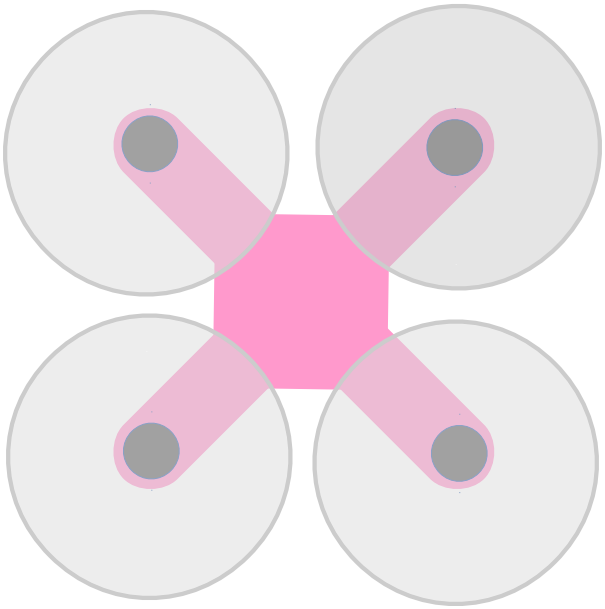


mwptools

Mission Planner, GCS, Log replay for inav

Jonathan Hudson

Table of contents

1.1 Overview

Sweet dreams and flying machines[1]

mwp (originally "multi-wii planner") is a mission planner, ground control station and flight logger for MSP (Multiwiii Serial Protocol) compatible flight controller firmware (Multiwii and inav at least).

From its MultiWii origins mwp has evolved to support navigation capabilities in inav.

inav is now the main development target, however MultiWii mission planning and ground control remains a supported function.

1.1.1 Features

- **Mission Planner** : Support all inav and MultiWii mission planning functions, including all inav extensions.
- **Ground Control Station** : (Near) real time ground control monitoring, using a wide range of telemetry options. Audio status reports.
- **Monitoring and warning** of other airspace users (inav radar, manned aviation ADS-B)
- **Flight log replay** (Blackbox, OTX/ETX logs, BulletGCSS)
- **Embedded video** (live and replay)
- **Support** functions
  - inav Safehome editor
  - Automatic mission shape generation, block moves, animated mission preview.
  - Terrain Analysis with WP mission rewrite to safe margins
  - Favourite sites editor
  - KML/KMZ static overlays

Supported Protocols

mwp supports the following telemetry protocols :

- MSP (MultiWii Serial Protocol)
- LTM (Lightweight Telemetry)
- MAVLink (iNav subset)
- Smartport (direct / via inverter / or from Multi-protocol Module)
- Crossfire (CRSF)
- Flysky AA (via Multi-protocol Module)
- BulletGCCS MQTT

Monitoring

mwp also supports the real-time display of adjacent aircraft using:

- inav-radar (INAV UAS)
- MAVlink Traffic Report (e.g. full-size aviation, typically ADS-B via a device such as uAvionix PingRX)

Log replay formats

mwp supports replay of:

- mwp log files (logged by GCS)
- Blackbox logs
- OpenTX CSV (sdcard) logs
- BulletGCSS logs
- Ardupilot (`.bin`) log

Log replay requires tools from the flightlog2x project.

1.1.2 Platforms and OS

The tools are designed to be portable and as far as possible platform and architecture agnostic. The suite is developed on Arch Linux and is tested on Debian (Bullseye, Sid), Ubuntu (latest and most recent LTS), Fedora (current) and FreeBSD (current release). mwp also runs on MS Windows, with Windows 11 / WSL-g is almost on feature parity with Linux / FreeBSD. Other (older) OS are unsupported, but may work (i.e. Debian 10 is used for the "release" builds).

1.1.3 Build and installation

Build and installation is described in the following sections:

- Generic build and installation Linux, FreeBSD, Windows / WSL
  - Windows additional information (Win11, Win10 and earlier)

Installation Tutorial

Somewhat outdated, if you follow this, please note that some of is much simplified by the later Generic build and installation article.

Watch on Vimeo

---

1. *James Taylor, Fire and Rain. Full line is 'sweet dreams and flying machines in pieces on the ground', you may skip the final part.* ↩

1.2 Running mwp

1.2.1 Video Tutorials

There is an slightly outdated video that describes dock usage and some post-install actions:

Watch on Vimeo

> **Update**
>
> - More useful than I remember!
> - The dock is now installed populated.
> - WP editor switch is enabled by default
> - There is now a graphical "favourite places" editor
> - The build system is no longer `make`

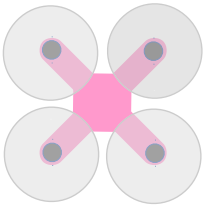
Apart from that, it's quite informative.

Tutorial Playlist

All the developer's tutorial videos are in a youtube playlist.

1.2.2 Graphical User Interface

Once you've built and / or installed mwp.



The install process installs an desktop icon and `mwp.desktop` application file

The `desktop` file tells the window manager where to find mwp and on modern desktop environments (e.g. Gnome Shell, xfce, kde), mwp will be added to the system application menu and / or 'finder'. * It is also possible to run mwp from a terminal, passing additional options if required. * Such options can be added to a configuration file for persistence or use from the graphical icon.

1.2.3 Command line options

mwp's command line options may be displayed with the `--help` option:

```
mwp --help
Usage:
  mwp [OPTION…]

Help Options:
  -h, --help                        Show help options
  --help-all                        Show all help options
  --help-gapplication               Show GApplication options
  --help-gtk                        Show GTK Options

Application Options:
  -m, --mission=file-name           Mission file
  -s, --serial-device=device_name   Serial device
  -d, --device=device-name          Serial device
  -f, --flight-controller=fc-name   mw|mwnav|bf|cf
  -c, --connect                     connect to first device (does not set auto flag)
  -a, --auto-connect                auto-connect to first device (sets auto flag)
  -N, --no-poll                     don't poll for nav info
  -T, --no-trail                    don't display GPS trail
```



```
  -r, --raw-log                     log raw serial data to file
  --ignore-sizing                   ignore minimum size constraint
  --full-screen                     open full screen
  --ignore-rotation                 legacy unused
  --dont-maximise                   don't maximise the window
  --force-mag                       force mag for vehicle direction
  --force-nav                       force nav capaable
  -l, --layout                      Layout name
  -t, --force-type=type-code_no     Model type
  -4, --force4                      Force ipv4
  -3, --ignore-3dr                  Ignore 3DR RSSI info
  -H, --centre-on-home              Centre on home
  --debug-flags                     Debug flags (mask)
  -p, --replay-mwp=file-name        replay mwp log file
  -b, --replay-bbox=file-name       replay bbox log file
  --centre=position                 Centre position
  --offline                         force offline proxy mode
  -S, --n-points=N                  Number of points shown in GPS trail
  -M, --mod-points=N                Modulo points to show in GPS trail
  --rings=number,interval           Range rings (number, interval(m)), e.g. --rings 10,20
  --voice-command=command string    External speech command
  -v, --version                     show version
  --build-id                        show build id
  --really-really-run-as-root       no reason to ever use this
  --forward-to=device-name          forward telemetry to
  --radar-device=device-name        dedicated inav radar device
  --perma-warn                      info dialogues never time out
  --fsmenu                          use a menu bar in full screen (vice a menu button)
  -k, --kmlfile=file-name           KML file
  --relaxed-msp                     don't check MSP direction flag
  --smartport                       Unsupported
  --display=DISPLAY                 X display to use
```

Bash completion

mwp installation also installs a 'bash completion' script (and also a `blackbox_decode` completion script). Note this is only available after you log in, so on first install, it's only available after the *next* login.

This facilitates automatic command completion, so you don't have to remember all the options or be always typing `mwp --help`.

Typing `mwp` and then `<TAB>` will first display the option lead `--`; then a subsequent `<TAB><TAB>` will display all the options. If one then typed `ra<TAB><TAB>`, it would complete to:

```
$ mwp --ra
--radar-device  --raw-log
```

Further entry (e.g. `d`) would complete the command (`--radar-device`).

Adding options to a running mwp

Certain options, like `--replay-bbox`, `--mission` allow you to add a file to a running mwp. So if mwp was running, either from the command line or Desktop Environment icon, then (for example):

```
mwp --mission file-i-forgot.mission
```

would load the mission `file-i-forgot.mission` into the running mwp rather than starting a new instance.

Drag and Drop

You can *drag and drop* relevant files onto the mwp map:

- Blackbox Logs
- Mission Files
- KML Overlays

Clean and unclean exits

If you exit mwp from the **Quit** menu (or Control-Q key shortcut), then the current dock layout will be saved; if you close mwp from the Window Manager `close` title bar button, or CLI `kill` command, the layout is not saved; this is a feature.

1.3 User interface

1.3.1 Main Window

The mwp main window and the main user interface elements are:

1. Menu bar. The menu options are described later.
2. Map and Mission settings
3. Communications and telemetry settings
4. Map window
5. Dock Bar
6. Dock Items (Docklets)
7. Mouse location (user preference units, cursor or map centre location)
8. Flight controller information
9. Sensor status and flight timer

In the sections that follow, there will be a brief summary of each part; more detail will then provided in subsequent sections.

1.3.2 Menu Bar (1)

The following tables summarise the available menu options. Where usage is not obvious, operation will be described later on.

File Menu

Item	Usage
Open Mission	Offers a dialog to open a mission file
Append Mission file	Appends a mission to the current mission set (creates a multi-mission element)
Save Mission	Saves the mission to the current mission file, overwriting any extant content
Save Mission As	Saves the mission to a user selected file. For a multi-mission the user can choose not to save specified mission segments.
Download Mission from FC	Downs a (multi-) mission from the flight controller
Upload Mission to FC > Upload Active Mission	Uploads the current mission segment to the flight controller
Upload Mission to FC > Upload All Missions	Uploads all mission segments to the flight controller
Restore Mission from EEPROM	Restores the EEPROM stored mission from the flight controller
Save Mission to EEPROM	Saves the current mission segment(s) to the flight controller. The current active mission segment (in a multi-mission) is set as the active mission in the FC
Replay mwp log	Replay a mwp (JSON) log file
Load mwp log	Loads a mwp (JSON) log file (i.e, as fast as practical, ignoring timings)
Replay blackbox log	Replays a Blackbox log file
Load blackbox log	Loads a Blackbox log file (i.e, as fast as practical, ignoring timings)
Replay OTX log	Replays an OpenTX / EdgeTX CSV log file. (Also BulletGCSS and Ardupilot logs where available)
Load OTX log	Loads an OpenTX / EdgeTX CSV log file. (Also BulletGCSS and Ardupilot logs where available)
Stop Replay	Stops a running replay
Static Overlay > Load	Loads a static KML format overlay file
Static Overlay > Remove	Removes a loaded KML file from the display
Safe Homes	Invokes the inav safe-home editor
Quit	Cleanly quits the application, saving the display layout

Edit Menu

Item	Usage
Preferences	Displays the preferences dialogue
Multi Mission Manager	Display the multi-mission dialogue to remove segments from a multi-mission
CLI serial terminal	Displays the inav CLI using the current connection
Nav Config	(Legacy MW) MW Nav Configuration
Get FC Mission Info	Display the mission status from a connected FC
Seed current map	Shows a dialogue to seed the map cache for offline (field) use
Reboot FC	Reboots a connected flight controller
Audio Test	Reads out the mwp version number as an audio test

View Menu

Item	Usage
Zoom to Mission	Zooms the map to the currently loaded mission
Set location as default	Sets the current location as the default (startup) location
Centre on position ...	Shows the "Centre on Position" selector and "favourite places" editor"
Map Source	Displays a dialogue with information on the selected map source
GPS Statistics	Displays FC GPS status (rate, packets, errors, timeouts, HDOP/EPV/EPH)
Mission Editor	Adds the Mission Editor (tabular view) to the dock (default)
MW Nav Status	Adds the (legacy MW) Nav Status docklet to the dock
GPS Status	Adds the (legacy MW) GPS Status docklet to the dock
Radio Status	Adds the radio status docklet to the dock (default)
Battery Monitor	Adds the Battery Status docklet to the dock (default)
Telemetry Status	Adds the Telemetry Status docklet to the dock
Artificial Horizon	Adds the Artificial Horizon docklet to the dock
Direction View	Adds the Direction View (mag v. GPS) docklet to the dock
Flight View	Adds the Flight View docklet to the dock (default)
Vario View	Adds the Vario docklet to the dock
Radar View	Displays the Radar (inav radar / ADS-B) view
Flight Statistics	Display the flight statistic dialogue (also automatic on disarm)
Layout Manager > Save	Saves the current dock layout
Layout Manager > Restore	Restores a saved dock layout
Video Stream	Opens the (live_ video stream window
GCS Location	Displays the indicative GCS location icon

Help Menu

Item	Usage
Shortcut keys list	Displays the short cut keys list
About	Displays version, author and copyright information

1.3.3 Map and Mission Settings (2)

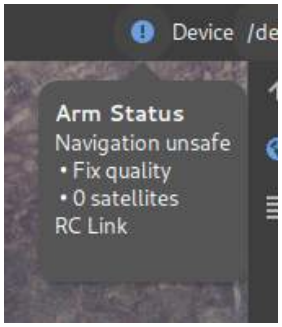
A number of different map provides are available. mwp offers the mapping library (`libchamplain`) defaults, Bing Maps (Bing Proxy) using a bespoke mwp API key, and user defined options, for example anonymous maps.

The zoom level may be selected from the control here, or by zooming the map with the mouse wheel. The **+Edit WPs** button enables mission edit mode (click on the map to create a WP, drag to move, right mouse button for properties). Graphical WP editing may be augmented by the table orientated mission table view, which allows additional control (altitude, speed, special functions, for example fly-by-home waypoints).

The "Active Mission" drop down supports inav 4.0+ multi-mission. There is also a **multi-mission manager** under the **Edit** menu.

1.3.4 Communications and telemetry settings (3)

There is a (blue "!" in the example) 'navigation safe' status icon. If this icon is shown (i.e. navigation is *unsafe*, then clicking on the item will provide more information:



The **Device** drop-down offers detected and pre-set (**Preferences**) devices for the FC / telemetry port. The device syntax is described the Device and Protocol definition article.

The **Protocol Selection** drop-down (here showing **Auto**) allows the user to provide a hint as to communication protocols available on **Device**. These are further described in the Device and Protocol definition article.

The **Connect / Disconnect** button connects / disconnects the displayed device.

The **auto** button causes mwp to automatically attempt to connect to the nominated device.

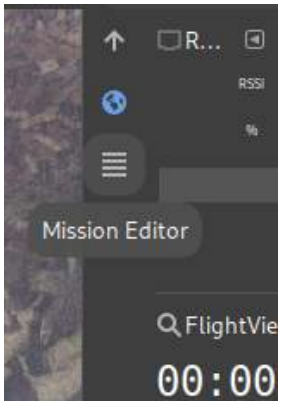
1.3.5 Map Area (4)

The map area displays the currently selected map at the desired zoom level. The map may be managed using familiar controls (drag, scroll wheel etc).

> **Graphics Requirement**
>
> The map API used my mwp requires OpenGL / 3D accelerated graphics. Performance with software rendering is likely to be disappointing and CPU intensive.

1.3.6 Dock Bar (5)

The **Dock Bar** contains essentially minimised **Docklets**, selected from the **View** menu. In the illustration, these are the **Vario** view, **Telemetry** statistics, and **Mission Editor**. Hovering the mouse over the icon will reveal its function:



1.3.7 Docklets (6)

Docklets are display items that can be docked, iconised, hidden or displayed in floating windows. See Dock Management. In the **main window screen shot** (left to right, top to bottom) we have:

- Radio status (RSSI or LQ)
- Artificial horizon
- Direction Status (Heading (Position Estimator/Compass v. GPS). Useful to diagnose mag EMF interference on multi-rotors).
- Flight View. General geo-spatial information.
- Battery status. Current usage is also shown when available.

1.3.8 Location (7)

The location (of the mouse pointer), user setting "pos-is-centre" for either mouse pointer or map centre, and format (**Preferences**).

1.3.9 FC Information (8)

Displays the firmware, version and build with API information, profile and flight mode.

1.3.10 Sensors and flight status (9)

- Follow : user setting "auto-follow". whether the map always displays the aircraft icon (required GPS).
- In View : Scrolls the map to keep the aircraft in view; otherwise the map is centred on the aircraft (requires GPS).
- Logger : Generate mwp logs (JSON format).
- Audio : user setting "audio-on-arm". Whether to "speak" status information.

The green / red bars show gyro / acc / baro / mag / gps / sonar sensor status. If a required sensor fails, a map annotation will be displayed, together with an audible alarm.

1.4 Dock Concepts and Usage

1.4.1 Dock Overview

The **dock**, items 5 and 6 in the main window guide provides an area for optional widgets.

This slighty outdated video that describes dock usage probably better than words might do.

Watch on Vimeo

> **Current Status**
>
> - The dock is now installed populated.
> - WP editor switch is enabled by default
> - There is now a graphical "favourite places" editor
> - The build system is no longer `make`

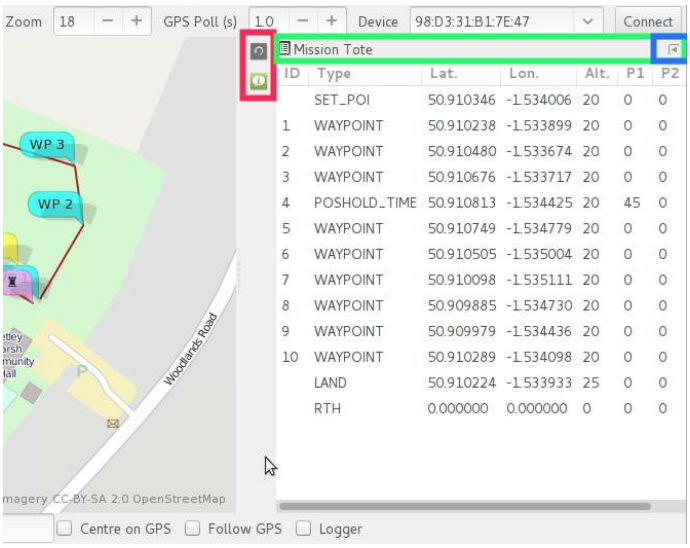
1.4.2 Dock Usage

mwp uses the GNOME Docking Library (`gdl`) to provide a dock capability. Items in the dock may be hidden, iconified or torn off into a separate window (that may then be returned to the dock). This section explains how use gdl in mwp. There is also an ancient short (silent) video illustrating dock actions.

- Load a mission into the mission tote
- Load the Nav Status into the dock bar
- Click the Nav Status icon to view nav status in the dock
- Move the Nav status view into a window
- Drag the Nav Status window back into the dock, selecting one of dock locations offered
- Minimise the Nav Status back to the dock bar (the little arrow)
- Reopen the Nav Status into the dock
- Hide the Nav Status
- Restore the Nav Status as a dock icon
- Reopen Nav Status in the dock.

> **Caveat updates**
>
> If a mwp update expands the dock adding new dock items, any previously saved dock layouts are invalidated, and you will have to manually recreate them. Fortunately, this is a rare occurrence.

The main dock controls are shown below:



> **This is an old image from c. 2015.**

- Highlight in **red** : the dock icons. Clicking on these will restore the window (either to the dock, or as a separate window)
- Hightlight in **green** : the dock item bar. Where multiple items are in the dock, the tab icon may be dragged to reposition the docked window. In also has a pop-up menu, that allows the item to be completely hidden (but recoverable from the View menu), and
- Highlight in **blue** : a iconify widget that will add the item to dock icon bar (the red highlight).

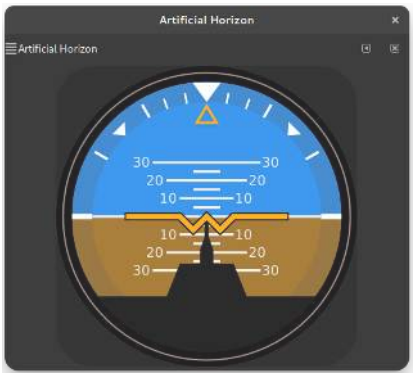
If the item bar icon (left-most in the green area) is dragged from the dock, the item will appear as a separate window. The detached window may be added back to the dock by dragging the window's “item bar” back into the dock, or added back to the dock icon bar using the iconify button (the left facing arrow to the right of the window's “item bar”. If the detached window is closed, then it becomes hidden, and may be reattached to the dock (as an iconified dock item) from the View menu.

> **Wayland Display API**
>
> When docklets are dragged around to reposition then, an "target" landing area is shown on the dock area. Unfortunately, the some older versions of the "modern" **Wayland** display manager breaks this in a way that only the upstream maintainers can fix. The workaround is to temporarily force X11 mode:
>
> ```
> # In a terminal
> $ GDK_BACKEND=x11 mwp
> # Drag dock items around
> $ mwp # items moved, Wayland again
> ```

1.4.3 Dock Items (Dockets)

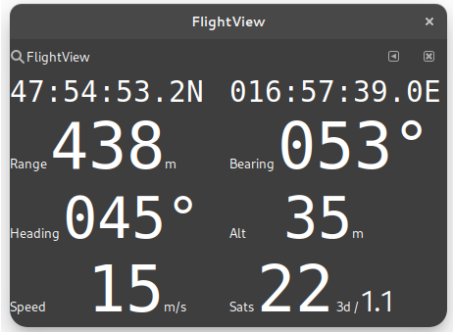
The following items are provided.

Artificial Horizon

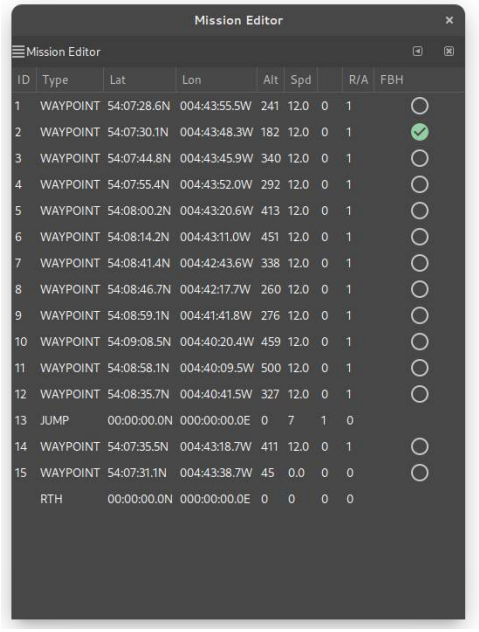


Direction View

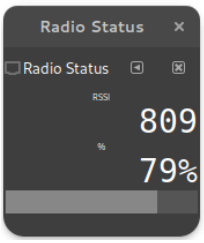
Flight View



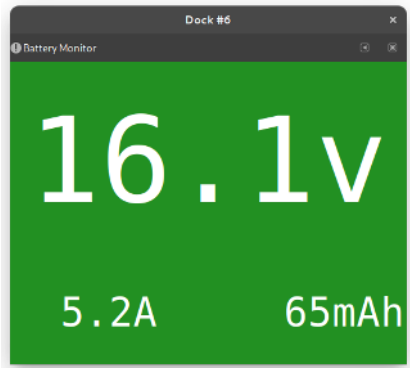
Mission Editor



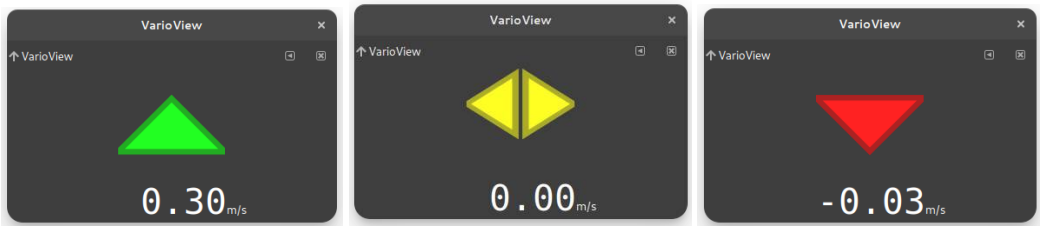
Radio Status



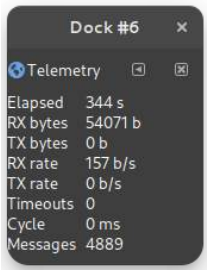
Battery Monitor



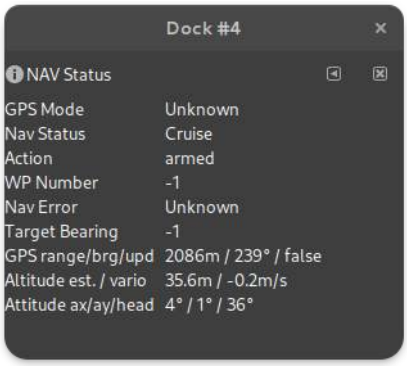
Vario View



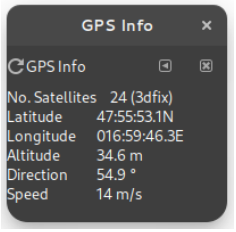
Telemetry View



MW Nav Status



MW GPS Status



1.5 Mission Editor

1.5.1 Overview

Another slightly outdated video, generic misson editing.

Watch on Vimeo

> **Current situation**
>
> - inav now supports 120 waypoints
> - inav now supports `SET_POI` and other multiwii waypoint types.
> - Delete from the map popup context menu behaves as it does in the tabular editor; it removes the RTH state.

Please also refer to the following articles that provide specific information for advanced inav topics:

- inav multi-missions
- inav fly-by-home

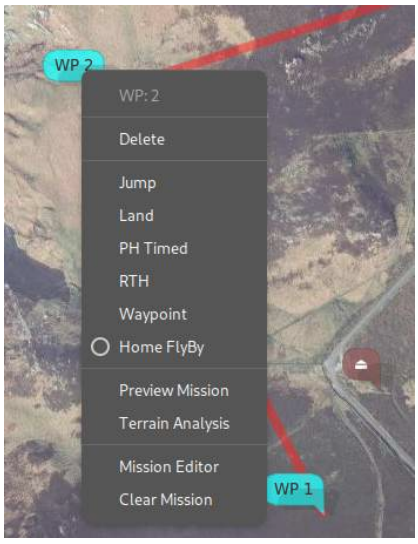
1.5.2 Map Features

Missions are edited on the map by enabling mission edit mode:

This will:

- Display a notional home location (brown icon)
- Allow new WPs to be created by clicking on the map
- Provide a context popup menu by right click on a WP icon

The context menu depends on the type of the current WP, for example:



The use of more advanced functions, for example setting parameter values, moving multiple WP, mission preview etc. requires the tabular mission editor

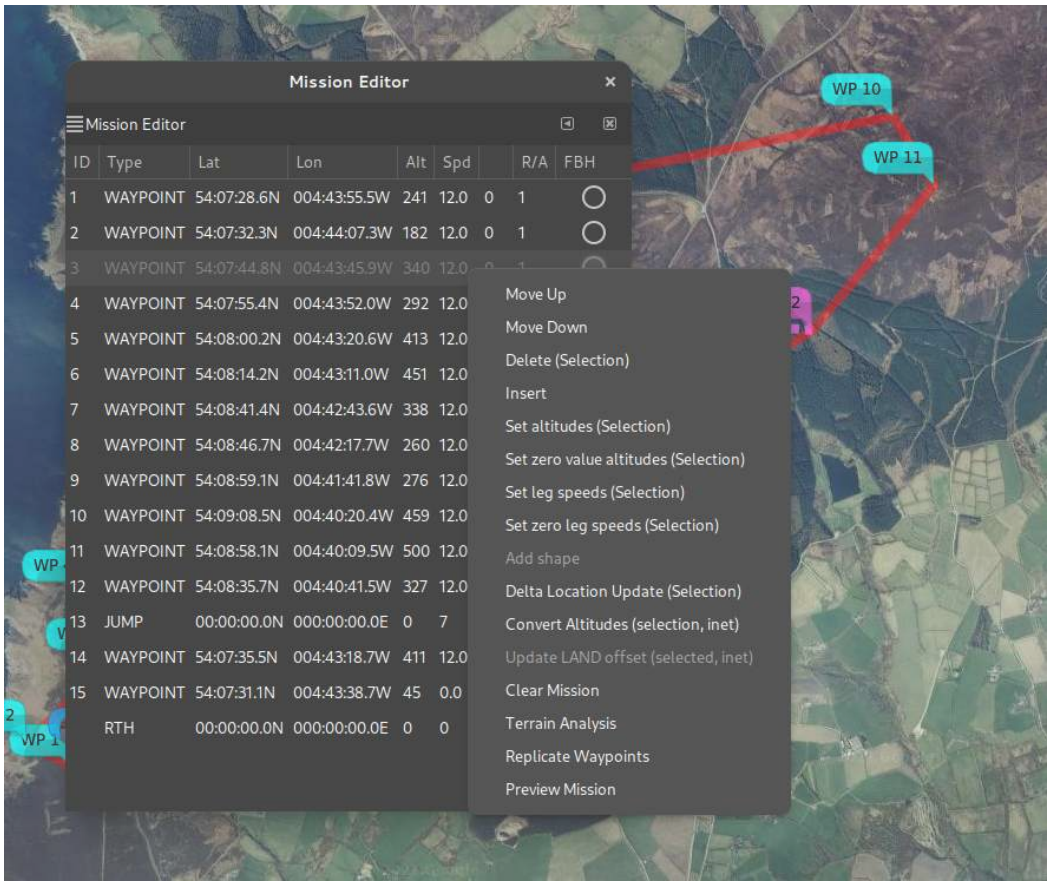
1.5.3 Mission Editor

The mission editor may be invoked from the dock or from a WP context menu.

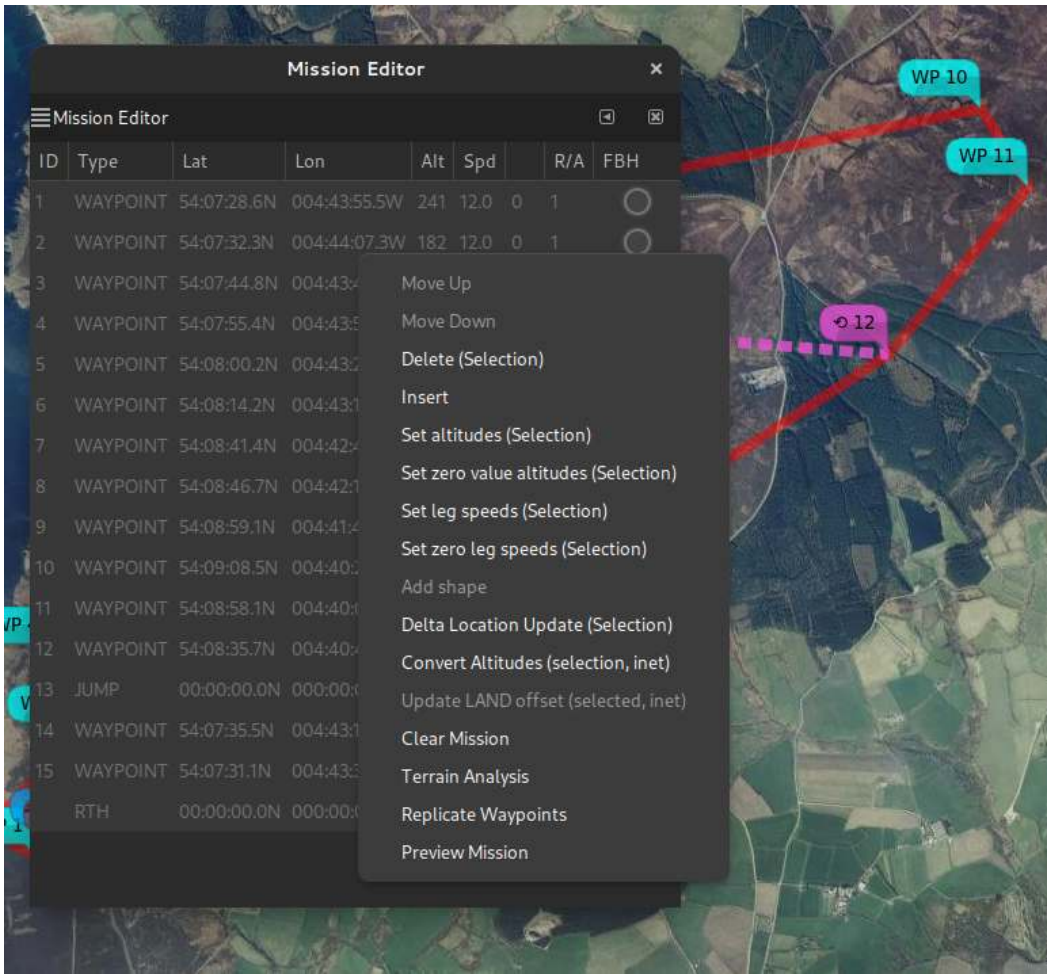
It provides the following functions:

- Create, delete, modify, reorder waypoints.
- Inline editing of parameters
- Context sensitive column titles for parameter editing
- Bulk updates (altitudes, speeds, position offsets)
- Automated path (polygon around a shape) generation.
- Terrain Analysis, automated altitude correction.

There is a right mouse context menu, the availability of items depending on whether zero, one or multiple items are selected.



Single selection context menu



Multiple selection context menu

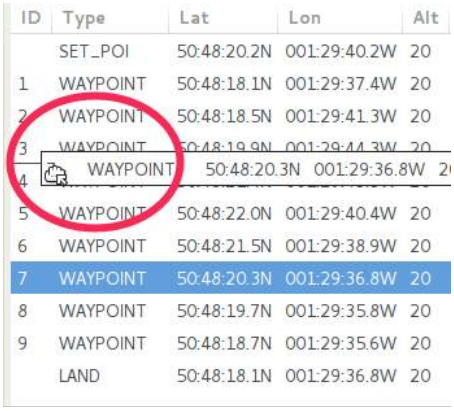
Common Operations

Many of the operations described below are shown in the videos, which probably provide a clearer explanation that any textual description could.

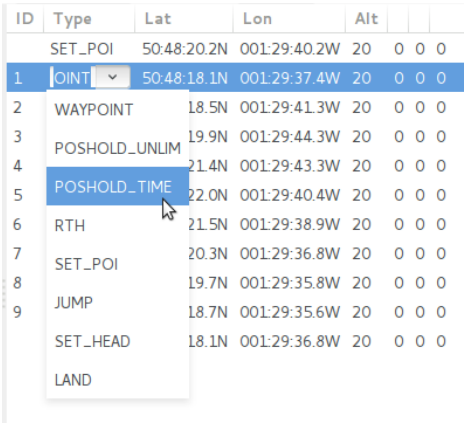
EDITING

Way points can be edited Mission Editor. When a row is selected, the column headers will change to indicate the data fields appropriate to the point type (in particular the “parameters” P1,P2,P3 whose interpretation is dependent on the point type.

- Position. The position of a way point may be changed by dragging the way point icon on the map or editing in the list.
- Order. The order of way points may be changed by either:
  - Using the “Move Up” and “Move Down” entries from the mission pop-up menu; or
  - Dragging the list item to the desired position. In order to drag, the entry must be 'grabbed' on the ID column. In that screen-shot (below), way point 7 is being dropped between way points 3 and 4.
  - At the end of the drop, the list and markers on the map will be re-ordered.



- Type. The way point type may be selected from a drop down menu embedded in the "Type" column of the list:



Once the type has been changed, default parameters for that way point type or action will be set. The type may also be set by a right mouse button click on the map symbol.

- Altitude. New points are created with the default altitude (from the "Preferences". Some basic validation is performed
- Parameters P1, P2 and P3. The parameters P1,P2 and P3 are integer values that have a meaning specific to the way-point type or action. For example, for action type of JUMP, P1 is the point to which to jump, and P2 is the number of repeats. This usage is documented in the inav wiki.
- Delete. The delete action will delete the selected (highlighted) way point(s). If no way point is selected, this option has no affect.

ADD SHAPE

If a SET POI point is added to the mission, (there may also be other extant way-points), this option will display a dialogue to enter the number of points in a shape, the radial distance (from the SET POI to each point), an offset angle and the direction of rotation. i.e this defines a polygon around the POI.

- The offset is relative to North. If you wanted the lines to be horizontal / vertical, specify an offset of 45° for a square.
- Shape points are appended to any extant mission points, and the shape tool may be invoked multiple times, for example to create 'concentric' circles.
- The `SET_POI` point may be deleted, unless you really want `SET_POI` functionality.

LOCATION UPDATES

Bulk location updates may be applied to selected waypopints.

If an item if left black (or 0), then no adjustment is applied to that axis. Offsets are in metres, regardless of the user's preference distance unit.

SPEED AND ALTITUDE UPDATES

Bulk speed and altitude updates may be applied to selected waypopints.

CONVERT ALTITUDES

From inav 3.0, inav supports both relative and AMSL altitudes. This, and the mwp features for managing this, are described in a separate article

REPLICATE WAYPOINTS

This item facilitates the cloning of waypoints. Since inav now supports the JUMP waypoint type, this option is less useful.

PREVIEW MISSION

"Flys" an aircraft icon around the mission; this may be useful for predicting the behaviour of embedded JUMPs.

CLEAR MISSION

The Clear Mission option clears the mission. There is no confirmation, so be sure you really want to do this.

1.5.4 Advanced WP types / Video Tutorials

JUMP, POSHOLD TIMED, LAND

Video example setting up JUMP, POSHOLD TIMED and LAND waypoints.

Watch on Youtube

SET_POI, SET_HEAD as mission elements

Video example SET_POI and SET_HEAD (real mission usage).

Youtube video

Mission Preview

Video example of preview for a complex (multiple jumps, timed POSHOLD) mission (preview from the first video).

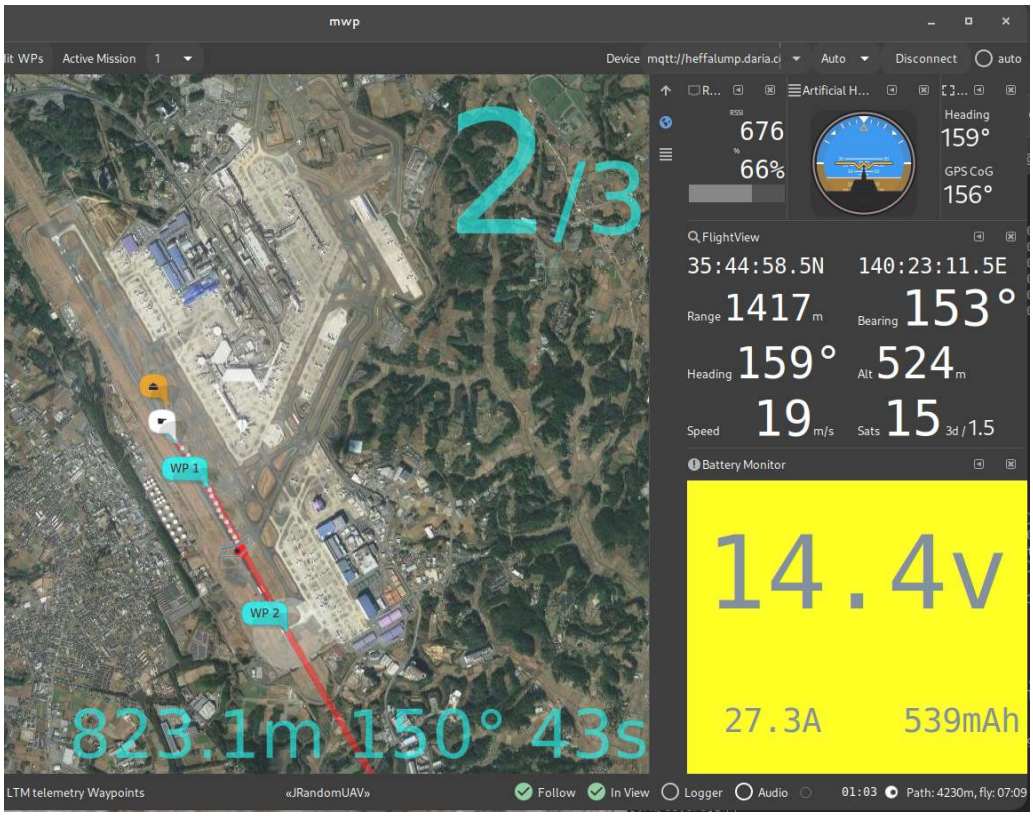
Watch on Youtube

1.6 Ground Control Station Features

1.6.1 GCS Usage

OSD information

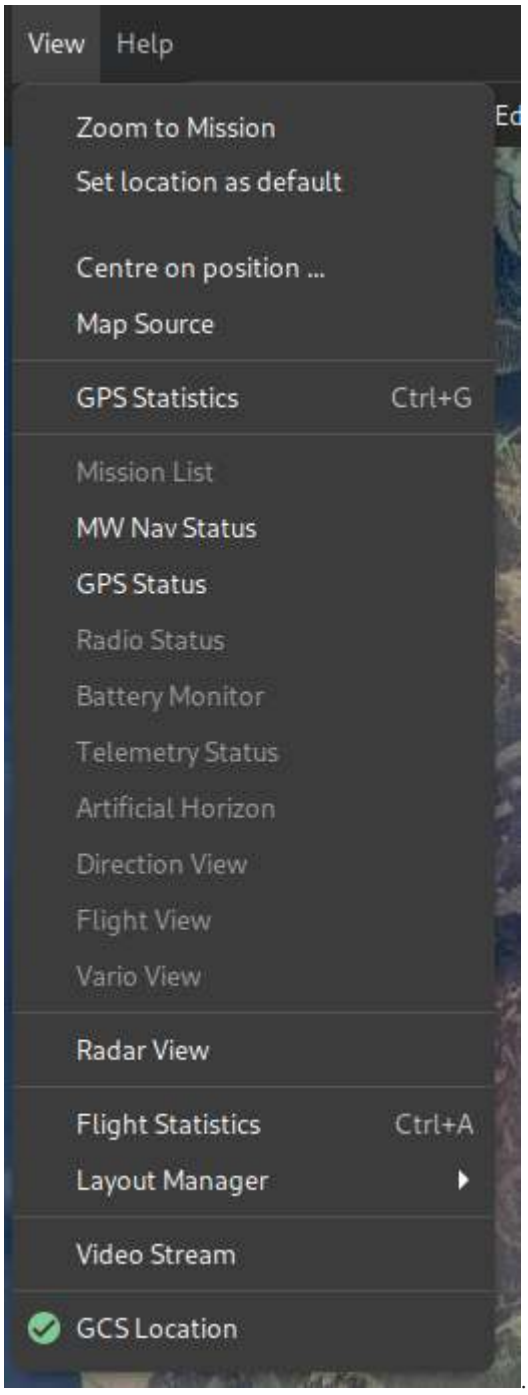
When flying waypoints, if the mission is also loaded into mwp, mwp can display some limited OSD information.



Various settings (colour, items displayed etc.) are defined by settings.

GCS Location Icon

A icon representing the "somewhat static" GCS location can be activated from the **View/GCS Location**" menu option:



By default, it will display a tasteful gold star which one may drag around. It has little purpose other than showing some user specificed location (but see below).

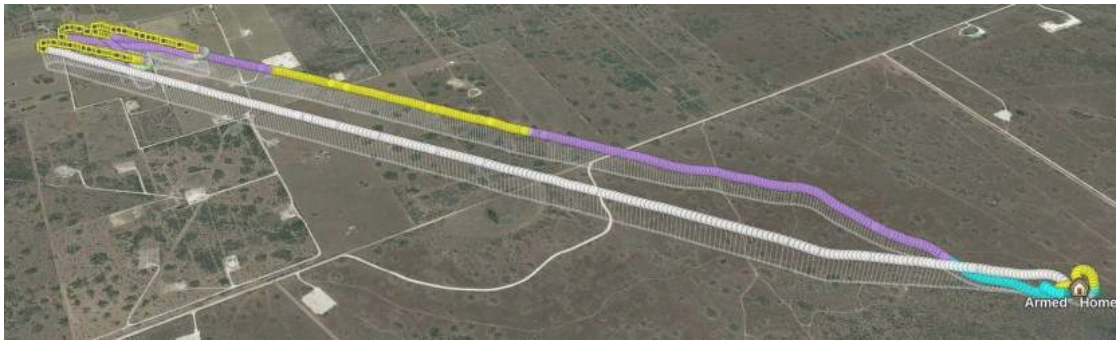
If you don't like the icon, you can override it by creating your own icon in `~/.config/mwp/pixmaps/` (as one can for *any* mwp specific icon), with the same file name as the default in `$prefix/share/mwp/pixmaps/`; i.e. override `$prefix/share/mwp/pixmaps/`

`gcs.svg` with `~/.config/mwp/pixmaps/gcs.svg` (which could be a PNG, vice SVG; we're a real OS and file "extensions" are an advisory illusion).

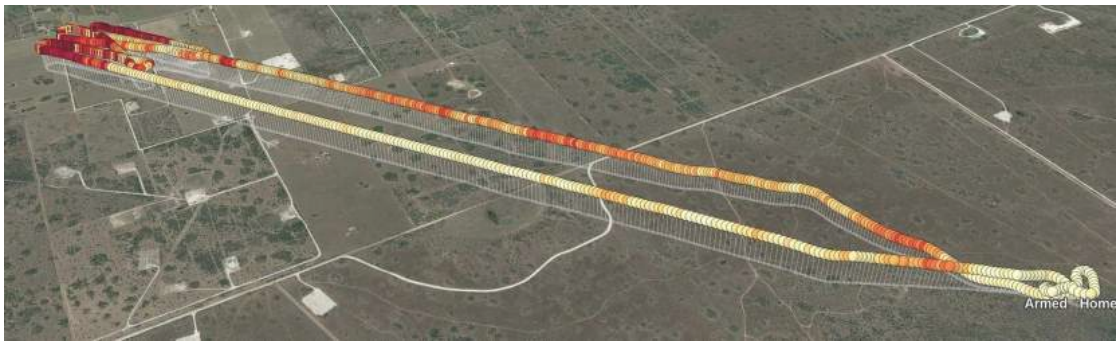
- If `gpsd` is detected (on `localhost`), then the position will be driven by `gpsd`, as long as it has a 3D fix.
- The one usage is when inav-radar is active; if the GCS icon is enabled (either by manual location or driven by `gpsd`), then rather than being a passive 'GCS' node, mwp will masquerade as an 'INAV' node and advertise the GCS (icon) location to other nodes. This implies that you have sufficent LoRa slots to support this node usage.

1.7 Replay Tools

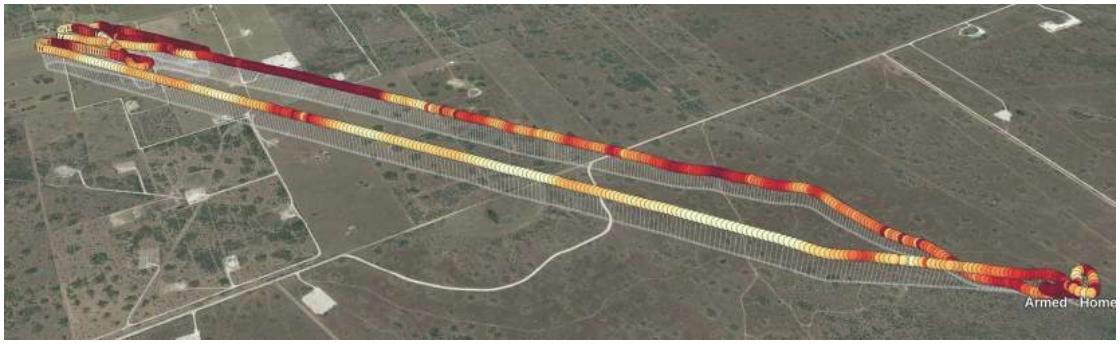
in order to replay log files, mwp has a number of external dependencies, in particular the **flightlog2x** `fl2ltm` tool provided by the bbl2kml repository. As well as providing replay tools for mwp, you also get the facility to generate attractive animated KML / KMZ files for visualisation in **google-earth**.



Flight mode view



RSSI view



Efficiency view

> **Analysis**
>
> The RSSI view shows why the aricraft is playing "failsafe ping-pong" at the right extreme of flight

Binary packages are provided for many popular platforms.

1.7.1 Blackbox replay

In order to replay blackbox logs, you additionally need inav blackbox tools, specifically `blackbox_decode`). Binary packages are provided for many popular platforms. The minimum required version in 0.4.4, the latest release is recommended.

1.7.2 OpenTX / EdgeTX logs (Smartport)

OpenTX enables the storage of Smartport telemetry logs on a transmitter's SD-Card. These logs contain all the (Frsky) telemetry information transmitted from the flight controller.

mwp can replay these logs, in a similar manner to the replay of Blackbox or mwp logs, albeit with less detail and typically at lower data rates.

- Enable Frsky telemetry on the FC
- Enable telemetry logging on the TX
- Post flight, transfer the log from the LOGS directory of the SD card to your computer
- Replay the log using the Replay OTX Log (or Load OTX Log for a "fast-forward" rendering)
- Limited support is available of TX logs from Ardupilot.

No addition software requirements.

1.7.3 BulletGCSS Logs

Requires that mwp is built with MQTT support.

No addition software requirements.

1.7.4 Ardupilot logs

Requires Ardupilot's mavlogdump.py and dependencies.

1.7.5 mwp JSON logs

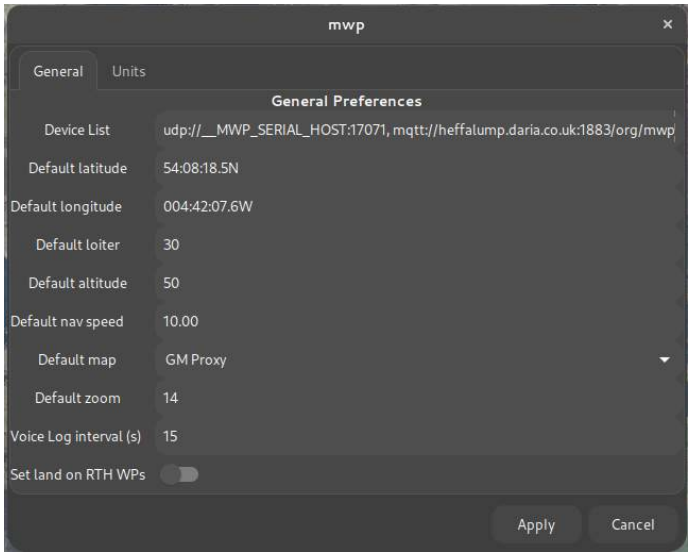
No addition requirements.

1.8 Miscellaneous Ui Elements

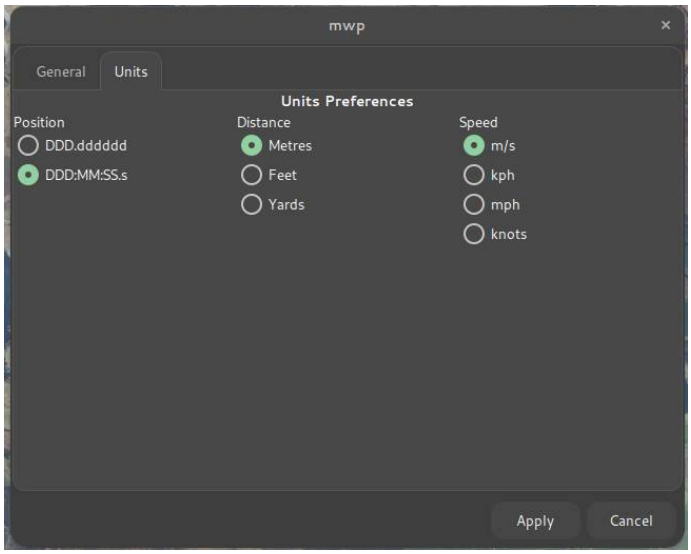
1.8.1 Preferences

The "Edit > Preferences" menu provides a UI for some `gsetting` / `dconf` settings. The setting here applied immediately if 'Apply' is clicked.

General Preferences



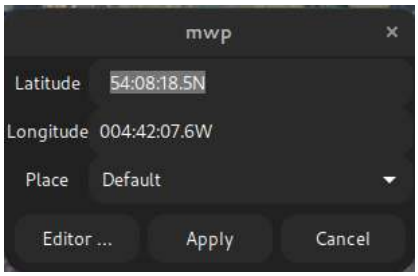
Units Preferences



Unit preferences should be instantly reflected in the UI when 'Apply' is clicked.

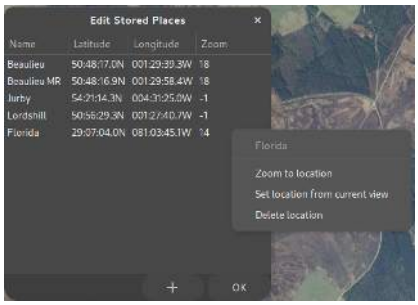
Favourite Places

mwp maintains a list of favourite places, from "View > Centre on Location" menu item.



The "Place" combo menu holds all places defined in `~/.config/mwp/places` (see the configuration reference).

For convenience, clicking the 'Editor ...' button will load the "Places Editor".



- New items are added with the **+** button.
- Locations can be edited in place
- The context (right mouse button) menu:
  - Zoom to location : Zooms to the place
  - Set location from current view : Sets the location to the centre of the current map view
  - Delete location : Deletes the location without question.
- **OK** Saves the locations to `~/.config/mwp/places`
- Closing using the window manager **X** icon closes without saving.

1.9 Mission Elevations

1.9.1 Overview

Prior to inav 3.0, mission altitudes are relative to the HOME (arming) location, which is not part of a mission definition. As a result, the pilot has to be ensure by some other means that the mission will clear any raised elevations on the mission path. For inav 3.0, missions may be either relative to home or absolute (above a datum, see below).

mwp includes a `mwp-plot-elevations` tool that performs mission and terrain analysis. Prior to 2021-05-03, this was provided by a ruby script in `mwptools/samples`; since 2021-05-03 there is a Go program (in `mwptools/mwp-plot-elevations`) which is an enhanced version, and supports inav 3.0 absolute altitude missions. If you're running an older version of **mwp**, or you haven't installed the Go compiler, you can use the older, less functional ruby version, but the Go version is recommended as:

- It supports inav 3.0 absolute altitude waypoints
- It can update LAND waypoints to offset the difference between the home ground elevation and the LAND WP ground elevation.
- It's much faster
- Its usage is compatible with the deprecated ruby version.
- Bug fixes and improvements

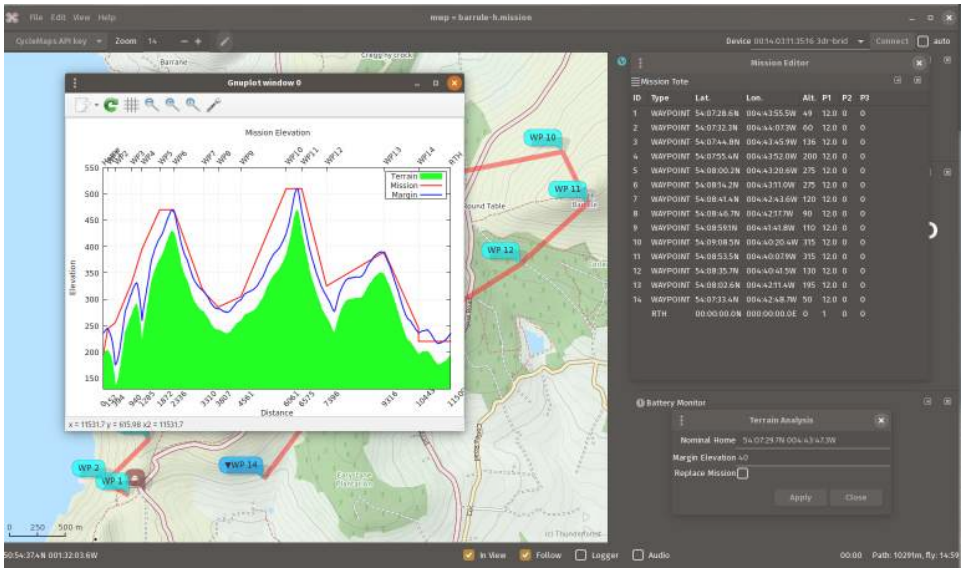
Both the ruby application and the Go application are platform independent and can be used without mwp for mission terrain analysis.

> **Obsolescence Note**
>
> Prior to 2021-05, the ruby version was installed as `mwp-plot-elevations.rb`; now it's installed as plain `mwp-plot-elevations` in order that the superior Go version is a drop in replacement.

`mwp-plot-elevations` can rewrite the mission file with new elevations to provide a specified ground clearance.

1.9.2 Video Tutorial & UI integration

From of 2018-12-06, `mwp-plot-elevations` is integrated into the `mwp` application.



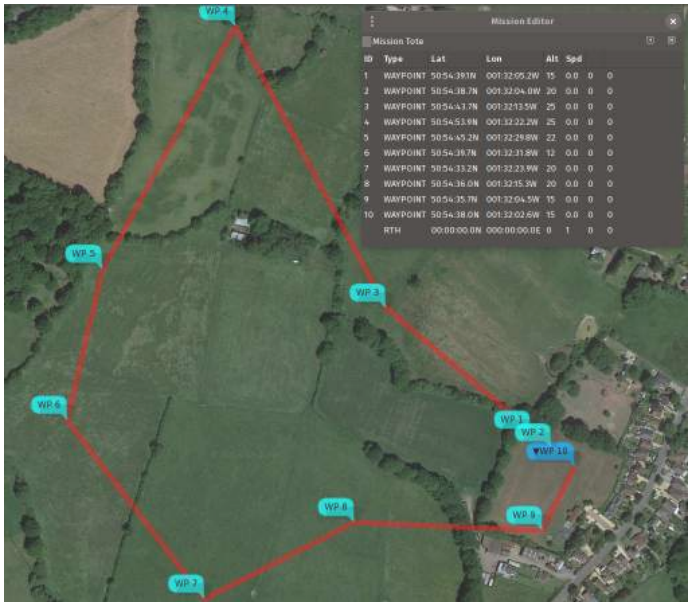
There is a video tutorial.

> **Obsolescence Note**
>
> The video uses the older ruby application, but that doesn't really affect *basic* functionality.

Watch on Youtube

1.9.3 Sample output

Given the mission shown below:



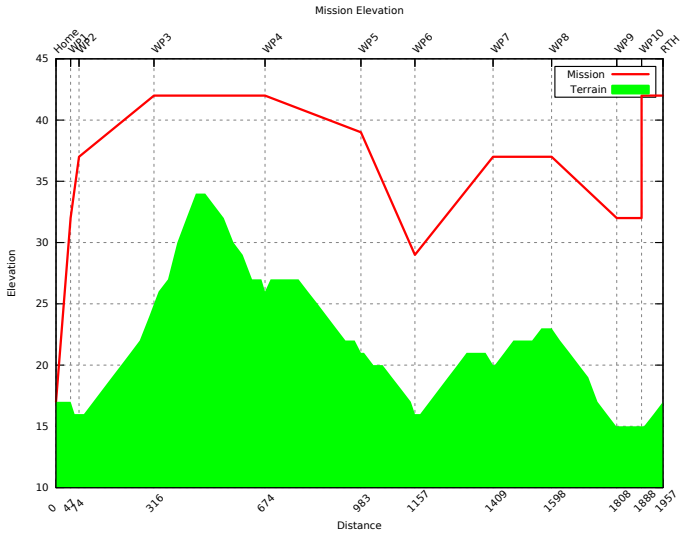
and knowing that the land rises to the north and west, we can check that we do indeed have adequate clearance with the planned route and elevations:

```
# for decimal '.' locales
$ mwp-plot-elevations -- home 50.9104826,-1.5350745 --plotfile profile.svg  west_field.mission
# for decimal ',' locales
$ mwp-plot-elevations --home "50,9104826 -1,5350745" --plotfile profile.svg  west_field.mission
```

where:

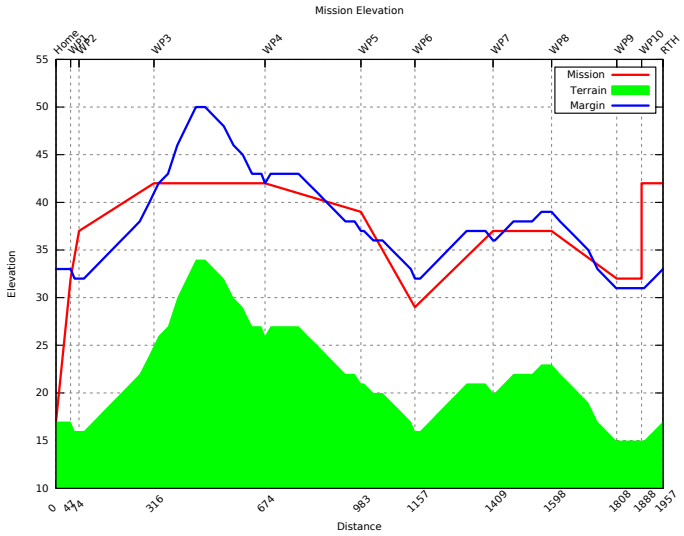
- `west_field.mission` is the MW-XML mission file (via mwp, ezgui, mission planner for inav or impload)
- the `--home lat,lon` option defines the home position (which may also be set by the environment variable `MWP_HOME`), the command line having preference.
- The graphical output is `profile.svg`, via the `--plotfile` option.

The result from this command is an SVG file, which can be displayed with common image tools (`eog`, ImageMagick `display` et al). It can also be converted to a raster image using e.g. `rsvg-convert`); a sample is shown below:



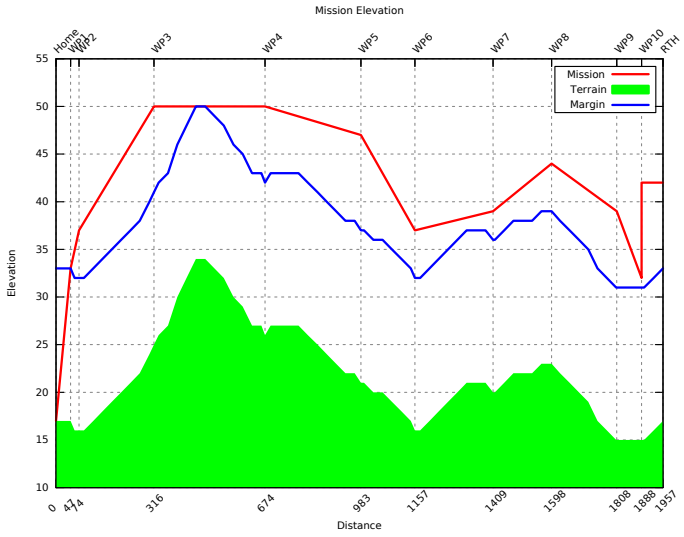
The red line represents the planned mission altitudes (which are defined relative to the estimated home location), and the green area represents the terrain. As we can see, we clear the hill (and other terrain), but cannot guarantee that we have LOS to lowest point of the mission, or that we're clear of the trees.

We can also specify a "clearance" option, in the image below this was set to 16m. Where the blue line is above the red line, one should review that the mission elevations are adequate.



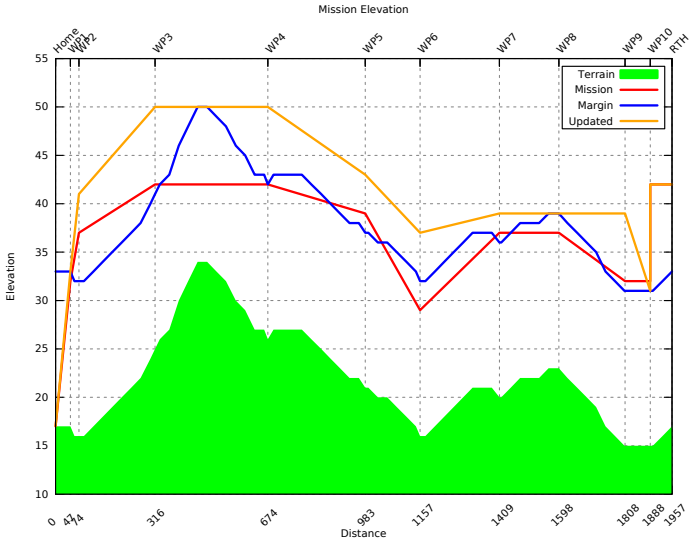
1.9.4 Creating a new mission file

It is also possible (see command line options below) to write out a new mission that takes into account the clearance (`margin` parameter). If we then plot this new mission file, we can see that we are at least `margin` (in this example 16m) distance clear of the terrain.



Note that the original mission elevations are still taken into account. We can also ignore these, so we end up the absolute clearance distance above the terrain.

```
$ mwp-plot-elevations nm_west_field.mission --output /tmp/p1.mission --no-mission-alts
```



1.9.5 Dependencies

The `mwp-plot-elevations` has **NO** dependency on mwp or Linux / FreeBSD, it can just as easily be run on MacOS or even MS Windows. It does however has some dependencies:

Go version

- Go compiler (1.13 or later)

Ruby version

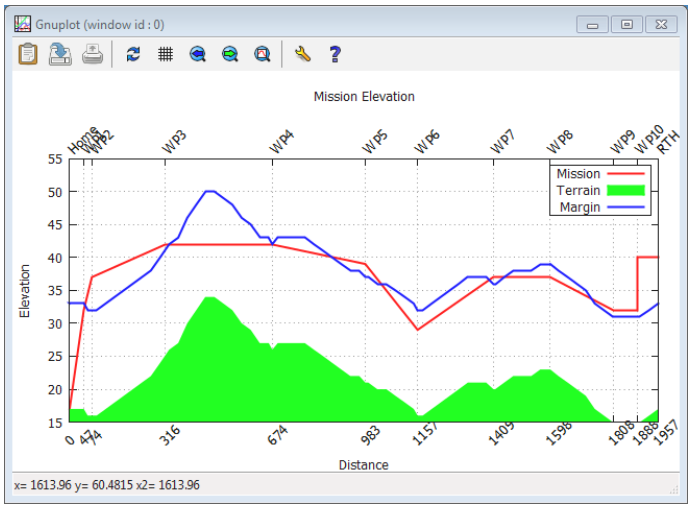
- ruby (2.0 or later)
- ruby 'gems' (libraries)
  - nokogiri
- gnuplot

`gnuplot` is easily provided (by your distro or from a binary download), and the `nokogiri` dependency is also easily satisfied by either the distro or Ruby's `gem` command:

```
$ apt install ruby-nokogiri
### or ###
> gem install nokogiri
## mwp Windows / Cygwin
$ cyg-apt install ruby-nokogiri
```

Using the package manager is recommended for non-proprietary operating systems.

On all operating systems, the terrain graph is also plotted interactively, regardless of whether the `-p` (save SVG plot) option has been specified. The following shows the UI on Windows (it's pretty much the same on other OS).



1.9.6 Caveats

- 3rd party terrain data is not guaranteed, either as to its absolute accuracy, nor to its coverage.
- Terrain data does not take into account other obstacles (trees, buildings, power lines etc).
- The tool does not faithfully model the vehicle motion. As multi-rotor and fixed-wing have different climb behaviours, this would be quite complex.
- RTH altitude has to specified if you wish to model it, and assumes 'AT LEAST' behaviour.

1.9.7 Datums

Digital elevation services can use the WGS84 Ellipsoid or "sea level"; survey maps typically use AMSL (Above Mean Sea Level); GPS can report either or both of WGS Ellipsoid and above MSL (mean sea level). The "sea level" used by Bing Elevations is computed from a magnetic anomaly / gravity database and may not be the same as the AMSL "sea level" used by the survey. **Caveat User**.

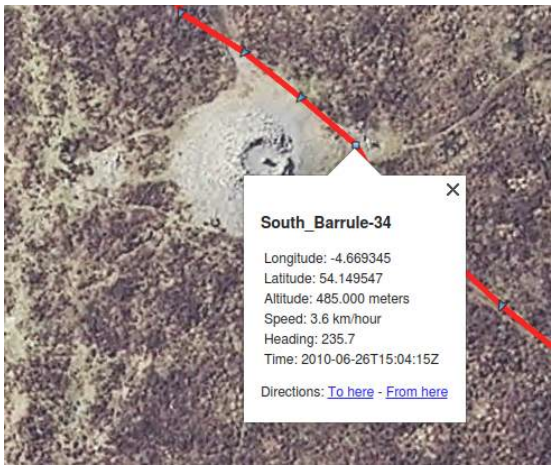
- mwp currently uses Bing "Sea level" to obtain elevations. The user should apply a suitable margin.
- inav firmware uses the GPS' AMSL value, so inav and mwp are consistent on this.
- The inav configurator uses Bing's Ellipsoid values (by default, it can be changed).

Due to the granularity of the AMSL grid used by GPS and the gravity based Bing Sea Level, there may be a significant difference between ASML, "sea level", WGS84 Ellipsoid and Survey heights, for example, for a test point of 54.149461 -4.669315 (summit of South Barrule, Isle of Man):

- Google Earth : 470m
- Ordnance Survey (OS) Map (official survey): 483m
- Bing Ellipsoid (Configurator): 526m
- Bing "Sea Level" (mwp): 470m

1.9.8 So who's right?

Many years ago, I took a GPS up South Barrule.



It reads 485m, this pretty much agrees with the OS (Survey) height (AMSL). So the real issue is with the DEM available online (either Bing or Google). The 'sea-level" height DEM reports **for this location** is c. 13m below Ordnance Survey AMSL value whilst the WGS84 ellipsoid value is 43m above the OS AMSL value.

1.9.9 Command line help and options

```
$ mwp-plot-elevations --help
Usage of mwp-plot-elevations [options] missionfile
  -dump
        Dump  internal data,exit
  -home string
        home as DD.dddd,DDD.dddd (default "50.910476,-1.535038")
  -margin int
        Clearance margin (m) (default 16)
  -no-mission-alts
        Ignore extant mission altitudes
  -no-plot
        No interactive plot
  -output string
        Revised mission file
  -plotfile string
        SVG graph file
  -rth-alt int
        RTH altitude (m) (default 25)
  -upland
        Update landing elevation offset
```

Note that Go considers `-foo` and `--foo` to the equivalent. The ruby script requires the `--` notation.

1.9.10 Configuration File

As well as specifying options such as home location, clearance margin and RTH altitude on the command line (or as an environment variable), some or all of these options may be set in a configuration file.

`mwp-plot-elevations` looks for options in one of the following (in order) `./.elev-plot.rc` (i.e. current directory), `$HOME/.config/mwp/elev-plot`, and `$HOME/.elev-plot.rc`. The configuration file is a simple text file containing `key=value` pairs. Blank lines and lines beginning with `#` are ignored; the following example illustrates the recognised keys. Note that `$HOME/.config/mwp/elev-plot` is the preferred location, as this is also used by `mwp` to populate its graphical dialogue to launch the analysis tool.

```
# settings for mwp-plot-elevations
margin = 16
home = 50.910476,-1.535038
# for ',' locales
# home = 50,910476 -1,535038
rth-alt=25
# 'sanity' is the home -> WP1 distance check; default if not set here is 100m
sanity = 200
```


1.9.11 Usage Examples

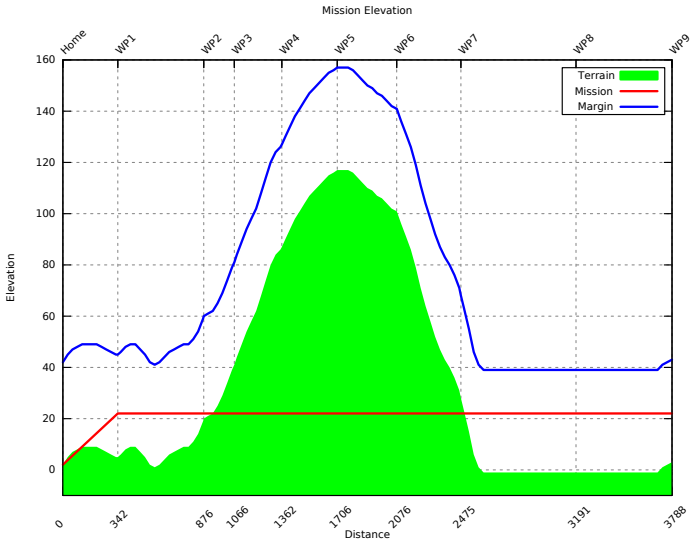
```
# Interactive plot, using the above configuration file:
$ mwp-plot-elevations nm_west_field.mission

# Interactive plot. save SVG file
$ mwp-plot-elevations --plotfile /tmp/mission.svg nm_west_field.mission

# Interactive plot. save SVG file, rewrite mission file
$ mwp-plot-elevations --plotfile /tmp/mission.svg --output new_west_field.mission nm_west_field.mission

# Interactive plot. save SVG file, rewrite mission file, override clearance margin (20m)
$ mwp-plot-elevationsb --plotfile /tmp/mission.svg --outout new_west_field.mission --margin 20 nm_west_field.mission

# Interactive plot. save SVG file, rewrite mission file,
# override clearance margin (20m), reduce RTH altitude (22m)
$ mwp-plot-elevations --plotfile /tmp/mission.svg --output new_west_field.mission --margin 20 --rth-alt 22 nm_west_field.mission
```

Another contrived example ... create a mission in Google Earth (tied to ground), save as KMZ, convert to MWXML mission file with impload (0 altitude). Use `mwp-plot-elevations.rb` to calculate a safe mission.

KMZ planned in Google Earth

Conversion tools

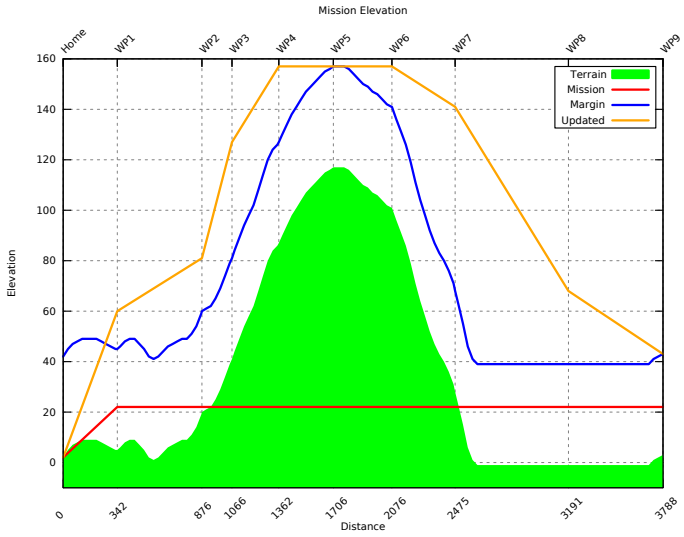
```
# convert the saved KMZ file to a MWXML mission file
$ impload convert  /tmp/IOM.kmz /tmp/perwick.mission

# Verify the elevations and clearance with plot-elevations.rb
$ mwp-plot-elevations.rb -h  54.068826,-4.735472   -m 40 /tmp/perwick.mission
```

Looks OK (well, apart from the flying through the hill, due to imload's default altitude of 20m).

If we specify that a new mission file be generated (`--output`), the updated mission is also plotted, and we can see that this clears the hill.

```
mwp-plot-elevations --home  54.068826,-4.735472 --margin 40 --output /tmp/perwick-ok.mission /tmp/perwick.mission
```



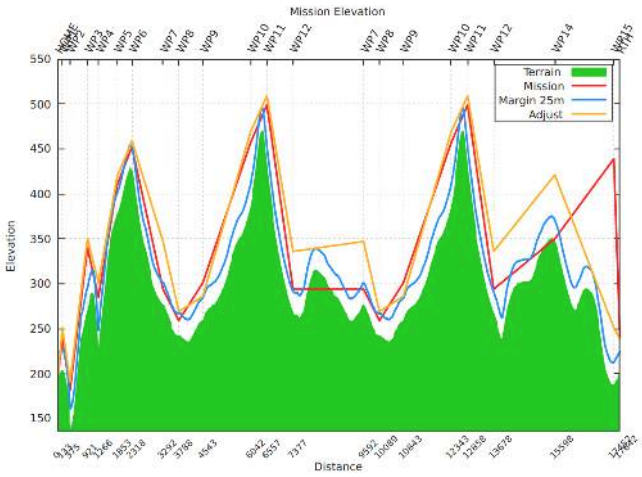
It's not yet perfect, we could be more aggressive in reaching just the clearance altitude, but we clear the hill!.

1.9.12 Climb and Dive Angle Report

Mission used for climb /dive example

As of 2021-06, it's also possible to get climb and dive angles for the calculated mission. Before I added the WP12 => WP7 jump in the mission shown below, it was almost OK; below the desired clearance in a couple of places and just failing to clear the hill at WP15. After adding the JUMP, it hits the terrain pretty conclusively between WP12 and WP7. The modified mission is interesting, as it has to adjust the WPs within the JUMP for the worst case (so the WP7, the second pass is definitive).

The final result:



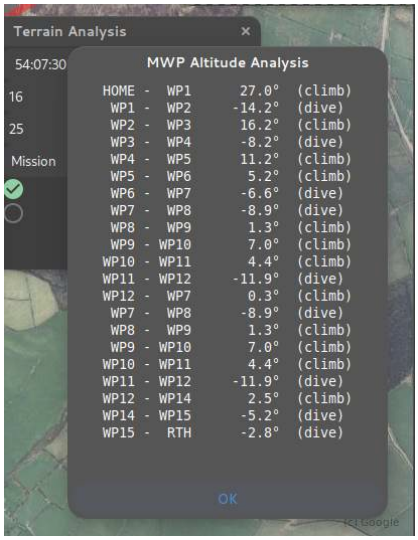
We also get a climb / dive report, currently to STDOUT and `$TMP/mwpmission-angles.txt` (tab separated for easy analysis).

```
$ mwp-plot-elevations --margin 25 -no-mission-alts --output /tmp/n.mission \
 --home 54.125205,-4.730322 -rth-alt 40 mwp/missions/IoM/barrule-jump.mission
HOME -  WP1  21.3°  (climb)
 WP1 -  WP2 -13.9°  (dive)
 WP2 -  WP3  16.2°  (climb)
 WP3 -  WP4  -8.1°  (dive)
 WP4 -  WP5  11.4°  (climb)
 WP5 -  WP6   4.9°  (climb)
 WP6 -  WP7  -6.6°  (dive)
 WP7 -  WP8  -8.9°  (dive)
 WP8 -  WP9   1.3°  (climb)
 WP9 - WP10   7.0°  (climb)
WP10 - WP11   4.4°  (climb)
WP11 - WP12 -11.9°  (dive)
WP12 -  WP7   0.3°  (climb)
 WP7 -  WP8  -8.9°  (dive)
```

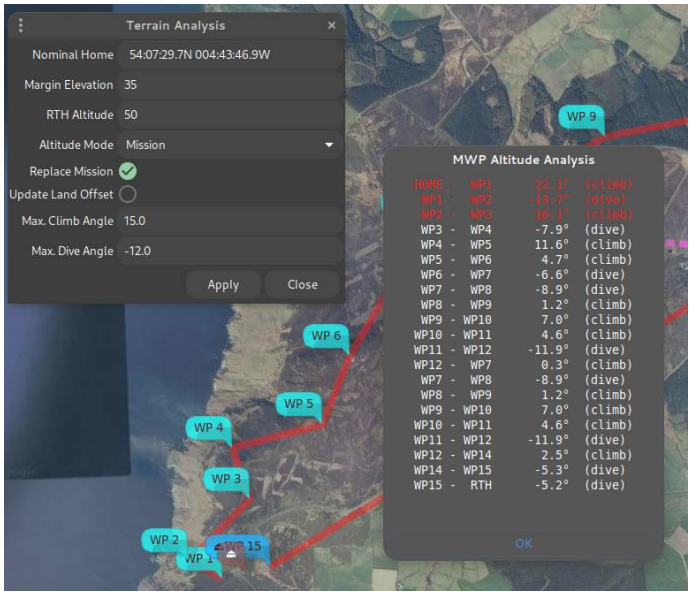


```
  WP8 -  WP9   1.3°  (climb)
  WP9 - WP10   7.0°  (climb)
 WP10 - WP11   4.4°  (climb)
 WP11 - WP12 -11.9°  (dive)
 WP12 - WP14   2.5°  (climb)
 WP14 - WP15  -5.2°  (dive)
 WP15 -  RTH  -3.6°  (dive)
```

If you run **mwp-plot-elevations** via **mwp**, the information is presented in a separate window.

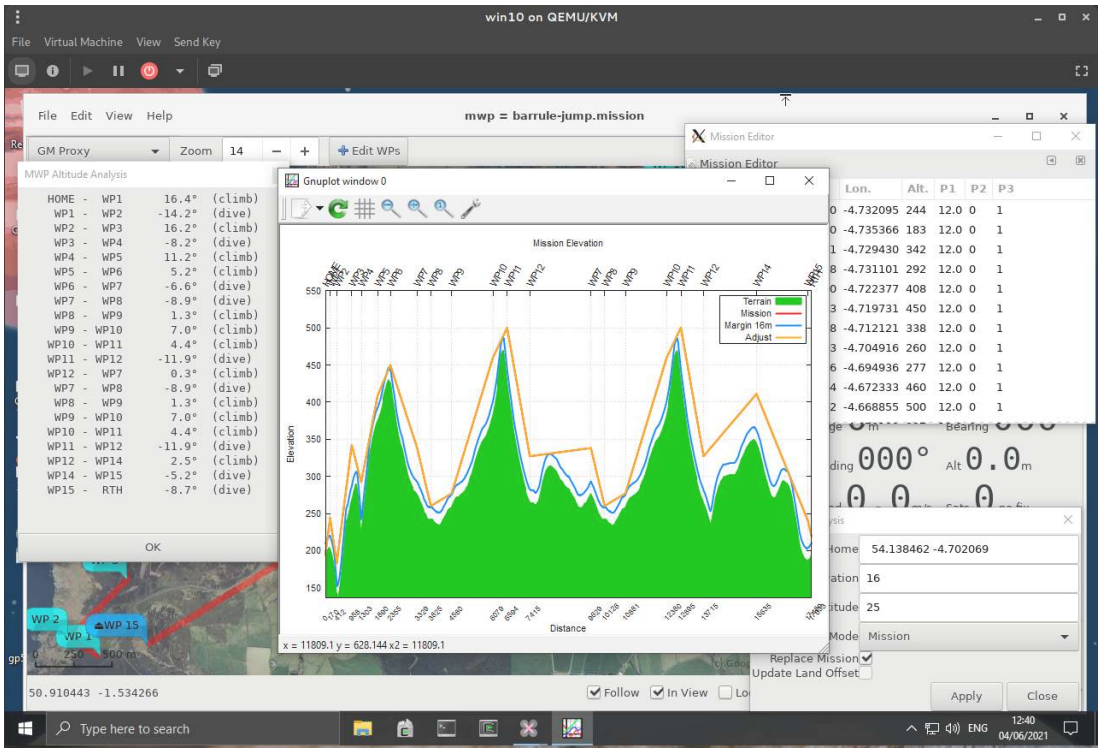


mwp can also highlight any legs that exceed user-defined (not 0) climb and dive angle limits. However, it's up to you to work out the best solution.



The steep hill and valley at the start are just too much here; best to reroute.

1.9.13 Finally ....



For Window 10 / Cygwin, you probably need to have the Windows `gnuplot`, vice the Cygwin version.

1.10 "Serial" device support

mwp supports a number of different data transports for "serial" protocols:

- Wired serial devices (USB TTL (VCP) etc.)
- Bluetooth
- IP (UDP and TCP)
- "Special" (e.g. BulletGCSS via the MQTT protocol).

Each of these requires a specific device name and *may* require a protocol selection.

1.10.1 Serial devices

Serial devices are defined by the operating system device node name and optionally include an embedded baud rate, for example:

```
# Linux, USB seral
/dev/ttyACM0
# Linux, USB serial with baud rate
/dev/ttyUSB0@57600
# Linux, RFCOM Bluetooth
/dev/rfcomm1
```

```
# FreeBSD
/dev/cuaU0
```

1.10.2 Bluetooth

Bluetooth may be specified by either an `rfcomm` device node (`/dev/rfcommX`) or by the device address (`BD_ADDR` , Linux only):

```
# BT RFCOMM device node
/dev/rfcomm1
/dev/rfcomm1@57600
# BT device address (note here baud rate is immaterial)
35:53:17:04:07:27
```

Serial permissions

It is necesary for the user to have read / write permission on serial devices. The installation guide provides instructions.

1.10.3 IP protocols (UDP and TCP)

mwp uses a pseudo-URL format for TCP and UDP connections `udp://host:port` and `tcp://host:port` (where `host` is either a hostname or an IP address as required).

Typically on one side of the connection you'll provide a hostname /IP and on the other you won't (as it can get the peer address from the first data packet).

Assuming the required UDP port is 43210

if mwp is the "listener" (doesn't need, *a priori*, to know the address of peer), set the "Device" to:

```
udp://:43210
```

i.e. the host part is empty.

If the remote device / application is the listener, and we know its IP address; in the following example "192.168.42.17", set the "Device" to:

```
udp://192.168.42.17:43210
```


Note that for TCP, mwp only supports the latter form (it expects to be the TCP client).

1.10.4 Special Cases

MQTT / BulletGCSS

See the mwp's MQTT support article for a detailed description of the URI format:

```
mqtt://[user[:pass]@]broker[:port]/topic[?cafile=file]
```

WSL UDP bridge

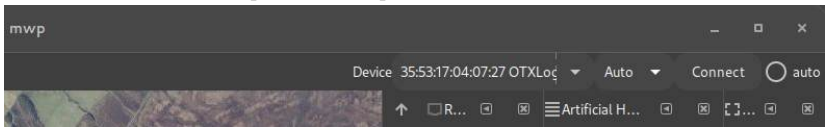
As WSL does not support serial connections, mwp provides a bespoke serial / UDP bridge using the pseudo-device name `udp://__MWP_SERIAL_HOST:17071`. See the WSL article for more detail.

1.10.5 Multi Protocol selection

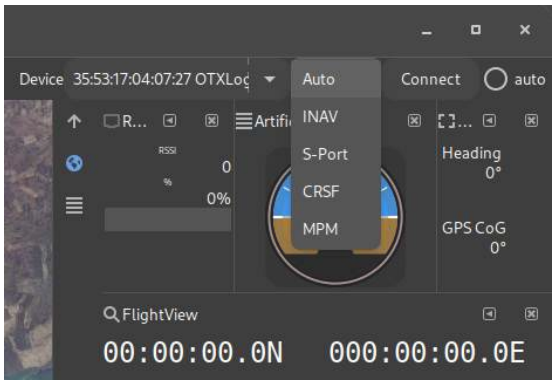
Overview

From 4.317.587 (2021-11-21), mwp does away with some of the weirdness around serial protocols (e.g. having to separately specify `--smartport` in order to use S-Port telemetry).

Instead, there is now a protocol drop-down that allows the user to select the in-use serial protocol.



Offering:



Usage

Item	Usage
Auto	Auto-detects the protocol from the serial data stream. Note that MPM cannot (yet) be auto-detected reliably, and must be explicitly selected).
INAV	INAV protocols, MSP, LTM and MAVLink. Legacy behaviours
S-Port	Smartport telemetry, previously required `--smartport` options. Expects a non-inverted stream
CRSF	Crossfire Telemetry.
MPM	Multi-Protocol-Module telemetry. The output from an EdgeTX / OpenTX radio with a multi-protocol module, FrSky Smartport or Flysky 'AA' via the EdgeTX / OpenTX "Telem Mirror" function. Is not auto-detected, must be explicitly selected.

NOTES

- For radar functions (inav-radar, ADSB), it is necessary to set the `--radar-device=`. Leave the protocol selector at 'Auto'.

- For telemetry forwarding, it is necessary to set the `--forward-to=`. Leave the protocol selector at 'Auto'.
- For FlySky MPM telemetry, the inav CLI setting `set ibus_telemetry_type = 0` is required; any other `ibus_telemetry_type` value will not work.

AUTO-DETECTION

- INAV (MSP, LTM, MAVLink) auto-detection should be reliable (legacy function).
- S-Port and CRSF may be less reliably detected.
- MPM is not auto-detected. This may change if EdgeTX merges an extant PR
- It is recommended that for S-Port, CRSF and MPM, the desired protocol is set explicitly (not left at "Auto").

1.11 mwp Configuration

1.11.1 Overview

mwp stores configuration in a number of places, to some degree at the developer's whim, but also in accordance with the data item's volatility.

- Command line options
- Configuration Files
- dconf / gsettings

Each type is further discussed below.

1.11.2 Command line options

Command line options provide a 'per instantiation' means to control mwp behaviour; the current set of command line options may be viewed by running mwp from the command line with the single option `--help`:

```
$ mwp --help
```

Where it is required to give permanence to command line options, they can be added to the configuration file `$HOME/.config/cmdopts`, which is described in more detail in the following section.

Debug flags

The `--debug-flags` option takes a numeric value defines areas where additional debug information may be output.

Value	Usage
1	Waypoints
2	Startup
4	MSP
8	ADHOC
16	RADAR
32	LOG REPLAY
64	SERIAL
128	VIDEO
256	GCS Location

Values may be added together (so 511 means all).

1.11.3 Configuration Files

mwp configuration files are stored in a standard directory `$HOME/.config/mwp`. This directory is created on first invocation if it does not exist. The following files may be found there:

1.11.4 `cmdopts`

The file `cmdopts` contains command line options that the user wishes to apply permanently (and conveniently when run from a launcher icon rather than the command line).

The file contains CLI options exactly as would be issued from the terminal. Options may be on separate lines, and blank lines and line prefixed with a hash '#' are ignored. For example:

In addition to options (`--`), the file may also contain environment variables e.g. `FOO=BAR` .

```
# Default options for mwp
--rings 50,20
#--voice-command "spd-say -t female2 -e"
#--debug-flags=2
--dont-maximise
#-S 8192

# set the anonymous tile file.
MWP_BLACK_TILE=/home/jrh/.config/mwp/mars.png
```

So here the only current, valid options are `--rings 50,20 -dont-maximise` , and the environment variable MWP_BLACK_TILE is set (for anonymous maps).

1.11.5 `.layout`

`.layout` contains the current arrangement of Dock items. You are advised not to manually edit this file (or other named, alternate layout files).

1.11.6 `sources.json`

`sources.json` facilitates adding non-standard map sources to mwp. See the manual and comments in the source files in the `qproxy` directory.

1.11.7 `vcol.css`

`vol.css` contains alternate CSS themeing for the battery voltage dock item that may work better on dark desktop themes. An example file is provided as `mwp/vcol.css` which can be copied into `.config/mwp/` .

1.11.8 `places`

The `places` (`~/.config/mwp/places`) file is a delimited (CSV) file that defines a list of "shortcut" home locations used by the "View / Centre on Position ..." menu item. It consists of a Name, Latitude, Longitude and optionally zoom level, separated by a `TAB` , `|` , `:` or `;` . Note that positions may be localised in the file and thus `.` is no longer recognised as a field separator.

Example `places`

```
# mwp places name,lat,lon [,zoom]
Beaulieu|50.8047104|-1.4942621|17
Jurby:54.353974:-4.523600:-1
```

The user may maintain these files manually if used, or use the graphic editor.

1.11.9 Dconf / gsettings

Linux has a facility for storing configuration items in a registry like facility. This is used extensively by mwp. The items can viewed and modified using a number of tools:

- mwp preference dialogue (for a small subset of the items)
- The `dconf-editor` graphical editor
- The command line `gsettings` tool

For `gsettings` and `dconf-editor` , the name-space is `org.mwptools.planner` , so to list of items:

```
$ gsettings list-recursively  org.mwptools.planner
```

and to list then set a single item:

```
$ gsettings get org.mwptools.planner log-save-path
..
$ gsettings set org.mwptools.planner log-save-path ~/flight-logs/
```


List of mwp settings

Name	Summary	Description	Default
adjust-tz	mwp should adjust TZ (and DST) based in the local clock	mwp should adjust TZ (and DST) based in the local clock	true
ah-invert-roll	Invert AH roll	Set to true to invert roll in the AH (so it becomes an attitude indicator)	false
ah-size	minimum size of artificial horizon	(private setting)	32
arming-speak	speak arming states	whether to reporting arming state by audio	false
atexit	Something that is executed at exit	e.g. `gsettings set org.gnome.settings-daemon.plugins.power idle-dim true`. See also `manage-power` (and consider setting to true).	""
atstart	Something that is executed at startup	e.g. `gsettings set org.gnome.settings-daemon.plugins.power idle-dim false`. See also `manage-power` (and consider setting to true).	""
audio-bearing-is-reciprocal	Announce bearing as reciprocal	Whether the audio bearing is the reciprocal (i.e. bearing from home to machine, rather than from machine to home)	false
audio-on-arm	start audio on arm	start audio on arm (and stop on disarm)	true
auto-follow	set auto-follow on start	set auto-follow on start	true
auto-restore-mission	Whether to automatically import a mission in FC memory to MWP	If the FC holds a valid mission in memory, and there is no mission loaded into MWP, this setting controls whether MWP automatically downloads the mission.	false
auto-wp-edit	Whether you direct WP editing is available	If true, the user can edit / create waypoints directly by clicking on the map, if false, it is necessary to toggle the WP Edit button to enable editing.	false
baudrate	Baud rate	Serial baud rate	115200
blackbox-decode	Name of the blackbox_decode application	Name of the blackbox_decode application (in case there are separate for iNav and betaflight)	"blackbox_decode"
centre-on	centre map on GPS as needed	centre map on GPS as needed	true
checkswitches	check switches	check switches (a JH sanity check)	false
compat-version	mw-nav compat version	Default mw-nav compat version in XML files. mwp doesn't care, older (MW) applications might.	"42.0"
dbox-is-horizontal	Geometry of the DirectionView box	If true, uses a horizontal organisation, rather than vertical	false
default-altitude	Default altitude	Default Altitude (m)	20
default-latitude	Default Latitude	Default Latitude when no GPS	50.909528

Name	Summary	Description	Default
default-layout	Default layout name	Default layout name. If not set, .layout is used.	""
default-loiter	Default Loiter time	Default Loiter time	30
default-longitude	Default Longitude	Default Longitude when no GPS	-1.532936
default-map	Default Map	Default map *key*	""
default-nav-speed	Default Nav speed	Default Nav speed (m/s). For calculating durations only.	2.5
default-zoom	Default Map zoom	Default map zoom	15
delta-minspeed	Minimum speed for elapsed distance updates	Minimum speed for elapsed distance updates (m/s). Default is zero, which means the elapsed distance is always updated; larger values will take out hover / jitter movements.	0.0
device-names	Device names	A list of device names to be added to those that can be auto-discovered	[]
display-distance	Distance units	0=metres, 1=feet, 2=yards	0
display-dms	Position display	Show positions as dd:mm:ss rather than decimal degrees	false
display-speed	Speed units	0=metres/sec, 1=kilometres/hour, 2=miles/hour, 3=knots	0
dump-unknown	dump unknown	dump unknown message payload	false
espeak-voice	Default espeak voice	Default espeak voice (see espeak documentation)	"en"
fctype	Force fc type	Forces fc type (mw,mwnav,bf,cf)	"auto"
fixedfont	Use a fixed font for Flight View	Use a fixed font for Flight View	true
flash-warn	Flash storage warning	If a dataflash is configured for black box, and this key is non-zero, a warning in generated if the data flash is greater than "flash-warn" percent full.	0
flite-voice-file	Default flite voice file	Default flite voice file (full path, *.flitevox), see flite documentation)	""
font-fv	flight view font scaling	Scales the flight view widget. Smaller screens may need a lower value	12
forward	Types of message to forward	Types of message to forward (none, LTM, minLTM, minMAV, all)	"minLTM"
geouser	User account on geonames.org	A user account to query geonames.org for blackbox log timezone info. A default account of 'mwptools' is provided; however users are requested to create their own account.	"mwptools"
gpsd-host	gpsd provider	Provider for GCS location via gpsd. Default is "localhost", can be set to other host name or IP address. Setting blank ("") disables.	"localhost"

Name	Summary	Description	Default
gpsintvl	gps sanity time (m/s)	gps sanity time (m/s), check for current fix	2000
heartbeat	Something that runs every minute (i.e. screensaver disable)	e.g. `xscreensaver-command -deactivate`. See also `manage-power` (and consider setting to true).	""
ignore-nm	Don't ever query Network Manager for network status	Set to true to always ignore NM status (may slow down startup)	false
kml-path	Directory for KML overlays	Directory for KML overlays	""
led	GPS LED colour	GPS LED colour as well know string or #RRGGBB	"#60ff00"
load-safehome	Load default set of safehomes	Set to file[,Y]. File defines a set of safehome lines (CLI format), optionally followed by a comma and Y. If the definition includes ",Y", then the safehome locations will be displayed.	""
log-on-arm	start logging on arm	start logging on arm (and stop on disarm)	false
log-path	Directory for replay log files	Directory for log files (for replay)	""
log-save-path	Directory for storing log files	Directory for log files (for save), default = current directory	""
mag-sanity	Enable mag sanity checking	mwp offers a primitive mag sanity checker that compares compass heading with GPS course over the ground using LTM (only). There are various hard-coded constraints (speed > 3m/s, certain flight modes) and two configurable parameters that should be set here in order to enable this check. The parameters are angular difference (º) and duration (s). The author finds a settings of 45,3 (i.e. 45º over 3 seconds) works OK, detecting real instances (a momentarily breaking cable) and not reporting false positives.	""
manage-power	manage power and screen	whether to manage idle and screen saver	false
map-sources	Additional Map sources	JSON file defining additional map sources	""
mavph	RC settings for Mav PH	RC settings for Mav PH (chanid:minval:maxval)	""
mavrth	RC settings for Mav RTH	RC settings for Mav RTH (chanid:minval:maxval)	""
max-climb-angle	Maximum climb angle highlight for terrain analysis	If non-zero, any climb angles exceeding the specified value will be highlighted in Terrain Analysis Climb / Dive report. Note that the absolute value is taken as a positive (climb) angle	0.0

Name	Summary	Description	Default
max-dive-angle	Maximum dive angle highlight for terrain analysis	If non-zero, any dive angles exceeding the specified value will be highlighted in Terrain Analysis Climb / Dive report. Note that the absolute value is taken as a negative (dive) angle	0.0
max-home-delta	home position delta (m)	Maximum variation of home position without verbal alert	2.5
max-radar-slots	Maximum number of aircraft reported by iNav-radar	Maximum number of aircraft reported by iNav-radar	4
max-wps	Maximum number of WP supported	Maximum number of WP supported	120
media-player	Media player for alerts	Blank means internal gstreamer, "false" or "none" means no beeps.	""
misc-icon-size	Miscellaneous icon size	Size for miscellaneous icons (radar, GCS location) in pixels. -1 means the image's natural size (no scaling).	32
mission-file-type	Preferred mission file type	m for XML (.mission), j for json (change at your peril)	"m"
mission-meta-tag	use meta vice mwp in mission file	If true, the legacy mwp tag is named meta	false
mission-path	Directory for mission files	Directory for mission files	""
osd-mode	Data items overlaid on the map	0 = none, 1 = current WP/Max WP, 2 = next WP distance and course. This is a mask, so 3 means both OSD items.	3
poll-timeout	Poll messages timeout (ms)	Timeout in milliseconds for telemetry poll messages. Note that timer loop has a resoltion of 100ms.	900
pos-is-centre	Determines position label content	Whether the position label is the centre or pointer location	true
pwdw-p	internal parameter	(private setting)	72
radar-alert-altitude	Altitude below which ADS-B alerts may be generated	Target altitude (metres) below which ADS-B proximity alerts may be generated. Requires that 'radar-alert-range' is also set (none zero). Setting to 0 disables. Note that ADS-B altitudes are AMSL (or geoid).	0
radar-alert-range	Range below which ADS-B alerts may be generated	Target range (metres) below which ADS-B proximity alerts may be generated. Requires that 'radar-alert-altitude' is also set (none zero). Setting to 0 disables.	0
radar-list-max-altitude	Maximum altitude for targets to show in the radar list view	Maximum altitude (metres) to include targets in the radar list view. Targets higher than this value will show only in the map view. This is mainly for ADS-B receivers where there is no need for high altitude targets to be shown. Setting to 0 disables. Note that ADS-B altitudes are AMSL (or geoid).	0

Name	Summary	Description	Default
require-telemetry	Whether to warn the operator if telemetry is disabled in iNav	if set, and telemetry is disabled, a non-timeout dialogue is displayed	false
rings-colour	range rings colour	range rings colour as well know string or #RRGGBBAA	"#ffffff20"
rth-autoland	Automatically assert land on RTH waypoints	Automatically assert land on RTH waypoints	false
say-bearing	Whether audio report includes bearing	Whether audio report includes bearing	true
set-head-is-b0rken	set head bearing as reciprocal	Whether the set head bearing is the reciprocal (i.e. ancient bug in mw nav)	false
smartport-fuel-unit	User selected fuel type	Units label for smartport fuel (none, %, mAh, mWh)	"none"
speak-amps	When to speak amps/hr used	none, live-n, all-n n=1,2,4 : n = how often spoken (modulus basically)	"none"
speak-interval	Interval between voice prompts	Interval between voice prompts, 0 disables	15
speech-api	API for speech synthesis	espeak, speechd, flite. Only change this if you know you have the required development files at build time	"espeak"
speechd-voice	Default speechd voice	Default speechd voice (see speechd documentation)	"male1"
stats-timeout	timeout for flight statistics display (s)	Timeout before the flight statistics popup automatically closes. A value of 0 means no timeout.	30
tote-float-p	Do Mission tote float	(private setting)	true
uc-mission-tags	Upper case mission XML tags	If true, MISSION, VERSION and MISSIONITEM tags are upper case (for interoperability with legacy Android applications)	false
uilang	Language Handling	"en" do everything as English (UI numeric decimal points, voice), "ev" do voice as English (so say 'point' for decimals even when shown as 'comma')	""
use-legacy-centre-on	If true, uses legacy centre-on	If true, uses legacy centre-on mode rather than the new "In View" mode.	false
vlevels	Voltage levels	Semi-colon(;) separated list of *cell* voltages values for transition between voltage label colours	""
wp-dist-size	Font size (points) for OSD WP distance display	Font size (points) for OSD WP distance display	56.0
wp-spotlight	Style for the 'next waypoint' highlight	Defines RGBA colour for 'next way point' highlight	"#ffffff60"

Name	Summary	Description	Default
wp-text-style	Style of text used for next WP display	Defines the way the WP numbers are displayed. Font, size and RGBA description (or well known name, with alpha)	"Sans 144/ #ff000080"
zone-detect	Application to return timezone from location	If supplied, the application will be used to return the timezone (in preference to geonames.org). The application should take latitude and longitude as parameters. See samples/tzget.sh	""

1.12 mwp and inav safehome

One of the great features of inav 2.6 was the `safehome` capability. The user can define of set of up to eight locations, and if any of these is within 200m (configurable up to 650m in inav 2.7), then that is used as the home location for RTH (and RTH failsafe).

1.12.1 inav setting

`safehome` is set in inav usng the CLI, here's an example:

```
# safehome
safehome 0 1 508047750 -14948970
safehome 1 1 509102384 -15344850
safehome 2 1 509390336 -14613540
safehome 3 1 509149619 -15337365
safehome 4 0 508054891 -14961431
safehome 5 0 543545392 -45219430
safehome 6 0 540954148 -47328458
safehome 7 0 0 0
```

As you see, it's not too user friendly; the parmeters are

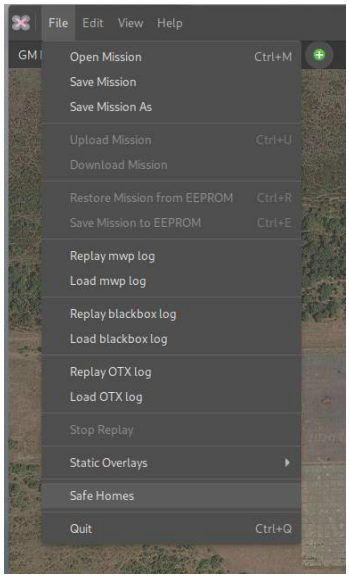
- Index (0 - 7)
- Status (0 = don't use, 1 = can use)
- Latitude as degrees * 10,000,000 (i.e. 7 decimal places)
- Longitude as degrees * 10,000,000 (i.e. 7 decimal places)

It can be error prone to get locations into the correct format, particularly when a common source (Google Maps) only provides 6 decimal places of precision.

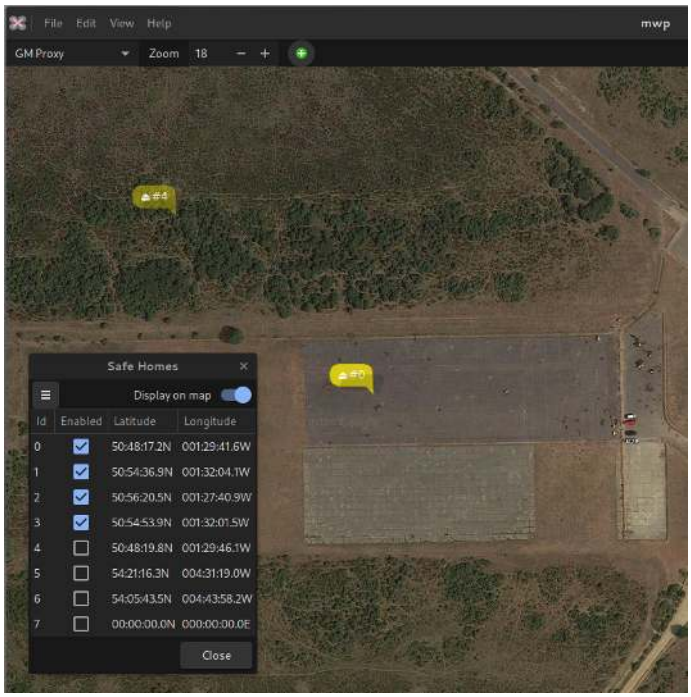
1.12.2 mwp solution

Graphical User Interface

mwp now offers a `Safe Homes` menu option:



This will launch the `Safe Home` window:



From here it is possible to:

- Load safehomes from a file in CLI format. A CLI diff or dump can be used.
- Save safehomes to a file in CLI format. If a CLI diff or dump is selected, then only the safehomes stanza is changed; other information in the diff / dump is preserved.
- Display safehomes on the map. Active safehomes are displayed with greater opacity than inactive locations.
- Change the status (active, inactive). If a previously unused item is enabled, an icon is placed on the centre of the map for positioning.
- Clear (unset) one or all safehomes.

Note that editing functions are only available when the `Safe Homes` window is active; if the windows is dismissed with icons displayed, then the icons remain on the map, but are not editable.

Display safehomes at startup

It also is possible to set a `gsettings` key to define a file of safehomes to load at startup, and optionally display (readonly) icons.

```
gsettings set org.mwptools.planner load-safehome ~/.config/mwp/safehome.txt,Y
```

This sets the default safehomes file to `~/.config/mwp/safehome.txt` and the appended `,Y` means display the icons on the map.

Example

The image below shows a blackbox replay. Note that the flight home location (brown icon) is coincident with the pale orange safehome icon.

1.13 Radar View

mwp supports the display of "radar" contacts. This provides a view of adjacent aircraft obtained from a number of sources:

- inav-radar. inav radar works in conjunction with inav flight controllers to broadcast the location of UAS fitted with an ESP32 LoRa module. mwp can listen to one of these modems in ground station mode to display the positions of the rest of the 'swarm' (up to 4 UAS); technical / MSP details.
- **Full size aircraft** reported by the MAVLink 'Traffic Report' message. An example is the uAvionix PingRX, a compact device that receives ADS-B location data from full sized aircraft and publishes the locations as MAVLink. For a ground based installation, this device has around a 40Km detection radius. MAVLink ICD.
- Proximity alerts (visual and audible) for manned (ADS-B) aircraft, based on planned or actual home location.

1.13.1 mwp Configuration

mwp can receive the 'radar' data over one or two connections, either or both may be active, and mwp can receive and display 'own vehicle' telemetry (MSP, LTM or Smartpost), 'inav-radar' and 'MAVlink Traffic' data simultaneously. Radar data may be received over:

- The main serial port device (see caveat for inav-radar) or
- device(s) defined by the `radar-device` CLI or configuration parameter (MAVLink Traffic, inav-radar)

The `radar-device` option is defined by the standard mwp naming scheme:

- A serial device node, with optional baud rate, e.g.:
  - `/dev/ttyACM0`, `/dev/ttyUSB4@567600`, `/dev/rfcomm3`
  - Serial defaults to 115200 baud, but may be set in the device name (@baudrate)
- A Bluetooth address (for BT bridges)
  - `00:0B:0D:87:13:A2`
- An IP address, e.g. for simulation, recording replays or serial multiplexer.
  - `udp://:30001` local UDP listener.

The specific (not shared with the main serial port) radar device(s) may be defined on the command line, or in the static command options file (`~/.config/mwp/cmdopts`):

- `mwp --radar-device udp://:30001`
- `$ cat ~/.config/mwp/cmdopts`

```
# Default options for mwp
# using udev rule to associate a specifc USB-TTL adaptor to a name
--radar-device=/dev/pingRX@57600
```

Multiple devices may be defined, e.g.

- As separate options, `--radar-device=/dev/pingRX@57600 --radar-device= /dev/inavradar@115200`
- As a comma separated list: `--radar-device=/dev/pingRX@57600,/dev/inavradar@115200`

Any bespoke `radar-device` is started automatically on startup (or when it shows up). It is not managed via the serial `Connect` button.

1.13.2 Using the main serial port

The main serial port may be used for MavLink Traffic without any further configuration. For inav-radar, to use the main msp port for inav-radar (vice using `--radar-device`), it is still necessary to add a command option to mwp; it needs to told to relax the default inbound MSP direction check.

This is enabled as

```
mwp --relaxed-msp
```

which should be 'mainly harmless' for normal operations. It's entirely acceptable to put this in `~/config/mwp/cmdopts` to make it the default, as the protocol check dilution is slight.

1.13.3 Settings

The following `dconf` setting affect the radar function:

Setting	Usage
`radar-list-max-altitude`	Maximum altitude (metres) to show targets in the radar list view; targets higher than this value will show only in the map view. Setting to 0 disables. Note that ADS-B altitudes are AMSL (or geoid)
`radar-alert-altitude`	Target altitude (metres) below which ADS-B proximity alerts may be generated. Requires that 'radar-alert-range' is also set (none zero). Setting to 0 disables. Note that ADS-B altitudes are AMSL (or geoid).
`radar-alert-range`	Target range (metres) below which ADS-B proximity alerts may be generated. Requires that 'radar-alert-altitude' is also set (none zero). Setting to 0 disables.

Note that proximity alerts require that both the `radar-alert-altitude` and `radar-alert-range` values are set, and that there is a planned or actual home location.

1.13.4 Usage

Once the radar interface is open, radar tracks are displayed on the map and in a list available from the "View -> Radar View' menu option.

- The list view is sort-able on the `Id`, `Status`, `Last` (time) and `Range` columns.
- The map visualisation may be toggled by the `Hide Tracks` (`Show Tracks`) button.
- List and map views are updated in (near) real time.
- Preference for display units are used for positions, altitude and speed.

Name

Type	Usage
inav-radar	Node Id (typically 'A' - 'D')
Traffic Report	Callsign if reported, otherwise [ICAO number]

Status

Radar contacts have one of the following status values:

Status	Explanation
Undefined	Not shown in list or on the map
Stale	The last contact was more that 120s previous. Displayed in the list and shown on the map with reduced intensity or an inav-radar node has'lost' status
Armed	An active inav-radar contact
ADS-B	A live MAVLink Traffic report
Hidden	A MAVLink Traffic contact is between 5 and 10 minutes old. It remains in the list but is not displayed in the map. MAVLink Traffic Report tracks are removed from the list (and internal storage) after 10 minutes inactivity. inav-radar ground station

Stale / 'Lost' inav-radar contacts do not expire, as they may relate to a lost model.

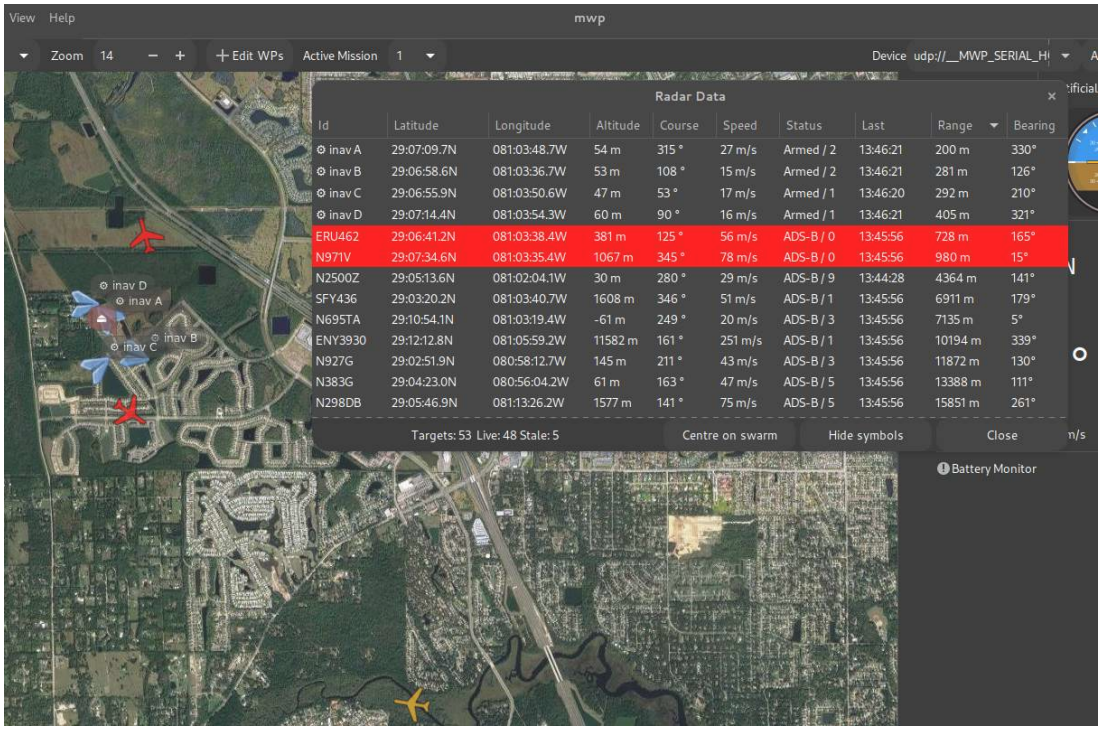
The number displayed after the status text is:

Type	Usage
inav-radar	The link quality
Traffic Report	Time since last communication in seconds

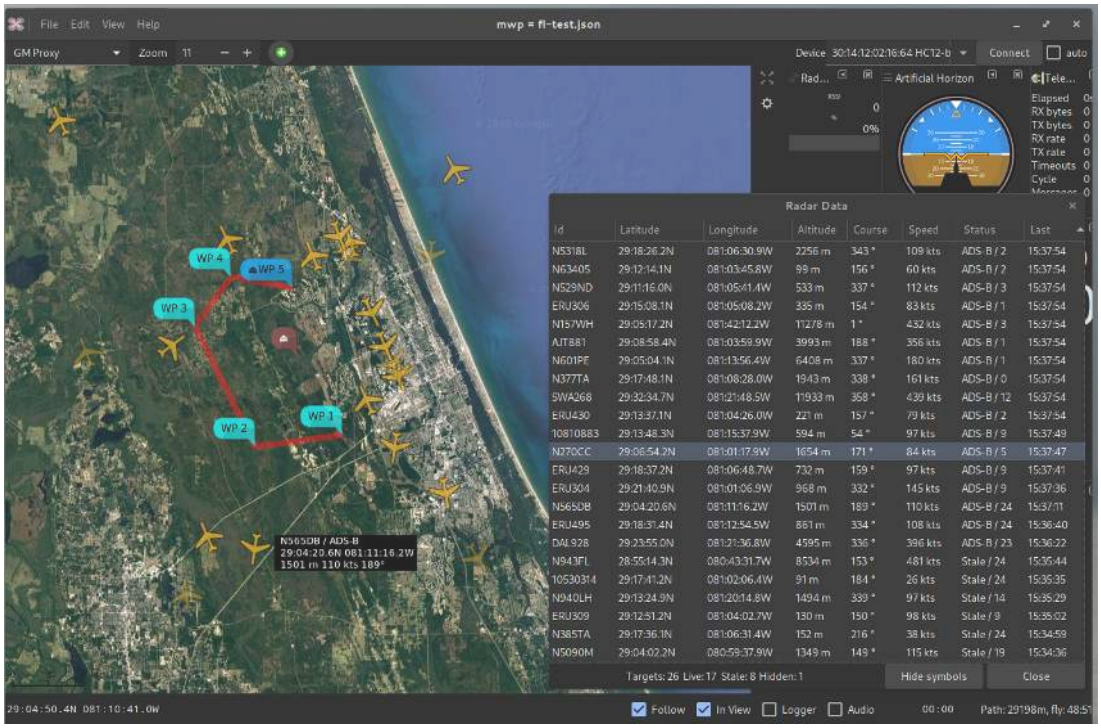
1.13.5 Examples

- Proximity Alerts
- Live and stale aircraft
- Aircraft tooltip
- Mission Plan
- List view

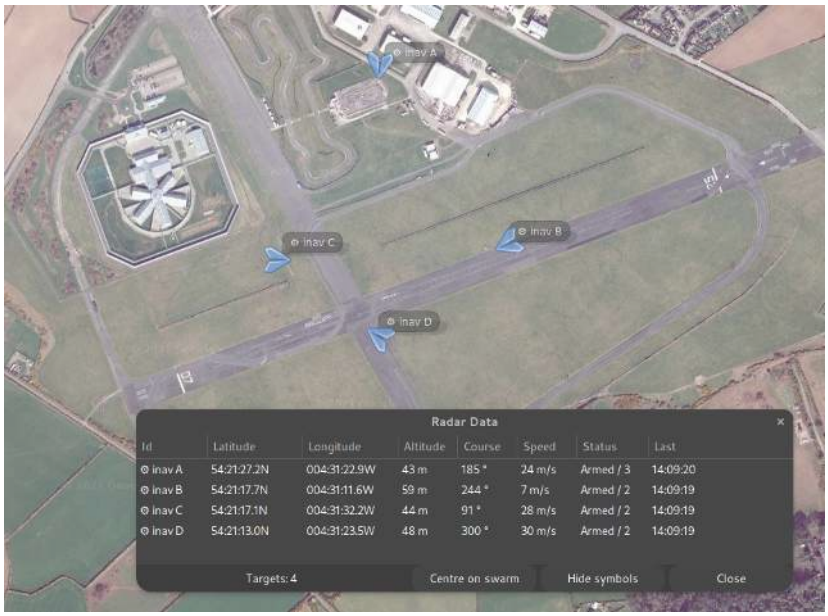
Live ADS-B and simulated inav targets, with proximity alerts (range < 3000m).



Local manned aircraft view over Florida (May 2020).



Simulated inav radar view



1.13.6 Simulators

There are simulators for both inav-radar and MAVLink 'Traffic Report' (e.g. uAvionix PingRX) in the `mwptools/samples/radar` directory.

1.13.7 Changing the Radar Symbols

Any map symbol used by mwp can be changed by the user; in the image above, the inav radar node symbol has been changed from the default stylised inav multirotor to a smaller version of the mission replay "paper plane" symbol as follows:

- All the default mwp icons / map symbols can be found in `$prefix/share/mwp/pixmaps/` (e.g. `~/.local/share/mwp/pixmaps` for a "local" installation).
- Create your own icon with the equivalent name in `~/config/mwp/pixmaps/`.

The icons in `~/config/mwp/pixmaps/` are found before the defaults.

```
mkdir -p `~/config/mwp/pixmaps
# copy the preview image
cp /usr/share/mwp/pixmaps/preview.png  ~/config/mwp/pixmaps/
# (optionally) resize it to 32x32 pixels
mogrify -resize 80% ~/config/mwp/pixmaps/preview.png
# and rename it, mwp doesn't care about the 'extension', this is not MSDOS:)
mv  ~/config/mwp/pixmaps/preview.png  ~/config/mwp/pixmaps/inav-radar.svg
# and verify ... perfect
file ~/.config/mwp/pixmaps/inav-radar.svg
/home/jrh/.config/mwp/pixmaps/inav-radar.svg: PNG image data, 32 x 32, 8-bit/color RGBA, non-interlaced
```

1.13.8 Protocol documentation

MAVLink 'Traffic Report' (e.g. uAvionix PingRX)

The MAVLink implementation is comprehensively documented by the vendor.

inav radar

The following is required by a device wishing to act as a ground node (it either masquerades as an inav FC, or declares itself a GCS)

- Receive and respond to the following MSP data requests:
  - MSP_FC_VARIANT (responding as `INAV` or (from 2021/05/06) `GCS` for generic ground control stations).
  - MSP_FC_VERSION (in `INAV` and `GCS` modes)
  - MSP_NAME (in `INAV` and `GCS` modes)
  - MSP_STATUS (in `INAV` mode)
  - MSP_ANALOG (in `INAV` mode)
  - MSP_BOXIDS (in `INAV` mode)
  - MSP_RAW_GPS (in `INAV` mode)
- Receive unsolicited
  - MSP2_COMMON_SET_RADAR_POS

Note that the device firmware assumes that MSP buffer sizes are "as specification"; exceeding the expected message buffer size may crash the device (mea culpa).

In `GCS` mode, the node is passive; it does not use a LoRa slot and does not attempt to broadcast a location. In `INAV` mode, the node takes up a LoRa slot and is expected to reply to the additional MSP queries.

mwp's behaviour is defined by the GCS Location

- If the GCS Location is defined (when the radar device is iniitilised, then mwp will respond as `INAV` and return the GCS Location, which may be driven by gpsd if required.
- Otherwise, mwp will respond as a passive `GCS`.

1.14 Playing Video in mwp

mwp provides support for live and replay video.

- In ground station mode, in order to repeat the FPV feed to the mwp screen, presumably for the enjoyment of spectators;
- During Blackbox replay, to show the FPV recorded video during the replay.

1.14.1 Dependencies and platform requirements

The video replay capability requires:

- Arch Linux `sudo pacman -S gst-plugins-base-libs`
- Debian / Ubuntu `sudo apt install libgstreamer-plugins-base1.0-dev`
- Fedora `sudo dnf install gstreamer1-plugins-base gstreamer1-plugins-base-devel`
- Other distro -- consult the package manager

And, if not installed:

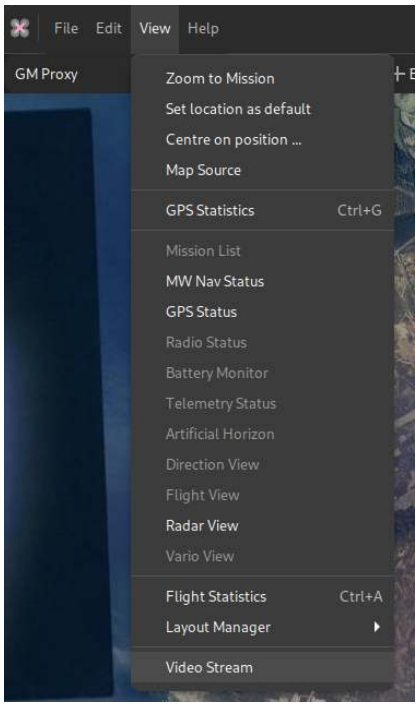
- Arch Linux `gst-plugins-good`
- Debian / Ubuntu `gstreamer1.0-plugins-good`
- Fedora `gstreamer1-plugins-good`
- Other distro -- consult the package manager

> **One off actions**
>
> These are documented for new installs (and provided by the 'easy' script).

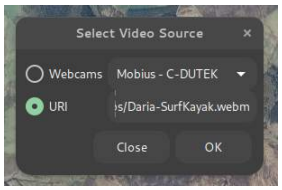
> **FreeBSD**
>
> Strictly, mwp requires `gstreamer1.0-plugins-gtk` which *should* be included in `gstreamer1.0-plugins-good`; on FreeBSD it is necessary to install `gstreamer1-plugins-gtk` explicitly.

1.14.2 Live stream mode (GCS)

There is now a **Video Stream** option under the view menu.



Selecting this option opens the source selection dialogue. Camera devices offering a "video4linux" interface (i.e most webcams) will be auto-detected. There is also the option to enter a URI, which could be a `http` / `https`, `rtsp` or other standard streaming protocol, or even a file.

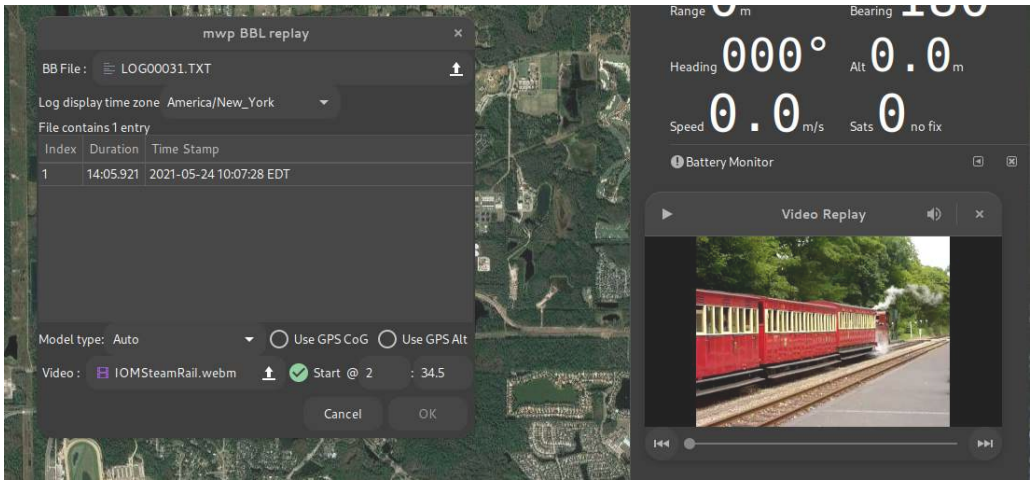


The selected source will then play in a separate window. This window will remain above the mwp application and can be resized, minimised and moved.

In stream mode, there are minimal video controls; a play/pause button and volume control. Note the volume is that of the video, the overall volume is controlled by the system volume control.

1.14.3 Blackbox replay mode (BBL replay)

The Blackbox log replay chooser also offers a video replay option.



Here the user can select a media file and start options, i.e. whether and when to start the video replay with respect to the start of the BB log replay.

- In order for mwp to start the replay, the **Start** check-button must be selected. If it is:
- The user can enter an optional time (minutes : seconds) that defines when the video starts relative to the start of the BB log:
  - No time is entered, or the time is 0:00 : The video starts at the start of the BBL replay.
  - The time is positive (e.g. 2:34.5 (two minutes, 34.5 seconds), as the example: Here the video would start when BB log starts, at an offset 2:34.5 into the video (i.e. the pilot started FPV recording 2m 34.5s before arming the aircraft).
  - If the time is negative (including "-0" minutes), then the start of the video is delayed by that amount; so -0:57 would delay the start of the video by 57 seconds relative to the start of BB log replay.
  - Pausing the replay will pause the video, and vice-versa.

When playing a file (vice a stream), the player gains a progress bar (which can be used to position the stream and "beginning" and "end" buttons.

1.14.4 Issues / Workarounds

If your camera does not work the `gstreamer` utilities, it is unlikely to work with mwp, as it uses `gstreamer` APIs for camera access.

You can easily test this using `gst-launch-1.0` which will closely emulate the way mwp works:

```
gst-launch-1.0 playbin uri=v4l2:///dev/video0
```

Where `/dev/video0` is the camera device node.

Fail example and resolution

A camera (an old Mobius) works on some computers and not others, including, annoyingly, the main mwp development box. The issue was an old USB2.0 (extension) hub that didn't provide enough bandwidth; so there was just a black screen shown.

Fixed by setting uvcvideo quirk 640: `UVC_QUIRK_FIX_BANDWIDTH` (0x80, 128) `UVC_QUIRK_RESTRICT_FRAME_RATE` (0x200, 512)

TEST FIX

```
sudo rmmod uvcvideo
sudo modprobe uvcvideo quirks=640
```

Now there is a proper picture, rather than a black screen.

PERMANENT SOLUTION

Add a file e.g. `/etc/modprobe.d/v4l2.conf` containing the line:

```
options uvcvideo quirks=640
```

or to any other `.conf` file under `/etc/modprobe.d/`

Helper tools

There are a couple of tools under `mwptools/src/samples/gst-video/`. These are not built / installed by default but may be built if required to enable diagnostics.

```
cd mwptools/src/samples/gst-video
make
```



```
# optionally, install to ~/.local/bin
make install
```

- `gst-devmon` provides the same video device monitoring as employed by mwp. It should report the insertion and removal of camera devices, together with their attributes.
- `gst-video-player` provides the same video replay capability as mwp
  - Camera stream : `gst-video-player v4l2:///dev/video0` . Assuming the camera, as reported by `gst-devmon` is `/dev/video0`.
  - File: `gst-video-player somefile.mp4`
  - Web stream `gst-video-player https://www.freedesktop.org/software/gstreamer-sdk/data/media/sintel_trailer-480p.webm`

1.14.5 Other OS

- FreeBSD. FreeBSD offers a video4linux emulation that works with mwp. Cameras are not auto-detected but will be recognised if plugged in before mwp is invoked. In any case, the URI `v4l2:///dev/video0` (for example) can be used in streaming mode if required.
- Windows 11/ WSLG: No support for cameras, probably works with files / URLs.

1.15 Fly By Home Waypoints

1.15.1 Introduction

For inav 4.0, there is a "FlyBy Home" (FBH) waypoint modifier.

This will set waypoints of types WAYPOINT, POSHOLD_TIME and LAND to execute at the arming home location (any safehome is ignored).

The flight controller applies FBH behaviour to waypoints having one (or both) of the following characteristics:

- The latitude and longitude are 0
- The mission item `flag` field is set to 0x48 (72 decimal, 'H')

In this case, the waypoint position is determined at run time (when the WP is actually used) and is set to the arming location. Note that the arming location must be set with a valid GPS fix.

As the waypoint location is determined during execution, it is not stored; so downloading a completed mission will return the original locations, not the locations used during the mission.

1.15.2 Implications for a graphical mission planner

inav (and mwp) do not require a planned homed location, so providing graphical support for waypoints whose location is indeterminate prior to mission execution is an interesting challenge. mwp incorporates a number of new features to support FBH.

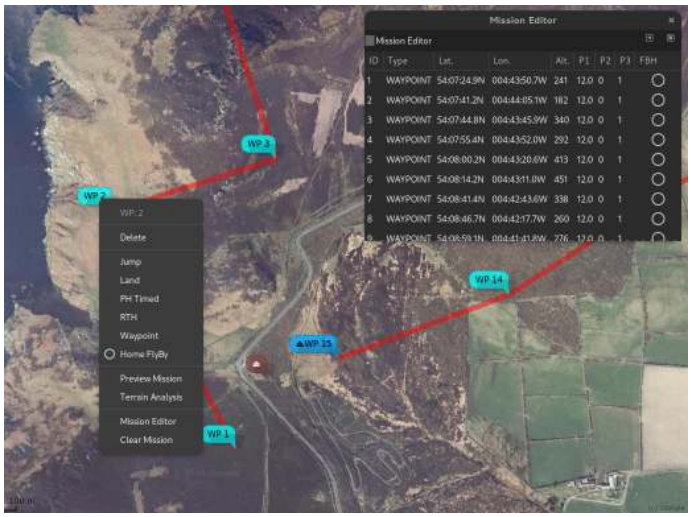
- The concept of a planned home location is embedded in the planning function. The planned home location is indicated by a brown icon.
- The planned home location is stored as metadata in the XML mission files.
- The `flag` attribute as been added the XML mission file schema.

The practical results being:

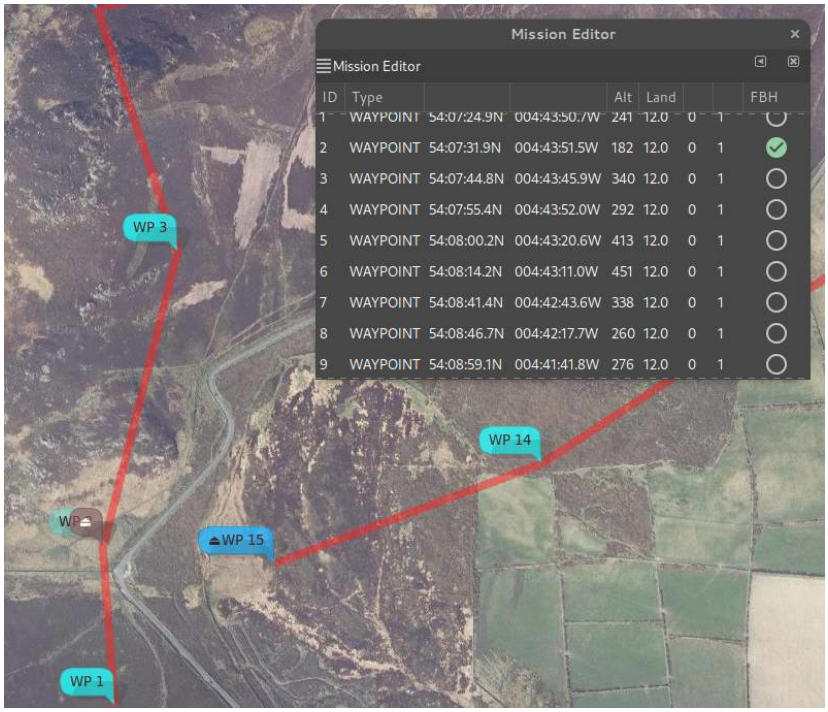
- A common mission file format continues to be used by mwp and the inav configurator planner; maintaining mission file interoperability between the two applications.
- The planned home is recorded and may be used for subsequent re-planning of a mission.
- FBH waypoints have a position (the planned home) and the `flag` set. This means they will behave predictably when uploaded to older firmware.

1.15.3 Usage in mwp

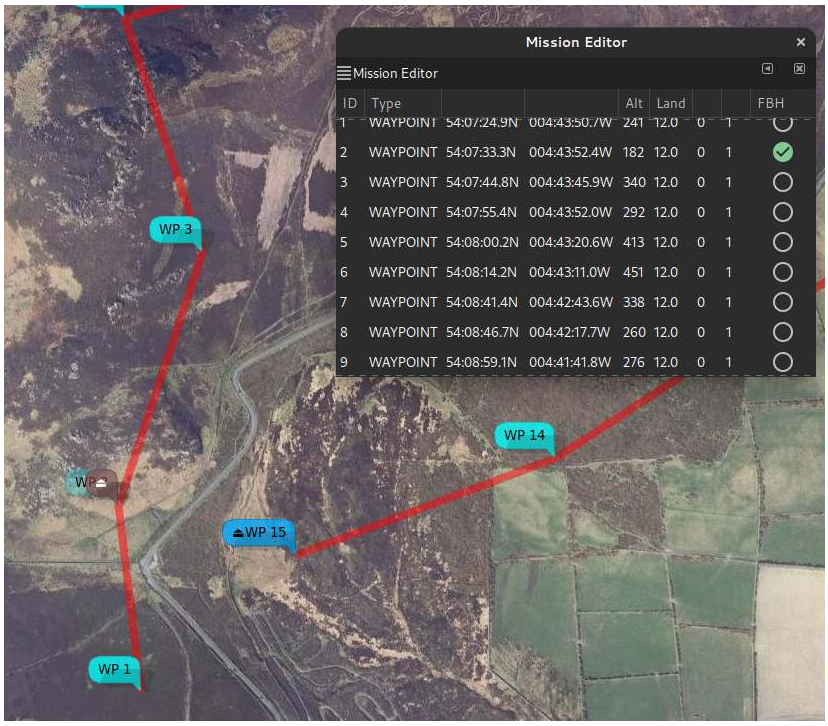
A waypoint may set set to FBH (or have FBH removed) from either the right mouse popup or the mission editor.



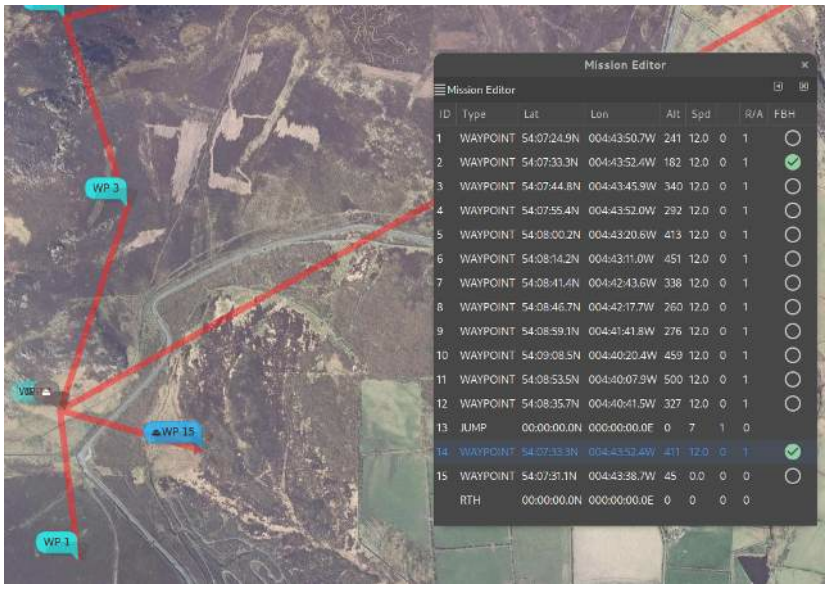
In the first image, no FBH waypoints have been set. We can see the planned home (the brown icon, which was read from the extant mission file), and the popup menu and mission editor. Note: the popup entry has since been renamed 'Fly By Home' for consistency.



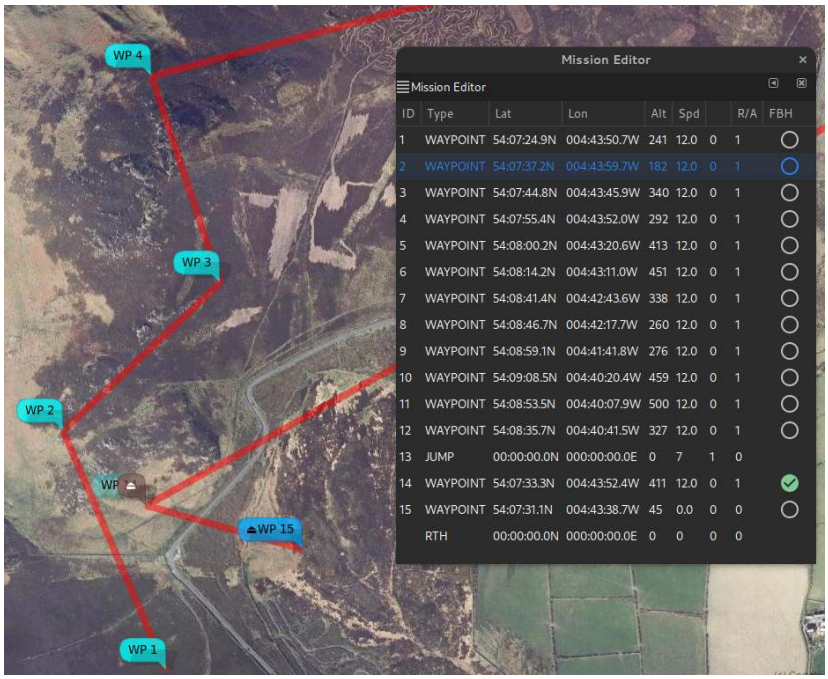
In the second image, WP2 has been made a FBH WP; we can see that it is now attached the home icon (and slightly faded). The home icon can be dragged, the attached FBH waypoint is no longer independently draggable.



In the third image, the planned home has been moved slightly north, WP2 has moved with it.



In the forth image, a second waypoint (WP14) has been set as FBH; it is also now locked to the planned home location.



In the fifth image, the FBH attribute as been cleared on WP2; it has been independently dragged to a new location.

1.15.4 mwp Ground Control Station and Replay modes

If a mission is loaded when mwp is used as ground control station or for log replay, and the mission contains FBH waypoints, then the mission will be redrawn with the actual home location when the home location is established.

1.16 Anonymous Maps

mwp povides a pseudo-map proxy that just gives you a black map (or user specified tile). This may be useful for a number of use-cases:

- privacy
- general obstinacy
- clarity of display

1.16.1 Building

This proxy is not build by default, it is necessary to build, install and configure the proxy manually.

```
cd mwptools/qproxy
make bproxy
# copy bproxy somewhere on the PATH
cp bproxy ~/.local/bin/
# or
sudo cp broxy /usr/local/bin
# or
sudo cp broxy /usr/bin
```

1.16.2 Configuration

That was the easy bit! Now it is necessary to tell mwp where to find the proxy. This involves a setting and a configuration file.

First of all, ensure that the `map-sources` setting is enabled:

```
$ gsettings get org.mwptools.planner map-sources
'sources.json'
# here this set to a file sources.json (in ~/.config/mwp/)
```

if this is not set, then set it:

```
$ gsettings set org.mwptools.planner map-sources 'sources.json'
```

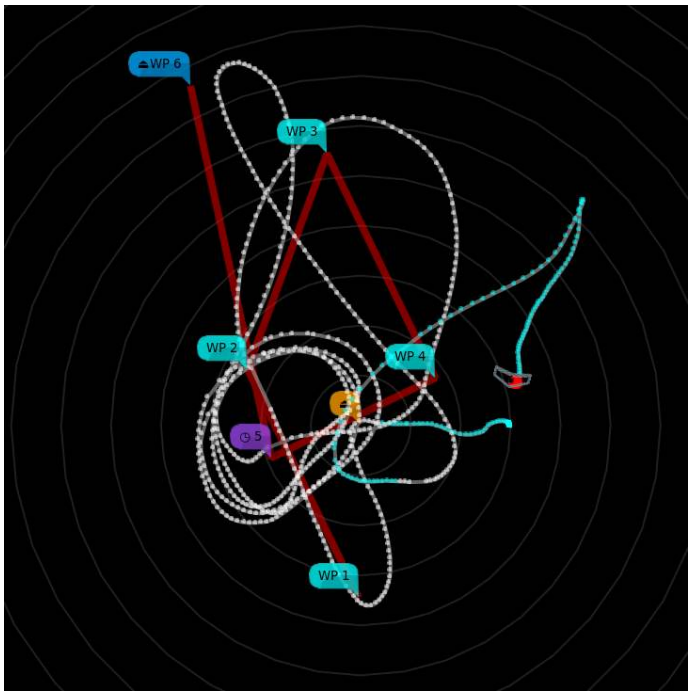
Now we need to edit the file `~/.config/mwp/sources.json`, there is a sample file in `mwptools/samples/sources.json`. you file needs a stanza like:

```
{
 "id": "Black",
 "name": "Black Tiles",
 "license": "(c) jh ",
 "license_uri": "http://daria.co.uk/",
 "min_zoom": 0,
 "max_zoom": 20,
 "tile_size": 256,
 "projection": "MERCATOR",
 "spawn" : "bproxy",
}
```

So a minimal `~/.config/mwp/sources.json` looks like:

```
{
   "sources" : [
      {
         "id": "Black",
         "name": "Black Tiles",
         "license": "(c) jh ",
         "license_uri": "http://daria.co.uk/",
         "min_zoom": 0,
         "max_zoom": 20,
         "tile_size": 256,
         "projection": "MERCATOR",
         "spawn" : "bproxy",
       }
   ]
}
```

On starting **mwp** you should see a new map option "Black Tiles".



1.16.3 Custom Tile

It's also possible to have a custom tile (which does not have to be black). The tile **must** be:

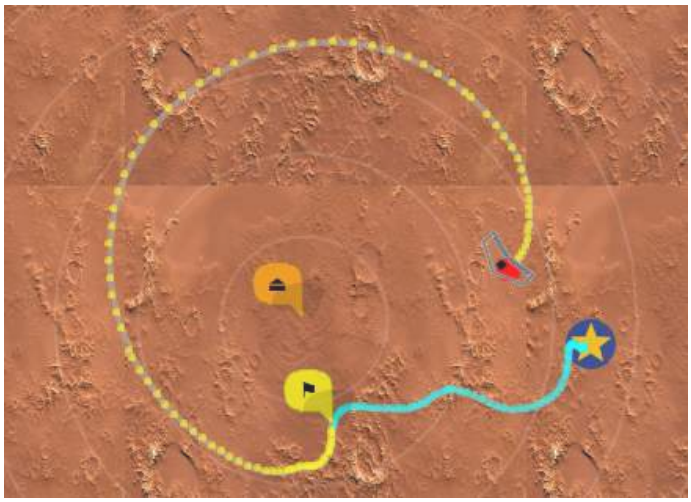
- 256x256 pixels
- PNG

The full path is provided in the environment variable `MWP_BLACK_TILE`, e.g.

```
# put this in e.g. ~/.bashrc to make it permanent
export MWP_BLACK_TILE=~/.config/mwp/mytile.png
```

The environment variable may instead be added to `~/.config/mwp/cmdopts`.

For example:



2.1 Building mwp (Generic)

2.1.1 Overview

The **mwptools** suite is built using the meson and ninja toolchain. For most users these will be automatically provided by a `build-essentials` type of package transparently to the user.

Prior to late May 2021, the build system used a convoluted `Makefile`.

For Debian and derivatives (Ubuntu, WSL etc.) there is a simple "one stop" installation script, as well as a x86_64 "Release" .deb archive.

For Arch Linux, you can install using the AUR package `mwptools-git`

2.1.2 Rationale

In its early days, `make` was a suitable build tool. As **mwptools** has gained in features and functionality, this has become un-maintainable. The migration to `meson` and `ninja` solves this problem and allows the project structure to be rationalised.

2.1.3 Usage

Migration

Please ensure your extant mwptools instance does not have untracked files:

```
git clean -fd -fx
git pull
```

First time

Set up the `meson` build system from the top level:

```
meson build --buildtype=release --strip [--prefix $HOME/.local`]
```

- For a user / non-system install, set `--prefix $HOME/.local`
  - This will install the binaries in `$HOME/.local/bin`, which should be added to `$PATH` as required.
- For a Linux system wide install, set `--prefix /usr`
- For FreeBSD (*BSD), for a system-wide install, don't set `--prefix` as the default (`/usr/local`) is suitable

Unless you need a multi-user setup, a local install is preferable, as you don't need `sudo` to install, and you'll not risk messing up build permissions.

- If you're using a really old OS (e.g. Debian 10), you may also need `export XDG_DATA_DIRS=/usr/share:$HOME/.local/share` for a local install.

Build and update

```
cd build
# for a local install (and cygwin)
ninja install
# for system install
ninja && sudo ninja install
```


Legacy

For now, some of the legacy `Makefiles` remain, and can be used similar to before, e.g. :

```
cd mwptools/src/mwp
make && sudo make install
```

At some stage, more of the Makefiles will be removed (or just rot into uselessness).

2.1.4 Files built / installed

Default

Application	Usage
`cliterm`	Interact with the CLI
`fc-get`, `fc-set` 1	Backup / restore CLI diff
`mwp`	Mission planner, GCS, log relay etc.
`mwp-area-planner`	Survey planner
`mwp-plot-elevations` 2	Mission elevation / terrain analysis
`qproxy`	Proxy for certain commercial TMS
`inav_states.rb`	Summarise BBL state changes, also installed `inav_states_data.rb`
`fcflash`	FC flashing tool, requires `dfu-util` and / or `stmflash32`

> **Notes:**
>
> 1. `fc-set` is a hard link to `fc-get`
>
> 2. This may either be the new Go executable or the legacy, less functional Ruby script.

Optional

These are only built by explicit target name; they will be installed if built.

```
# one of more of the following targets
ninja bproxy ublox-geo ublox-cli flashdl
sudo ninja install
```

Application	Usage
`flashdl`	Download, optionally delete BBL from flash
`bproxy`	Black tile map proxy, for those anonymous needs
`ublox-cli`	Ublox GPS tool
`ublox-geo`	Graphical Ublox GPS tool

Supporting data files

File	Target	Usage
`src/common/mwp_icon.svg`	`$prefix/share/icons/hicolor/scalable/apps/`	Desktop icon
`src/mwp/org.mwptools.planner.gschema.xml`	`$prefix/share/glib-2.0/schemas/`	Settings schema
`src/mwp/vcols.css`	`$prefix/share/mwp/`	Colours used by battery widget
`src/mwp/default.layout`	`$prefix/share/mwp/`	Default dock layout
`src/mwp/beep-sound.ogg`	`$prefix/share/mwp/`	Alert sound
`src/mwp/bleet.ogg`	`$prefix/share/mwp/`	Alert sound
`src/mwp/menubar.ui`	`$prefix/share/mwp/`	Menu definition
`src/mwp/mwp.ui`	`$prefix/share/mwp/`	UI definition
`src/mwp/orange.ogg`	`$prefix/share/mwp/`	Alert sound
`src/mwp/sat_alert.ogg`	`$prefix/share/mwp/`	Alert sound
`src/mwp/mwp.desktop`	`$prefix/share/applications/`	Desktop launcher
`src/mwp/mwp_complete.sh`	`$prefix/share/bash-completion/completions/`	bash completion for `mwp`
`src/mwp/pixmaps`	`$prefix/share/mwp/pixmaps/`	UI Icons
`src/mwp/blackbox_decode_complete.sh`	`$prefix/share/bash-completion/completions/`	bash completion for `blackbox-decode`
`src/samples/area-tool/mwp_area_icon.svg`	`$prefix/share/icons/hicolor/scalable/apps/`	Desktop icon
`src/samples/area-tool/mwp-area-planner.desktop`	`$prefix/share/applications/`	Desktop launcher
`docs/mwptools.pdf`	`$prefix/share/doc/mwp/`	(Obsolete) manual
`docs/debian-ubuntu-dependencies.txt`	`$prefix/share/doc/mwp/`	Debian / Ubuntu dependencies
`docs/fedora.txt`	`$prefix/share/doc/mwp/`	Fedora dependencies

Troubleshooting and Hints

MIGRATE FROM A SYSTEM INSTALL TO A USER INSTALL

```
cd build
sudo ninja uninstall
meson --reconfigure --prefix=$HOME/.local
ninja install
```

FIXING BUILD PERMISSIONS

If you install to system locations, it is possible that `sudo ninja install` will write as `root` to some of the install files, and they become non-writable to the normal user.

- In the `build` directory, run `sudo chown -R $USER .`
- Consider migrating to a local install

Easy first-time install on Debian and Ubuntu

- Download the first time build script
- Make it executable `chmod +x deb-install.sh`

- Run it `./deb-install.sh Y`
- Note that the script may ask for a password to install system packages
- The resulting executables are in `~/.local/bin`. Ensure this exists on `$PATH`
- If you get messages like `Removing /home/$USER/.config/mwp/.layout.xml 0` and `Failed to save layout, remains in /tmp/.mwp.xxxxxx.xml` you also need `export XDG_DATA_DIRS=$XDG_DATA_DIRS:$HOME/.local/share`; this is fixed in 206efe2 / 4.295.560 of 2021-10-22.

Help!!!!

ACCESSING THE SERIAL PORT

The user needs to have read / write permissions on the serial port in order to communicate with a flight controller. This is done by adding the user to a group:

- Arch Linux: `sudo usermod -aG uucp $USER`
- Debian / Ubuntu / Fedora (and derivatives): `sudo usermod -aG dialout $USER`
- FreeBSD: `sudo pw group mod dialer -m $USER`
- Windows/WSL: Not needed, no serial passthrough. Use the ser2udp bridge instead.

YOU'VE INSTALLED A NEW VERSION BUT YOU STILL GET THE OLD ONE!

If you used the `deb-install.sh` script, then it installed everything into `$HOME/.local/bin` (and other folders under `~/.local`). This is nice because:

- mwp does not pollute the system directories;
- you don't need `sudo` to install it.

Linux (like most other OS) has the concept of a `PATH`, a list of places where it looks for exec utable files). You can see this from a terminal:

```
## a colon separated list
echo $PATH
```

So check that `$HOME/.local/bin` is on `$PATH`; preferably near the front.

If it is, then the problem may be that the older mwp also exists elsewhere on the PATH, and the system will not re-evaluate the possible chain of locations if it previously found the file it wants.

So, maybe you have an old install. You didn't remove it (alas); so the system thinks that mwp is `/usr/bin/mwp`; in fact it's now `$HOME/.local/bin/mwp`

If `$HOME/.local/bin` is on the PATH before `/usr/bin`, the you have two choices:

```
# reset the path search
hash -r
# mwp, where art thou? Hopefully now is ~/.local/bin
which mwp
# From **this terminal** executing mwp will run the location reported by `which mwp`
```

or

Log out, log in. The PATH will be re-evaluated.

If `$HOME/.local/bin` is not on PATH. then it needs to be added to a login file (`.profile`, `.bashrc`, `.bash_profile` etc.). Modern distros do this for you, however if you've updated an older install you may have to add it yourself.

```
# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi

# set PATH so it includes user's private bin if it exists
if [ -d "$HOME/.local/bin" ] ; then
    PATH="$HOME/.local/bin:$PATH"
fi
```


If an older (perhaps Makefile generated) mwp exists; then you should remove all evidence of an earlier system install.

```
find /usr -iname \*mwp\*
```

review the list and as root, delete the old files. Do similar for blackbox-decode.

If you're content with the list, then (*caveat emptor*):

```
sudo find /usr -iname \*mwp\* -delete
```

You'll still have to remove non-empty directories manually.

"NINJA: ERROR: LOADING 'BUILD.NINJA': NO SUCH FILE OR DIRECTORY

Something, or persons unknown has removed this file.

```
cd mwptools
meson setup --reconfigure  build --prefix ~/.local
cd build
ninja install
```

ERROR: DEPENDENCY "?????" NOT FOUND, TRIED PKGCONFIG

mwp requires a new dependency. This will be documented in the wiki Recent Changes document.

- Install the newly required dependencies
- Rerun your build

2.2 Windows 11 / WSL-G

2.2.1 Intro

As a result of user interest in running mwp on Windows 11 / WSL-G, here's an experiment to see if it's possible. By a Windows neophyte, so if I can install mwp on WSL, anyone can.

There is also an excellent you-tube video tutorial from Marc Hoffmann (in English and German).

Watch on Youtube

2.2.2 Environment

Tested with Windows 11 VM hosted on Arch Linux by the developer.

2.2.3 WSL Installation

- Installed default Ubuntu
- Note that serial ports **still** don't appear to work in WSL (workaround described below)

Windows / WSL Pre-requisites

None other than the serial port issue, Wayland (GUI) and sound just work. The serial port problem can be mitigated by a "serial to IP" solution; mwptools provides `ser2udp` for this purpose.

2.2.4 mwp Installation

Use one of the following:

(a) Install the current release from github.

- Down load the `.deb` file
- `cd` to where ever you saved the `.deb` file
- In the wsl terminal `sudo apt install mwptools_x.y.z_amd64.deb`

Example: using `curl` to download ...

```
$ curl -LO https://github.com/stronnag/mwptools/releases/download/x.y.z/mwptools_x.y.z_amd64.deb
$ sudo apt install ./mwptools_x.y.z_amd64.deb
```

Where `x.y.z` represents the build tag.

(B) UNIFIED FIRST-TIME BUILD SCRIPT (BUILD AND INSTALL FROM SOURCE)

For the initial installation, there is a unified / simplified install / build / install script: Instructions

This installs mwptools and blackbox-tools-inav to `$HOME/.local/bin`.

(C) TRADITIONAL BUILD PROCESS (BUILD AND INSTALL FROM SOURCE)

If you want more control over build options.

If `git` is not pre-installed in WSL, then it will be necessary to install it.

```
sudo apt update && sudo apt upgrade
sudo apt install git
```

Note: `/etc/sudoers` (via `visudo`) was edited to allows the WSL user to run commands as root without asking for a password.

Then it was just a case of cloning the mwp repository and following mwp's instructions (`mwptools/docs/debian-ubuntu-dependencies.txt`), to install the dependencies, thusly:

```
cp mwptools/docs/debian-ubuntu-dependencies.txt /tmp/u.sh
chmod +x /tmp/u.sh
# edit /tmp/u.sh for any optional items ...
sudo /tmp/u.sh Y # "Y" bypasses interactive query / responses
```

Then build and install mwp and optionally the blackbox tools (as `mwptools/docs/debian-ubuntu-dependencies.txt`). Build documentation.

For the optimal blackbox replay, install the flightlog2x tools, either from the Github release or build from source in Linux/WSL.

2.2.5 Running mwp

Compared to Win10/WSL or Cygwin, there is no longer any need to mess around the `DISPLAY` or `udev` settings. No 3rd party X-server, Windows 11 handles all the GUI.

One off changes

- WSL installs a very cut down icon theme that does not provide the all the system / standard icons used by mwp. Fix this by:

  ```
  sudo apt install adwaita-icon-theme-full
  ```

- If you wish to replay blackbox / OTX / BulletGCSS logs, it is necessary to have an IPv6 definition of `localhost` ; WSL's `/etc/hosts` does not provide this:

```
# updated in /etc/hosts for ipv6
::1   localhost ip6-localhost ip6-loopback
```

Note: This was caused by an unnecessary assumption in flightlog2x's `fl2ltm` which is corrected in flightlog2x release (> 0.11.0)

- Then tell WSL to please not break your `hosts` file again

  ```
  ### Add the following entry to /etc/wsl.conf:
  [network]
  generateHosts = false
  ```

- Due font differences, it may be necessary to reduce the font scaling in the mwp 'Flight View' docklet.

```
gsettings set org.mwptools.planner font-fv 10
# if you still have resizing problems, try 9 ....
```

Then you are ready to run mwp.

```
mwp
```

Serial devices

In order to use a serial device, it is necessary to run a "serial to IP" bridge on the Windows side. This application will need to be white-listed in the Windows firewall.

There are a number of existing solutions that may work; **mwp** provides a simple, dedicated `ser2udp` tool that works well.

Building mwp's `ser2udp`

On the **Linux/WSL side**:

- `cd mwptools/src/samples/s2n`
- `make ser2udp.exe`
- copy `ser2udp.exe` to the ~~dark~~ Windows side

On the Windows side:

- Use the Windows firewall settings to allow `ser2udp.exe` to accept UDP traffic.
- Run `ser2udp.exe` ; it will autodetect your serial port. By default this listens on UDP port 17071, you can change this by supplying a second parameter, e.g., to use port 34567. In this case, either define the serial port or use `auto` (auto-detect).

```
> ser2udp.exe auto :34567
## or just let ser2udp autodetect
> ser2udp.exe
External address: fe80::1439:d6de:efcb:97e1%7
External address: 172.29.32.1
```

  The colon is required to define an alternative port.
- `ser2udp` will survive removal of USB devices and attempt to re-connect (e.g. if the FC is rebooted).
- `ser2udp` will *only* attempt to automatically acquire STM32 USB devices (`0483:5740` vid:pid)
- You need to terminate `ser2udp` when you're done with it (e.g. to use the inav configurator in Windows).

USING SER2UDP IN MWP

- On the Linux side, we need to know the IP address (or have a hostname for) the Windows WSL endpoint. Fortunately this happens to be Linux's default gateway, so we can handle it fairly transparently.

It is easily automated by using the magic `__MWP_SERIAL_HOST` name in the serial device.

```
mwp -d udp://__MWP_SERIAL_HOST:17071
# recognised by other tools as well ...
cliterm udp://__MWP_SERIAL_HOST:17071
```

`__MWP_SERIAL_HOST` is resolved as:

- If an environment variable `$MWP_SERIAL_HOST` exists, it is used; else
- The default gateway (which on WSL is the Windows host IP) is used; else
- It will fail, as the literal name is unlikely to exist as a resolvable host name (not even a RFC legal host name).

Thus:

- For WSL and `ser2udp`, in mwp preferences, set the serial device to `udp://__MWP_SERIAL_HOST:17071`
- Or in the shell, for some other scenario, `export MWP_SERIAL_HOST=foobox.org` in the event that you have a valid use case

Launch ser2udp and MWP in one go

- Create a new `txt` file in the same folder where `ser2udp.exe` is located and copy the following lines into that file:

```
@echo off
echo Launching MWP Mission Planner
start wslg.exe -d Ubuntu mwp
echo Waiting for WSL System to boot up
timeout 5
echo Launching Serial to UDP Tool
start "Serial2UDP" cmd /c ser2udp.exe -verbose 1
exit
```

- rename the file with any name and change the extension to `.cmd`
- Create a shortcut anywhere on your PC or in `C:\Users\<username>\AppData\Roaming\Microsoft\Windows\Start Menu\Programs` to pin it to your Start Menu
- Replace the shortcut symbol with the MWP icon from here

BATch file considerations

- The `timeout` value may need changing (or not be needed at all). YMMV.
- Consider adding the `/min` to `cmd` to minimise the `ser2udp` window on startup.

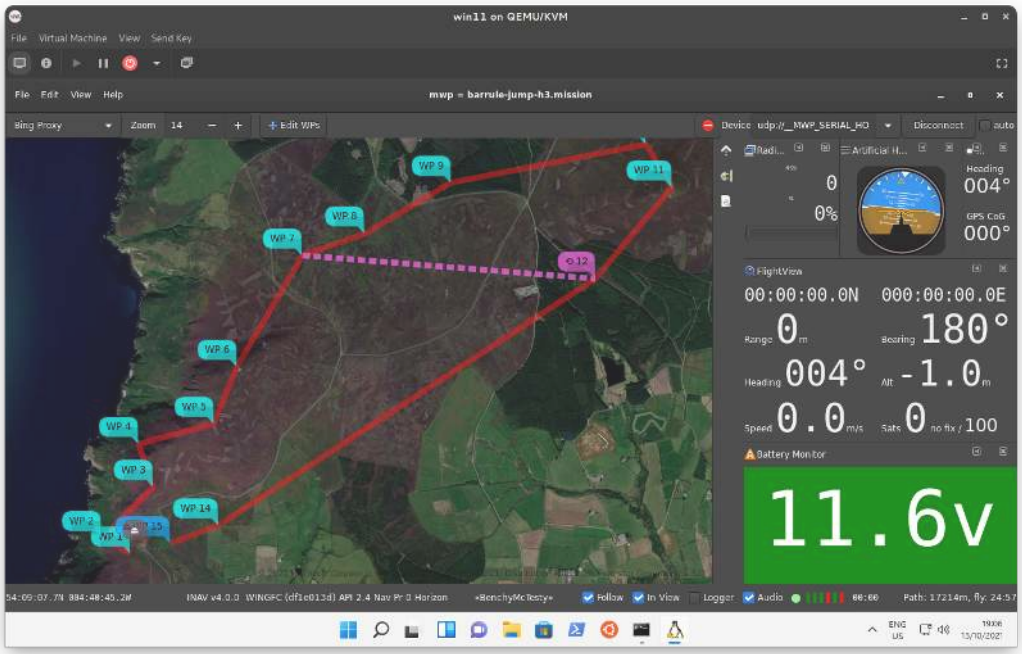
2.2.6 Other packages for additional functionality.

- To replay blackbox logs, you need
  - inav blackbox tools, mandatory
  - flightlog2x / bbl2kml. Provides a much better blackbox replayer than the default shipped with mwp (and you can generate really pretty Google Earth files from blackbox / opentx / bulletgcss logs).
- Terrain Analysis
  - Gnuplot. Check the installer script that it's enabled.

2.2.7 Summary

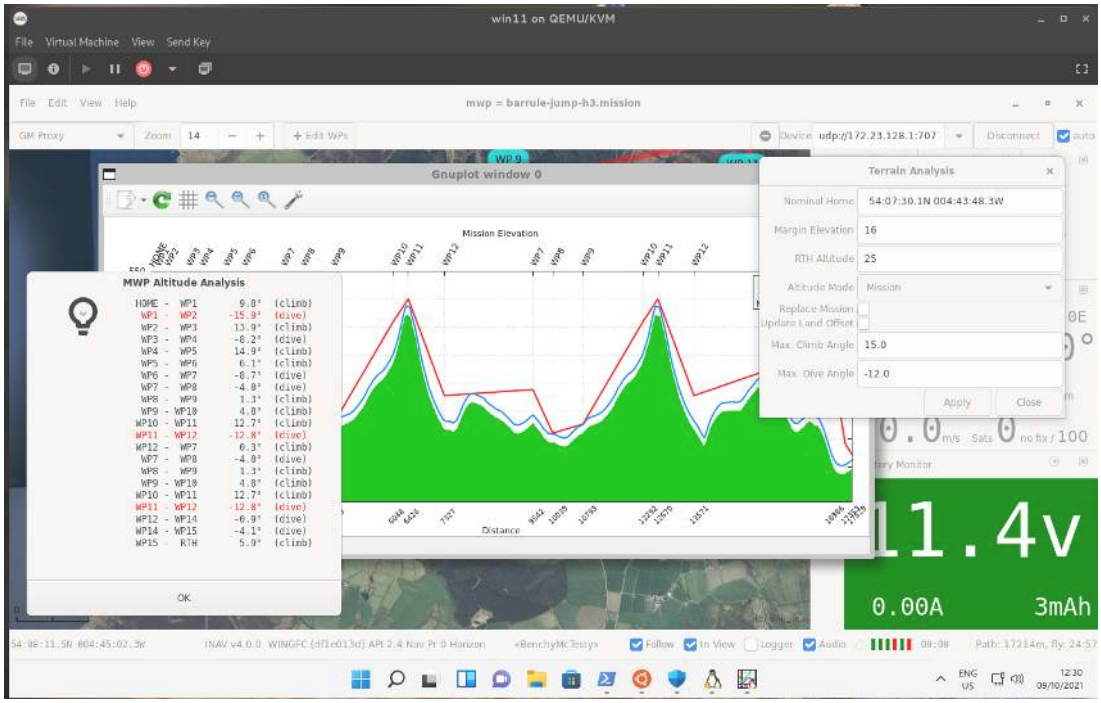
- Much, much better than the prior WSL instances, pity about the lack of support for serial ports (still). Overall, the seamless WSL-g experience is impressive.

Connection via `ser2udp` bridge

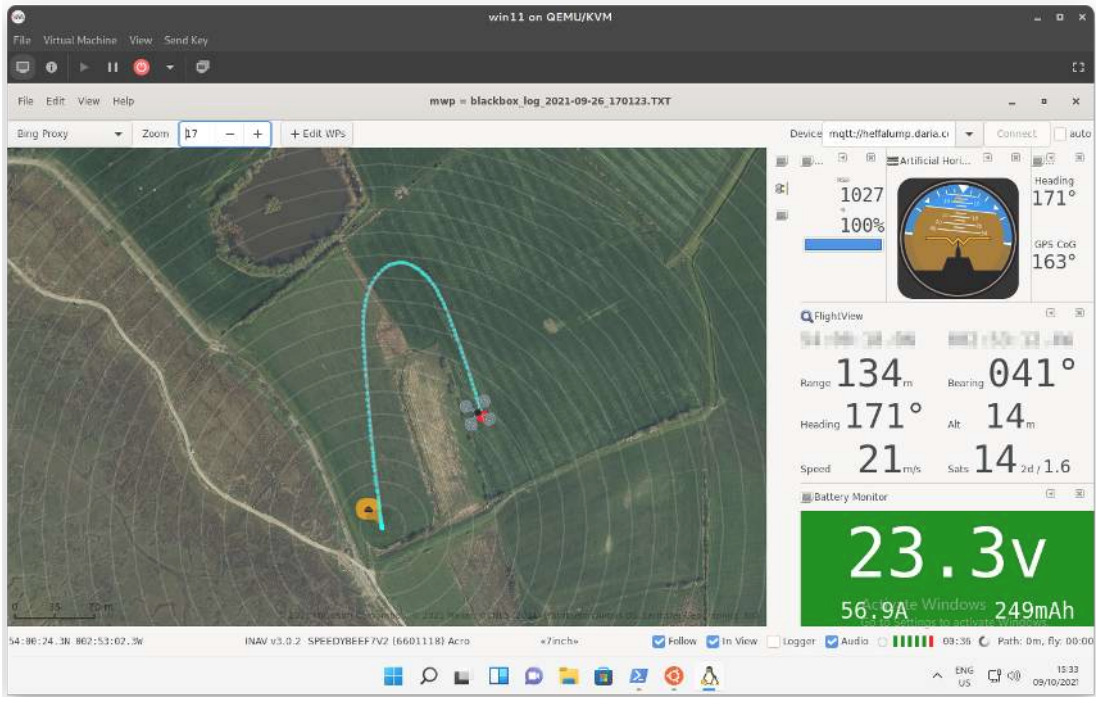


Dark theme, correct system icons installed, connected to FC via `ser2udp`.

Terrain / elevation analysis



Blackbox replay



> **Good enough!**
>
> The user's compass seems good enough for navigation functions (top right widget comparing GPS CoG v. compass heading).

3.1 Power and screen management

There are a number of ways of managing the screen (inhibit screen saver etc.)

- Use an external screen-saver manager such as caffeine
- Use the legacy mwp settings options, for example:

```
org.mwptools.planner atexit 'gsettings set org.gnome.settings-daemon.plugins.power idle-dim true'
org.mwptools.planner atstart 'gsettings set org.gnome.settings-daemon.plugins.power idle-dim false'
```

- Allow mwp to manage screen and power settings, controlled by a setting:

```
gsettings set org.mwptools.planner manage-power true
```

In the first two cases, the setting is somewhat coarse, either requiring the user to click on something and applying to the whole mwp session.

The final case applies only when mwp is receiving push telemetry (LTM, Mavlink, MQTT). Inhibiting IDLE and SUSPEND is performed using the GTK inhibit() API and will thus work with most window managers.

3.2 INAV 4.0 Multi-Mission Support

3.2.1 Overview

In inav 4.0, the FC supports "multi-missions", that is allowing the user to upload and store multiple missions.

The mission to be executed may be set when the mission set is uploaded, or selected by OSD command (or stick command).

3.2.2 mwp support

The means by which this function is provided by the FC is a little inconvenient (for the planner) but expedient; it's hard to see how else it could have been implemented.

In general and in summary, the functionality allows multiple missions to exist in a single "mission file" and either one or all of those mission can be uploaded to the FC.

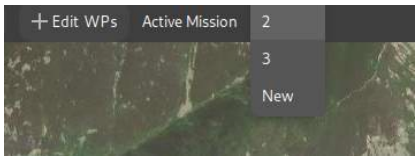
When a "multi-mission" set is downloaded from the FC, mwp will set the active mission to that set as active in the FC.

When a "multi-mission" set is uploaded to the FC, mwp will set the active FC mission to its active mission.

3.2.3 mwp changes

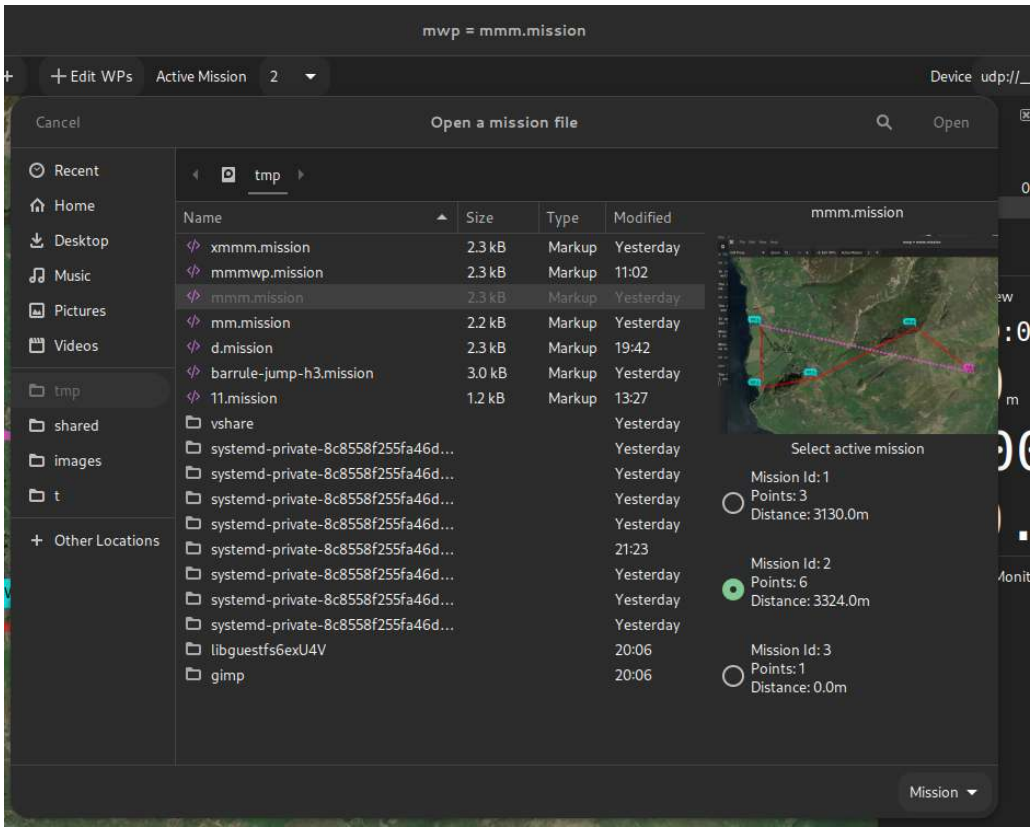
Top Bar

The top bar how includes an "Active Mission" item. This always has mission 1 (the legacy mission) and offers "New", allowing multiple missions to be maintained in one mwp session.



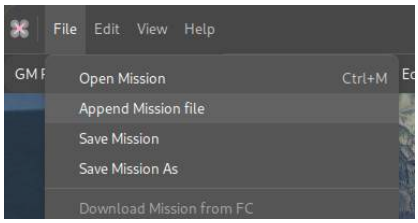
Open Mission file

The file open dialog has a preview pane that displays the missions in a multi-mission file. The user can select the mission to be the active mission.



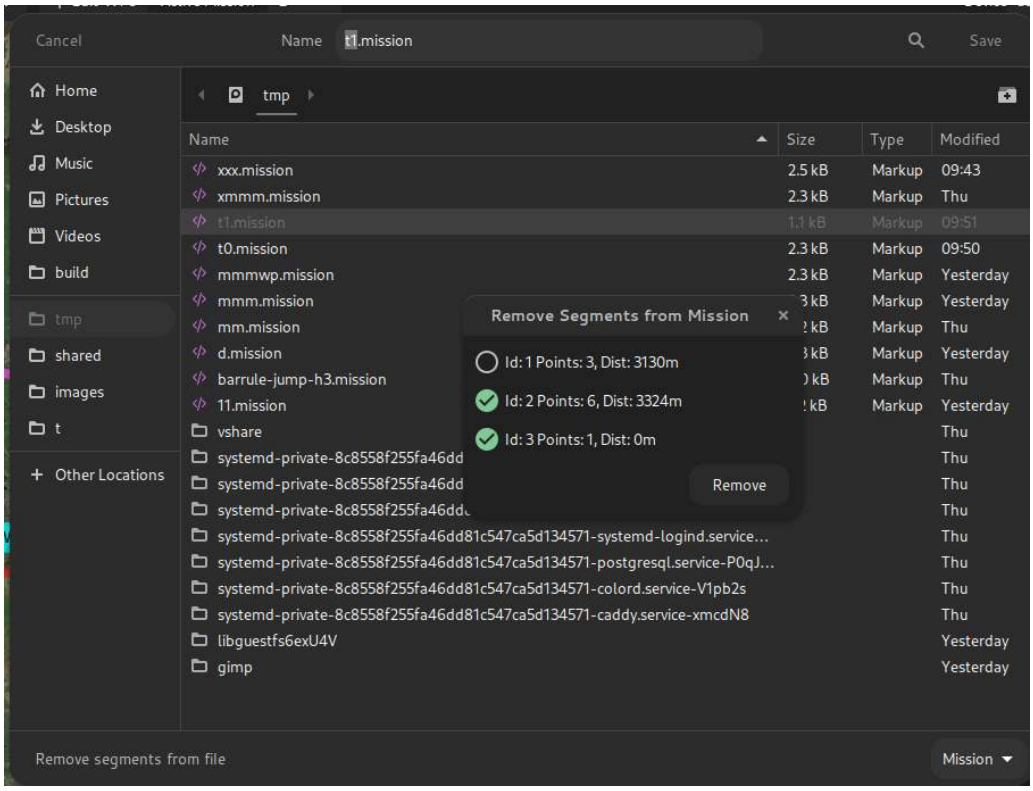
Apppend Mission File

It is now possible to append an existing mission file (which may hold multiple missions) into a multi-mission set. This uses same dialog as Open Mission File.



Save As Mission file

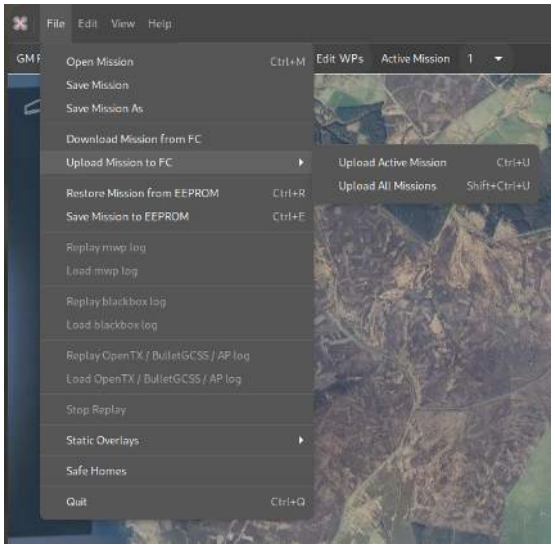
The file "Save as" dialog has an option to exclude specific segments from a multi-mission (via the **Remove Segments from file** button in the following image). Note that "Save" will always save all mission segments.



In this case, only segment 1 of the multi-mission would be saved.

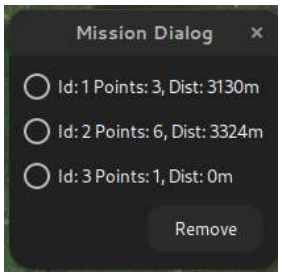
3.2.4 Upload / Download Menu Options

The menu options reflect the new capability to upload all or the active mission. The "Save to EEPROM" option may also change to this pattern in future.



Multi-Mission Manager

The **Edit** menu has a **Multi Mission Manager** option. This allows the user to delete one or more missions from a multi-mission scenario.



3.2.5 FC Limits

inav 4.0 limits the total number of waypoints to 120 and the number of mission segments within a multi-mission scenario to 9.

mwp will allow the user to exceed these limits while creating / editing multi-mission scenarios, but enforces the limits for upload. So it would be possible to open / append files containing a total of (for example) 11 mission segments and 150 WPs. It would be necessary to reduce the mission set to the FC limits before it could be uploaded.

3.2.6 Legacy

mwp still supports prior FC firmware, including MW. It is a bug if this is not the case. However, the user needs to be aware of the capabilities of the FC firmware.

3.2.7 Caveats

- This is all quite novel and has required some significant changes in mwp; however it appears quite stable.
- By default, mwp writes mission files in "reset / per segment metadata" style.
- Multi-mission files may be written in the (IMO) ugly / confusing "sequential" style required by the configurator if the environment variable CFG_UGLY_XML is set (to any value). See the schema definition for details. mwp can read either style.

3.2.8 Example XML multi-mission file

```
<?xml version="1.0" encoding="utf-8"?>
<mission>
  <!--mw planner 0.01-->
  <version value="42"></version>
  <mwp save-date="2021-11-11T07:22:43+0000" zoom="14" cx="-3.2627249" cy="54.5710168" home-x="-3.2989342" home-y="54.5707123" generator="mwp
(mwptools)"><details><distance units="m" value="3130"></distance><nav-speed units="m/s" value="10"></nav-speed><fly-time units="s" value="319"></fly-
time><loiter-time units="s" value="0"></loiter-time></details></mwp>
  <missionitem no="1" action="WAYPOINT" lat="54.5722109" lon="-3.2869291" alt="660" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="2" action="WAYPOINT" lat="54.5708178" lon="-3.2642698" alt="755" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="3" action="WAYPOINT" lat="54.5698227" lon="-3.2385206" alt="513" parameter1="0" parameter2="0" parameter3="0" flag="165"></missionitem>
  <mwp save-date="2021-11-11T07:22:43+0000" zoom="15" cx="-3.2778311" cy="54.5568837" home-x="-3.2983737" home-y="54.5622331" generator="mwp
(mwptools)"><details><distance units="m" value="9029"></distance><nav-speed units="m/s" value="10"></nav-speed><fly-time units="s" value="929"></fly-
time><loiter-time units="s" value="0"></loiter-time></details></mwp>
  <missionitem no="1" action="WAYPOINT" lat="54.5599696" lon="-3.2958555" alt="236" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="2" action="WAYPOINT" lat="54.5537978" lon="-3.2958555" alt="136" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="3" action="WAYPOINT" lat="54.5547933" lon="-3.2864141" alt="238" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="4" action="WAYPOINT" lat="54.5597705" lon="-3.2695913" alt="570" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="5" action="WAYPOINT" lat="54.5552910" lon="-3.2598066" alt="502" parameter1="0" parameter2="0" parameter3="0" flag="0"></missionitem>
  <missionitem no="6" action="JUMP" lat="0.0000000" lon="0.0000000" alt="0" parameter1="1" parameter2="1" parameter3="0" flag="165"></missionitem>
  <mwp save-date="2021-11-11T07:22:43+0000" zoom="20" cx="-3.2501935" cy="54.5714148" generator="mwp (mwptools)"><details><distance units="m" value="0"></
distance></details></mwp>
  <missionitem no="1" action="WAYPOINT" lat="54.5714148" lon="-3.2501935" alt="50" parameter1="0" parameter2="0" parameter3="0" flag="165"></missionitem>
</mission>
```

Download sample mission

3.3 BulletGCSS Telemetry

3.3.1 mwp requirements

mwp works with the web-based Ground Control Station BulletGCSS MQTT protocol, tested with both a `fl2mqtt` simulation and a recorded live session.

The MQTT component is build if either `paho-mqtt` or `mosquitto` libraries are detected; `paho-mtqq` is preferred.

```
## Debian / Ubuntu ##
### Debian testing / Ubuntu 20.10 + for paho ###
sudo apt install libpaho-mqtt-dev
# or #
sudo apt install libmosquitto-dev

## Arch ##

yay -S paho-mqtt-c-git  ## or you favourite AUR helper
# or #
sudo pacman -S mosquitto

## Fedora ##

dnf install paho-c-devel
# or #
dnf install mosquitto-devel

## FreeBSD ##

## paho-mqtt
# Clone github repo and build from source. Configure with cmake -DPAHO_WITH_SSL=true ..
git clone https://github.com/eclipse/paho.mqtt.c.git
cd paho.mqtt.c
mkdir build
cd build
cmake -DPAHO_WITH_SSL=true ..
make && sudo make install

# or #
sudo pkg install mosquitto
```

If you have both `paho-mqtt` and `mosquitto` installed, then `paho-mqtt` is preferred.

3.3.2 Usage

Once mwp is built with a MQTT library, you can use an MQTT URL as a device name, for example for the demo that runs every other hour (00:00, 02:00 .. 22:00) UTC on `broker.emqx.io` with topic `org/mwptools/mqtt/otxplayer`, the mqtt URI for mwp would be:

```
mqtt://broker.emqx.io/org/mwptools/mqtt/otxplayer
```

Or in general:

```
mqtt://[user[:pass]@]broker[:port]/topic[?cafile=file]
```

Note: * port is the mqtt port (typically and by default 1883), not the websocket port. * if you want to use TLS, then the port will be different, often 8883, and you might need to provide the broker's CA file. * As mwp uses a pseudo-URL for the broker,topic etc, the topic should comply with rules for a URL rather than the more relaxed MQTT topic specification. This is a feature.

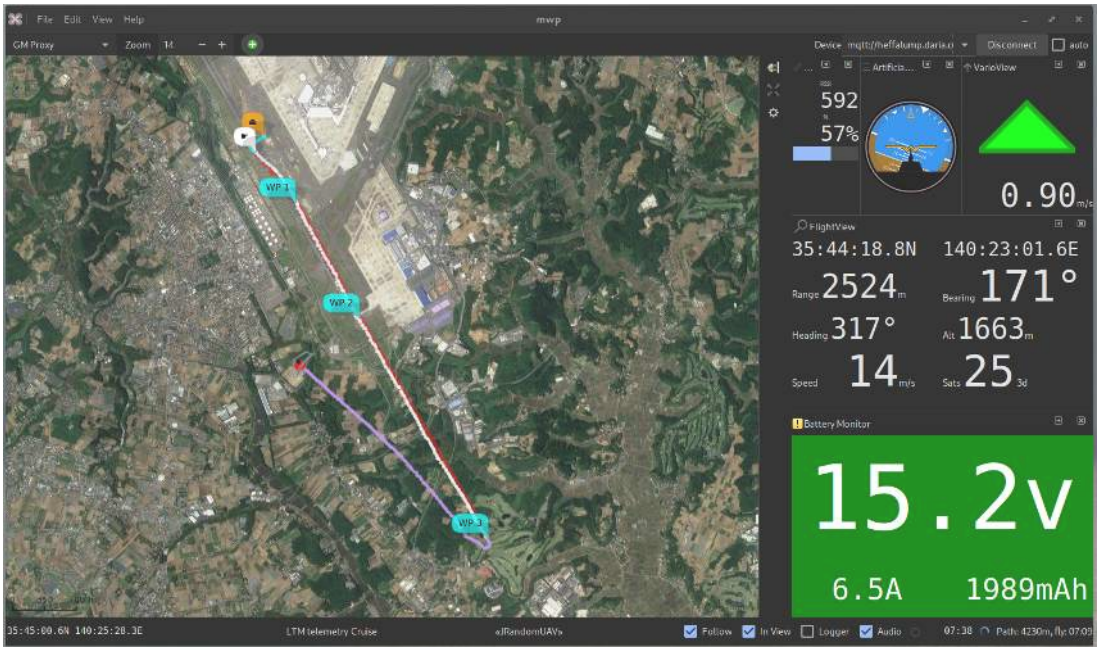
The scheme part (`mqtt://`) in the example is interpreted as:

- `ws://` - Websocket (vice TCP socket), ensure the websocket port is also specified, requires 'paho-mqtt' as the provider.
- `wss://` - Encrypted websocket, ensure the TLS websocket port is also specified. TLS validation is performed using the operating system. Not supported by `mosquitto`; requires `paho-mqtt` 1.39 or later.
- `mqtts://`, `ssl://` - Secure (TLS) TCP connection. Ensure the TLS port is specified. TLS validation is performed using the operating system, unless `cafile` is provided.
- `mqtt://` - TCP connection. If `?cafile=file` is specified, then that is used for TLS validation (and the TLS port should be specified).

MQTT looks like an incredibly elegant solution to long range telemetry.

More information on the BulletGCSS website and BulletGCSS wiki

See also fl2mqtt, a tool to replay Blackbox and OpenTx logs as MQTT and BulletGCSS mosquitto hosting guide for hosting your own MQTT broker.



3.4 mwp and inav 3.0 Mission Updates

3.4.1 Overview

inav 3.0 adds a couple of changes to inav mission planning:

- Absolute WP altitudes
- Land WP ground elevation setting

Absolute WP altitudes

For Multiwii and inav prior to 3.0, waypoint altitudes were always relative to the arming location. If you always fly in a flat area, or always arm at the same point, this wasn't really an issue; you could always use mwp's terrain analysis to check that you'd clear any obstructions.

However, if you armed some (vertical) distance from the arming point assumed when the plan was created, the absolute, (AMSL) elevation of the WP would differ by the ground difference between the assumed arming point at planning time and the actual arming point at take off. In the worst case (arming at an 'zero' absolute elevation well below the 'assumed at planning time' location), this could result in automated flight into terrain, which is generally undesirable.

Absolute mission altitudes addresses this issue, as the AMSL elevation of the WP is fixed and does not depend on arming location.

Land WP ground elevation setting

A similar issue existed prior to inav 3.0 for the LAND WP; the initial implementation assumed that the LAND WP site ground elevation was at approximately the same ground elevation as the arming location. inav computes landing behaviour based on relative altitude from home; if the actual LAND site was lower than home, then the descent would be slow; if it was higher, then slowdown might not occur and there would be a hard landing (for MR). For FW the final approach and motor-off would be sub-optimal.

The required land elevation uses the `P2` WP parameter, **in metres.**

- If LAND is a relative altitude WP, then this is the altitude difference between the assumed home and the LAND location.
- If LAND is an absolute altitude WP, then this is the absolute (AMSL) altitude of the LAND location.

3.4.2 mwp support for 3.0 features

mwp supports the new feature in the Mission Editor and Terrain Analysis.

Mission Editor

The mission editor gains two new context message options:

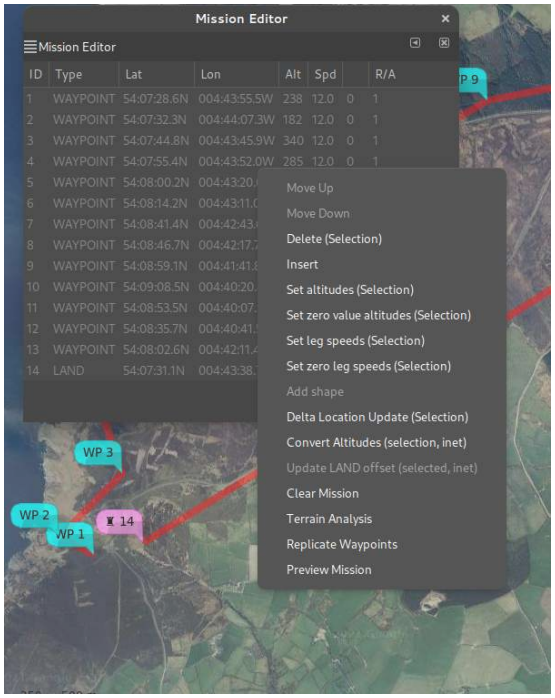
- Convert Altitudes (selection, inet)
- Update LAND offset (selected, inet)

The text in parentheses indicating that a selection of point and an internet connection is potentially needed.

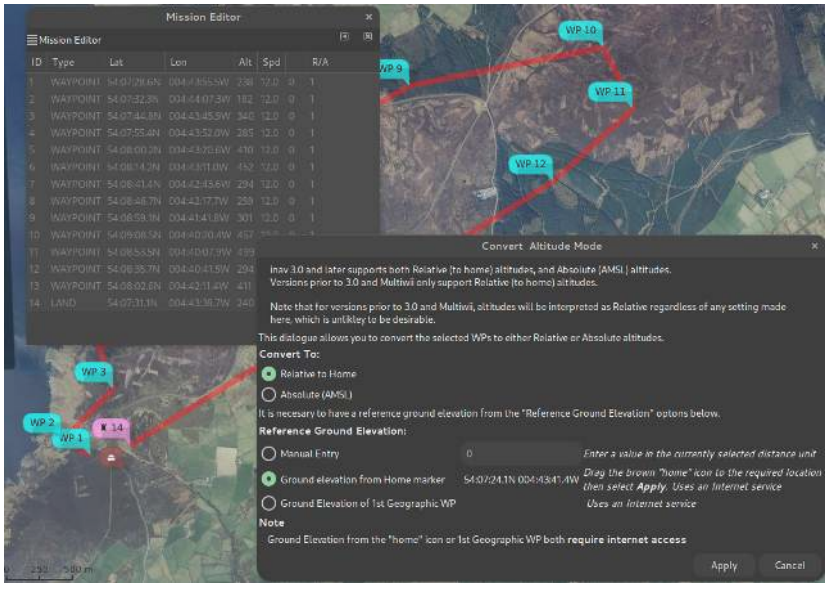
- Internet connectivity is needed in order to perform conversion between absolute and relative modes, unless manual entry of the home elevation is chosen.
- Internet connectivity is needed for automatic LAND elevation adjustment, as mwp needs to known the LAND site ground elevation.
- However, the values can all the edited manually if necessary:

In the image below:

- The R/A column indicates the altitude mode (**R**elative to home, **A**bsolute).
- "Convert Altitudes ..." is enabled, because geospatial WPs are selected.
- "Update LAND offset ..." is not enabled; it requires a single LAND WP to be selected.



When "Convert Altitudes ..." is invoked, the user is presented with the following:

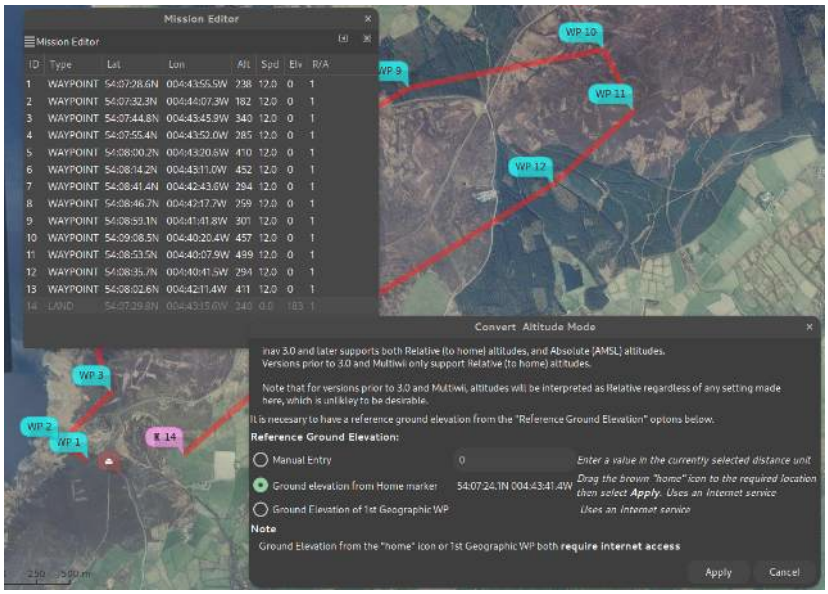


- The user can select to convert the selected WPs to either Relative or Absolute. Only geospatial WPs are converted, and if the WP is already of the selected mode, it will be ignored.
- The user can select the reference home altitude by:
  - Entering a manual value, does not require an internet connection.
  - Dragging the brown "home" icon to the required position
  - Using the position of the 1st geographic WP, which does not have to be in the conversion selection.

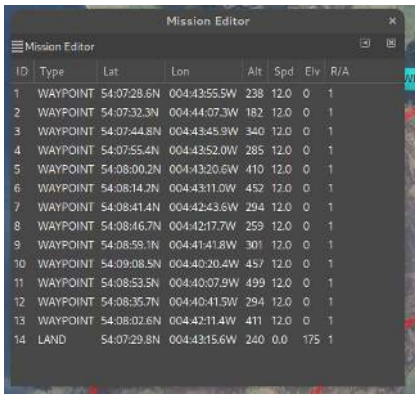
If "Apply" is clicked, the conversion proceeds, downloading elevation data from the internet as required. Cancel closes the dialogue and clears the selection from the Mission Editor.

When "Update LAND offset ..." is invoked, the user is presented with a similar dialogue, without the Altitude Mode selection, as that's implicit from the selected waypoint.

In the image below, WP14 has been moved down the valley:



When this is applied, the WP14 value (parameter 2, "Elv" in the cell headers), should decrease, which it does, from 183m to 175m (AMSL).

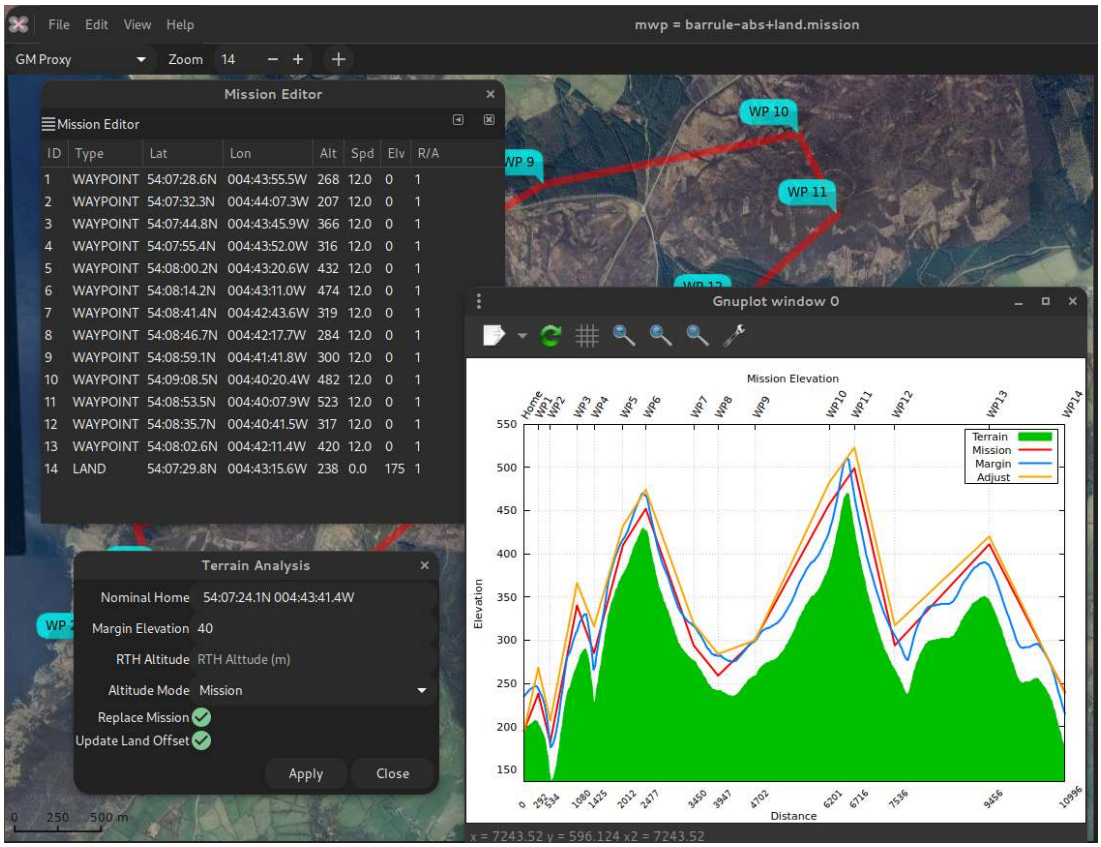


ID	Type	Lat	Lon	Alt	Spd	Elv	R/A
1	WAYPOINT	54:07:28.6N	004:43:55.5W	238	12.0	0	1
2	WAYPOINT	54:07:32.3N	004:44:07.3W	182	12.0	0	1
3	WAYPOINT	54:07:44.8N	004:43:45.9W	340	12.0	0	1
4	WAYPOINT	54:07:55.4N	004:43:52.0W	285	12.0	0	1
5	WAYPOINT	54:08:00.2N	004:43:20.6W	410	12.0	0	1
6	WAYPOINT	54:08:14.2N	004:43:11.0W	452	12.0	0	1
7	WAYPOINT	54:08:41.4N	004:42:43.6W	294	12.0	0	1
8	WAYPOINT	54:08:46.7N	004:42:17.7W	259	12.0	0	1
9	WAYPOINT	54:08:59.1N	004:41:41.8W	301	12.0	0	1
10	WAYPOINT	54:09:08.5N	004:40:20.4W	457	12.0	0	1
11	WAYPOINT	54:08:53.5N	004:40:07.9W	499	12.0	0	1
12	WAYPOINT	54:08:35.7N	004:40:41.5W	294	12.0	0	1
13	WAYPOINT	54:08:02.6N	004:42:11.4W	411	12.0	0	1
14	LAND	54:07:29.8N	004:43:15.6W	240	0.0	175	1

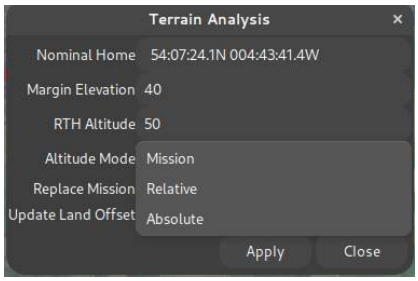
Terrain Analysis

mwp's terrain analysis function has been upgraded to handle inav 3.0 features (Relative / Absolute Elevations, Land Ground Elevation). If you're using the older (ruby) terrain analysis tool, you won't see the new features. The mwp terrain analysis article also describes the new analysis tool.

In the image below, the dialogue has been enhanced to allow selection of the altitude mode and adjustment of LAND elevation. The orange graph line shows the generated mission with a 40m clearance of all obstacles.



The user can select the following altitude modes:



- Mission - use the altitude mode from the mission
- Relative to home
- Absolute (AMSL).

3.4.3 Attribute editing

Of course, it's not necessary to use the new dialogues to set or change the new inav 3.0 features.

- The `parameter3` value sets the altitude mode 0 = relative to home (legacy default), 1 = Absolute.
- The `altitude` value is interpreted according to `parameter3`
- For a LAND WP `parameter2` defines the LAND WP ground elevation; if `parameter3` is 0, then it's relative to home, if `parameter3` is 1, then it's absolute (AMSL).

3.4.4 Further reading

The inav wiki describes WP mission parameters in some detail.

Discussion of the meaning of "sea level". It's confusing.

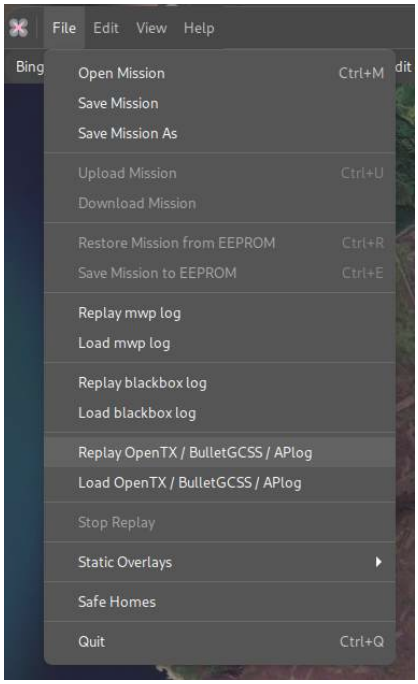
3.5 Ardupilot log replay

3.5.1 Requirements

It is possible to replay Ardupilot logs in the same way as one can replay blackbox, OpenTX / EdgeTX and BulletGCCS logs. This also requires flightlog2x tools of the same (0.11.0) or more recent vintage.

- It is necessary to install an Ardupilot tool to decode the logs mavlogdump.py.

As the author does not have any (useful) AP logs, contributions are welcome.



3.6 Flite Text to Speech

3.6.1 Overview

mwp can use the `flite` text to speech engine (as well as **espeak** or **speech-dispatcher**. Flite is enabled if:

- You have the flite development files installed

Flite is available at run-time if:

- The flite version is 2.0 or later.

Unfortunately, it is non-trivial to detect the flite version at build time.

Flite provides reasonable quality voices with low overhead, including some female voices.

3.6.2 Configuration

Flite is configured using two `gsettings` keys:

Key	Usage
`speech-api`	Defines the speech api to be used, one of `none`, `espeak`, `speechd` or `flite`
`flite-voice`	The voice file to be used. If not specified, the internal `slt` (female) voice is used. The value takes the absolute path name to a voice file, optionally followed by a `,` and a floating point speed factor (see below)

```
$ gsettings set org.mwptools.planner speech-api flite
$ gsettings set org.mwptools.planner flite-voice-file /home/jrh/.config/mwp/cmu_us_clb.flitevox,0.9
```

3.6.3 Discussion

Voice Files

flite can use external voice files that provide better quality than the built-in voices. Your distro may provide these voice files in an optional package, or you can download from http://www.festvox.org, eg. for flite 2.1 http://www.festvox.org/flite/packed/flite-2.1/voices/ (replace 2.1 with 2.0 etc., not all the 2.1 voices may exist for 2.0). The following script will bulk download the non-Indic voices; you can test them out with the `flite` application, or mwp's `ftest` application).

```
#!/bin/bash

BASE=http://www.festvox.org/flite/packed/flite-2.1/voices

for V in cmu_us_aew.flitevox cmu_us_ahw.flitevox cmu_us_aup.flitevox \
  cmu_us_awb.flitevox cmu_us_axb.flitevox cmu_us_bdl.flitevox \
  cmu_us_clb.flitevox cmu_us_eey.flitevox cmu_us_fem.flitevox \
  cmu_us_gka.flitevox cmu_us_jmk.flitevox cmu_us_ksp.flitevox \
  cmu_us_ljm.flitevox cmu_us_lnh.flitevox cmu_us_rms.flitevox \
  cmu_us_rxr.flitevox cmu_us_slp.flitevox cmu_us_slt.flitevox
do
  wget -P . $BASE/$V
done
```

Replay Speed

The default replay speed for some flite voices is rather slow. The optional rate setting in the gsettings `flite-voice-file` key may be used to increase the rate.

3.6.4 Test

`mwptools/samples/flite` provides a test programme for assessing flite voices.


```
$ cd  mwptools/samples/flite
$ make
$ ./ftest < mwp.txt # speak mwp like phrases using default voice
$ ./ftest cmu_us_clb.flitevox,0.9 < mwp.txt # speak mwp like phrases using external voice file, with relative rate (0.9)
```

Note: this test programme will work with flite 1.x; though you can only use the default 'kal' voice (you cannot load 'better' voices).

3.7 DBus API

3.7.1 Introduction

mwp provides a Dbus API to permit remote control or monitoring of mwp by third party applications.

Dbus is a common Linux API for inter-process communications, and can be used from most programming languages. `mwptools/samples` provides examples in `python`, `ruby` and `bash`.

It is intended that that the `ruby` examples cover the majority of the API and provide canonical examples of usage.

As this is a developer topic, please raise Github issues if clarification is needed or you have a use case that would benefit from extending the API.

Please also note that the definitive definition of the DBus API is provided by DBus inspection.

3.7.2 DBus object and interface

The mwp Dbus API exists on the session bus when mwp is running.

- Object Path: `/org/mwptools/mwp`
- Interface: `"org.mwptools.mwp"`

3.7.3 Flight Status and geo-location information

A set of APIs is provided for synchronous and asynchronous (signals, event by event) notification of vehicle status and location. A use case might be to drive an antenna tracker.

Flight status and geo-location methods

GETSTATENAMES

Returns human-readable names for the FC 'state' returned by `GetState`, as an array of strings. The size of the array is the return value.

```
int GetStateNames(out string[] states_names)
```

```
<method name="GetStateNames">
  <arg type="as" name="names" direction="out"/>
  <arg type="i" name="result" direction="out"/>
</method>
```

GETSTATE

Returns the FC 'state'. 0 if unarmed. Human-readable state names are provided by `GetStateNames()`.

```
int GetState()
```

```
<method name="GetState">
  <arg type="i" name="result" direction="out"/>
</method>
```

GETHOME

Returns the home location as latitude (WGS84 decimal degrees), longitude (WGS84 decimal degrees) and relative altitude (metres, which should always be 0).

```
void GetHome(out double latitude, out double longitude, out int32 altitude)
```

```
<method name="GetHome">
  <arg type="d" name="latitude" direction="out"/>
  <arg type="d" name="longitude" direction="out"/>
```



```
  <arg type="i" name="altitude" direction="out"/>
</method>
```

GETLOCATION

Returns the vehicle location as latitude (WGS84 decimal degrees), longitude (WGS84 decimal degrees) and relative altitude (metres).

```
void GetLocation(out double latitude, out double longitude, out int32 altitude)
```

```
<method name="GetLocation">
  <arg type="d" name="latitude" direction="out"/>
  <arg type="d" name="longitude" direction="out"/>
  <arg type="i" name="altitude" direction="out"/>
</method>
```

GETSATS

Returns the number of satellites and the fix type (0=nofix, 1=undefined, 2=2D fix, 3=3D fix).

```
void GetSats(out uint8 number_satellites, uint8 fix_type)
```

```
<method name="GetSats">
  <arg type="y" name="nsats" direction="out"/>
  <arg type="y" name="fix" direction="out"/>
</method>
```

GETVELOCITY

Returns the vehicle speed (m/s) and course (degrees), GPS provided.

```
void GetVelocity(out uint32 speed, out uint32 course)
```

```
<method name="GetVelocity">
  <arg type="u" name="speed" direction="out"/>
  <arg type="u" name="course" direction="out"/>
</method>
```

GETPOLARCOORDINATES

Returns the vehicle location as polar coordinates relative the home position: Range (m), Bearing (degrees) **from home to vehicle**, azimuth (elevation angle, degrees).

```
void GetPolarCoordinates(out uint32 range, out uint32 direction, out uint32 azimuth)
```

```
<method name="GetPolarCoordinates">
  <arg type="u" name="range" direction="out"/>
  <arg type="u" name="direction" direction="out"/>
  <arg type="u" name="azimuth" direction="out"/>
</method>
```

GETWAYPOINTNUMBER

Returns the next WP number (en-route to) or -1 if not flying WPs.

```
int GetWaypointNumber()
```

```
<method name="GetWaypointNumber">
  <arg type="i" name="result" direction="out"/>
</method>
```

Flight status and geo-location signals

A number of signals (asynchronous event by event notifications) are issues for changes in state and location. This avoids applications having to poll for changes. In general, the data returned is that for the eponymous Get* methods.

All location signals may be rate limited by the `DbusPosInterval` property in order to avoid excessive DBus traffic.

HOMECHANGED

Notifies that the home position has changed.


```
signal void HomeChanged (double latitude, double longitude, int altitude)
```

```
<signal name="HomeChanged">
  <arg type="d" name="latitude"/>
  <arg type="d" name="longitude"/>
  <arg type="i" name="altitude"/>
</signal>
```

LOCATIONCHANGED

Notifies that the vehicle position has changed (geographic coordinates).

```
signal void location_changed (double latitude, double longitude, int altitude)
```

```
<signal name="LocationChanged">
  <arg type="d" name="latitude"/>
  <arg type="d" name="longitude"/>
  <arg type="i" name="altitude"/>
</signal>
```

POLARCHANGED

Notifies that the vehicle position has changed relative to home (polar coordinates).

```
signal void polar_changed(uint32 range, uint32 direction, uint32 azimuth)
```

```
<signal name="PolarChanged">
  <arg type="u" name="range"/>
  <arg type="u" name="direction"/>
  <arg type="u" name="azimuth"/>
</signal>
```

VELOCITYCHANGED

Notifies that the vehicle velocity (course or speed) has changed.

```
signal void velocity_changed(uint32 speed, uint32 course)
```

```
<signal name="VelocityChanged">
  <arg type="u" name="speed"/>
  <arg type="u" name="course"/>
</signal>
```

STATECHANGED

Notifies that the vehicle 'state' has changed.

```
signal void StateChanged(int32 state)
```

```
<signal name="StateChanged">
  <arg type="i" name="state"/>
</signal>
```

SATSCHANGED

Notifies that the satellite status has changed.

```
signal void SatsChanged(uint8 nsats, uint8 fix)
```

```
<signal name="SatsChanged">
  <arg type="y" name="nsats"/>
  <arg type="y" name="fix"/>
</signal>
```

WAYPOINTCHANGED

Notifies that the current WP number has changed.

```
signal void WaypointChanged(int32 wp)
```



```
<signal name="WaypointChanged">
  <arg type="i" name="wp"/>
</signal>
```

Application Status

QUIT

The `Quit` signal is issued when mwp exits, allowing a dependent application to close down gracefully or take action to wait for the bus to reappear.

```
Quit()
```

```
<signal name="Quit">
</signal>
```

3.7.4 Properties

DbusPosInterval

```
uint dbus_pos_interval
```

Defines rate limiting for all position related signals. The value represents the minimum update interval in 0.1s intervals.

- 0 disables rate limiting
- 2 is the default, and matches the best LTM rate of 5Hz
- a large value (e.g. 999999, greater than a realistic flight time), would effectively disable event by event positional updates.

3.7.5 Serial Port and Mission management

A set of APIs is provided for remote serial port and mission management.

3.7.6 Serial Ports

GetDevices

The `GetDevices` API returns a list of the serial devices known to the mwp instance, as an array of strings.

```
void GetDevices(out string[]device_names)
```

```
<method name="GetDevices">
  <arg type="as" name="devices" direction="out"/>
</method>
```

ConnectionStatus

The `ConnectionStatus` API returns a boolean status as to whether `mwp` is connected to a serial device, and if connected, the name of the device.

```
bool ConnectionsStatus(out string device_name)
```

```
<method name="ConnectionStatus">
  <arg type="s" name="device" direction="out"/>
  <arg type="b" name="result" direction="out"/>
</method>
```

ConnectDevice

The `ConnectDevice` API attempts connection to the given device, and returns the status of the operation (`true` => connected).

```
bool ConnectDevice(string device_name)
```



```
<method name="ConnectDevice">
  <arg type="s" name="device" direction="in"/>
  <arg type="b" name="result" direction="out"/>
</method>
```

3.7.7 Mission Management

Somewhat inconsistent set of mission management APIs. Note these are not yet multi-mission aware.

ClearMission

Clears the current mission from mwp.

```
void ClearMission()
```

```
<method name="ClearMission">
</method>
```

SetMission

Opens a mission in mwp from an XML or JSON document, returns the number of mission points.

```
int SetMission(string mission)
```

```
<method name="SetMission">
  <arg type="s" name="mission" direction="in"/>
  <arg type="u" name="result" direction="out"/>
 </method>
```

LoadMission

Opens a mission in mwp from an mission file, returns the number of mission points.

```
int LoadMission(string filename)
```

```
<method name="LoadMission">
  <arg type="s" name="filename" direction="in"/>
  <arg type="u" name="result" direction="out"/>
</method>
```

UploadMission

Loads the current mwp mission into the flight controller, optionally saving to it EEPROM. Returns the number of mission points.

```
int UploadMission(bool to_eeprom)
```

```
<method name="UploadMission">
  <arg type="b" name="to_eeprom" direction="in"/>
  <arg type="i" name="result" direction="out"/>
</method>
```

3.7.8 Examples

- `samples/mwp-dbus-test.sh`
- `samples/mwp-dbus.rb`
- `samples/mwp-dbus.py`
- `samples/mwp-dbus-loc.rb`
- `samples/mwp-dbus-loc.py`
- `samples/mwp-dbus-to-gpx.rb`

3.7.9 Introspection

Not withstanding the state of the documentation, it is possible introspect the API. Note that mwp must be running for the API to exist. The document returned by DBus introspection **is** the definitive definition of the API.

```
# Note samples/mwp-dbus-loc.rb also provides introspection.
$ samples/mwp-dbus-test.sh introspect
<!DOCTYPE node PUBLIC "-//freedesktop//DTD D-BUS Object Introspection 1.0//EN"
                      "http://www.freedesktop.org/standards/dbus/1.0/introspect.dtd">
<!-- GDBus 2.60.3 -->
<node>
  <interface name="org.freedesktop.DBus.Properties">
    <method name="Get">
      <arg type="s" name="interface_name" direction="in"/>
      <arg type="s" name="property_name" direction="in"/>
      <arg type="v" name="value" direction="out"/>
    </method>
    <method name="GetAll">
      <arg type="s" name="interface_name" direction="in"/>
      <arg type="a{sv}" name="properties" direction="out"/>
    </method>
    <method name="Set">
      <arg type="s" name="interface_name" direction="in"/>
      <arg type="s" name="property_name" direction="in"/>
      <arg type="v" name="value" direction="in"/>
    </method>
    <signal name="PropertiesChanged">
      <arg type="s" name="interface_name"/>
      <arg type="a{sv}" name="changed_properties"/>
      <arg type="as" name="invalidated_properties"/>
    </signal>
  </interface>
  <interface name="org.freedesktop.DBus.Introspectable">
    <method name="Introspect">
      <arg type="s" name="xml_data" direction="out"/>
    </method>
  </interface>
  <interface name="org.freedesktop.DBus.Peer">
    <method name="Ping"/>
    <method name="GetMachineId">
      <arg type="s" name="machine_uuid" direction="out"/>
    </method>
  </interface>
  <interface name="org.mwptools.mwp">
    <method name="GetStateNames">
      <arg type="as" name="names" direction="out"/>
      <arg type="i" name="result" direction="out"/>
    </method>
    <method name="GetVelocity">
      <arg type="u" name="speed" direction="out"/>
      <arg type="u" name="course" direction="out"/>
    </method>
    <method name="GetPolarCoordinates">
      <arg type="u" name="range" direction="out"/>
      <arg type="u" name="direction" direction="out"/>
      <arg type="u" name="azimuth" direction="out"/>
    </method>
    <method name="GetHome">
      <arg type="d" name="latitude" direction="out"/>
      <arg type="d" name="longitude" direction="out"/>
      <arg type="d" name="altitude" direction="out"/>
    </method>
    <method name="GetLocation">
      <arg type="d" name="latitude" direction="out"/>
      <arg type="d" name="longitude" direction="out"/>
      <arg type="d" name="altitude" direction="out"/>
    </method>
    <method name="GetState">
      <arg type="i" name="result" direction="out"/>
    </method>
    <method name="GetSats">
      <arg type="y" name="nsats" direction="out"/>
      <arg type="y" name="fix" direction="out"/>
    </method>
    <method name="SetMission">
      <arg type="s" name="mission" direction="in"/>
      <arg type="u" name="result" direction="out"/>
    </method>
    <method name="LoadMission">
      <arg type="s" name="filename" direction="in"/>
      <arg type="u" name="result" direction="out"/>
    </method>
    <method name="ClearMission">
    </method>
    <method name="GetDevices">
      <arg type="as" name="devices" direction="out"/>
    </method>
    <method name="UploadMission">
      <arg type="b" name="to_eeprom" direction="in"/>
      <arg type="i" name="result" direction="out"/>
    </method>
```



```
    <method name="ConnectionStatus">
      <arg type="s" name="device" direction="out"/>
      <arg type="b" name="result" direction="out"/>
    </method>
    <method name="ConnectDevice">
      <arg type="s" name="device" direction="in"/>
      <arg type="b" name="result" direction="out"/>
    </method>
    <signal name="HomeChanged">
      <arg type="d" name="latitude"/>
      <arg type="d" name="longitude"/>
      <arg type="i" name="altitude"/>
    </signal>
    <signal name="LocationChanged">
      <arg type="d" name="latitude"/>
      <arg type="d" name="longitude"/>
      <arg type="i" name="altitude"/>
    </signal>
    <signal name="PolarChanged">
      <arg type="u" name="range"/>
      <arg type="u" name="direction"/>
      <arg type="u" name="azimuth"/>
    </signal>
    <signal name="VelocityChanged">
      <arg type="u" name="speed"/>
      <arg type="u" name="course"/>
    </signal>
    <signal name="StateChanged">
      <arg type="i" name="state"/>
    </signal>
    <signal name="SatsChanged">
      <arg type="y" name="nsats"/>
      <arg type="y" name="fix"/>
    </signal>
    <signal name="Quit">
    </signal>
    <property type="u" name="DbusPosInterval" access="readwrite"/>
  </interface>
</node>
```


3.8 Support Policy

3.8.1 How, where

- Github issues preferred
- RCG, inav discord and telegram
  - Most likely you will be requested to raise a Gihub issue for non-trivial cases.

3.8.2 Supported OS

- Arch Linux
- Debian Stable and later (`testing` , `sid`)
- Ubuntu latest and latest LTS (prior release where latest is also LTS).
- Fedora latest
- FreeBSD latest `RELEASE`

3.8.3 Supported infrastructure

- Native hardware (x64_x86, ia32, aarch64).
- Non-proprietary video driver.
- qemu/kvm virtualisated instances.
- Little endian (big endian never tested).

3.8.4 Unsupported

- Anything else!

Problem reports on non-supported platforms are unlikely to be dismissed without *some* consideration, however it's unlikely that too much time be expended on such environments unless the problem can also be demonstrated on a supported platform.

3.9 Licence

GPL v3 or later

3.10 Alternative Tools

In addition to mwp, the following inav mission planners (and GCS in some cases) exist, in various states of usefulness, at least:

- Inav Configurator (for inav 2.x), limited planning support
- Inav Configurator (for inav 3.x and later), supports all current WP types. Preview builds, may be augmented with impload to upload missions to 2.x firmware.
- Drone Helper (Windows 10)
- Ezgui, MissionPlanner for Inav (Android) Unsupported, obsolete. May not work with either contemporary Android or inav firmware.
- Mobile Flight (IOS) Unsupported, obsolete. May not work with either contemporary IOS or inav firmware.
- Apmplanner2 with impload. Ardupilot planner, missions can be uploaded to inav using impload.
- qgroundcontrol with impload. Ardupilot planner, missions can be uploaded to inav using impload.
- Side-Pilot with impload (untested). Ardupilot mission planner and telemetry viewer for IOS.

The following alternatives exist for **mwp-area-planner** :

- iforce2d's online planner
- qgroundcontrol with impload. Generic surveys and corridor plans are supported. Example images.

