

NAVIEDIT

ASCII IMPORTER - HEADER TEMPLATE

Last update: 13/03/2023
Version: 8.7

Contents

| | | |
|----------|---------------------------|----------|
| 1 | Introduction | 3 |
| 2 | Usage | 3 |
| 2.1 | Syntax | 4 |
| 2.2 | Example | 5 |

1 Introduction

The Ascii Importer lets you define a template for import of tide, sound velocity, XYZ, pressure and CTD ascii files. In this template it is possible to define a Header Template where you by means of regular expression can specify Date, Time, and Position for the ascii file being imported. This document describes how to use this Header Template.

2 Usage

During import of an ascii file you first select the data type to import, then you can either create a new template or edit an existing template for this data type:

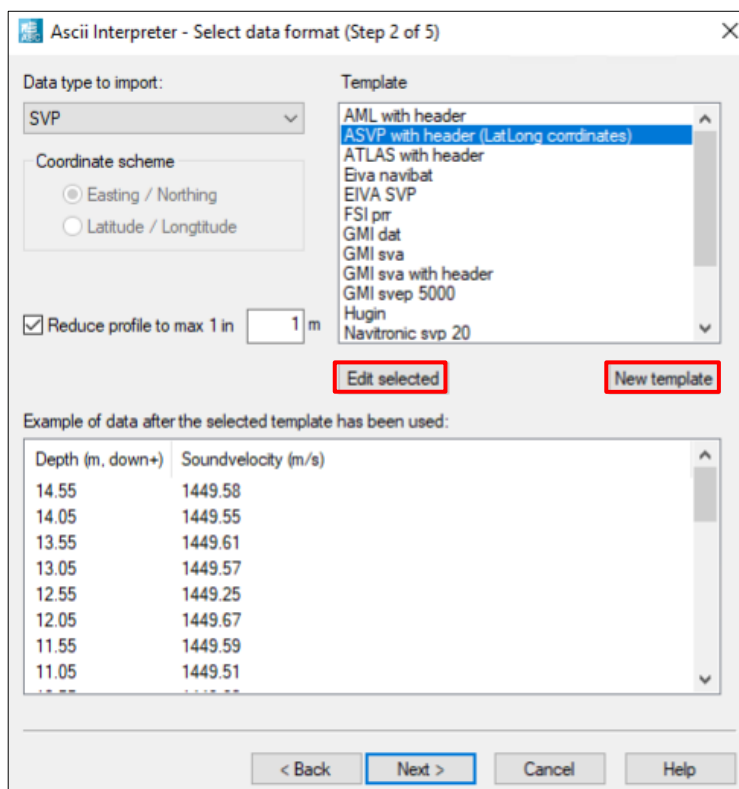


Figure 1 Ascii Interpreter - Select data format

For a detailed description on how to configure the other parts of the ascii importer, see <https://eiva.freshdesk.com/>

When you select either **Edit selected** or **New template** you will get the option to edit the Header Template:

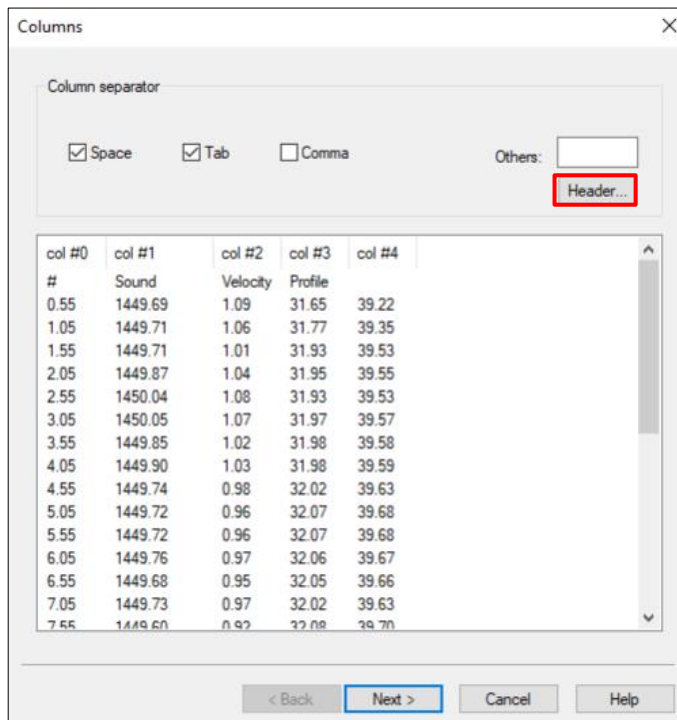


Figure 2 Columns Header

The Header Template lets you specify Date, Time, and Position by use of regular expressions:

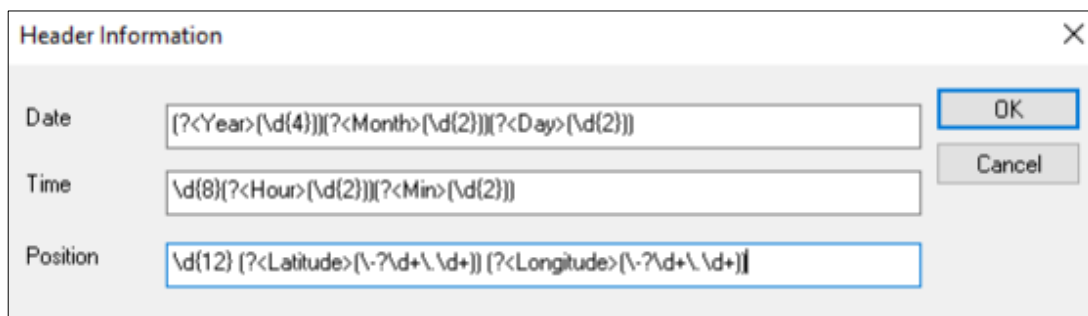


Figure 3 Header Information

The pre-installed templates are located at **C:\EIVA\Setup\AsciiFileTemplates**. If you want to copy a template from one NaviEdit installation to another you should copy the relevant file from this directory.

2.1 Syntax

The syntax used to identify Date, Time and Position is regular expressions. Regular

expressions come in different flavour, NaviEdit uses a perl derived version with a **named group** extension.

The **named group** is the keyword which is used to identify which expression is the Year, Month and Day part of the Header.

The syntax for a named group is

```
(?<NamedGroup>(RegularExpression))
```

The regular expression inside the parenthesis is searching for the first match of the regular expression.

The following named groups can be used:

- Year
- Month
- Day
- Hour
- Min
- Sec
- Easting
- Northing
- Latitude
- Longitude

Regular expressions are very powerful, but the syntax is a bit difficult. This guide will not at all try to explain the general syntax. The links below describes the syntax:

http://en.wikipedia.org/wiki/Regular_expressions

<http://www.regular-expressions.info/reference.html>

<http://www.addedbytes.com/cheat-sheets/download/regular-expressions-cheat-sheet-v2.pdf>

2.2 Example

The Date field of the following **ASVP with header (LatLong corrdinates).aft** template is used as an example:

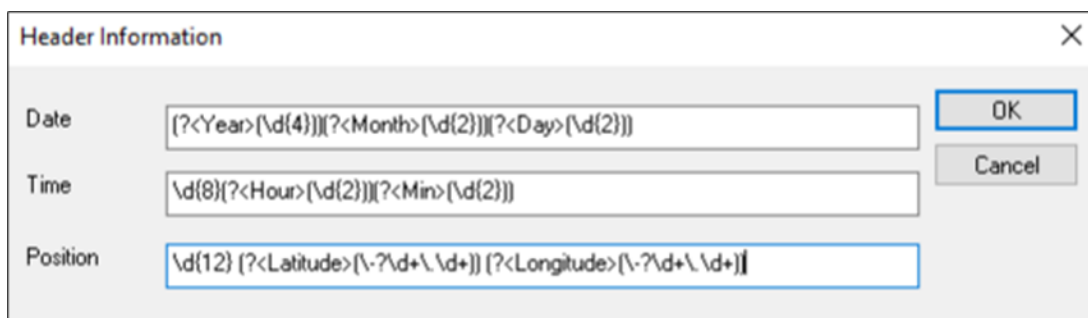


Figure 4 Header example

```
matchdate=(?<Year>\d{4})(?<Month>\d{2})(?<Day>\d{2})
matchtime=\d{8}(?<Hour>\d{2})(?<Min>\d{2})
matchposition=\d{12} (?<Latitude>(\-?\d+\.\d+)) (?<Longitude>(\-?\d+\.\d+))
```

The following sound velocity profile file is used as input:

```
( SoundVelocity 1.0 0 201005311222 56.87388889 12.30722222 -1 0 0 MVS10_00021 P 0232 )
0 1472.43
1.93 1472.43
1.98 1472.43
...
```

The regular expression for the Date part consists of 3 named groups:

```
(?<Year>\d{4}) (?<Month>\d{2}) (?<Day>\d{2})
```

The first named group **Year** will look for the first 4 consecutive digits, and find **2010**.

The next named group **Month** will continue the search just after the first match, and search for the first occurrence of 2 consecutive digits and find **05**.

The third and last named group **Day** will continue the search just after the previous match, and search for the first occurrence of 2 consecutive digits, and find **31**.

The regular expression in both the Date, Time, and Position fields will begin the search from the very beginning of the ascii file being imported and search the first million characters in the ascii file for matches.

This implies that the header part of an ascii file being import does not need to be on a single line.

In the example above the second run **Time** of the regular expression search will again begin searching from the beginning of the ascii file to be imported. It will first search for 8 consecutive digits and find **20100531**, which are not 'remembered' in any named group – it is the Year, Month and Date which we have already match in the Date field. Then the regular expression will search for the first occurrence of consecutive digits and find **12** which are the named expression **Hour**.