General Csv

Overview

Imports time series data from files in CSV format with one header line containing a column heades of the time series:

- The first line contains the column names (fields) in the csv file, the line is used to determine the field separator and to determine the names of the data columns
- All other lines contain the date-time as field and the values for each time series.
- Values between -1000.0 and -999.0 (inclusive) are regarded as missing values.

The CSV files can be supplied in a ZIP file.

Import type

The import type is *generalCSV*. There is no particular file extension required.

Example

Here is a simple example:

```
Time,Waterstand,Pomp-1 Born
04-05-2011 03:24,0.000000,-0.450000
04-05-2011 03:44,0.000000,-0.450000
04-05-2011 03:54,0.000000,-0.440000
.....
```

for configuration of the table layout see Table Layout

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0/timeSeriesImportRun.
xsd">
        <import>
                <general>
                        <importType>generalCSV</importType>
                        <folder>$IMPORT_FOLDER$/OBS</folder>
                        <failedFolder>$IMPORT_FAILED_FOLDER$</failedFolder>
                        <backupFolder>$IMPORT_BACKUP_FOLDER$/OBS</backupFolder>

                        <table>
                                <dateTimeColumn name="Time" pattern="dd-MM-yyyy HH:mm"/>
                                <valueColumn unit="m" locationId="Bosscheveld" parameterId="H.meting" name="
Waterstand"/>
                                <valueColumn unit="min" locationId="Bosscheveld" parameterId="DT.meting" name="
Pomp-1 Born"/>
                        </table>
                        <idMapId>IdImportOBS</idMapId>
                        <unitConversionsId>ImportUnitConversions</unitConversionsId>
                        <importTimeZone>
                                <timeZoneOffset>+00:00</timeZoneOffset>
                        </importTimeZone>
                </general>
        </import>
</timeSeriesImportRun>
```

Another example, skipFirstLinesCount

Example to read from CSV files where unfortunately the first line contains a key instead of the column headers.
The CSV files look like:

```
[DATA]
Tagname,TimeStamp,Value,DataQuality
WDD.MEM_BER0001_01_LT01_MW,2010-04-01 12:21:00,-0.000,GOOD
WDD.MEM_BER0001_01_LT01_MW,2011-01-12 10:34:05,-0.001,GOOD
WDD.MEM_BER0001_01_LT01_MW,2011-01-12 10:35:00,-0.011,BAD
WDD.MEM_BER0001_01_LT01_MW,2011-01-12 10:36:00,-0.003,GOOD
WDD.MEM_BER0001_01_LT01_MW,2011-01-12 10:37:00,-0.000,GOOD
WDD.MEM_BER0001_01_LT01_MW,2011-01-12 10:38:00,-0.000,GOOD
WDD.MEM_BER0001_01_LT01_MW,2011-01-12 10:39:00,-0.000,GOOD
```

In this example the first line should be skipped, so skipFirstLinesCount = 1

```
<general>
        ....
<gotoLineWhichStartWith>Tagname,TimeStamp,Value,DataQuality</gotoLineWhichStartWith>
</general>
```

Example gotoLineWhichStartsWith (Since 2016.02)

In the previous example, the first line should be skipped. A more flexible way to do this is to use gotoLineWhichStartsWith where a string can be specified with the start of the header text. This configuration is usefull if the position of the header changes in the CSV file.

```
<general>
        ...
        <gotoLineWhichStartsWith>Tagname,TimeStamp,Value,DataQuality</gotoLineWhichStartsWith>
</general>
```

Another example, no header at all (since 2012.02)

The CSV files look like:

```
WDD.MEM_BER0001_01_LT01_MW,A,2010-04-01 12:21:00,-0.000,GOOD
WDD.MEM_BER0001_01_LT01_MW,A,2011-01-12 10:34:05,-0.001,GOOD
WDD.MEM_BER0001_01_LT01_MW,A,2011-01-12 10:35:00,-0.011,BAD
WDD.MEM_BER0001_01_LT01_MW,A,2011-01-12 10:36:00,-0.003,GOOD
WDD.MEM_BER0001_01_LT01_MW,A,2011-01-12 10:37:00,-0.000,GOOD
WDD.MEM_BER0001_01_LT01_MW,A,2011-01-12 10:38:00,-0.000,GOOD
WDD.MEM_BER0001_01_LT01_MW,A,2011-01-12 10:39:00,-0.000,GOOD
```

```
<general>
        <importType>generalCSV</importType>
        <folder>$IMPORT_FOLDER_KETEN$</folder>
        <fileNamePatternFilter>*.csv</fileNamePatternFilter>
        <failedFolder>$IMPORT_FAILED_FOLDER_KETEN$</failedFolder>
        <backupFolder>$IMPORT_BACKUP_FOLDER_KETEN$</backupFolder>
        <table>
                <locationColumn/>
                <skippedColumn/>
                <dateTimeColumn pattern="yyyy-MM-dd HH:mm:ss"/>
                <valueColumn unit="SI"/>
                <flagColumn/>
        </table>
        <idMapId>IdKETEN</idMapId>
        <flagConversionsId>ImportKETENFlagConversions</flagConversionsId>
        <importTimeZone>
                <timeZoneOffset>+01:00</timeZoneOffset>
        </importTimeZone>
        <dataFeedId>CSV files</dataFeedId>
</general>
```

Value column

The valueColum can be used in many different ways by using its attributes:

- **parameterId**: determines which parameter this value belongs to, only used when there is no parameterColumn
- **locationId**: determines which location this value belongs to, only used when there is no locationColumn
- **ignoreNumericalParameters:** when set to true, this column will not be read for numerical parameters. Useful when the value is provided in 2 columns, 1 for numerical and 1 for enumeration (i.e. alphanumeric). It prevents values beings overwritten with an undesired value from the wrong column.
- **ignoreEnumerationParameters**: when set to true, this column will not be read for enumeration parameters. Useful when the value is provided in 2 columns, 1 for numerical and 1 for enumeration. It prevents values beings overwritten with an undesired value from the wrong column.
- **requireNumericalParameters:** when set to true, the import will fail when this column is filled for non-numerical parameters (enumerations). Useful when the value is provided in 2 columns, 1 for numerical and 1 for enumeration and when the wrong column is filled the import file will be rejected so it can be corrected before the data will be imported.
- **requireEnumerationParameters**: when set to true, the import will fail when this column is filled for non-enumeration parameters (numerical). Useful when the value is provided in 2 columns, 1 for numerical and 1 for enumeration and when the wrong column is filled the import file will be rejected so it can be corrected before the data will be imported.

Flag column

A flag column <flagColumn name="Y"/> can be configured to store the quality flag with the value.

Importing multiple flag columns (since 2016.02)

Since 2016.02 it is possible to import multiple flag columns. This works only in combination with multiple value columns. For each flag column a parameter id and/or location id can be specified, this should be done in the same way as the value column so the flag will match the correct value.

Example config multiple flag columns

```
<general>
        <importType>generalCSV</importType>
        <folder>import</folder>
        <table>
                <locationColumn name="NoClimato"/>
                <dateTimeColumn name="Date_Heure" pattern="yyyy-MM-dd HH:mm:ss"/>
                <valueColumn name="TempMin_degC" parameterId="T.m" unit="DEGC"/>
                <flagColumn name="Code_TMin" parameterId="T.m"/>
                <valueColumn name="TempMax_degC" parameterId="T.smelt" unit="DEGC"/>
                <flagColumn name="Code_TMax" parameterId="T.smelt"/>
        </table>
        <flagConversionsId>FlagConversionsMultipleFlagColumns</flagConversionsId>
        <missingValue>-999</missingValue>
</general>
```

Example import file multiple flag columns

```
NoClimato;Date_Heure;TempMin_degC;Code_TMin;TempMax_degC;Code_TMax;TempMoy_degC;Code_TMoy;PrecipTotale_mm;
Code_PTotale;PrecipLiquide_mm;Code_PLiquide;PrecipSolide_mm;Code_PSolide
H-2001;2017-02-03 06:00:00;-27.09;0;-26.66;3;-26.91;1;0;1;0;1;0;1
H-2001;2017-02-03 07:00:00;-14.01;6;-13.27;6;-13.69;1;-999;-999;-999;-999;-999;-999
H-2001;2017-02-03 08:00:00;-15.89;3;-15.64;0;-15.76;1;0;1;0;1;0;1
H-2001;2017-02-03 09:00:00;-17.69;0;-16.14;0;-16.94;1;0;1;0;1;0;1
```

Comment Column

A column can be configured to import a comment for a value: <commentColumn name="columnX"/>.

Importing comments only (since 2014.02)

For all imports a value column is required with the exception of importing comments. If the importer is configured without a valueColumn and a commentColumn was specified, the values will be set to missing values and the comments will be imported.

The CSV files look like:

```
WLOCATION;DATE;TIME;VALUE;TYPE
peilschaal_1;16-12-2014;04:01:22;29.20;Value
peilschaal AFW-2096M;17-12-2014;06:22:31;27.20;Waterstand
schorfkoelerbeek peilschaal_1;17-12-2014;06:24:17;27.20;Waterstand
schorfkoelerbeek peilschaal_1;17-12-2014;06:24:17;gemaaid;Opmerking
```

In this example the value column is marked as a commentColumn. There is no valueColumn configured for this import.

```
<import>
                <general>
                        <importType>generalCSV</importType>
                        <folder>%REGION_HOME%/Import/testImport</folder>
                        <fileNamePatternFilter>*.csv</fileNamePatternFilter>
                        <failedFolder>%REGION_HOME%/ImportFailed/testImport</failedFolder>
                        <backupFolder>%REGION_HOME%/ImportBackup/testImport</backupFolder>
                        <table>
                                <locationColumn name="LOCATION"/>
                                <dateColumn name="DATE" pattern="dd-MM-yyyy"/>
                                <timeColumn name="TIME" pattern="HH:mm:ss"/>
                                <commentColumn name="VALUE"/>
                                <parameterColumn name="TYPE"/>
                        </table>
....
```

To import only comments into FEWS, after the general section the timeSeries have to be configured that contain the comments. Suppose that in the “TYPE” column of the CSV file all comments are marked as “COMMENT”, then the parameter "COMMENT" can be used to import the comment lines. All comments will be stored into FEWS and the values will be set to missing.

Too many comments

Due to the current (technical) handling of the comments within Delft FEWS, it is only possible to have 254 unique comments per time series in memory. Any new comment (e.g. unique comment 255) will not be stored, but instead will get the value "too many comments". This is a known limitation of the import of comments. Similarly, when using a transformation to produce a time series with more than 254 unique comments, the new series will not contain more than 254 unique comments and substitute additional comments with "too many comments". The same holds for viewing the entire series in the TimeSeriesDisplay: the table will show unique comments for 254 values, and for the others it will show "too many comments".

A way to work around this issue is to only manually import series for a period with less than 254 unique comments at once. In this way, for each period all the comments are read, and stored in the datastore (as can be checked through the database viewer). When viewing the entire series in the TimeSeriesDisplay, the table will still show "too many comments" for any comment past the 254 unique comments, even if the comment is stored correctly in the datastore. To show the comment as stored in the datastore for a specific value use the shortcut Ctrl + H or right click --> show history on a value in the table. This will still show the stored comment in the datastore, even if the table display shows "too many comments".

Date time formatting

For the date time formatting please follow the Java conventions.

For example to parse a date 31-12-2023 01:10:23 you would use the following pattern:

```
dd-MM-yyyy HH:mm:ss
```

In case the datetime string contains a T or Z or any part that shouldn't be interpreted use single quotes (') to mark this.

https://docs.oracle.com/javase/8/docs/api/java/text/SimpleDateFormat.html says: Date and time formats are specified by *date and time pattern* strings. Within date and time pattern strings, unquoted letters from `'A'` to `'Z'` and from `'a'` to `'z'` are interpreted as pattern letters representing the components of a date or time string. Text can be quoted using single quotes (`'`) to avoid interpretation. `"'"` represents a single quote. All other characters are not interpreted; they're simply copied into the output string during formatting or matched against the input string during parsing.

Importing flagSourceColumns (since 2015.02)

Below is given an example on how to configure a generalCSV import with flagSource columns.

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:
schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0/timeSeriesImportRun.xsd">
    <import>
        <general>
            <importType>generalCSV</importType>
            <folder>../junit_test_output/nl/wldelft/fews/system/plugin/dataImport/TimeSeriesImportTestData
/import/generalcsv</folder>
            <table>
                <dateTimeColumn name="DATE" pattern="dd-MM-yy HH:mm"/>
                <locationColumn name="LOC"/>
                <unitColumn name="UNIT"/>
                <parameterColumn name="PARAM"/>
                <flagSourceColumn id="A" name="FS_A"/>
                <!-- column with name fs a will be mapped to the flag source column with identifier A -->
                <flagSourceColumn id="B" name="FS_B"/>
                <flagSourceColumn id="C" name="FS_C"/>
                <flagSourceColumn id="D" name="FS_D"/>
                <flagSourceColumn id="E" name="FS_E"/>
                <valueColumn name="VALUE"/>
            </table>
            <logWarningsForUnmappableLocations>true</logWarningsForUnmappableLocations>
            <logWarningsForUnmappableParameters>true</logWarningsForUnmappableParameters>
            <logWarningsForUnmappableQualifiers>true</logWarningsForUnmappableQualifiers>
            <maxLogWarnings>1000</maxLogWarnings>
            <missingValue>-999</missingValue>
            <importTimeZone>
                <timeZoneOffset>+01:00</timeZoneOffset>
            </importTimeZone>
            <dataFeedId>generalCSV</dataFeedId>
        </general>
    </import>
</timeSeriesImportRun>
```

The CSV file for this example could look like this:

```
DATE,LOC,VALUE,UNIT,PARAM,FS_A,FS_B,FS_C,FS_D,FS_E01-01-81 00:00,H-2001,2.3,m,P.m,OK,OK,OK,OK,OK3110040601;
Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:00;
Tubificidae;2;n;MACEV2010;AANTL_MEA;gemeten
```

Note that during import, the mapping goes from 'name' to 'id'. When importing the data, the flagSourceColumnId's should be configured in the flagSourceColumns configuration file.

Importing dates and times with separate year, month, day, hour, minute and second columns (since 2017.01)

There are several options for reading the date and time at which a value should be stored:

- A single dateTimeColumn (with a pattern) can be used for csv files containing a single column with both date and time.
- A separate dateColumn and timeColumn (both with patterns) can be used for csv files containing two columns, one containing the date and the other containing the time.
- A yearColumn, monthColumn and dayColumn can be used (instead of a dateColumn) when a csv file contains separate columns with the year, month and day respectively. Note that when used, all three columns must be specified and present in the csv file, together the columns must result in a valid date.
- An hourColumn, minuteColumn and/or secondColumn can be used (instead of a timeColumn) when a csv file contains separate columns with the hour, minutes and seconds respectively. Note that the hour value must be between 0 and 24 (where 24 is interpreted as 0 the next day), and the minute and second values must be between 0 and 59. The minute and second columns are optional, when an hour column is used without a minute and second column, the minutes and seconds are always set to 0. When an hour column and minute column are used without a second column, the seconds are always set to 0. It is not possible to use an hour and second column withour also specifying a minute column.

Note that the option to specify a time through separate hour, minute and/or second columns, can be combined with a dateColumn, and the option to specify the date through separate year, month and day columns can be combined with a timeColumn. Ultimately, each of the values needed to obtain the full date and time (year, month, day, hour, minute, second) must only be present in one of the configured columns, i.e., you can not use both a dateTimeColumn and a dateColumn (there would be two values for year, month and day), you can not use both an hourColumn and a timeColumn, etc.

A config example using the separate year, month, day, hour, minute and second columns (new in 2017.01):

```
<?xml version="1.0" encoding="UTF-8"?>
<timeSeriesImportRun xmlns="http://www.wldelft.nl/fews" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                xsi:schemaLocation="http://www.wldelft.nl/fews http://fews.wldelft.nl/schemas/version1.0
/timeSeriesImportRun.xsd">
   <import>
      <general>
         <importType>generalCSV</importType>
         <folder>../junit_test_output/nl/wldelft/fews/system/plugin/dataImport/TimeSeriesImportTestData/import
/generalcsvyearmonthdayhourminutesecond</folder>
         <table>
            <dayColumn name="Day"/>
            <monthColumn name="Month"/>
            <yearColumn name="Year"/>
            <hourColumn name="Hour"/>
            <minuteColumn name="Minute"/>
            <secondColumn name="Second"/>
            <locationColumn name="Location"/>
            <parameterColumn name="Parameter"/>
            <valueColumn name="Value"/>
         </table>
         <missingValue>-999</missingValue>
         <importTimeZone>
            <timeZoneName>CET</timeZoneName>
         </importTimeZone>
         <columnSeparator> </columnSeparator>
         <decimalSeparator>.</decimalSeparator>
      </general>
   </import>
</timeSeriesImportRun>
```

The CSV file for this example would look like this:

```
Year Month Day Second Hour Minute Parameter Value Location
1980 01 1 02 4 59 H.m 1.23 H-2001
1980 02 04 33 2 32 H.m 4.56 H-2001
1981 12 31 22 18 0 H.m 7.89 H-2001
```

Importing with Julian Days

Where Julian Days (1-366) are used to defined the day of year, then following configuration can be used:

```
<dateColumn name="Year" pattern="yyyy"/>
<timeColumn name="Day" pattern="DDD"/>
```

The year and date columns of the CSV file for this example would look like this:

```
Year Day
1980 360
1980 361
1981 362
```

Defining column separator and decimal separator

Since 2016.01 it is possible to choose from multiple column separators: comma "," or semi-colon ";" or pipe "|" or tab "	" or space " "

When specifying a column separator it is compulsory to also specify the decimal separator as comma "," or point "."

Example configuration for usage of colum separator and decimal separator

```
<general>
   <importType>generalCSV</importType>
   <folder>importfolder</folder>
   <table>
      <dateTimeColumn name="DATE" pattern="dd-MM-yy HH:mm"/>
      <locationColumn name="LOC"/>
      <unitColumn name="UNIT"/>
      <parameterColumn name="PARAM"/>
      <valueColumn name="VALUE"/>
   </table>
   <missingValue>-999</missingValue>
   <importTimeZone>
      <timeZoneOffset>+01:00</timeZoneOffset>
   </importTimeZone>
   <columnSeparator>|</columnSeparator>
   <decimalSeparator>.</decimalSeparator>
</general>
```

This enables importing for example the format below:

```
DATE|LOC|VALUE|UNIT|PARAM
01-01-81 00:00|H-2001|2.3|m|P.m
```

Samples Import

For the import of ecological data based on samples, the general csv import is used extensively because there is a lot of information per timeseries.

Besides columns for location, parameter, date/time, value and unit, there is a column for sample id and multiple columns for qualifiers and sample properties.

The CSV files have just a single time step of a single time series per line, for example:

```
SMP_CODE;SMP_NAME;COORD_X_S;COORD_Y_S;ANAL_CODE;EMP_NR_S;PROD_CODE;METH_SAMP;METH_ANAL;EMP_NR_A;LOC_CODE;
DATE_SMP;PAR_REF;waarde;eenheid;BRON;parameter_id;type
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Tubifex tubifex;2;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Tubificidae;2;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Hypania invalida;5;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Gammarus tigrinus;10;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Dicrotendipes pulsus;3;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Microtendipes chloris agg.;3;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Polypedilum;5;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Procladius;1;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Pseudochironomus prasinatus;1;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Stictochironomus;1;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Ceratopogonidae;3;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Caenis horaria;1;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Ecnomus tenellus;1;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Corbicula fluminea;6;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Dreissena polymorpha;268;n;MACEV2010;AANTL_MEA;gemeten
3110040601;Spiegelplas 4;133054;475032;MEA;WP_CAS_ZUY;OW;Van Veenhapper;SPV7030;WP_CAS_ZUY;SBP130;06-04-10 00:
00;Potamopyrgus antipodarum;1;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Piscicola;1;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Tubificidae;1;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Hygrobates longipalpis;1;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Piona;1;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Chelicorophium curvispinum;23;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Gammarus tigrinus;22;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Asellus aquaticus;3;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Limnomysis benedeni;29;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Ablabesmyia;1;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Chironomus;5;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Chironomus riparius agg.;9;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Clinotanypus nervosus;4;n;MACEV2010;AANTL_MEA;gemeten
3110040602;Spiegelplas 1;132546;476323;MEA;WP_CAS_ZUY;OW;macrofaunanet;SPV7030;WP_CAS_ZUY;SBP175;06-04-10 00:00;
Corynoneura scutellata agg.;2;n;MACEV2010;AANTL_MEA;gemeten
```

With corresponding import xml:

```
                <general>
                        <importType>generalCSV</importType>
                        <folder>$IMPORT_FOLDER$/HydroBiologie</folder>
                        <table>
                                <dateTimeColumn name="DATE_SMP" pattern="dd-MM-yyyy HH:mm"/>
                                <locationColumn name="LOC_CODE"/>
                                <unitColumn name="Eenheid"/>
                                <parameterColumn name="PARAMETER_ID"/>
                                <qualifierColumn name="PAR_REF" prefix="PAR_REF_"/>
                                <qualifierColumn name="PROD_CODE" prefix="PROD_CODE_"/>
                                <qualifierColumn name="ANAL_CODE" prefix="ANAL_CODE_"/>
                                <qualifierColumn name="METH_ANAL" prefix="METH_ANAL_"/>
                                <qualifierColumn name="METH_SAMP" prefix="METH_SAMP_"/>
                                <qualifierColumn name="TYPE" prefix="TYPE_"/>
                                <sampleIdColumn name="SMP_CODE"/>
                                <propertyColumn name="SMP_NAME" key="SMP_NAME"/>
                                <propertyColumn name="COORD_X_S" key="COORD_X_S"/>
                                <propertyColumn name="COORD_Y_S" key="COORD_Y_S"/>
                                <propertyColumn name="EMP_NR_S" key="EMP_NR_S"/>
                                <propertyColumn name="EMP_NR_A" key="EMP_NR_A"/>
                                <propertyColumn name="BRON" key="BRON"/>
                                <valueColumn name="Waarde"/>
                        </table>
                        <idMapId>IdImport_HydroBiologie</idMapId>
                        <dataFeedId>generalCSV</dataFeedId>
                </general>
```

SampleId prefix

Since 2017.01 an optional prefix attribute can be specified for the sampleIdColumn:

```
                                <sampleIdColumn name="SMP_CODE" prefix="prefix_"/>
```

If present the given prefix will be added to the front of each sample id in the file. This allows you to differentiate between samples from different imports which would otherwise have the same sampleId, preventing the samples from another import from being overwritten.

Qualifier columns

Multiple qualifiers are used per time series divided over different colums. The column a qualifier originated from is important to keep track of. That is why a "prefix" attribute can be configured for a qualifier column. This prefix is added to the content of the column to form the external qualifier id.

With the use of id mapping, this prefix helps mapping to the correct qualifiers using a qualifier id function <qualifierIdFunction externalQualifierFunction=" @ExternalQualifierId@"/> and qualifiers configured in a csv file:

```
        <csvFile>
                <file>par_ref_qualifiers.csv</file>
                <id>%qualifierId%</id>
                <name>%taxonname%</name>
                <attribute id="ExternalQualifierId">
                        <text>PAR_REF_%taxonname%</text>
                </attribute>
        </csvFile>
```

The content out of the qualifier column with prefix "PAR_REF_" will in this way be used to map to a qualifier in par_ref_qualifiers.csv that has equal content as "taxonname".

In the same way the content out of the qualifier column with prefix "ANAL_CODE_" will be mapped to a qualifier in analyse_code.csv that has equal content as "qualifierId" using the next qualifier configuration:

```
        <csvFile>
                <file>analyse_code.csv</file>
                <id>%qualifierId%</id>
                <name>%qualifierName%</name>
                <attribute id="ExternalQualifierId">
                        <text>ANAL_CODE_%qualifierId%</text>
                </attribute>
        </csvFile>
```

With the use of the above configuration it will be possible to export the qualifiers to the same columns again using the same qualifier if function <qualifierIdFunction externalQualifierFunction="@ExternalQualifierId@"/> and "prefix" attributes in the qualifier columns in the export xml:

```
                <general>
                        <exportType>generalCsv</exportType>
                        <folder>$EXPORT_FOLDER$/HydroBiologie</folder>
                        <exportFileName>
                                <name>ExportGeneralCsv.csv</name>
                        </exportFileName>
                        <table>
                                <dateTimeColumn name="DATE_SMP" pattern="dd-MM-yyyy HH:mm"/>
                                <locationColumn name="LOC_CODE"/>
                                <unitColumn name="Eenheid"/>
                                <parameterColumn name="PARAMETER_ID"/>
                                <qualifierColumn name="PAR_REF" prefix="PAR_REF_"/>
                                <qualifierColumn name="PROD_CODE" prefix="PROD_CODE_"/>
                                <qualifierColumn name="ANAL_CODE" prefix="ANAL_CODE_"/>
                                <qualifierColumn name="METH_ANAL" prefix="METH_ANAL_"/>
                                <qualifierColumn name="METH_SAMP" prefix="METH_SAMP_"/>
                                <qualifierColumn name="TYPE" prefix="TYPE_"/>
                                <sampleIdColumn name="SMP_CODE"/>
                                <propertyColumn name="SMP_NAME" key="SMP_NAME"/>
                                <propertyColumn name="COORD_X_S" key="COORD_X_S"/>
                                <propertyColumn name="COORD_Y_S" key="COORD_Y_S"/>
                                <propertyColumn name="EMP_NR_S" key="EMP_NR_S"/>
                                <propertyColumn name="EMP_NR_A" key="EMP_NR_A"/>
                                <propertyColumn name="BRON" key="BRON"/>
                                <valueColumn name="Waarde"/>
                        </table>
                        <idMapId>IdExport_HydroBiologie</idMapId>
                </general>
```

Property columns

Property columns can be used to define in which columns values can be found for specific sample properties where a "key" needs to be configured. These properties will be stored with the sample and not the time series itself. Because a sample will contain values for time series at different time steps and for each time step the properties are defined there is some redundancy. For each imported time step the property values will be checked wheter they are valid and are the same for all time steps in the sample.

Validation and logging

During the import the data for each time step is validated in different ways, failures will be logged as warning.

Property validation failure warnings:

- Sample properties should be the same for all values with the same sample id
- Property <key> with value <value> not present in current sample properties
- Property <key> has different value <value> than in current sample properties <currentValue>
- Property <key> missing, but present in current sample properties with value <value>
- Sample property key <key> not found in SampleMetadataSchema.xml
- Enumeration value <value> for property <key> not found in SampleMetadataSchema.xml

Location validation failure warnings when <logWarningsForUnmappableLocations>true</logWarningsForUnmappableLocations>:

- No FEWS location for external id <locationId> (without id map)
- External location <locationId> can not be mapped to FEWS location with id map <idMap>

Parameter validation failure warnings when <logWarningsForUnmappableParameters>true</logWarningsForUnmappableParameters>:

- No FEWS parameter for external id <parameterId> (without id map)
- External parameter <parameterId> can not be mapped to FEWS parameter with id map <idMap>

Qualifier validation failure warnings when <logWarningsForUnmappableQualifiers>true</logWarningsForUnmappableQualifiers>:

- No FEWS qualifier for external id <qualifierId> (without id map)
- External qualifier <qualifierId> can not be mapped to FEWS qualifier with id map <idMap>

Because sample data files can contain many lines of data each consisting of many properties and qualifiers lots of validation failre warnings can be logged. A maximum amount of log warnings can be configured to overrule the default of 5 <maxLogWarnings>100000</maxLogWarnings>. It is also possible to write all warnings to a separate file (instead of the FEWS log) in the failed folder with identical name to the import file but ending with .log by configuring <logWarningsToSeparateFile>true</logWarningsToSeparateFile>. This can be very useful to not have too many warnings in the FEWS database and have a separate file as feedback to the either the configurator of the FEWS system or the supplier of the import file.

Re-importing samples

When a sample is reimported the default behaviour is that the entire sample will be overwritten. So all old data will be deleted.

When <mergeWithExistingSampleData>true<mergeWithExistingSampleData> is configured a reimported sample will be merged with the old data. Only the time series that where already present in the sample are overwritten the rest is added. Be careful with changing qualifiers when merging because changing a qualifier will result in a different time series. This "new" time series will be added to the sample and the "old" will remain, instead of overwriting the old with the new. Also sample properties can not be changed when merging because sample properties should be the same for the entire sample, this also applies to already and newly imported data for the same sample.

When re-importing samples make sure the location and time are exactly the same, since it is used as an internal id. Different samples with the same location time combination are handled internally by applying a behind the scene time shift in the storage, when samples are read this time shift is automatically corrected so the data will not suffer from this shift.

rejectCompleteSampleOnUnmappableId

When true, if part of an imported sample cannot be mapped to a time serie in FEWS the whole sample will be rejected, when false part of a sample can be imported.

rejectCompleteSampleOnDuplicateValues

When true, if for the same time step within a time series multiple values are imported the sample will be rejected and a warning logged

Details of the import format

If the first line contains a comma, the decimal separator is taken to be a period (.), otherwise it is supposed to be a semicolon (;) and the decimal separator is taken to be a comma. This way locale-specific CSV files are supported.

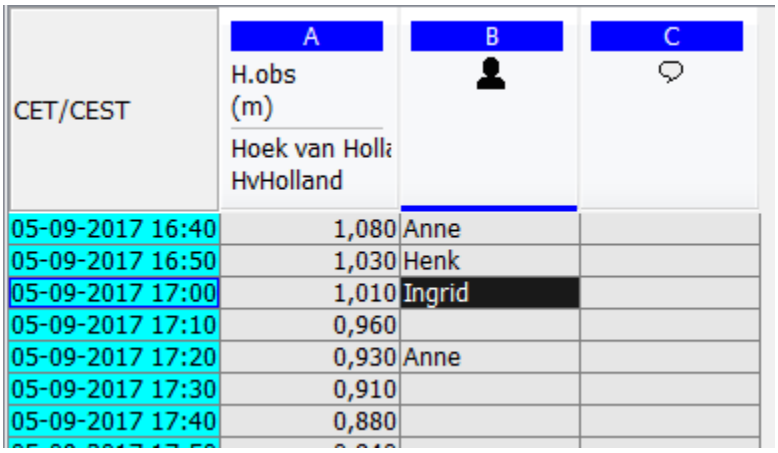
DATE	LOC	VALUE	UNIT	PARAM	FS_A	FS_B	FS_C	FS_D	FS_E
01-01-81 00:00	H-2001	2.3	m	P.m	OK	OK	OK	OK	OK

userColumn

Since 2016.02 there is a column that allows for specifying a user with each data point.

Example configuration for usage of userColumn

```
<general>
   <importType>generalCSV</importType>
   <folder>importfolder</folder>
   <table>
      <dateTimeColumn name="DATE" pattern="dd-MM-yy HH:mm"/>
      <locationColumn name="LOC"/>
      <unitColumn name="UNIT"/>
      <parameterColumn name="PARAM"/>
      <valueColumn name="VALUE"/>
      <userColumn name="USER"/>
   </table>
</general>
```



CET/CEST	A H.obs (m) Hoek van Holl HvHolland	B	C
05-09-2017 16:40	1,080	Anne	
05-09-2017 16:50	1,030	Henk	
05-09-2017 17:00	1,010	Ingrid	
05-09-2017 17:10	0,960		
05-09-2017 17:20	0,930	Anne	
05-09-2017 17:30	0,910		
05-09-2017 17:40	0,880		

limitSymbolColumn

Since 2014.02 there is a column that allows for specifying a limit symbol with each data point.

Example configuration for usage of userColumn

```
<general>
   <importType>generalCSV</importType>
   <folder>importfolder</folder>
   <table>
      <dateTimeColumn name="DATE" pattern="dd-MM-yy HH:mm"/>
      <locationColumn name="LOC"/>
      <unitColumn name="UNIT"/>
      <parameterColumn name="PARAM"/>
      <valueColumn name="VALUE"/>
      <limitSymbolColumn name="SYMBOL"/>
   </table>
</general>
```

This column can recognise < as below detection range, or > as above detection range.

Column(s) not available for this import type

attributeColumn (only available for export since attribute values can only be configured, not imported)

ensembleColumn (only available for export, not supported for import yet)

ensembleMemberColumn (only available for export, not supported for import yet)