Advanced Serial Data Logger

Trust in Confidence!

NMEA data parser plugin module

NMEA data parser plugin module

Printed: Ñåí òÿáðü 2007

Publisher

AGG Software

Production

© 2006 AGG Software
http://www.aggsoft.com

Table of Contents

1 Introduction

The National Marine Electronics Association (NMEA) has developed a specification that defines the interface between various pieces of marine electronic equipment. An NMEA standard defines an electrical interface and data protocol for communications between marine instrumentation. (They may also have standards for other things.)

NMEA 0183 devices are designated as either **talkers** or **listeners** (with some devices being both), employing an asynchronous serial interface with the following parameters: **Baud rate**: 4800, **Number of data bits**: 8 (bit 7 is 0), **Stop bits**: 1 (or more), **Parity**: none, **Handshake**: none. NMEA 0183 allows a single talker and several listeners on one circuit.

GPS receiver communication is defined within this specification. Most computer programs that provide real time position information understand and expect data to be in NMEA format. This data includes the complete PVT (position, velocity, time) solution computed by the GPS receiver. The idea of NMEA is to send a line of data called a **sentence** that is totally self contained and independent from other sentences. There are standard sentences for each device category and there is also the ability to define proprietary sentences for use by the individual company. All of the standard sentences have a two letter prefix that defines the device that uses that sentence type. (For GPS receivers the prefix is GP.) which is followed by a three letter sequence that defines the sentence contents. In addition NMEA permits hardware manufactures to define their own proprietary sentences for whatever purpose they see fit. All proprietary sentences begin with the letter P and are followed with 3 letters that identifies the manufacturer controlling that sentence. For example a Garmin sentence would start with PGRM and Magellan would begin with PMGN.

Our module parse each sentence begins with a '$' and ends with CRLF (a carriage return/line feed sequence). The data is contained within this single line with data items separated by commas. The data itself is just ASCII text and may extend over multiple sentences in certain specialized instances but is normally fully contained in one variable length sentence. The data may vary in the amount of precision contained in the message. For example time might be indicated to decimal parts of a second or location may be show with 3 or even 4 digits after the decimal point. There is a provision for a checksum at the end of each sentence which may or may not be checked by the unit that reads the data. The checksum field consists of a '*' and two hex digits.

Our parser module splits all data to variables and this variables can be used in data export modules.

2 System requirements

To run the program one of the following operation systems is needed:

- Windows 95;
- Windows 95 OSR2;
- Windows 98;
- Windows Me;
- Windows NT4, Windows 2000, Windows XP, Windows 2003 (administrator rights needed).

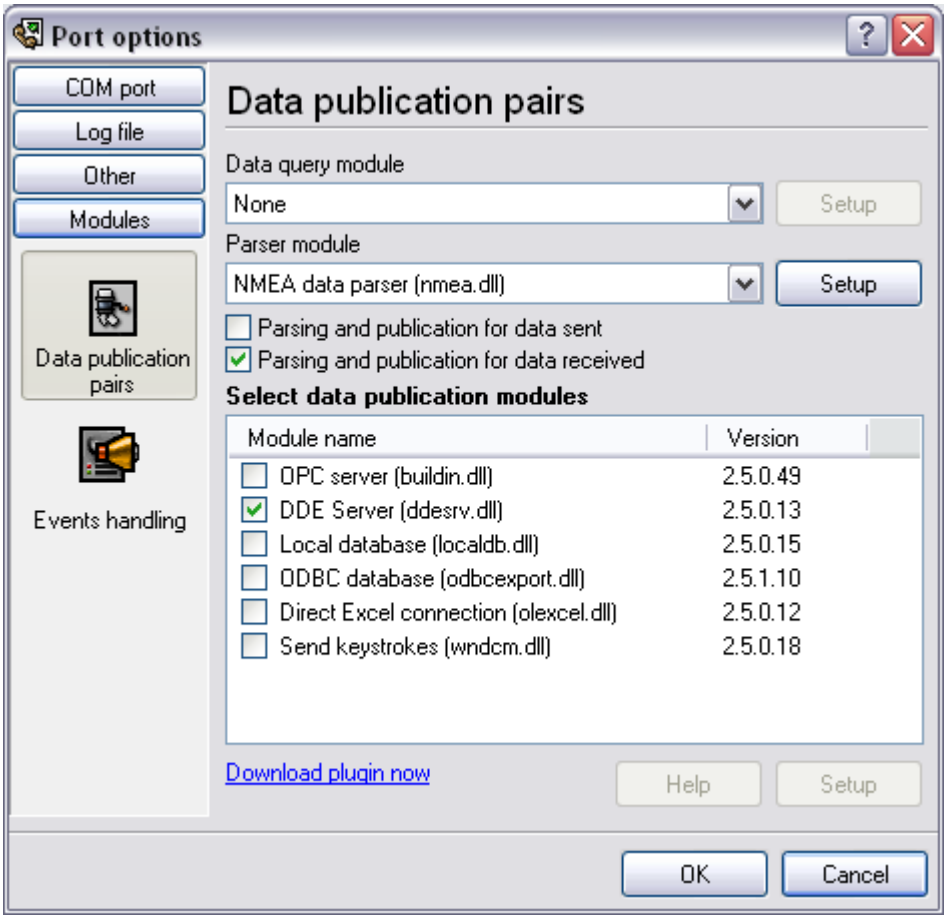
Advanced NMEA Data Logger installed on your computer.

3 Installation

1. Close main application, if it was run;
2. Copy module distributive file to your hard disk;
3. Run file on execution, clicking twice in Explorer on its name;
4. Execute setup wizard's instructions. In common case, if earlier was installed main application, enough to press several times button "Next". All necessary for installation data, wizard will get himself;
5. Restart main application. In case of successful installation module name will show up in options, on tab "Modules".

If , due to some reasons, You couldn't install publication module, so consult technical support service on address support@aggsoft.com and our professionals will be glad to help you.

If module is compatible with our program, its brief description You will see in module list (pic.1). Some modules may demand additional setting. To output module setting dialog select it from the list and press button "Setup".



Pic.1. Installed plugin modules

4 User Manual

4.1 Supported talkers

AG - Autopilot - General
AP - Autopilot - Magnetic
CD - Communications – Digital Selective Calling (DSC)
CR - Communications – Receiver / Beacon Receiver
CS - Communications – Satellite
CT - Communications – Radio-Telephone (MF/HF)
CV - Communications – Radio-Telephone (VHF)
CX - Communications – Scanning Receiver
DF - Direction Finder
EC - Electronic Chart Display & Information System (ECDIS)
EP - Emergency Position Indicating Beacon (EPIRB)
ER - Engine Room Monitoring Systems
GP - Global Positioning System (GPS)
HC - Heading – Magnetic Compass
HE - Heading – North Seeking Gyro

HN - Heading – Non North Seeking Gyro
II - Integrated Instrumentation
IN - Integrated Navigation
LC - Loran C
P - Proprietary Code
RA - RADAR and/or ARPA
SD - Sounder, Depth
SN - Electronic Positioning System, other/general
SS - Sounder, Scanning
TI - Turn Rate Indicator
VD - Velocity Sensor, Doppler, other/general
DM - Velocity Sensor, Speed Log, Water, Magnetic
VW - Velocity Sensor, Speed Log, Water, Mechanical
WI - Weather Instruments
YX - Transducer
ZA - Timekeeper – Atomic Clock
ZC - Timekeeper – Chronometer
ZQ - Timekeeper – Quartz
ZV - Timekeeper – Radio Update, WWV or WWVH

4.2 Supported sentences

AAM - Waypoint arrival alarm
- **AAM_ARIV_ENT** - Arrival circle entered
- **AAM_PERP_PASS** - Perpendicular passed
- **AAM_CIRCLE_RAD** - Circle radius
- **AAM_CIRCLE_RAD_UNIT** - Circle radius units
- **AAM_WPTNAME** - Waypoint name

ALM - GPS Almanac data
- **ALM_SENT_NUM** - Number of sentences
- **ALM_SENT_CNT** - Sentence count
- **ALM_PRN_ID** - Satellite PRN number
- **ALM_WEEK_NO** - GPS week number
- **ALM_SV_HEALTH** - SV health
- **ALM_ECCENTRICITY** - Eccentricity
- **ALM_REF_TIME** - Almanac reference time
- **ALM_INC_ANGLE** - Inclination angle
- **ALM_RA_RATE** - Rate of right ascension
- **ALM_AXIS_ROOT** - Root of semi-major axis
- **ALM_PEREGREE_ARG** - Argument of perigee
- **ALM_NODE_LONG** - Longitude of ascension node
- **ALM_MEAN_ANN** - Mean anomaly
- **ALM_F0_CLOCK** - F0 clock parameter
- **ALM_F1_CLOCK** - F1 clock parameter

APA - Auto pilot A sentence
- **APA_STATUS1** - Loran-C blink/SNR warning, general warning
- **APA_STATUS2** - Loran-C cycle warning
- **APA_CROSS_TRACK_RAD** - Cross-track error distance
- **APA_STEER** - Steer to correct
- **APA_CROSS_TRACK_RAD_UNIT** - Cross-track error units
- **APA_ARIV_ALRM_C** - Arrival alarm - circle

APA_ARIV_ALRM_P - Arrival alarm - perpendicular
APA_MAG_BEAR_OD - Magnetic bearing, origin to destination
APA_MAG_BEAR_OD_UNIT - Magnetic bearing unit
APA_DEST_WPTID - Destination waypoint ID

APB - Auto pilot B sentence
APB_STATUS1 - Loran-C blink/SNR warning, general warning
APB_STATUS2 - Loran-C cycle warning
APB_CROSS_TRACK_RAD - Cross-track error distance
APB_STEER - Steer to correct
APB_CROSS_TRACK_RAD_UNIT - Cross-track error units
APB_ARIV_ALRM_C - Arrival alarm - circle
APB_ARIV_ALRM_P - Arrival alarm - perpendicular
APB_MAG_BEAR_OD - Magnetic bearing, origin to destination
APB_MAG_BEAR_OD_UNIT - Magnetic bearing unit
APB_DEST_WPTID - Destination waypoint ID
APB_MAG_BEAR_PD - Magnetic bearing, present position to destination
APB_MAG_BEAR_PD_UNIT - Magnetic bearing unit
APB_MAG_BEAR_HS - Magnetic heading to steer
APB_MAG_BEAR_HS_UNIT - Magnetic heading unit

BEC - Bearing and distance to waypoint – dead reckoning
BEC_UTC - UTC time of fix
BEC_WPT_LAT - Latitude of waypoint
BEC_WPT_LAT_H - Latitude hemisphere
BEC_WPT_LONG - Longitude of waypoint
BEC_WPT_LONG_H - Longitude hemisphere
BEC_BEARING - Bearing to waypoint
BEC_BEAR_TYPE - Bearing to waypoint type
BEC_DIST - Distance to waypoint
BEC_DIST_UNIT - Distance to waypoint units
BEC_WPTID - Waypoint ID

BOD - Bearing origin to destination
BOD_BEARING - Bearing from START to DEST, degrees
BOD_BEAR_TYPE - Bearing from START to DEST type
BOD_DEST_WPTID - Destination waypoint ID
BOD_ORIG_WPTID - Origin waypoint ID

BWC - Bearing using great circle route
BWC_DEPTH - Depth
BWC_DEPTH_UNIT - Depth unit

DBS - Depth below surface
DBS_DEPTH - Depth, meters
DBS_OFFSET - Offset from transducer

FSI - Frequency set information
FSI_TX_FREQ - Transmitting frequency
FSI_RX_FREQ - Receiving frequency
FSI_COMM_MODE - Communications mode
FSI_POWER_LEVEL - Power Level

GGA - GPS fix data
GGA_TAKEN_AT - Fix taken at
GGA_LATITUDE_DEG - Latitude
GGA_LATITUDE_DEG_H - Latitude hemisphere
GGA_LONGITUDE_DEG - Longitude
GGA_LONGITUDE_DEG_H - Longitude hemisphere
GGA_QUALITY - Fix quality

- **GGA_SAT_NUM** - Number of satellites being tracked
- **GGA_HOR_DIL** - Horizontal dilution of position
- **GGA_ALTITUDE** - Altitude above mean sea level
- **GGA_ALTITUDE_UNIT** - Altitude units
- **GGA_HEIGHT_OF_GEOID** - Height of geoid (mean sea level) above WGS84 ellipsoid
- **GGA_HEIGHT_OF_GEOID_UNIT** - Height of geoid units
- **GGA_TIME_SNC_DGPS** - Time in seconds since last DGPS update
- **GGA_DGPS_ID** - DGPS station ID number

GLC - Geographic position, Loran-C
- **GLC_GRI_MS** - GRI Microseconds
- **GLC_TOA_MS** - Master TOA microseconds
- **GLC_TOA_STATUS** - Master TOA signal status
- **GLC_TIME_DIFF_MS** - Time difference in microseconds
- **GLC_TIME_DIFF_STATUS** - Time difference signal status

GLL - Geographic position, lat/lon data
- **GLL_LATITUDE_DEG** - Latitude
- **GLL_LATITUDE_DEG_H** - Latitude hemisphere
- **GLL_LONGITUDE_DEG** - Longitude
- **GLL_LONGITUDE_DEG_H** - Longitude hemisphere
- **GLL_TAKEN_AT** - Fix taken at
- **GLL_STATUS** - Status

GSA - Overall satellite data
- **GSA_AUTO_SEL** - Auto selection of 2D or 3D fix
- **GSA_3D_FIX** - 3D fix
- **GSA_SAT_PRN** - Sat used for fix
- **GSA_PDOP** - Dilution of precision
- **GSA_HDOP** - Horizontal dilution of precision
- **GSA_VDOP** - Vertical dilution of precision

GSV - Detailed satellite data
- **GSV_SENT_NUM** - Number of sentences
- **GSV_SENT_CNT** - Sentence count
- **GSV_SAT_IN_VIEW** - Number of satellites in view
- **GSV_SAT_PRN** - Satellite PRN number
- **GSV_ELEVATION** - Elevation, degrees
- **GSV_AZIMUTH** - Azimuth, degrees
- **GSV_SNR** - SNR - higher is better

GTD - Geographic location in time differences
- **GTD_TIME_DIFF** - Time difference

HDG - Heading, deviation and variation
- **HDG_MAG_HEAD** - Magnetic sensor heading in degrees
- **HDG_MAG_DEV** - Magnetic deviation in degrees
- **HDG_MAG_DEV_DIR** - Magnetic deviation direction
- **HDG_MAG_VAR** - Magnetic variation in degrees
- **HDG_MAG_VAR_DIR** - Magnetic variation direction

HDM - Heading, magnetic
- **HDM_HEADING** - Heading in degrees
- **HDM_HEADING_UNIT** - Heading unit

HDT - Heading, true
- **HDT_HEADING** - Heading in degrees
- **HDT_HEADING_UNIT** - Heading unit

LCD - Loran-C signal data
- **LCD_GRI_MS** - GRI Microseconds
- **LCD_MR_SNR** - Master relative SNR

LCD_MR_ECD - Master relative ECD
LCD_TIME_DIFF_MS - Time difference in microseconds
LCD_TIME_DIFF_STATUS - Time difference signal status

MSK - Send control for a beacon receiver
MSK_FREQ - Frequency
MSK_FREQ_MODE - Frequency mode
MSK_BITRATE - Bitrate
MSK_BITRATE_MODE - Bitrate mode
MSK_FREQ_STATUS - Frequency for MSS message status

MSS - Beacon receiver status information
MSS_SIGNAL_S - Signal strength in dB
MSS_SIGNAL_N - Signal to noise ratio in dB
MSS_BEACON_FREQ - Beacon frequency in KHz
MSS_BEACON_BITRATE - Beacon bitrate in bps

MTW - Water temperature
MTW_DEGREES - Degrees
MTW_DEGREES_UNIT - Unit of measurement

MWV - Wind speed and angle
MWV_ANGLE - Wind angle
MWV_REF - Reference
MWV_SPEED - Wind speed
MWV_SPEED_UNIT - Wind speed unit
MWV_STATUS - Status

OSD - Own ship data
OSD_HEADING - Heading true, degrees
OSD_STATUS - Status
OSD_VESSEL - Vessel ñourse true, degrees
OSD_VESSEL_REF - Course reference
OSD_VESSEL_SPEED - Vessel speed
OSD_SPEED_REF - Speed reference
OSD_VESSEL_SET - Vessel set true, degrees
OSD_VESSEL_DRIFT - Vessel drift true, degrees
OSD_VESSEL_DRIFT_UNIT - Vessel drift unit

ROO - Waypoints in active route
ROO_WPT_ID - Waypoint identifier

RMA - Recommended minimum navigation information
RMA_STATUS - Status
RMA_LATITUDE_DEG - Latitude
RMA_LATITUDE_DEG_H - Latitude hemisphere
RMA_LONGITUDE_DEG - Longitude
RMA_LONGITUDE_DEG_H - Longitude hemisphere
RMA_TIME_DIFF_A - Time difference A
RMA_TIME_DIFF_B - Time difference B
RMA_SPEED - Speed over the ground in knots
RMA_TRACK_ANGLE - Track angle in degrees
RMA_MAGN_VAR - Magnetic variation
RMA_MAGN_VAR_H - Magnetic variation hemisphere

RMB - Recommended minimum navigation information
RMB_STATUS - Status
RMB_CROSS_TRACK_ERR - Cross-track error
RMB_CROSS_TRACK_ERR_DIR - Cross-track error steer
RMB_ORIG_WPTID - Origin waypoint ID
RMB_DEST_WPTID - Destination waypoint ID

- **RMB_WPT_LAT** - Latitude of destination waypoint
- **RMB_WPT_LAT_H** - Latitude hemisphere
- **RMB_WPT_LONG** - Longitude of destination waypoint
- **RMB_WPT_LONG_H** - Longitude hemisphere
- **RMB_RANGE** - Range to destination, nautical miles
- **RMB_BEAR** - True bearing to destination
- **RMB_BEAR** - Velocity towards destination, knots
- **RMB_ARIV_ALRM** - Arrival alarm

RMC - Recommended minimum navigation information
- **RMC_TAKEN_AT** - Fix taken at
- **RMC_STATUS** - Status
- **RMC_LATITUDE_DEG** - Latitude
- **RMC_LATITUDE_DEG_H** - Latitude hemisphere
- **RMC_LONGITUDE_DEG** - Longitude
- **RMC_LONGITUDE_DEG_H** - Longitude hemisphere
- **RMC_SPEED** - Speed over the ground in knots
- **RMC_TRACK_ANGLE** - Track angle in degrees
- **RMC_DATE** - Date
- **RMC_MAGN_VAR** - Magnetic variation
- **RMC_MAGN_VAR_H** - Magnetic variation hemisphere

ROT - Rate of turn
- **ROT_RATE_OF_TURN** - Rate of turn, degrees per minute
- **ROT_STATUS** - Status

RPM - Revolutions
- **RPM_SOURCE** - Source
- **RPM_NUM** - Engine or shaft number
- **RPM_SPEED** - Speed, revolutions per minute
- **RPM_PITCH** - Propeller pitch, % of maximum
- **RPM_STATUS** - Status

RSA - Rudder sensor angle
- **RSA_SR_SENSOR** - Starboard (or single) rudder sensor
- **RSA_STATUS** - Starboard rudder sensor status
- **RSA_PR_SENSOR** - Port rudder sensor
- **RSA_STATUS** - Port rudder sensor status

RSD - Radar system data
- **RSD_CURSOR_RANGE** - Cursor range from own ship
- **RSD_CURSOR_BEARING** - Cursor bearing CW from zero, degrees
- **RSD_RANGE_SCALE** - Range scale
- **RSD_RANGE_UNIT** - Range units

RTE - Route message
- **RTE_SENT_NUM** - Number of sentences
- **RTE_SENT_CNT** - Sentence count
- **RTE_TYPE** - Type
- **RTE_TYPE_NAME** - Type name
- **RTE_ID** - Route identifier
- **RTE_WPT_ID** - Waypoint identifier

SFI - Scanning frequency information
- **SFI_SENT_NUM** - Number of sentences
- **SFI_SENT_CNT** - Sentence count
- **SFI_FREQ** - Frequency
- **SFI_MODE** - Mode

STN - Multiple data ID
- **STN_ID** - Talker ID number

TTM - Tracked target message
- **TTM_TARGET_NUM** - Target number
- **TTM_TARGET_DIST** - Target distance
- **TTM_BEARING** - Bearing from own ship
- **TTM_BEAR_TYPE** - Bearing units
- **TTM_TARGET_SPEED** - Target speed
- **TTM_TARGET_COURSE** - Target course
- **TTM_COURSE_UNIT** - Course units
- **TTM_DIST_CPA** - Distance of closest-point-of-approach
- **TTM_TIME_CPA** - Time until closest-point-of-approach '-' means increasing
- **TTM_SIGN** - '-' means increasing
- **TTM_TARGET_NAME** - Target name
- **TTM_TARGET_STATUS** - Target status
- **TTM_REF_TARGET** - Reference target

VBW - Dual ground/water speed
- **VBW_WATER_LONG_SPEED** - Longitudinal water speed
- **VBW_WATER_TRAV_SPEED** - Transverse water speed
- **VBW_WATER_STATUS** - Water speed status
- **VBW_GROUND_LONG_SPEED** - Longitudinal ground speed
- **VBW_GROUND_TRAV_SPEED** - Transverse ground speed
- **VBW_GROUND_STATUS** - Ground speed status

VDR - Set and drift
- **VDR_DEGRESS** - Degress
- **VDR_DEGRESS_TYPE** - Degress type
- **VDR_SPEED** - Speed
- **VDR_SPEED_UNIT** - Speed units

VHW - Water speed and heading
- **VHW_DEGRESS** - Degress
- **VHW_DEGRESS_TYPE** - Degress type
- **VHW_SPEED** - Speed
- **VHW_SPEED_UNIT** - Speed units

VLW - Distance traveled through water
- **VLW_TOTAL** - Total cumulative distance
- **VLW_TOTAL_UNIT** - Total cumulative distance unit
- **VLW_RESET** - Distance since Reset
- **VLW_RESET_UNIT** - Distance since Reset unit

VPW - Speed, measured parallel to wind
- **VPW_SPEED** - Speed
- **VPW_SPEED_UNIT** - Speed units

VTG - Vector track an speed over the ground
- **VTG_MAG_TRACK** - Track made
- **VTG_MAG_TRACK_TYPE** - Track made type
- **VTG_SPEED** - Ground speed
- **VTG_SPEED_UNIT** - Ground speed units

VWR - Relative wind speed and angle
- **VWR_WIND_DIR** - Wind direction magnitude in degrees
- **VWR_WIND_DIR_TYPE** - Wind direction type
- **VWR_SPEED** - Speed
- **VWR_SPEED_UNIT** - Speed units

WCV - Waypoint closure velocity
- **WCV_VELOCITY** - Velocity
- **WCV_VELOCITY_UNIT** - Velocity units
- **WCV_WPT_ID** - Waypoint identifier

WNC - Distance, waypoint to waypoint
- **WNC_DISTANCE** - Distance
- **WNC_DISTANCE_UNIT** - Distance units
- **WNC_DEST_WPTID** - Destination waypoint ID
- **WNC_ORIG_WPTID** - Origin waypoint ID

WPL - Waypoint information
- **WPL_LATITUDE_DEG** - Latitude
- **WPL_LATITUDE_DEG_H** - Latitude hemisphere
- **WPL_LONGITUDE_DEG** - Longitude
- **WPL_LONGITUDE_DEG_H** - Longitude hemisphere
- **WPL_WPTNAME** - Waypoint name

XDR - Multiple cross rrack error, dead reckoning
- **XDR_TRANS_TYPE** - Transducer type
- **XDR_MEASURE_DATA** - Measurement data
- **XDR_MEASURE_UNIT** - Measurement data units
- **XDR_TRANS_NAME** - Name of transducer

XTE - Measured cross track error
- **XTE_GEN_WARN** - General warning flag
- **XTE_LORAN_LOCK** - Loran-C cycle lock flag
- **XTE_CROSS_TRACK_DIST** - Cross track error distance
- **XTE_STEER** - Steer
- **XTE_DIST_UNIT** - Distance units

XTR - Cross rrack error, dead reckoning
- **XTR_TRANS_TYPE** - Transducer type
- **XTR_MEASURE_DATA** - Measurement data
- **XTR_MEASURE_UNIT** - Measurement data units
- **XTR_TRANS_NAME** - Name of transducer

ZDA - Date and Time
- **ZDA_TIME** - Time
- **ZDA_DAY** - Day
- **ZDA_MONTH** - Month
- **ZDA_YEAR** - Year
- **ZDA_ZONE_HOUR** - Local zone hours
- **ZDA_ZONE_MIN** - Local zone minutes

ZFO - UTC and time to destination waypoint
- **ZFO_TIME** - Time
- **ZFO_TIME_REMAIN** - Time remaining
- **ZFO_WPT_ID** - Waypoint identifier

GRMC - Sensor configuration information
- **GRMC_MODE** - Fix mode
- **GRMC_ALT** - Altitude above/below mean sea level
- **GRMC_DATUM_INDEX** - Earth datum index
- **GRMC_DATUM_AXIS** - User earth datum semi-major axis
- **GRMC_DATUM_FACTOR** - User earth datum inverse flattening factor
- **GRMC_DATUM_DELTA_X** - User earth datum delta x earth centered coordinate
- **GRMC_DATUM_DELTA_Y** - User earth datum delta y earth centered coordinate
- **GRMC_DATUM_DELTA_Z** - User earth datum delta z earth centered coordinate
- **GRMC_DIFF_MODE** - Differential mode
- **GRMC_BAUD_RATE** - NMEA Baud rate
- **GRMC_FILTER_MODE** - Filter mode
- **GRMC_PPS_MODE** - PPS mode

GRME - Estimated position error
- **GRME_HPE** - Estimated horizontal position error (HPE)

- **GRME_HPE_UNIT** - HPE units
- **GRME_VPE** - Estimated vertical error (VPE)
- **GRME_VPE_UNIT** - VPE units
- **GRME_OSEPE** - Overall spherical equivalent position error (OSEPE)
- **GRME_OSEPE_UNIT** - SEPE units

GRMF - Position fix sentence
- **GRMF_WEEK_NO** - GPS week number
- **GRMF_SEC_NUM** - GPS seconds
- **GRMF_UTC_DATE** - UTC date of position fix
- **GRMF_UTC_TIME** - UTC time of position fix
- **GRMF_LEAP_SEC_NUM** - GPS leap second count
- **GRMF_LATITUDE_DEG** - Latitude
- **GRMF_LATITUDE_DEG_H** - Latitude hemisphere
- **GRMF_LONGITUDE_DEG** - Longitude
- **GRMF_LONGITUDE_DEG_H** - Longitude hemisphere
- **GRMF_MODE** - Mode
- **GRMF_FIX_TYPE** - Fix type
- **GRMF_SPEED** - Speed over ground, km/h
- **GRMF_COURSE** - Course over ground, degrees
- **GRMF_DIL_POS** - Position dilution of precision
- **GRMF_TIME_DIL_POS** - Time dilution of precision

GRMI - Sensor initialisation information
- **GRMI_LATITUDE_DEG** - Latitude
- **GRMI_LATITUDE_DEG_H** - Latitude hemisphere
- **GRMI_LONGITUDE_DEG** - Longitude
- **GRMI_LONGITUDE_DEG_H** - Longitude hemisphere
- **GRMI_UTC_DATE** - Current UTC date
- **GRMI_UTC_TIME** - Current UTC time

GRMM - Map datum
- **GRMM_DATUM** - Currently active horizontal datum

GRMO - Output sentence enable/disable
- **GRMO_NAME** - Target sentence description
- **GRMO_MODE** - Target sentence mode

GRMV - 3D velocity
- **GRMV_EAST_VEL** - True east velocity
- **GRMV_NORTH_VEL** - True north velocity
- **GRMV_UP_VEL** - Up velocity

GRMZ - Altitude information
- **GRMZ_ALT** - Altitude
- **GRMZ_ALT_UNIT** - Altitude units
- **GRMZ_POS_FIX_DIM** - Position fix dimensions

SLIB - Differental GPS beacon receiver control
- **SLIB_FREQ** - Frequency
- **SLIB_BITRATE** - Bit rate
- **SLIB_REQ_TYPE** - Request type

SRF150 - OK to send
- **SRF150_STATUS** - Status

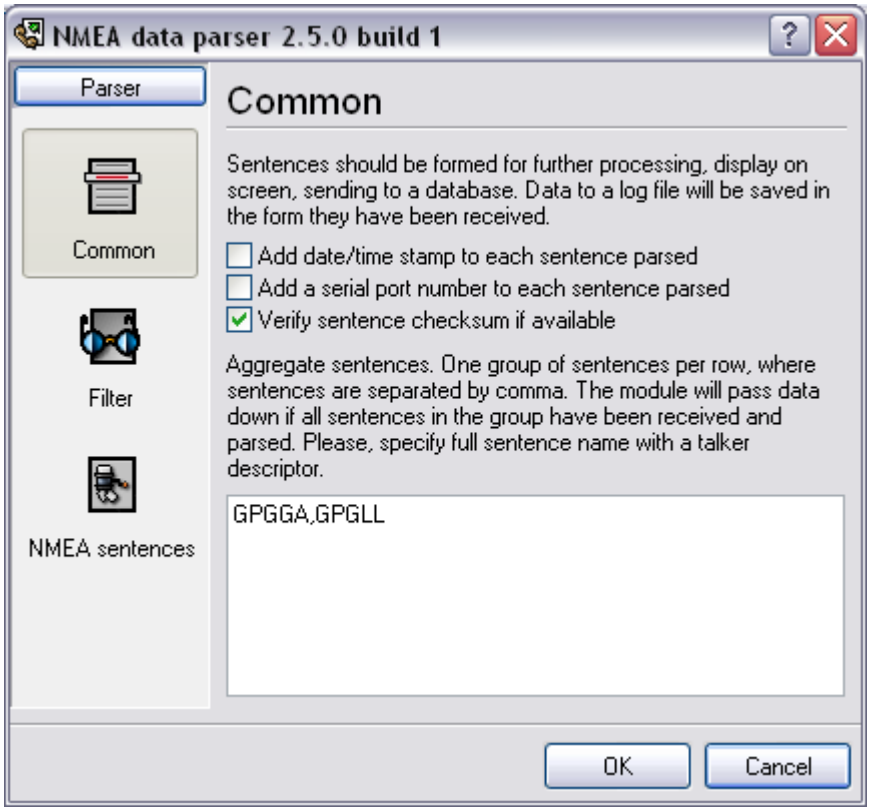
SRF161 - OK to send
- **SRF161_ANT_STATUS** - Antenna status
- **SRF161_AGC** - AGC

4.3 Common parameters

These parameters are used for data parsing (pic.2).

1. **Add date/time stamp to each sentence parsed** - the parser will add an additional stamp value to other values, that the parser will extract from a data block;
2. **Add serial port number to each sentence parsed** - the parser will add an additional value with serial port number, that received this data block. You can use it in a multi port configuration, for identifying sentences from different serial ports.
3. **Verify sentence checksum if available** - the parser will calculate a checksum and verify it for each sentence that will contain '*' characters at the end of sentence:

   Sentence example: `GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,*47`



Pic.2. Common parameters.

Aggregate sentences

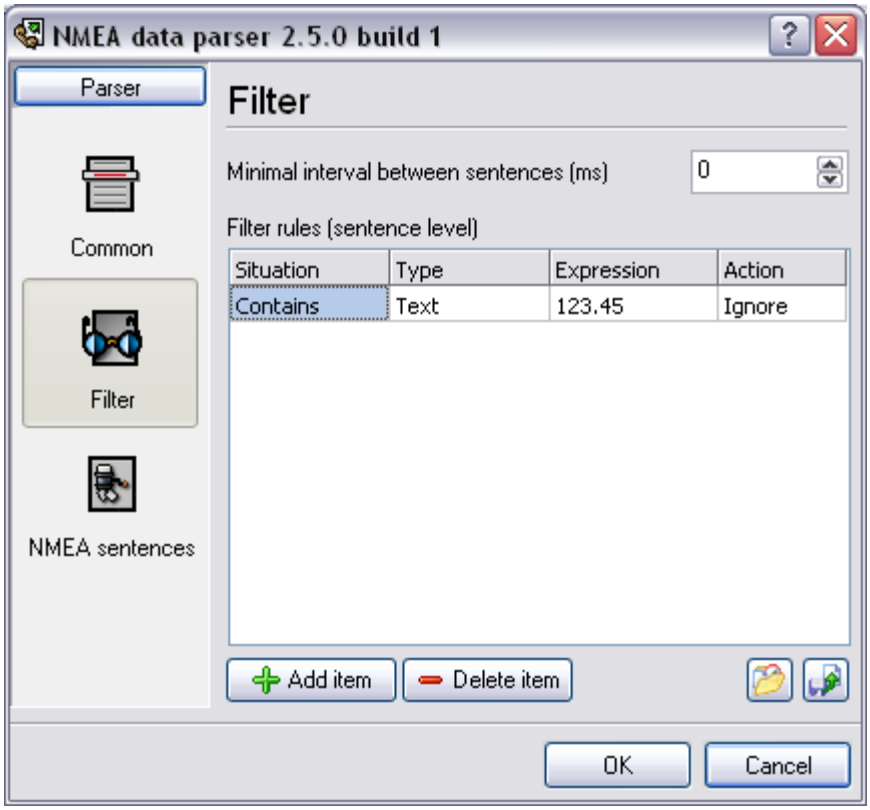
This option is very useful if your talker sends more than one sentence and you want to save data to a file at single row. You can aggregate two or more sentences and data of these sentences will be send to data export modules at same moment with one date time stamp. If you'll specify sentence names then the module will store all data in a temporary buffer, while all sentences isn't received. When all data is received the module sends data to a data export module, clears the buffer and starts waiting for new data.

You can specify one or more different aggregate groups. Simply add sentence name to different

rows. Sentences in the row should be separated by comma and a sentence should contain a talker name.

4.4 Filter

Filter are intended for ignoring some packages of the data. The ignored data's packages will not be published with a data publication plugins.



Pic.3 Filter rules

For filtration it's possible to set one or several rules. At execution of any of rules it will be fulfilled operation selected in the field "***Action***".

Action types

- ***Ignore*** - current sentence will be ignore and data from this sentence will not export
- ***Parse*** - current sentence will be parse and data from this sentence will export

That operation is fulfilled observance of the condition set in the field "***Situation***" was necessary.

Situation types

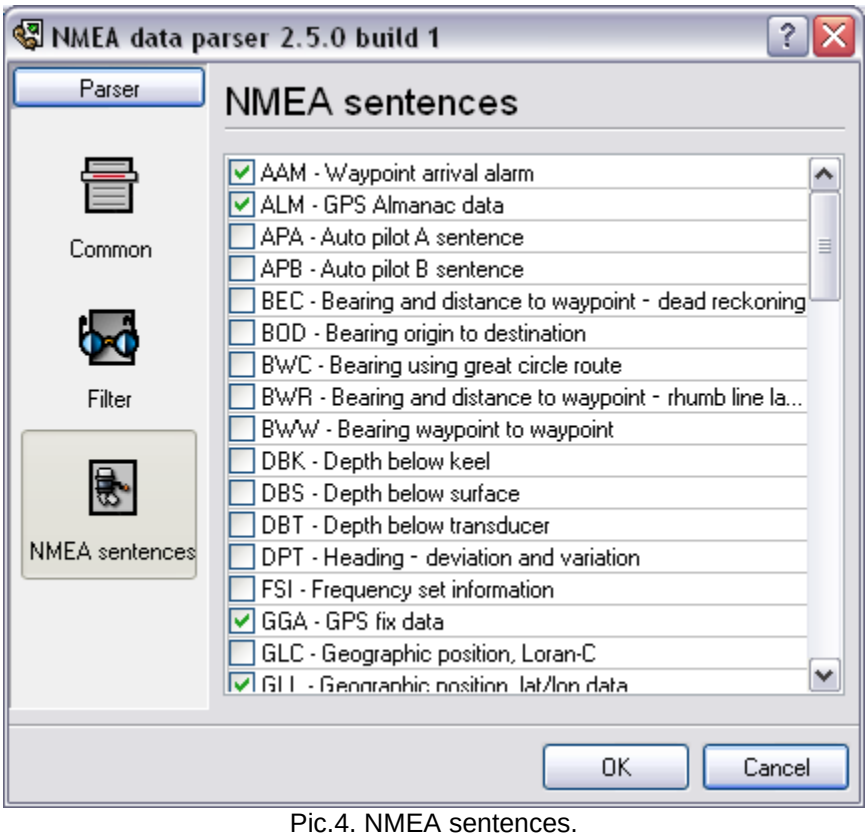
- ***Disabled*** - this rule have been disabled;
- ***Contains*** - this rule will be true if sentence contains string, specified in the filed "***Expression***";
- ***Not contains*** - this rule will be true if sentence not contains string, specified in the filed "***Expression***";

Expression, specified in the field "Expression" can be two types:

Text - the program will search, data specified in the field "Expression", in a sentence. This searching is case sensitive.
Reg. expr. - the program will search data use regular expression, specified in the field "Expression", in a sentence. This searching is case insensitive.

4.5 NMEA sentences parser

If you want to export to any target, then you should configure a parser module. The ASCII data parser allows you to extract data from data flow, that contains a ASCII characters. The parser module splits data flow to data block and extracts data values from each data block. On the "sentence" tab (pic.4) you should specify sentences, that the parser will parse. Other sentences will be ignored.



Pic.4. NMEA sentences.

The full list of supported sentences and variables that parsed from each sentence is listed here.

Our software create variables with following types:

- **String** - Character array with length from 1 to 65535 characters;
- **Boolean** - Logical value (Truth/False) - 0 or 1;
- **Float** - Broken (natural) number - value range: -2.9 x 10^-39 .. 1.7 x 10^38
- **Integer** - Integer value: -2147483648..2147483647;

- **DateTime** - Date and time.

Note: Our modules doesn't support Time and Date data types. Therefore time variables, that exists in a sentence contains current date, but with time from the sentence.

4.6 Syntax of Regular Expressions

Introduction

Regular Expressions are a widely-used method of specifying patterns of text to search for. Special **metacharacters** allow You to specify, for instance, that a particular string You are looking for occurs at the beginning or end of a line, or contains **n** recurrences of a certain character.

Regular expressions look ugly for novices, but really they are very simple, handy and powerful tool.

Let's start our learning trip!

Simple matches

Any single character matches itself, unless it is a **metacharacter** with a special meaning described below.

A series of characters matches that series of characters in the target string, so the pattern "bluh" would match "bluh" in the target string. Quite simple, eh ?

You can cause characters that normally function as **metacharacters** or **escape sequences** to be interpreted literally by 'escaping' them by preceding them with a backslash "\", for instance: metacharacter "^" match beginning of string, but "\^" match character "^", "\\" match "\" and so on.

Examples:

`foobar`	*matchs string 'foobar'*
`\^FooBarPtr`	*matchs '^FooBarPtr'*

Escape sequences

Characters may be specified using a **escape sequences** syntax much like that used in C and Perl: "\n" matches a newline, "\t" a tab, etc. More generally, \xnn, where nn is a string of hexadecimal digits, matches the character whose ASCII value is nn. If You need wide (Unicode) character code, You can use '\x{nnnn}', where 'nnnn' - one or more hexadecimal digits.

`\xnn`	*char with hex code nn*
`\x{nnnn}`	*char with hex code nnnn (one byte for plain text and two bytes for Unicode)*
`\t`	*tab (HT/TAB), same as \x09*
`\n`	*newline (NL), same as \x0a*
`\r`	*car.return (CR), same as \x0d*
`\f`	*form feed (FF), same as \x0c*
`\a`	*alarm (bell) (BEL), same as \x07*

`\e`	*escape (ESC), same as \x1b*

Examples:

`foo\x20bar`	*matchs 'foo bar' (note space in the middle)*
`\tfoobar`	*matchs 'foobar' predefined by tab*

Character classes

You can specify a **character class**, by enclosing a list of characters in [], which will match any **one** character from the list.

If the first character after the "[" is "^", the class matches any character **not** in the list.

Examples:

`foob[aeiou]r`	*finds strings 'foobar', 'foober' etc. but not 'foobbr', 'foobcr' etc.*
`foob[^aeiou]r`	*find strings 'foobbr', 'foobcr' etc. but not 'foobar', 'foober' etc.*

Within a list, the "-" character is used to specify a **range**, so that a-z represents all characters between "a" and "z", inclusive.

If You want "-" itself to be a member of a class, put it at the start or end of the list, or escape it with a backslash. If You want ']' you may place it at the start of list or escape it with a backslash.

Examples:

`[-az]`	*matchs 'a', 'z' and '-'*
`[az-]`	*matchs 'a', 'z' and '-'*
`[a\-z]`	*matchs 'a', 'z' and '-'*
`[a-z]`	*matchs all twenty six small characters from 'a' to 'z'*
`[\n-\x0D]`	*matchs any of #10,#11,#12,#13.*
`[\d-t]`	*matchs any digit, '-' or 't'.*
`[]-a]`	*matchs any char from ']'..'a'.*

Metacharacters

Metacharacters are special characters which are the essence of Regular Expressions. There are different types of metacharacters, described below.

Metacharacters - line separators

`^`	*start of line*
`$`	*end of line*
`\A`	*start of text*
`\Z`	*end of text*
`.`	*any character in line*

Examples:

`^foobar`	*matchs string 'foobar' only if it's at the beginning of line*
`foobar$`	*matchs string 'foobar' only if it's at the end of line*
`^foobar$`	*matchs string 'foobar' only if it's the only string in line*
`foob.r`	*matchs strings like 'foobar', 'foobbr', 'foob1r' and so on*

The "^" metacharacter by default is only guaranteed to match at the beginning of the input string/text, the "$" metacharacter only at the end. Embedded line separators will not be matched by "^" or "$".
You may, however, wish to treat a string as a multi-line buffer, such that the "^" will match after any line separator within the string, and "$" will match before any line separator.

The "." metacharacter by default matches any character.

Note that "^.*$" (an empty line pattern) doesnot match the empty string within the sequence \x0D\x0A, but matchs the empty string within the sequence \x0A\x0D.

Metacharacters - predefined classes

`\w`	*an alphanumeric character (including "_")*
`\W`	*a nonalphanumeric*
`\d`	*a numeric character*
`\D`	*a non-numeric*
`\s`	*any space (same as [\t\n\r\f])*
`\S`	*a non space*

You may use \w, \d and \s within custom **character classes**.

Examples:

`foob\dr` *matchs strings like 'foob1r', "foob6r' and so on but not 'foobar', 'foobbr' and so on*

`foob[\w\s]r` *matchs strings like 'foobar', 'foob r', 'foobbr' and so on but not 'foob1r', 'foob=r' and so on*

Metacharacters - iterators

Any item of a regular expression may be followed by another type of metacharacters - **iterators**. Using this metacharacters You can specify number of occurences of previous character, **metacharacter** or **subexpression**.

`*`	*zero or more ("greedy"), similar to {0,}*
`+`	*one or more ("greedy"), similar to {1,}*
`?`	*zero or one ("greedy"), similar to {0,1}*
`{n}`	*exactly n times ("greedy")*
`{n,}`	*at least n times ("greedy")*
`{n,m}`	*at least n but not more than m times ("greedy")*
`*?`	*zero or more ("non-greedy"), similar to {0,}?*
`+?`	*one or more ("non-greedy"), similar to {1,}?*
`??`	*zero or one ("non-greedy"), similar to {0,1}?*
`{n}?`	*exactly n times ("non-greedy")*
`{n,}?`	*at least n times ("non-greedy")*
`{n,m}?`	*at least n but not more than m times ("non-greedy")*

So, digits in curly brackets of the form {n,m}, specify the minimum number of times to match the item n and the maximum m. The form {n} is equivalent to {n,n} and matches exactly n times. The form {n,} matches n or more times. There is no limit to the size of n or m, but large numbers will chew up more memory and slow down r.e. execution.

If a curly bracket occurs in any other context, it is treated as a regular character.

Examples:

`foob.*r`	*matchs strings like 'foobar', 'foobalkjdflkj9r' and 'foobr'*
`foob.+r`	*matchs strings like 'foobar', 'foobalkjdflkj9r' but not 'foobr'*
`foob.?r`	*matchs strings like 'foobar', 'foobbr' and 'foobr' but not 'foobalkj9r'*
`fooba{2}r`	*matchs the string 'foobaar'*
`fooba{2,}r`	*matchs strings like 'foobaar', 'foobaaar', 'foobaaaar' etc.*
`fooba{2,3}r`	*matchs strings like 'foobaar', or 'foobaaar' but not 'foobaaaar'*

A little explanation about "greediness". "Greedy" takes as many as possible, "non-greedy" takes as few as possible. For example, 'b+' and 'b*' applied to string 'abbbbc' return 'bbbb', 'b+?' returns 'b', 'b*?' returns empty string, 'b{2,3}?' returns 'bb', 'b{2,3}' returns 'bbb'.

Metacharacters - alternatives

You can specify a series of **alternatives** for a pattern using "|" to separate them, so that fee|fie|foe will match any of "fee", "fie", or "foe" in the target string (as would f(e|i|o)e). The first alternative includes everything from the last pattern delimiter ("(", "[", or the beginning of the pattern) up to the first "|", and the last alternative contains everything from the last "|" to the next pattern delimiter. For this reason, it's common practice to include alternatives in parentheses, to minimize confusion about where they start and end.
Alternatives are tried from left to right, so the first alternative found for which the entire expression matches, is the one that is chosen. This means that alternatives are not necessarily greedy. For example: when matching foo|foot against "barefoot", only the "foo" part will match, as that is the first alternative tried, and it successfully matches the target string. (This might not seem important, but it is important when you are capturing matched text using parentheses.)
Also remember that "|" is interpreted as a literal within square brackets, so if You write [fee|fie|foe] You're really only matching [feio|].

Examples:

`foo(bar|foo)`	*matchs strings 'foobar' or 'foofoo'.*

Metacharacters - subexpressions

The bracketing construct (...) may also be used for define r.e. subexpressions.

Subexpressions are numbered based on the left to right order of their opening parenthesis. First subexpression has number '1'

Examples:

`(foobar){8,10}`	*matchs strings which contain 8, 9 or 10 instances of the 'foobar'*
`foob([0-9]|a+)r`	*matchs 'foob0r', 'foob1r' , 'foobar', 'foobaar', 'foobaar' etc.*

Metacharacters - backreferences

Metacharacters \1 through \9 are interpreted as backreferences. \<n> matches previously matched **subexpression** #<n>.

Examples:

`(.)\1+`	*matchs 'aaaa' and 'cc'.*

`(.+)\1+` *also match 'abab' and '123123'*
`(['"]?)(\d+)\1` *matchs '"13" (in double quotes), or '4' (in single quotes) or 77 (without quotes) etc*

Modifiers

Modifiers are for changing behaviour of parser.

There are many ways to set up modifiers.
Any of these modifiers may be embedded within the regular expression itself using the (?...) construct.

i
Do case-insensitive pattern matching (using installed in you system locale settings).
m
Treat string as multiple lines. That is, change "^" and "$" from matching at only the very start or end of the string to the start or end of any line anywhere within the string.
s
Treat string as single line. That is, change "." to match any character whatsoever, even a line separators, which it normally would not match.
g
Non standard modifier. Switching it Off You'll switch all following operators into non-greedy mode (by default this modifier is On). So, if modifier /g is Off then '+' works as '+?', '*' as '*?' and so on
x
Extend your pattern's legibility by permitting whitespace and comments (see explanation below).

The modifier /x itself needs a little more explanation. It tells the parser to ignore whitespace that is neither backslashed nor within a character class. You can use this to break up your regular expression into (slightly) more readable parts. The # character is also treated as a metacharacter introducing a comment, for example:

```
(
(abc) # comment 1
| # You can use spaces to format r.e. - parser ignores it
(efg) # comment 2
)
```

This also means that if you want real whitespace or # characters in the pattern (outside a character class, where they are unaffected by /x), that you'll either have to escape them or encode them using octal or hex escapes. Taken together, these features go a long way towards making regular expressions text more readable.

How to change modifiers

(?imsxr-imsxr)
You may use it into r.e. for modifying modifiers by the fly. If this construction inlined into subexpression, then it effects only into this subexpression

Examples:
`(?i)New-York` *matchs 'New-york' and 'New-York'*

`(?i)New-(?-i)York`	*matchs 'New-York' but not 'New-york'*
`(?i)(New-)?York`	*matchs 'New-york' and 'new-york'*
`((?i)New-)?York`	*matchs 'New-York', but not 'new-york*

(?#text)
A comment, the text is ignored. Note that parser closes the comment as soon as it sees a ")", so there is no way to put a literal ")" in the comment.