REFERENCE GUIDE

IMPORTING CUSTOM FILE FORMATS INTO TRIMBLE ACCESS™

Revision C
June 2021

Introduction

You can import points (coordinates) and lines in custom file formats into Trimble Access™ software using import format definition files. Import format definition files must be defined as XML files. This document provides the format definition rules.

A number of standard import format definition files are provided with the Trimble Access software, in the **Trimble Data** folder. You can use these files to help create new import format definition files.

This document applies to all versions of Trimble Access.

File definition prerequisites

For point file importing there are the following prerequisites:

- A point name is required for each point.
- For grid points, a northing and easting value or an elevation value is required, that is, as well as three-dimensional points, two-dimensional or one-dimensional points are allowed.
- For Global points, the latitude, longitude, and height is required, that is, only three-dimensional points are allowed.
- For local ellipsoid points, the latitude and longitude or a height value is required, that is, as well as three-dimensional points, two-dimensional or one-dimensional points are allowed.

For line file importing there are the following prerequisites:

- A line name is required for each line.
- Each line must have either the start point and end point defined, or the start point and an azimuth and distance provided.

Importing custom format files

The format definitions for custom point and line import files are held in XML format files with the file extension "IXL". Each format definition file should be assigned a file name that clearly indicates the point or line file format the definition has been created to support. A separate import definition file is required for each defined point or line format. The Trimble Access software automatically searches for all the files in the **Trimble Data** folder with the "IXL" file extension and presents the file names as the selectable import formats. The file format definitions are as follows:

The file starts with a standard XML file header:

<?xml version="1.0" encoding="UTF-8"?>

NOTE – *Within an XML file, all element names are case sensitive so make sure that all element names are exactly right when you create a format definition.*

Following the header element is the **<ImportFormat version="1.0">** element that contains the file format definition. This element has a version attribute associated with it. Within the **<ImportFormat>** element, include the appropriate elements to define how the data items are to be read from the file being imported.

The following elements can be included in an **<ImportFormat>** element to provide the format definition. Some elements are required and others are optional – indicated for each element.

<FormatType> (required)

The **<FormatType>** element defines the file format the definition is designed for. The values are:

- **Fixed**, where the file contains fixed format lines with the values to be read occupying specific locations in each line
- **CSV**, where the file contains comma separated values
- **Delimited**, where the files contains values separated by a specified delimiting character or characters using the **<Delimiter>** element

<Encoding> (optional)

The **<Encoding>** element defines the way imported data will be interpreted. The values are:

- **UTF-8**, where strings are already UTF-8 encoded
- **MBCS**, where strings are ASCII / Multi-byte encoded and will be converted to UTF-8

<EntityType> (optional)

The **<EntityType>** element specifies the type of element to be imported. This can be set to **Point** or **Line**. If this element is omitted the system will assume that point records are being defined.

<CommaAsDecimalSeparator> (optional)

The **<CommaAsDecimalSeparator>** element can have only have two values—**Yes** or **No**. If this element is omitted the definition will treat this setting as if it had been set to **No**. In a definition with the **<Type>** element set to **CSV**, and this element set to **Yes**, the importing routines automatically expect that the separating character for the file being imported is a semi-colon rather than a comma (thus avoiding conflicts about where fields start and end).

<Delimiter> (optional)

The **<Delimiter>** element specifies the character(s) to be used as the field delimiter (used to separate the fields in an import file) for definitions set up as a **Delimited <FormatType>**. The **<Delimiter>** defaults to a comma character if the **<Delimiter>** element is not defined in a **Delimited <FormatType>** definition. If you

are defining a format for a file that uses the Tab character (ASCII character 9) as the delimiter, you must define it using **<Delimiter>Tab</Delimiter>**. If you need to handle a specific format, you can specify more than a single character as the delimiter.

<FileSpecification> (required)

Use the **<FileSpecification>** element to specify the type of files that are shown as available for import for this format definition. For example, if this element is defined as **<FileSpecification>*.csv;*.txt</FileSpecification>**, then all files in the **Trimble Data** folder with the file extensions **csv** and **txt** are available for selection. If more than one file type specification is included in this element, the specifications must be separated with semi-colon characters.

<HeaderLinesToSkip> (optional)

As some file formats may have one or more header lines before the start of the data that is to be imported, the **<HeaderLinesToSkip>** element is provided to let you specify how many lines should be ignored at the start of a file. If this element is omitted from a definition, it is assumed that no header lines need to be skipped. If there are five header lines to be skipped, this element would be defined as **<HeaderLinesToSkip>5</HeaderLinesToSkip>**.

<Match> (optional)

Multiple **<Match>** elements (up to a maximum of 20) can be used to define specific text that must be matched on a line before the data from the line will be accepted. This is useful when a file format to be imported has different types of data in it and you want to ensure that you import the data only from the appropriate lines.

Within each **<Match>** element, you can use one or more **<Component>** elements to define one or more text matches to be carried out. If there is only a single text match to be carried out, you can omit the **<Component>** element but it is required when more than one text match is to be applied. Defining more than one text match enables you to test that more than one text sequence is present before the line (or lines) containing the data to be imported is read. If more than one text match is specified, each match must be true before the point data will be read.

The following elements define the text matching to be carried out:

- **<StartColumn>**

  Use the **<StartColumn>** element to specify the starting character position of the text from the import file to be matched.

- **<Length>**

  Use the **<Length>** element to define how many characters (starting from the specified **<StartColumn>**) are to be read from the import file for comparison purposes.

- **<Text>**

  Use the **<Text>** element to specify the text to be matched. The text matching carried out is case sensitive and any leading or trailing spaces in the text read from the file, or in the specified match text will be stripped before the matching operation.

- **<Line>**

  If the file format of the data being imported spans multiple lines you can use the **<Line>** element to specify which line, in the group of lines from which the point data is being extracted, the text for matching is to be read from.

- **<CSVField>**

  If the file format of the data being imported is delimited, use this element to specify the field containing the text to be matched.

<Match> examples

This **<Match>** definition reads the data for a point if the third and fourth characters on the first line of a two line point definition are '11' and if the eighth, ninth and tenth characters on the second line are '212':

```
<Match>
    <Component>
        <StartColumn>3</StartColumn>
        <Length>2</Length>
        <Text>11</Text>
        <Line>1</Line>
    </Component>
    <Component>
        <StartColumn>8</StartColumn>
        <Length>3</Length>
        <Text>212</Text>
        <Line>2</Line>
    </Component>
</Match>
```

This <Match> definition reads the data for a point if the first two characters on the line are 'XX':

```
<Match>
    <StartColumn>1</StartColumn>
```

```
        <Length>2</Length>
        <Text>XX</Text>
</Match>
```

<Omit> (optional)

Multiple **<Omit>** elements (up to a maximum of 20) can be used to define specific text that, when matched in a line (or lines) containing data, will result in the data being omitted from the import operation. This is useful when a file format to be imported has different types of data in it, and you want to omit data that you do not want imported.

Within each **<Omit>** element, you can use one or more **<Component>** elements to define one or more text matches to be carried out. If there is only a single text match to be carried out, you can omit the **<Component>** element, but it is required when more than one text match is to be applied. Defining more than one text match enables you to test that more than one text sequence is present before the line (or lines) containing the data is omitted. If more than one text match is specified, each match must be **true** before the point data will be omitted.

Each **<Omit>** element must contain a **<NumberOfLinesToOmit>** element that specifies how many lines are to be omitted, based on the defined text match.

The following elements define the text matching to be carried out:

- **<StartColumn>**

  Use the **<StartColumn>** element to specify the starting character position of the text from the import file to be matched.

- **<Length>**

  Use the **<Length>** element to define how many characters (starting from the specified **<StartColumn>**) are to be read from the import file for comparison purposes.

- **<Text>**

  Use the **<Text>** element to specify the text to be matched. The text matching carried out is case sensitive and any leading or trailing spaces in the text read from the file, or in the specified match text will be stripped before the matching operation.

- **<Line>**

  If the file format of the data being imported spans multiple lines you can use the **<Line>** element to specify which line, in the group of lines from which the point data is being extracted, the text for matching is to be read from.

- **<CSVField>**

  If the file format of the data being imported is delimited, use this element to specify the field containing the text to be matched.

<Omit> examples

This **<Omit>** definition omits the two lines of data for a point if the first and second characters on the first line of a two line point definition are '11' and if the 20th, 21st, and 22nd characters on the second line are 'DEL':

```
<Omit>
    <Component>
        <StartColumn>1</StartColumn>
        <Length>2</Length>
        <Text>11</Text>
        <Line>1</Line>
    </Component>
    <Component>
        <StartColumn>20</StartColumn>
        <Length>3</Length>
        <Text>DEL</Text>
        <Line>2</Line>
    </Component>
    <NumberOfLinesToOmit>2<NumberOfLinesToOmit>
</Omit>
```

This **<Omit>** definition will omit the single line of data for a point if the first character on the line is '!':

```
<Omit>
    <StartColumn>1</StartColumn>
    <Length>1</Length>
    <Text>!</Text>
    <NumberOfLinesToOmit>1<NumberOfLinesToOmit>
</Omit>
```

<DefaultDataLineMatch> (optional)

A single **<DefaultDataLineMatch>** element can be used to define a match definition for identifying lines in a file that contain default data that will be assigned to following points. This is useful when a file format to be imported has a specific structure that allows repeated data in following points to be included a single time prior to the points. Note that this repeated data can only be handled in a **single** line that precedes the points that are to use this data. When defining the import format, the record definitions must be structured so that the repeated data is treated as if it is present in every record.

The following elements define the text matching to be carried out as part of the **<DefaultDataLineMatch>** element:

- **<StartColumn>**

  Use the **<StartColumn>** element to specify the starting character position of the text from the import file to be matched.

- **<Length>**

  Use the **<Length>** element to define how many characters (starting from the specified **<StartColumn>**) are to be read from the import file for comparison purposes.

- **<Text>**

  Use the **<Text>** element to specify the text to be matched. The text matching carried out is case sensitive and any leading or trailing spaces in the text read from the file, or in the specified match text will be stripped before the matching operation.

- **<Line>**

  If the file format of the data being imported spans multiple lines you can use the **<Line>** element to specify which line, in the group of lines from which the point data is being extracted, the text for matching is to be read from.

- **<CSVField>**

  If the file format of the data being imported is delimited, use this element to specify the field containing the text to be matched.

<DefaultDataLineMatch> example

For example, the following file with repeated Code, Desc1, and Desc2 data values can be handled as a space delimited file using the **<Field>** definitions following this data table:

Code	FCE	Desc1: P&W	Desc2: 10
1	299350.279	522734.975	15.263

2	299362.583	522712.673	18.265
3	299355.653	522742.151	12.845
Code	PP	Desc1: SS	Desc2: 1.3
4	299349.378	522741.250	22.358
5	299343.953	522786.482	16.352
6	299356.554	522735.876	15.846
Code	PP	Desc1: Light	Desc2: 8
7	299371.233	522735.456	14.964
8	299363.007	522734.275	16.832
9	299373.154	522722.074	15.436
Code	TREE	Desc1: OAK	Desc2: 5.5
10	299382.165	522762.168	12.698
Code	TREE	Desc1: PINE	Desc2: 8
11	299356.398	522787.958	12.836
Code	TREE	Desc1: BIRCH	Desc2: 6.5
12	299321.769	522735.925	18.102

In the following definitions, each point element is considered as consisting of two lines and when required the current default line is effectively inserted to complete the two-line definition.

```
<Fields>
    <PointName>
        <Component>
            <Line>2</Line>
            <CSVField>1</CSVField>
        </Component>
    </PointName>
    <Northing>
        <Line>2</Line>
        <CSVField>2</CSVField>
    </Northing>
    <Easting>
```

```
        <Line>2</Line>
        <CSVField>3</CSVField>
</Easting>
<Elevation>
        <Line>2</Line>
        <CSVField>4</CSVField>
</Elevation>
<Code>
        <Line>1</Line>
        <CSVField>2</CSVField>
</Code>
<Description1>
        <Line>1</Line>
        <CSVField>4</CSVField>
</Description1>
<Description2>
        <Line>1</Line>
        <CSVField>6</CSVField>
</Description2>
<DefaultDataLineMatch>
        <Component>
                <Line>1</Line>
                <CSVField>1</CSVField>
                <Text>Code:</Text>
        </Component>
        <Component>
                <Line>1</Line>
                <CSVField>3</CSVField>
```

```
            <Text>Desc1:</Text>
        </Component>
        <Component>
            <Line>1</Line>
            <CSVField>5</CSVField>
            <Text>Desc2:</Text>
        </Component>
    </DefaultDataLineMatch>
</Fields>
```

<LatLongFormat> (optional)

The **<LatLongFormat>** element is available to let you specify the format used for latitude and longitude values in the imported file. This element can be set to **DMSDegrees** or **DecimalDegrees** and if the element is not included then the default expected format is **DMSDegrees**. When reading **DMSDegrees** values the system parses the latitude and longitude values by looking for the degrees(°), single quote ('), and double quote (") identifiers, or a decimal point, to provide the split between the degrees and minutes values. This means that values of the forms **ddd°mm'ss.ssss"** or **ddd.mmssssss** can be imported.

<Fields> (required)

The **<Fields>** element contains the individual definitions for the data to be read from the file being imported. Each of the elements under the **<Fields>** element identifies the data type that can be read into the Trimble Access software.

For importing point files, the data type elements are:

<PointName>

<Northing>

<Easting>

<Elevation>

<Code>

<WGS84Latitude>

<WGS84Longitude>

<WGS84Height>

<LocalLatitude>

<LocalLongitude>

<LocalHeight>

<Note>

<Description1>

<Description2>

For importing line files, the data type elements are:

<LineName>

<StartPointName>

<EndPointName>

<StartStation>

<StationInterval>

<Azimuth>

<Distance>

<Code>

<Note>

<Description1>

<Description2>

Within each of these data type elements the following elements are available for defining the data to be read:

- **<StartColumn>**

  Use the **<StartColumn>** element to specify the starting character position for the value to be read.

- **<Length>**

  Use the **<Length>** element to define how many characters (starting from the specified **<StartColumn>**) are to be read for the value.

- **<Line>**

  If the file format of the data being imported spans multiple lines you can use the <Line> element to specify which line, in the group of lines from which the point data is being extracted, the value is to be read from.

- **<CSVField>**

  When the format type is CSV or Delimited, use the **<CSVField>** element to specify which comma separated or delimited field (on the appropriate line if specified) the value is to be read from. Even with CSV or Delimited type files, you can still use the **<StartColumn>** and **<Length>** elements to define a

particular portion of a field to be extracted. In this case the **<StartColumn>** value defines the starting column within the field itself.

- **<ImpliedDecimalPlaces>**

  When reading numeric values from a file that has values output without their decimal point character, use the **<ImpliedDecimalPlaces>** element to define where the decimal point is inserted into the value (how many decimal places there will be).

  For example, if you set the **<ImpliedDecimalPlaces>** element to "3" (**<ImpliedDecimalPlaces>3</ImpliedDecimalPlaces>**), the value "803265184" will be interpreted as "803265.184".

- **<NullValue>**

  Often, a special value is used in a file format to indicate a null value. Use the **<NullValue>** element in a numeric field definition to specify the actual value or string that is to be interpreted as a null value.

  For example, if a file format uses "-999999" as the null value for elevations, then include **<NullValue>-999999</NullValue>** in the **<Elevation>** element definition to have any points with elevation values of "-999999" interpreted as null values when a file of this type is imported.

For the **<PointName>**, **<Code>**, **<Note>**, **<Description1>**, **<Description2>**, **<StartPointName>**, and **<EndPointName>** data types you can specify multiple **<Component>** elements to allow the imported point name, code or note to be 'assembled' from multiple fields within the data being read.

In association with the ability to assemble the **<PointName>**, **<Code>**, **<Note>**, **<Description1>**, **<Description2>**, **<StartPointName>**, and **<EndPointName>** from multiple fields two further elements can be used within each **<Component>** element to specify how the fields are assembled:

- **<Prefix>**

  The **<Prefix>** element lets you specify some text that will precede the data extracted by this **<Component>** as the point name, code or note is assembled.

- **<Suffix>**

  The **<Suffix>** element lets you specify some text that will follow the data extracted by this **<Component>** as the point name, code or note is assembled.

The **<Component>** element is optional and may be included or left out when the **<PointName>**, **<Code>**, **<Note>**, **<Description1>**, **<Description2>**, **<StartPointName>**, or **<EndPointName>** includes a single definition.

<Fields> examples

The following definition extracts the first eight characters from the first line in a pair, then the 32nd to the 35th characters from the second line, and assembles them as the point name with a hyphen (-) character

inserted between the two portions of the name:

```
<PointName>
    <Component>
        <StartColumn>1</StartColumn>
        <Length>8</Length>
        <Line>1</Line>
        <Suffix>-</Suffix>
    </Component>
    <Component>
        <StartColumn>32</StartColumn>
        <Length>4</Length>
        <Line>2</Line>
    </Component>
</PointName>
```

The following definition builds the code from the sixth and fifth CSV fields and inserts an underscore character between the text from the two fields:

```
<Code>
    <Component>
        <CSVField>6</CSVField>
    </Component>
    <Component>
        <CSVField>5</CSVField>
        <Prefix>_</Prefix>
    </Component>
</Code>
```

The following definition simply uses the first field from a CSV file as the point name:

```
<PointName>
    <CSVField>1</CSVField>
</PointName>
```

General notes

When importing data from formats where the fields for a single point are spread over more than one line, the importing routines determine the number of lines that constitute a point definition based on the largest **<Line>** element value in the definition. During the import process, the file is read as a series of groups of lines based on this largest **<Line>** element value.

When an import definition covering multiple lines also defines text matching, and a text match fails, the import process increments only a single line rather than the complete group of lines based on the largest **<Line>** element value. This approach allows a new match to be established as soon as possible.

Lines from line definition files are not imported unless the **<StartPoint>** and **<EndPoint>** (if appropriate) already exist in the job that the lines are being imported into. Therefore, make sure that the appropriate points have been added to, or imported into, the job prior to importing the lines.

Any **<Azimuth>** and **<Distance>** values in a line definition file are interpreted according to the current Angles and Distance and grid coordinates settings in the Properties of current job option.