c2x-cmd

Manual

(Regarding version 1.8.0)

Document history

Version	Date	Author	Reason/Changes
0.1	11/10/2011	J. Gödeke	Initial creation
0.2	01/12/2012	J. Gödeke	Removed minor spelling mistakes

Inhaltsverzeichnis

1 CSV2XML 4

1.1 Prologue 4
1.2 License 4
1.3 Contact 4
1.4 FAQ 4

2 OVERVIEW 5

2.1 Starting c2x-cmd from c2x-wizard 5
2.2 Starting c2x-cmd from a command line 5
2.3 Installation and Execution 6
2.3.1 Windows 6
2.3.2 All operating systems based on Unix-/Linux (incl. MAC OS X) 7

3 C2X-CMD OVERVIEW 8

3.1 Generally annotations concerning the options 8
3.2 Additive options 8
3.2.1 As parameter sequence 8
3.2.2 As option sequence 8
3.3 Overview of options 8
3.3.1 Options which are supported by c2x-wizard 8
3.3.2 Options that are not supported by c2x-wizard: 9
3.3.3 Configuration files 9
3.4 She-Bang Execution 10
3.5 c2x-cmd options 10
3.5.1 Option : Source filename (m) / -s 10
3.5.2 Option : Target filename (m) / -t 11
3.5.3 Option : Separators by plain text / -sep 11
3.5.4 Option : Separator by ASCII code / -sep 11
3.5.5 Option : Encoding / -enc 12
3.5.6 Option : Convert command line parameters / -cc 12
3.5.7 Option : XML Mode / -m 12
3.5.7.1 Mode 1 : <eRecord><COLname>CELLvalue</ColName></eRecord> 13
3.5.7.2 Mode 2 : <eRecord><COLname aValue="CELLvalue"/></eRecord> 13
3.5.7.3 Mode 3 : <eRecord><eItem aName="COLname">CELLvalue</eItem></eRecord> 13
3.5.7.4 Mode 4 : <eRecord><eItem aName="COLname" aValue="CELLvalue"/></eRecord> 14
3.5.7.5 Mode 5 : <eRecord COLname="CELLvalue"/> 14
3.5.8 Option : suppress empty elements / -sup 15
3.5.9 Option : write „num“ attribute to record-element / -wn 15
3.5.10 Option : write xml:id=“id." attribute to record-element / -wid 15
3.5.11 Option : Stylesheet(s) / -styles 16
3.5.12 Option : Validation / -xsd & -dtd 16
3.5.13 Option : Use Namespace / -xmlns 17
3.5.14 Option : Alias / -alias 17
3.5.15 Option : Version information / -V 18
3.5.16 Option : Usage / -? 18
3.6 More examples 18

4 APPENDIX 19

4.1 References 19

1 csv2xml

1.1 Prologue

I'm always surprised how often CSV files have to be converted to XML. Thereby the first version of my converter csv2xml which I had created in 2002 was only a demo for a new CSV parser class which I had written in C++. I had released the software on my homepage and was amazed how popular it got. Soon the first change requests had reached me and favorable was the wish to use csv2xml on non-Windows systems. In 2003 I divided csv2xml in two tools. One was the command line converter csv2xml and the other was its calling graphical application csv2xml_win. It was the first time that csv2xml was also released for Linux. In 2007 I've renamed csv2xml into c2x-cmd, because in between its first release and 2007 many tools like csv2xml were published. c2x-cmd was the first version that had been compiled for popular other operating systems like AIX and MAC OS X. The problem was the replacement of csv2xml_win. This took about 4 years, cause my family and my job didn't offer me enough time for that. The other problem was that I had to find a suitable programming language which would enable me to write a graphical application that works on each operating system that was supported by c2x-cmd and its ansi c++ code. In the end I decided do use Java and Oracles (former SUN) integrated development environment Netbeans. I hope that the resulting application is impressive enough for you that you are willing to install an Java Runtime Environment if you haven't already done that.

1.2 License

c2x-cmd & c2x-wizard are subject of a FREEWARE License. That means that you could use these applications for free whether in a commercial or non-commercial purpose. You must not sell or change these applications. I don't give any warranty for damages that happen by using these applications.

1.3 Contact

If you have further questions, change requests or error reports don't hesitate to get in contact with me.

Jens Gödeke

Please use my contact page to get in contact with me.
http://www.jens-goedeke.eu/contact

You'll find the newest version of the applications or the documentation on the homepage of c2x-cmd :
http://www.jens-goedeke.eu/tools/csv2xml

1.4 FAQ

Not yet.

2 Overview

c2x-cmd is the command line converter which could be started in four different ways (Figure 1).
Hint : Only the first three ways are supported by Windows.

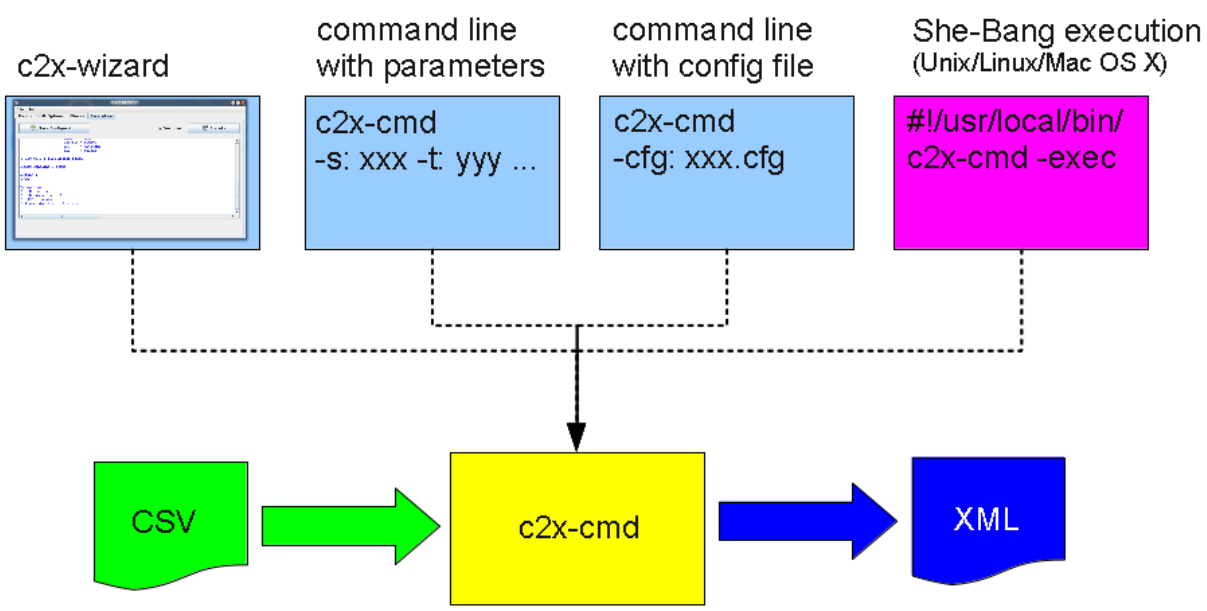

Figure 1: Different calling types of c2x-cmd

2.1 Starting c2x-cmd from c2x-wizard

c2x-wizard is the generator tool which should help the user to set the right options with an graphical application. c2x-wizard has the following tasks:

- Supporting the user so that he is able to set the c2x-cmd options correctly
- Option to save the user settings as configuration files
- Option to send the user settings directly to c2x-cmd and execute c2x-cmd immediately afterward.

You find more details on how to use c2x-wizard in its own documentation. See c2x-wizard manual.

2.2 Starting c2x-cmd from a command line

The advantage of a direct call of c2x-cmd from a command line is that you may run c2x-cmd in an automatic process like a shell or a batch script.
The execution over a command line works:

- if you give the necessary parameters as a sequence
- by giving a configuration file (maybe generated with c2x-wizard) or

unfortunately only under OS which are based on Linux or Unix:

- by using She-Bang[1]-Execution

See chapter 3 for details.

[1] see http://en.wikipedia.org/wiki/Shebang_%28Unix%29

2.3 Installation and Execution

2.3.1 Windows

You may download the ZIP file from my homepage and extract its content. If you want to start c2x-cmd from a batch or another script type you have to add c2x-cmd to your search path.

You find the right option if you go under Windows 7 or Vista to:
Start → Control Panel → (System and Security) → System → Advanced System Settings → Tab „Advanced“ → Button „Environment Variables“

If you use Windows 2K or XP you have to go to:
Start → Control Panel → System → Tab „Advanced“ → Button „Environment Variables“

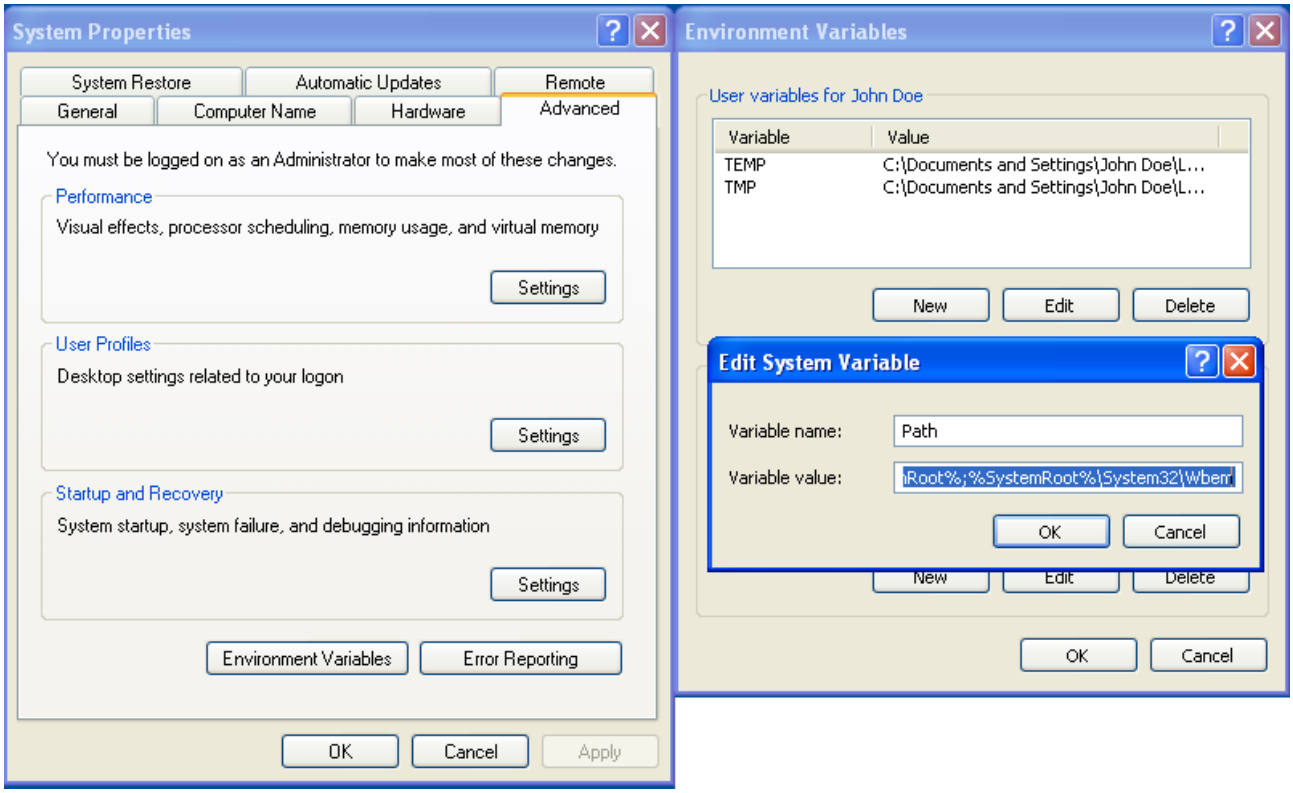

Figure 2: Editing the search path

Append a semicolon to the end of the existing search path and add the absolute path in which c2x-cmd has been extracted to.

Open a new command line in a directory in which c2x-cmd is **not** installed and enter:

```
c2x-cmd -V
```

If your former inserted path is added successfully to your search path c2x-cmd should print a version information:

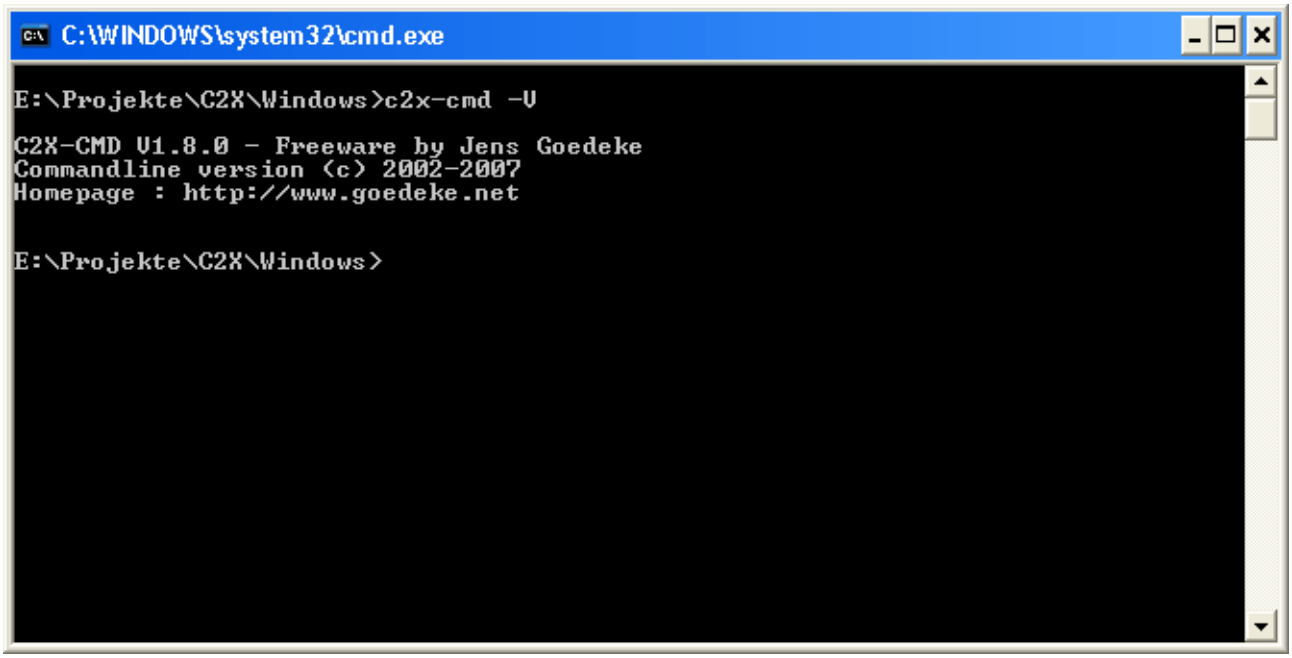

Figure 3: Successful result of a c2x-cmd version request

2.3.2 All operating systems based on Unix-/Linux (incl. MAC OS X)

You may download the tar.gz file from my homepage and extract its content. If you want to start c2x-cmd from a bash or another script type you'll have to add c2x-cmd to your search path.
If your system isn't mentioned on my homepage you might send me an email.

If you have root rights on your pc you may move c2x-cmd to one of your standard search path directories (e.g. „/usr/local/bin" etc.). The advantage of this action is that everyone on your system can use c2x-cmd without storing c2x-cmd in its own home directory.

Execution of c2x-cmd is possible over a Unix-/Linux-Shell (SH, KSH, BASH etc.).
If you run the following command in the directory where you have installed c2x-cmd

```
./c2x-cmd -V
```

you should see a version information.

Figure 4: Requesting c2x-cmd version information under Linux/Unix

3 c2x-cmd overview

c2x-cmd is a command line converter and doesn't have an graphical interface. Therefore c2x-cmd is perfect suitable for automated conversion of source files.

3.1 Generally annotations concerning the options

Options are lead in with a minus character. If the option is no simple switch the following parameter is separated by a colon (without any spaces).

3.2 Additive options

The options -alias and -styles are additive options. These options can be given in two different ways.

3.2.1 As parameter sequence

In this case each single given value is separated by a comma.
Example:

```
-alias:eRecord=My_Record,aValue=My_Value
```

3.2.2 As option sequence

In an option sequence each value is given as an attachment of a new option. This kind of notation is sensible if you use configuration files because it gives the chance to write comments to each given parameter. It can be used also in a command line call but it is wasting more space.

```
-alias:eRecord=My_Record -alias:aValue=My_Value
```

3.3 Overview of options

3.3.1 Options which are supported by c2x-wizard

These are the options which are supported by c2x-wizard (alphabetical) :

Option	Meaning	Mandatory	Example
-alias:	Renaming of Elements & Attributs	no	-alias:eItem=Test,eRecord=xx
-cc:i2u	Convert command line parameters from ISO-8859-1 to UTF-8	no	
-cc:u2i	Convert command line parameters from UTF-8 to ISO-8859-1	no	
-dtd:	Embed DTD validation information	no	-dtd:dtd-file
-enc:	Set encoding	no	-enc:ISO-8859-1
-m:	XML mode	no	-m:2
-s:	Select source file	yes	-s:source-file
-sep:	Set text separator(s)	no	-sep:,_-
-sep:::	Set Ascii separator	no	-sep::9:
-styles:	Embed stylesheet(s)	no	-styles:css-file,xsl-file
-sup	Suppress empty elements and attributes	no	
-t	Set target file name from source file name	yes	
-t:	Set target file name	yes	-t:target-file
-v	Show information while processing (Verbose)	no	
-wid	Write XML:ID into the target file	no	
-wn	Write record number into the target file	no	
-xmlns:	Set XML namespace	no	-xmlns:ns=uri
-xsd:	Embed XSD validation information	no	-xsd:xsd-file

3.3.2 Options that are not supported by c2x-wizard:

The following options are not supported by c2x-wizard:

Option	Bedeutung	Pflicht	Example
-V	Show version information	no	
-?	Show usage information	no	
-cfg:	Get options from configuration file.	no	-cfg:cfg-file
-exec	She-Bang execution	no	
-s:-	Get source data from STDIN	yes	
-t:-	Write target data to STDOUT	yes	

3.3.3 Configuration files

If you use many options they may exceed the character capabilities of your command line. To prevent this, c2x-cmd supports plain text configuration files. The format of a configuration file is:

```
[# Remark]|[Empty line]
Option 1|[Empty line]
[# Remark]|[Empty line]
[Option 2]|[Empty line]
...
```

You may use c2x-wizard to generate a configuration file for you. Otherwise you must ensure that you use a source code editor and not a text program like Word, Open-Office etc.). Here are some examples for Source code editors ...

for Windows:

```
Notepad, Notepad++, Ultraedit, ...
```

for Unix/Linux:

```
vi, emacs, gedit, kate, ...
```

Usage:

```
-cfg:Configuration_file
```

Example for a configuration file which has been created by c2x-wizard.
(The options are explained afterward)

```
#!c2x-cmd.exe -exec

# CSV source filename
-s:E:\Projekte\C2X\Windows\examples\source\example1.csv

# XML target filename must be optained from the csv source filename
-t

# Plain text separator(s)
-sep:;

# XML mode : <eRecord><COLname>CELLvalue</ColName></eRecord>
-m:1

# Encoding
```

```
-enc:ISO-8859-1

# Validation:
-xsd:E:\Projekte\C2X\Windows\examples\extend\mode3-schema.xsd

# Namespace
-xmlns:mode3-schema.xsd

# Aliases
-alias:eRoot=Wurzeldokument
-alias:eRecord=Datensatz

# Suppress empty elements:
-sup

# Write "num" attribute to record-element
-wn

# Write xml:id="id." attribute to record-element
-wid

# Verbose:
-v
```

3.4 She-Bang Execution

This function is only supported on Unix-/Linux- based operating systems.
With this OS specific function you have to write the name of the interpreter (which should process the current file) into the header of the file. You may know that from your (bash) shell scripts:

```
#!/bin/bash
```

After that you may execute the script without calling its interpreter directly.
The same functionality works with c2x-cmd. But you have to add the `-exec` option:

Usage:

```
#![Path to c2x-cmd]/c2x-cmd -exec
```

If you have generated the configuration file with c2x-wizard, c2x-wizard fills in the path in which c2x-cmd has been found within the search path. Before you can run the conversion you have to set the execute flag to your configuration file. After that you can test the conversion:

```
> chmod u+x ./myconfig.cfg
> ./myconfig.cfg
```

3.5 c2x-cmd options

The options of c2x-cmd were explained in detail in the following chapter.

3.5.1 Option : Source filename (m) / -s

This option sets the path to the source file or stream.

Usage:
This may be done in two different ways:
a) You define a file as the source:

```
-s:filename.csv
```

b) If the source data should be read from the standard input stream you have to use a minus character (instead of the file name):

```
-s:-
```

Example:

```
more sourcefile.csv | c2x-cmd -s:- -t:-
```

This example reads its data though a pipe from the more command an send its output to STDOUT.

3.5.2 Option : Target filename (m) / -t

This option sets the path to the target file or stream.

Usage:

This may be done in three different ways:

a) You define a file as the source:

```
-t:filename.xml
```

b) If the target data should be written to the standard output stream you have to use a minus character (instead of the file name):

```
-t:-
```

c) If the target file name should be optained from source file name

```
-t
```

Example:

```
c2x-cmd -s:my-file.csv -t
```

This generates a target file named „my-file.xml“

3.5.3 Option : Separators by plain text / -sep

With option you can set one or more separators. The default separator is a semicolon.

Usage:

```
-sep:CHAR-1[CHAR-2]...[CHAR-N]
```

Example 1:

```
-sep:,
```

This sets a comma as separator

Example 2:

```
-sep:,=
```

This sets a comma and a equal sign as separators

3.5.4 Option : Separator by ASCII code / -sep

If you want to set a single separator which could not be given in plain text format you can also define the separator by its ASCII code. For example : If your CSV file uses a tabulator as separator the correct ASCII value is „9“. The value of a space character is 32 and so on. ASCII separators have to be embedded between to colons.

Usage:

```
-sep::ASCII-VALUE:
```

Under this link you may have a look at a ASCII map:
http://en.wikipedia.org/wiki/Ascii#ASCII_control_characters

Example 1:

```
-sep::9:
```
This sets a tabulator as the separator.

Example 2:
```
-sep::32:
```
This sets a space as the separator.

3.5.5 Option : Encoding / -enc
With the encoding option you may define the encoding of the target file.
Two values are valid:

- ISO-8859-1
  This encoding uses one byte for each character and matches mostly with the Windows „1252“ character set.
- UTF-8
  This is the default Linux character set and it uses one or two bytes for each character.

Most applications that can handle UTF-8 files write a so called BOM (Byte-Order-Marker) into the header of the file (the first three bytes). c2x-cmd can detect this BOM. In case of an differing encoding c2x-cmd does automatic conversion to the selected encoding so that the target file complains to the selected encoding.
Usage:
```
-enc:ISO-8859-1|UTF-8
```

Example 1:
```
-enc:ISO-8859-1
```
Target file encoding is set to ISO-8859-1.

Example 2:
```
-enc:UTF-8
```
Target file encoding is set to UTF-8.

3.5.6 Option : Convert command line parameters / -cc
If you want to convert the command line parameters to a different encoding you can use this options. For example: If you have selected the UTF-8 encoding under Windows and your parameters include special characters you should use this option.
Usage:
Command line conversion is supported for two directions:
```
-cc:u2i
```
This is a conversion from UTF-8 to ISO-8859-1 and
```
-cc:i2u
```
is a conversion from ISO-8859-1 to UTF-8.

3.5.7 Option : XML Mode / -m
C2x-cmd supports 5 different xml modes. For a better explanation all examples use the following CSV source file content:
```
Name;Type;Year-1987;Year-1995;Year-2001;Year-2006
Berlin;Cnty;BE;3.260.000;3.471.418;3.388.434;3.404.037
```
The conversion is always processed with default options.

Hint:

The id's with a leading small character (e.g. *eRecord*) indicate that this id's can be renamed. Their default values are shown in square brackets afterward. Example: *eRecord [record]*. You may refer chapter 3.5.14 for more details.

3.5.7.1 Mode 1 : <*eRecord*><COLname>CELLvalue</ColName></*eRecord*>

Hint:
This mode was former named „Elements with Element Content (default)".

Usage:

```
-m:1
```

Description:
Within each record exists one element per parameter with its parameter id and its parameter value.

```
    <record>
        <Name>Berlin</Name>
        <Type>Cnty</Type>
        <Year-1987>BE</Year-1987>
        <Year-1995>3.260.000</Year-1995>
        <Year-2001>3.471.418</Year-2001>
        <Year-2006>3.388.434</Year-2006>
    </record>
```

3.5.7.2 Mode 2 : <*eRecord*><COLname *aValue*="CELLvalue"/></*eRecord*>

Hint:
This mode was former named „Elements with Attributes Values".

Usage:

```
-m:2
```

Description:
Within each record exists one element per parameter with its parameter id and one attribute which is named *aValue [value]*. The value of the attribute is the parameter value.

```
    <record>
        <Name value="Berlin" />
        <Type value="Cnty" />
        <Year-1987 value="BE" />
        <Year-1995 value="3.260.000" />
        <Year-2001 value="3.471.418" />
        <Year-2006 value="3.388.434" />
    </record>
```

3.5.7.3 Mode 3 : <*eRecord*><*eItem aName*="COLname">CELLvalue</*eItem*></*eRecord*>

Hint:
This mode was former named „Attributes with Element Content".

Usage:

```
-m:3
```

Description:
Within each record exists one element with the parameter name *eItem [item]* and an attribute which is named *aName [name]*. The content of the element is the parameter value.

```
<record>
      <item name="Name">Berlin</item>
      <item name="Type">Cnty</item>
      <item name="Year-1987">BE</item>
      <item name="Year-1995">3.260.000</item>
      <item name="Year-2001">3.471.418</item>
      <item name="Year-2006">3.388.434</item>
 </record>
```

3.5.7.4 Mode 4 : <*eRecord*><*eItem aName*="COLname" *aValue*="CELLvalue"/></*eRecord*>

Hint:
This mode was former named „Attributes with Attribute Values“.

Usage:

```
-m:4
```

Description:
Within each record exists one element an element which is named *eItem [item]* and an attribute which is named *aName [name]*. The value of the parameter is stored in a second attribute which is named *aValue [value]*.

```
<record>
      <item name="Name" value="Berlin" />
      <item name="Type" value="Cnty" />
      <item name="Year-1987" value="BE" />
      <item name="Year-1995" value="3.260.000" />
      <item name="Year-2001" value="3.471.418" />
      <item name="Year-2006" value="3.388.434" />
</record>
```

3.5.7.5 Mode 5 : <*eRecord* COLname="CELLvalue"/>

Hint:
This mode was former named „Attributes with Values“.

Usage:

```
-m:5
```

Description:
This is the most space saving mode. Within each record exists one attribute per parameter name. The value of the parameter is stored in this attribute.

```
<record Name="Berlin" Type="Cnty" Year-1987="BE" Year-1995="3.260.000"
Year-2001="3.471.418" Year-2006="3.388.434" />
```

3.5.8 Option : suppress empty elements / -sup

Is the value of an element or an attribute empty c2x-cmd is able to suppress the output of this element or attribute.
Source data (the values of „Type“ & „Year-1987“ are empty):

```
Name;Type;Year-1987;Year-1995;Year-2001;Year-2006
Berlin;;;3.260.000;3.471.418;3.388.434;3.404.037
```

Usage:

```
-sup
```

Result : Section of the target file in xml mode 5 with deactivated option:

```
<record Name="Berlin" Type="" Year-1987="" Year-1995="3.260.000" Year-
2001="3.471.418" Year-2006="3.388.434" />
```

Result : Section of the target file in xml mode 5 with activated option:

```
<record Name="Berlin" Year-1995="3.260.000" Year-2001="3.471.418" Year-
2006="3.388.434" />
```

3.5.9 Option : write „num“ attribute to record-element / -wn

This option writes the current record number as an attribute into the record. The numeration starts with „1“.

Usage:

```
-wn
```

Result : Section with deactivated option:

```
<record>
...
</record>
```

Result : Section with activated option:

```
<record num="1">
...
</record>
```

3.5.10 Option : write xml:id=“id." attribute to record-element / -wid

This option writes the current record number as an XML-ID attribute into the record. The numeration starts with „1“.

Usage:

```
-wid
```

Result : Section with deactivated option:

```
<record>
...
</record>
```

Result : Section with activated option:

```
<record xml:id="id.1">
...
</record>
```

3.5.11 Option : Stylesheet(s) / -styles

This option enables stylesheet embedding into the header of the target file. This does not proceed any conversion. If you want to transform the target file you need a xslt processor or something similar.

Two kind of stylesheet classes can be embedded:

- CSS - Cascading Stylesheets
- XSL – XML Stylesheets

While CSS files are useful for a visual improvement XSL(T) stylesheets are used to transform the regarding file into an other format.

More information on CSS files: http://en.wikipedia.org/wiki/Cascading_Style_Sheets
More information on XSL files: http://en.wikipedia.org/wiki/Extensible_Stylesheet_Language

Usage:

```
-styles:STYLESHEET
-styles:STYLESHEET,STYLESHEET,...
-styles:STYLESHEET,[STYLESHEET],...
```

This is an additive option (see 3.2). The different stylesheets could be given divided by commas or by new option calls. Is the stylesheet name encapsulated within square brackets the value of the xml property “Alternative” is set to “yes”.

Example 1:

```
-styles:.\extend\mode5-stylesheet.css
```

Result:

```
<?xml-stylesheet type="text/css" href="./extend/mode5-stylesheet.css"
title="mode5-stylesheet" ?>
```

Example 2:

```
-styles:.\extend\mode5-stylesheet.css,[.\extend\mode1-stylesheet.xsl]
```

Result:

```
<?xml-stylesheet type="text/css" href=".\extend\mode5-stylesheet.css"
title="mode5-stylesheet" ?>
<?xml-stylesheet type="text/xsl" href=".\extend\mode1-stylesheet.xsl"
alternate="yes"  title="mode1-stylesheet" ?>
```

3.5.12 Option : Validation / -xsd & -dtd

This option enables embedding a Document-Type-Definition (DTD) or a XML-Schema-Definition (XSD) reference. By choosing XSD validation its additionally possible to define a namespace (targetNamespace).

Usage:
Embedding of a XSD reference:

```
-xsd:xsd-filename.xsd [-xmlns:targetNamespace]
```

Embedding of a DTD reference:

```
-dtd:dtd-filename.dtd
```

Example:

```
-xsd:.\extend\mode3-schema.xsd -xmlns:mode3-schema.xsd
```

Result:

```
...
<csv_data_records xsi:schemaLocation="mode3-schema.xsd .\extend\mode3-
schema.xsd"
                  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xmlns="mode3-schema.xsd"
...>
```

3.5.13 Option : Use Namespace / -xmlns

This option enables setting a namespace and an optional prefix.

Usage:

```
-xmlns:[PREFIX=]URI
```

Example 1:

```
-xmlns:MyURI
```

Result:

```
...
<csv_data_records xmlns="MyURI"
...
```

Example 2:

```
-xmlns:MyPrefix=MyURI
```

Result:

```
...
<MyPrefix:csv_data_records xmlns:MyPrefix="MyURI">
     <MyPrefix:record>
          <MyPrefix:Name>Aachen</MyPrefix:Name>
...
```

3.5.14 Option : Alias / -alias

With this option its possible to define aliases for element and/or attribute names. If the defined alias is not xml compliant c2x-cmd corrects this automatically.

Usage:

```
-alias:Name=Alias[,Name2=Alias2,...]
```

The following table lists all changeable parameters (Name) and their default values (Default).

To change the label of a parameter enter your prefered name in the "Alias" field.

Name	Type	Default	Alias
eRoot	element	csv_data_records	WurzelElement
eRecord	element	record	Datensatz
eItem	element	item	
aName	attribute	name	
aValue	attribute	value	
aSource	attribute	source	Quelle
aDT	attribute	datetime	Zeitstempel
aID	attribute	xml:id	
aNum	attribute	num	

Example:

```
-alias:eRoot=WurzelElement,eRecord=Datensatz,aSource=Quelle,aDT=Zeitstempel
```

Result:

```
...
<WurzelElement Quelle="example1.csv" Zeitstempel="2007-12-16T19:54:22+01:00" >
     <Datensatz>
      ...
```

3.5.15 Option : Version information / -V

This option prints the version information.

Usage:

```
-V
```

Result:

```
C2X-CMD V1.8.0 - Freeware by Jens Goedeke
Commandline version (c) 2002-2007
Homepage : http://www.goedeke.net
```

3.5.16 Option : Usage / -?

This option prints the usage information.

Usage:

```
-?
```

Result:

```
Usage:
c2x-cmd    -s:FILE [-t[:FILE]] [-v] [-wn] [-wid] [-sep:SEPARATORS|-sep::#:]
           [-m:#] [-sup] [-e:ENCODING] [-cc:i2u|-cc:u2i] [-xmlns:[NSPFX=]NSURI]
           [-dtd:FILE|-xsd:FILE] [-styles:FILE[,...]] [-alias:TOKEN=NAME[,...]]
           [-cfg:FILE] [-V] [-?]
...
```

3.6 More examples

If you have extracted the c2x-cmd package you could access a sub directory in which you will find several examples. This examples could be started by running the file „example.bat“ or „example.sh“. All source files a stored in the directory called „source” , additional files are stored in „extend“ and the target files are written to the directory „results“.

4 Appendix

4.1 References

Link	URL
c2x-wizard manual	http://www.goedeke.net/c2x-cmd/c2x-wizard-manual.pdf
c2x-cmd manual	http://www.goedeke.net/c2x-cmd/c2x-cmd-manual.pdf