRestart (since 20210322)	Register a stopCallback that will allow to disable the plugin.
unregisterUserApp (since 20210322)	unregister a previously registered user app
deregisterUsbHandler (since 20210322)	unregister an usb device id (see registerUsbHandler)

shouldStopMainThread (since 20210322)	can be used in the main loop of the plugin to check if it should stop. Do not call this from other threads as it always returns True.
sendRemoteCommand (since 20230426)	Send a remote control command, see the source code for details.
registerSettingsFile (since 20230426)	Register a file with store user settings. Those settings can be loaded by the user.
registerCommand (since 20230426)	Register a command that can be executed by AvNav. This can be used to replace existing commands or register new commands. See the source code or the AVNCommandHandler config for details.
registerConverter (since 20240520)	Register a converter for chart types. For an example refer to the ochartsng plugin
deregisterConverter (since 20240520)	Deregister a chart converter

Enabling and Disabling of System Plugins

(since 20230426)

To be able to disable plugins that are installed with debian packages, there is a script /usr/lib/avnav/plugin.sh.
You can call this script (as root) with to maintain the visibility of system plugins and even set some default parameters.
Just call the script with no parameters to get a help.

Special Functions for Raspberry Pi

(since 20230426)

Plugins that are provided as debian packages for the raspberry pi can provide a shell script "plugin-startup.sh".
This script will allow plugins to configure system parameters.
It will always be called during the boot process of the system.
whether a plugin script is called or not depends on a parameter in the avnav.conf file (see Image preparation). The parameter name is:
AVNAV_<PLUGIN>
with plugin being the plugin name (i.e. name of it's directory) all translated to upper case and all characters except digits and letters a-z being removed.
If thisparameter is set to "yes" the plugin script will be called.

There are 3 ways it can be called:

plugin-startup.sh enable

This will be used when the plugin becomes active (first boot with parameter set in avnav.conf) for the first time.
This plugin should now make all necessary changes on the system (if possible it should be able to revert them, see below).
Typically those are changes in /boot/config.txt or other files.
The script should return 1 to indicate the need for a reboot, 0 otherwise or < 0 for errors.

There are some helper functions available that can be used by this script.
You can include those helpers using

```
. "$AVNAV_SETUP_HELPER"
```

The environment variable AVNAV_SETUP_HELPER is set up before calling the script.
For an example script refer to the obp-plotterv3-plugin.

plugin-startup.sh disable

This call will be made when the parameter in avnav.conf changes from yes to anything else. The script should revert it's changes that have been made to the system - as far as possible.
Remark: As the process is normally only intended to be used once when the image starts for the first time it should not be a real problem if not all changes can be reverted.

plugin.startup.sh [no parameters]

This will be called on any boot. You should not try to modify system settings that require a reboot or some restarts afterwards as this could be very surprising to the user if on an arbitrary reboot suddenly things are changed on the system.

Mobile Atlas Creator Mapsources

2022/04/17
adapt to mobac 2.2.2

2021/08/22
adaptation for mobac 2.2x, new BSH layers

2020/01/26

Some minor modifications to fix access to BSH servers.

Sources

For Mobile Atlas Creator I created a couple of map sources that will give you more control when accessing online map sources with some xml config. You need to unpack the file avnav-mapsources.zip within the directory "mapsources" of the Mobile Atlas Creator.
For mobac version 2.2.1 please use avnav-mapsources-before222.zip.
For mobac versions < 2.2.1 please use avnav-mapsources-before22.zip.

You will get (among others) a "mashUp" with BSH chart services (see bsh-viewer) and OpenSeaMap ("BSH OpenSeaMap 2021 Extended"). Additionally BSH allone ("BSH 2021 Extended") or OpenSeaMap + OpenStreetMap ("OWS OpenSeaMap 2021").

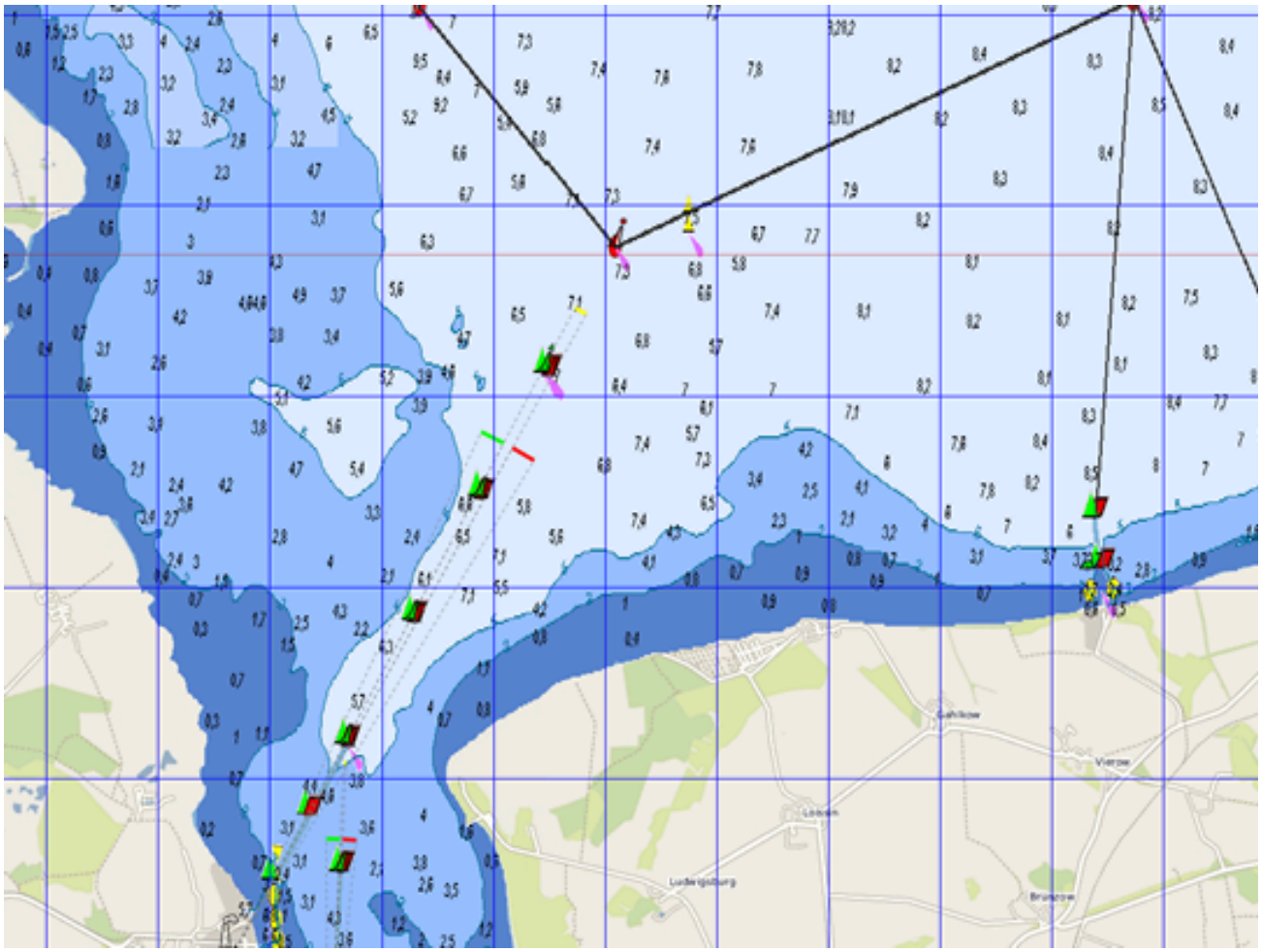
If you want to modify - just adapt the .exml files. You can try the BSH layers with my bsh-viewer (just edit the sources on the right side). Additionally you can adjust the colors (I tried to get some more contrast). To change the color mappings you can open one of the tiles with a grafics program and pick up the hex values for the colors you would like to change. Those must be adapted inside the .exml files.

Typically the download will run for a long time - the BSH servers are slow. If the download fails, just try again in MOBAC. Set the cache to a sufficient expiration time (e.g. one month). Always use "OsmdroidGEMF" as atlas format.

If you apply changes to the .exml files you need to delete the databases for this chartsource (on tilestore tab) and restart MOBAC.

Source files on github.

The result could look like this ... (approach to Greifswald):



The download:

- avnav-mapsources.zip (Mapsources BSH, BSH+OpenSeaMap)