Molconvert

Molconverter is a command line program in Marvin Suite and JChem that converts between various file types.

Syntax

```
molconvert [options] outformat[:
exportoptions] [files...]
```

The outformat stands for one of the supported formats.

Format type	Outformat

Document formats
 	
MRV ChemAxon Marvin Document
 	
ChemDraw sketch file (CDX)
 	
ISIS/Draw sketch file (SKC)
	
	`mrv`
	`cdx`
	`skc`

Molecule file formats
 	
MDL MOL
 	
RG file
 	
Rxn file
 	
SD file
 	
RD file
 	
ChemAxon Compressed Molfile
	
	`mol`
	`rgf`
	`rxn`
	`sdf`
	`rdf`
	`csmol`
	`csrxn`
	`cssdf`
	`csrdf`
	`smarts`

Chemaxon Compressed Rxn file	`cxsmarts`
ChemAxon Compressed SDfile	`smiles`
ChemAxon Compressed RDfile	`cxsmiles`
SMARTS	`abbrevgroup`
ChemAxon Extended SMARTS	`peptide:1`
SMILES	`peptide:3`
ChemAxon Extended SMILES	`sybyl`
ChemAxon SMILES Abbreviated Groups	`mol2`
1-letter Peptide Sequence	`pdb`
3-letter Peptide Sequence	`xyz`
Tripos SYBYL molfile	`inchi`
Tripos Mol2	`inchikey`
Protein Data Bank	`name`
MSC XYZ	`csv`
IUPAC InChI	`cube`
IUPAC InChIKey	`gjf`
IUPAC Name	
CSV	
Gaussian cube	
Gaussian input output	

Graphics formats
jpeg
msbmp
png
pov
svg
emf
tiff
eps
	`jpeg`
	`msbmp`
	`png`
	`pov`
	`svg`
	`emf`
	`tiff`
	`eps`

Compression and Encoding gzip Base64	`gzip` `base64`

Alternatively, use

```
molconvert [options] query-encoding
[files...]
```

to query the automatically detected encodings of the specified molecule files.

From files having doc, docx, ppt, pptx, xls, xls, odt, pdf, xml, html or txt format, Molconvert is able to recognize the name of compounds and convert it to any of the above mentioned output formats.

Options

Molconvert options can be specified in the format string. The format descriptor and the options are separated by a colon, the options by commas.

```
molconvert jpeg:w100,Q95,#ffff00 nice.mol -o nice.jpg
```

(creates 100x100 JPEG image on yellow background, with 95% quality)

`-o` file	Write output to specified file instead of standard output

`-m`	Produce multiple output files
`-e` charset	Set the input character encoding. The encoding must be supported by Java.
`-e` [in `]..[` out]	Set the input (in) and/or output (out) character encodings. Examples: UTF-8, ASCII, Cp1250 (Windows Eastern European), Cp1252 (Windows Latin 1), ms932 (Windows Japanese).
`-s` string	Read molecule from specified SMILES, SMARTS or peptide string (try to recognize its format)
`-s` string `{` format `:` options `}`	Read molecule from the string in the specified format (can be omitted), using the specified importoptions (can be omitted)
`-f` string	Specify the import format and options
`--peptide` string	Read molecule from specified peptide string
`-g`	Continue with next molecule on error (default: exit on error)
`-Y`	Remove explicit H atoms
`-I` <range>	process input molecules with molecule index (1-based) falling into the specified range (e.g. 5-8,15 refers to molecules 5,6,7,8,15)
`-U`	fuse input molecules and output the union
`-R` <file>[: <range>]	fuse fragments to input molecule(s) from file with specified mol index range range syntax: "-5,10-20,25,26,38-" (e.g. -R frags.mrv:20-)
`-R<i>` <file> [:<range>]	fuse R<i> definition members to input molecule(s) from file in specified index range (e.g. -R1 rdef1.mrv:5-8,19)
`-R<i>:` `<1|2>` <file> [:<range>]	fuse R<i> definition members to input molecule(s) from file in specified index range, filter molecules having 1 (2, resp.) attachment points (e.g. -R1:2 rdef1.mrv:-3,8-10)
`-F`	Remove small fragments, keep the largest
`-c"`f1 OP value&f2 OP value..."	Filtering by the values of fields in the case of SDF import. OP may be: =,<,>,<=,>=
`--mol-fields-to-records`	Convert molecule type fields to separate records.

`-v`	Verbose
`-vv`	Very verbose (print stack trace at error)
`-2 [:` options] [`:` F<i1><i2>..., <iN>]	Calculate 2D coordinates Options for coordinate calculation. Performs partial clean with fixed atom coordinates for atoms <i1><i2>..., <iN> (1-based indexes) if the Fparameter is specified.
`-3 [:` options]	Calculate 3D coordinates Options for coordinate calculation.
`-H3D`	Help on options for 3D calculations. Detailed list on Clean 3d Options

Import options can be specified between braces, in one of the following forms:

filename{options}	
filename{MULTISET,options}	to merge molecules into one that contains multiple atom sets
filename{format:}	to skip automatic format recognition
filename{format:options}	
filename{format:MULTISET, options}	

You can also pass options to JAVA VM when you run the application from command line.

Options for file formats:

`a, +a, +a_gen`	General aromatization	`mrv:a`
`a_bas`	Basic aromatization	`mrv:a_bas`
`a_loose`	Loose aromatization	`mrv:a_loose`
`a_ambig`	Ambiguous aromatization	`mrv:a_ambig`
`-a, -a_gen`	General Dearomatization	`mrv:-a`

`-a_huckel`	Huckel dearomatization	`mrv:-a_huckel`
`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure	`mrv:-a_huckel_ex`
`H, +H`	Add explicit Hydrogen atoms	`mrv:H`
`-H`	Remove explicit Hydrogen atoms	`mrv:-H`
`+numbering`	assigns atom numberings corresponding to the IUPAC name	`mrv:+numbering`

`a, +a, +a_gen`	General aromatization	`cdx:a`
`a_bas`	Basic aromatization	`cdx:a_bas`
`a_loose`	Loose aromatization	`cdx:a_loose`
`a_ambig`	Ambiguous aromatization	`cdx:a_ambig`
`-a, -a_gen`	General Dearomatization	`cdx:-a`
`-a_huckel`	Huckel dearomatization	`cdx:-a_huckel`
`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure	`cdx:-a_huckel_ex`
`H, +H`	Add explicit Hydrogen atoms	`cdx:H`
`-H`	Remove explicit Hydrogen atoms	`cdx:-H`
`+numbering`	assigns atom numberings corresponding to the IUPAC name	`cdx:+numbering`

`a, +a, +a_gen`	General aromatization	`skc:a`
`a_bas`	Basic aromatization	`skc:a_bas`

`a_loose`	Loose aromatization	`skc:a_loose`
`a_ambig`	Ambiguous aromatization	`skc:a_ambig`
`-a, -a_gen`	General Dearomatization	`skc:-a`
`-a_huckel`	Huckel dearomatization	`skc:-a_huckel`
`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure	`skc:-a_huckel_ex`
`H, +H`	Add explicit Hydrogen atoms	`skc:H`
`-H`	Remove explicit Hydrogen atoms	`skc:-H`
`+numbering`	assigns atom numberings corresponding to the IUPAC name	`skc:+numbering`

Export options

The argument of MolConverter, MolExporter and the `getMol`/`getM` functions (of the applets and beans) is the format string. The format specification ("cml") is followed by ":" and the selected option(s) for CML export.

CodeName	Explanation
`a, +a, +a_gen`	General aromatization.
`a_bas`	Basic aromatization.
`a_loose`	Loose aromatization.
`a_ambig`	Ambiguous aromatization.
`-a, -a_gen`	General Dearomatization.
`-a_huckel`	Huckel dearomatization.
`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure.

`H, +H`	Add explicit Hydrogen atoms.
`-H`	Remove explicit Hydrogen atoms.
`A`	Atom attributes are stored in arrays. For 2D molecules, only the x, y coordinates are stored. This is a more compact form of storage than the default (using `<atom>` tags).
`P`	Create human readable output: put new XML elements in new lines and indent for embedded elements.
`CN`	The accuracy of the exported coordinates can be given: *N* is the length of the decimals of the coordinate, 0 < *N* 9
`D`	This option is important if the molecule has parity information and has 0 dimension. By default during the export, a clean method is invoked on the structure and the generated coordinates and wedge information are exported into CML format but *NOT* the parity information. However, using this option coordinates and wedge information are not generated but parity information is exported. Attention: When a CML file containing parity information is imported to Marvin older than 5.8, the parity information will be displayed wrong!
`I`	Ignore unexportable molecule properties. Without this option the exporter will throw an exception when reach an unexportable property.
`BOM`	Write the UTF-8 *byte order mark* (BOM), if the given or the system's encoding is UTF-8.

For example: cml:A or cml:C5.

Import options

Codename	Explanation
`Xsg`	Expand all S-groups.
`Usg`	Ungroup all S-groups.
`Fsg`	Ungroup S-groups with 3 or more attachment points.

`bXXX`	Set the C-C bond length used in the molfile. The molecule file is supposed to store coordinates in 1.54Å/XXX units. Marvin uses Å units internally, thus coordinates are rescaled by factor 1.54/XXX at import if XXX is a nonzero number. If XXX = 0, then coordinates are not rescaled (default setting for 3D molecules if option 'b' is not used). If XXX = A, then coordinates are rescaled to transform the molfile's average C-C bond length to 1.54 Å (default setting for 2D molecules). Examples: "caffeine.mol{b0}" or "caffeine.mol{b1.54}" (bond lengths are in angstroms), "caffeine.mol{b0.825}" (bond lengths are in ISISDraw's units), "caffeine-V3.mol{bA}" (average bond length calculation, same as default).
`nomolp`	Read molecule type data fields (`$DTYPE $MFMT` and `$RFMT` in RDfiles) as strings instead of Molecule objects.
`skipMMRV`	Neglect ChemAxon/Marvin specific lines in the properties block. Such lines are in the following format: `M MRV` ... They should be skipped if the file is converted with non-ChemAxon software, which preserved them but made them invalid, e.g. by changing the total number of atoms and bonds.
`skipAtomValue`	Disables the import of "Atom values" from the given ctfile.
`z`	Carbon atoms with "Z" as alias string are converted to R-group attachment points.

Export options

Codename	Explanation	
`a, +a, +a_gen`	General aromatization.	`XXX:a`
`a_bas`	Basic aromatization.	`XXX:a_bas`
`a_loose`	Loose aromatization.	`XXX:a_loose`
`a_ambig`	Ambiguous aromatization.	`XXX:a_ambig`
`-a, -a_gen`	General Dearomatization.	`XXX:-a`
`-a_huckel`	Huckel dearomatization.	`XXX:-a_huckel`

`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure.	`XXX:-a_huckel_ex`
`H, +H`	Add explicit Hydrogen atoms.	`XXX:H`
`-H`	Remove explicit Hydrogen atoms.	`XXX:-H`
`+numbering`	assigns atom numberings corresponding to the IUPAC name	`XXX:+numbering`
`V2 or V3`	Force writing V2 or V3 (extended) molfiles. The default format is V2 for simple molecules, V3 if the number of atoms or bonds exceeds 999, in case of reactions with Rgroups or there is enhanced stereo in the molecule.	`mol:V3`
`strict`	When it is applied, the output strictly follows the CTFile format specification (i.e. names longer than 80 characters are omitted)	`mol:strict`
`P`	Write floating point numbers with maximum precision. Only meaningful for V3 molfiles.	`mol:V3P`
`bXXX`	Set C-C bond length. If XXX is nonzero, then the exported atom coordinates are rescaled by XXX / 1.54. If XXX = 0, then coordinates are not rescaled. Examples: "mol:b0" or "mol:b1.0" (bond lengths are in angstroms), "mol:b1.0a" (set bond length, aromatize). Default: 0.825 in V2 format for 2D molecules, 1.54 (Å units) (which means no rescaling) in any other case.	
`ec`	Convert to enhanced stereo representation, considering the chiral flag. Only meaningful with option V3. (Chiral centers are grouped into ABS or an AND stereo group, depending on the chiral flag. When the input molecule contained any enhanced stereo labels, the unlabeled stereo centers always will form a new AND group.)	`mol:V3ec`
`ea`	Convert to enhanced stereo representation, assuming absolute stereochemistry. Only meaningful with option V3. (Chiral centers are grouped into the ABS group. In case the input molecule already contains enhanced stereo labels, the behaviour is similar to the one described at option **ec** above.)	`mol:V3ea`

`cc`	Write CHIRAL flag if there are only ABS enhanced stereo labels in the molecule. Only meaningful with option V2.	`mol:V2cc`
`omitClean0D`	Omits the clean operation while exporting 0D molecules into ctfile format with V2 compatibility which is the default. This clean was introduced in 5.4 because the ctfile format cannot contain stereo information without coordinates.	`mol: omitClean0D`
`BOM`	Write the UTF-8 *byte order mark* (BOM), if the given or the system's encoding is UTF-8.	`mol:BOM`

Import options

`--smiles` string	Read molecule from specified SMILES string	
`--smarts` string	Read molecule from specified SMARTS string	
`f` {\f*FIELD1*,f*FIELD2*,...}	Import data fields from a multi-column file. The fields should be separated by tab character. The first column contains the SMILES/SMARTS strings, the second may contain the *molecule name* or the data field called *FIELD1*, the following columns contain the other fields.	`molconvert sdf "foo.smi{fname,fID}"` reads the smiles string, the name and the ID from the foo.smi file and converts it to SDF format.

d	Import with Daylight compatibility for query H. In daylight smarts, H is only considered as H atom when the atom expression has the syntax [<mass>H<charge><map>] (mass, charge and map are optional). Otherwise it is considered as query H count. Examples: [!H!#6] without d option is imported as an atom which is not H and not C. However with d option it is imported as an atom which has not one H attached, and which is not C. Use "H1" or "#1" or "#1A" instead of "H" to avoid ambiguous meaning of H. "H1" always means query H count. "#1" always means H atom, "#1A" means aliphatic H atom.	
c	Ignore fixing of double bond stereo information in small rings, also ignore fixing of aromatic bonds to aliphatic if necessary. Double bonds in small rings (ring size < 8) is imported automatically with CIS stereo information. If c options is set, the double bond stereo information is not changed to CIS during the import. By default the bond is aromatic between two aromatic atom. But this is not true e.g. in case of biphenyl where the bond connecting the two aromatic ring is single. If biphenyl is represented with the SMILES string: "c1ccc(cc1)c1ccccc1" then it is necessary to set the bond between the two rings to single. If the molecule is exported by ChemAxon tools, the single bond between two aromatic atom is always explicitly written to avoid any confusion, so fixing aromatic bonds to aliphatic can be avoided.	
Z	Import compressed smiles. The compressed format must be specified explicitly, as it is not recognized by the importer automatically.	

After importing SMILES, invoking of `MoleculeGraph.clearCashedInfo` method is recommended in order to remove cashed information which results increased molecule size.

Export options

Export options can be specified in the format string. The format descriptor and the options are separated by a colon.

Codename	Explanation	

`a, +a, +a_gen`	General aromatization.	
`a_bas`	Basic aromatization.	
`a_loose`	Loose aromatization.	
`a_ambig`	Ambiguous aromatization.	
`-a, -a_gen`	General Dearomatization.	
`-a_huckel`	Huckel dearomatization.	
`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure.	
`H, +H`	Add explicit Hydrogen atoms.	
`-H`	Remove explicit Hydrogen atoms.	

`0`	Do not include chirality (parity) and double bond stereo (cis/trans) information	`smiles:0`
		
		(not stereo)
		`smiles:a0`
		
		(aromatic, not stereo)

`q`	*Obsolete option.* Atom equivalences are checked by default using graph invariants at double bonds.	smiles -s "C/C=C(/C)C" results CC=C(C)C

`ri`	Smiles export rigorousness (*i* with the following values):
	
	• 1. Export the most information from the molecule to SMILES or SMARTS format. Don't check anything.
	• 5. Atoms, bonds and the molecule is checked for SMILES, SMARTS compatibility (*default*).
	• 7. In addition to the checks in case of value 5, double bonds in alternating single and double bond chain are checked for correct export.
		Let molecule.mrv file contain the molecule CC=CC=CC=CC where the two side double bonds are in TRANS configuration but the middle one has no CIS, TRANS information (crossed double bond, or double bond with wiggly bond).
		`molconvert smiles:r7 m.mrv`
		
		drops the exception: "Nonstereo double bond between active CIS TRANS stereo bonds. Not possible to export it correctly to SMILES"
		
		`molconvert smiles m.mrv`
		
		results C\C=C\C=C\C=C\C (which is incorrect in the sense that the middle bond became TRANS configuration).

`s`	Write query smarts. (don't write explicit H in bracket) (See query SMARTS for details.)	
`u`	Write unique smiles (considering chirality info also [2]). Note: Use this option if you want unique smiles export.	
`h`	Convert explicit H atoms to query hydrogen count.	

Tf1:f2:...	Export *f1*, *f2* ... SDF fields. The fields are separated by tab character. If '-' is given before the T option like '-Tf1:f2:...' then no header line is written. '*' character is used to export all fields (and name also) in the molecules. 'name' field is used to export molecule name (if no 'name' field in the molecule exists).	
t	Export terminal atom with single_or_aromatic bond.	Examples: instead of [#6]-c1ccccc1 export the molecule to [#6]c1ccccc1 instead of [#6]-[#6] export the molecule to [#6][#6]
n	Export molecule name (the first line of an MDL molfile).	
Z	Use compressed format, and compress the SMILES string. Note that the compressed format is not recognized by the import, so it should be specified explicitly.	
BOM	Write the UTF-8 *byte order mark* (BOM), if the given or the system's encoding is UTF-8.	

Import options

See also

Sequences - peptide, DNA, RNA

`--peptide <string>`	The **string** is a valid one or three letter sequence.	convert a one-letter sequence to a molfile: `molconvert --peptide FFKMLL mol -o peptide.mol`

Export options

`peptide:3`	three-letter sequence	• convert SMILES representation to a three-letter sequence
		`molconvert peptide:3 -s "C[C@H](N)C(O)=O"`
		• convert one-letter sequence to a three-letter sequence
		`molconvert --peptide GAG peptide:3`

`peptide:1`	one-letter sequence	convert the SMILES string to a one-letter sequence `molconvert peptide:1 -s "C[C@H](N)C(O)=O"`

Import Options

`H or +H`	Add explicit hydrogen atoms.	`PDB:H`
`-H`	Remove explicit hydrogen atoms.	`PDB:-H`
`c`	Omit CONECT records for hetero compounds. Bonds are detected by the PDB reader modul based on local geometry unless the b option is specified.	`pdb:c`

See also

Protein Data Bank (PDB) file format

b	Do not recognize bond order. All bonds either defined by CONECT records or generated by PDB import are represented as ANY bonds.	pdb:b

Export Options

H or +H	Add explicit hydrogen atoms.	PDB:H
-H	Remove explicit Hydrogen atoms.	PDB:-H

Export options

Codename	Explanation
H, +H	Add explicit Hydrogen atoms.
-H	Remove explicit Hydrogen atoms.
Srel	Force relative stereo.
SAbs	Force absolute stereo
NEWPS	Narrow end of wedge points to stereocenter (default: both)
RecMet	Include reconnected metals results
FixedH	Mobile H Perception Off (Default: On)
AuxNone	Omit auxiliary information (default: Include)
NoADP	Disable Aggressive Deprotonation (for testing only)
Compress	Compressed output
DoNotAddH	Don't add H according to usual valences: all H are explicit

`Key`	Exports the InChIKey as well
`Woff`	Do not display warnings

Export options

Codename	Explanation
`H, +H`	Add explicit Hydrogen atoms.
`-H`	Remove explicit Hydrogen atoms.
`Srel`	Force relative stereo.
`SAbs`	Force absolute stereo
`NEWPS`	Narrow end of wedge points to stereocenter (default: both)
`RecMet`	Include reconnected metals results
`FixedH`	Mobile H Perception Off (Default: On)
`AuxNone`	Omit auxiliary information (default: Include)
`NoADP`	Disable Aggressive Deprotonation (for testing only)
`Compress`	Compressed output
`DoNotAddH`	Don't add H according to usual valences: all H are explicit
`Key`	Exports the InChIKey as well
`Woff`	Do not display warnings

Import options

Codename	Explanation

ocr	converts names containing OCR (optical character recognition) error. *Example*: convert the defective name "3-rnethyl-l-methoxynaphthalene" to SMILES `molconvert 'smiles:T*' -s '3-rnethyl-l-methoxynaphthalene' -f name:ocr`
-systematic	disable conversion of systematic names
-common	disable conversion of common names (such as *aspirin*)
-elements	disable conversion of the name of chemical elements, for instance carbon, sodium, ... Even though "carbon" is not converted, "methane" still is, since it is a molecule name for CH_4, not an element.
-ions	disable conversion of atomic ion syntax, for instance "Ca2+".
-groups	disable conversion of groups and fragments, such as "oxo" or "methyl".
-cas	disable the conversion of CAS registry numbers
-casNames	disable the conversion of CAS names
nameField=FIELD	sets the field/property that stores the original name. By default, the molecule title is used.
dict=*PATH*	specify the location of the custom dictionary. Example: `name:dict=C:\Users\Me\MyDictionary.smi .`
webservice=*URL*	enable the usage of a custom webservice at the given URL

Some of these options are mainly useful when configuring which names Document to Structure recognizes.

To enable an option, a + sign can be used before the option name. For instance, both forms **ocr** and **+ocr** are accepted to enable this option.

Export options

Codename	Explanation

`t`	converts to traditional name. This option generates a common name if one is known for the structure. Otherwise, it generates a systematic name, but sometimes uses some traditionally accepted rules instead of the strict application of the IUPAC rules, when that generates a simpler name.
`i`	converts to IUPAC name (default).
`common`	generates the most popular common name of a structure. (It fails if none is known.)
`common,all`	generates all common names of a structure.
`source`	outputs the name present in the source data (no generation).
`CAS#`	Fetches the CAS Registry Number® of the structure. This option uses a public internet webservice, so do not use it with confidential structures. ChemAxon assumes no liability whatsoever. Read the notice about CAS Registry Numbers® for more information. In case the structure corresponds to several CAS Registry Numbers, they are all returned, separated by commas.
`singleCAS#`	Fetches the CAS Registry Number® of the structure. This option uses a public internet webservice, so do not use it with confidential structures. ChemAxon assumes no liability whatsoever. Read the notice about CAS Registry Numbers® for more information. In case the structure corresponds to several CAS Registry Numbers, the lowest one (historically assigned earlier) is returned.
`ascii`	Use only ASCII characters to encode the name.
`timeout=<N>`	Use at most N seconds for the name to be computed. The default timeout is currently 20 seconds (which should normally only be reached exceptionally, for very large structures). A value of 0 means no timeout.

code:**csv**

- Define Molecule column name:
- Define headless export
- Define export format
- Define exported column header names

Basic information about the format

CSV stands for "coma separated value" and it is very simple molecule format.

```
id,mol,registeting_user,note
1,C,anonymous@chemicalize.com,this is a rather common element
2,[H],h.canvenids@chemicalize.com,"I bet this is more common, how could you miss it?"
3,[He],pjc_janssen@chemicalize.com,This is boring il ne reagit pas avec quoi que ce soit!
```

In this file we have 3 molecules, and every of them has the following information:

- ID
- registering_user
- note

The molecule sources are in smiles. After import we get the following structures and properties:

- A simple Carbon, with:
  - ID = 1
  - registering_user = anonymous@chemicalize.com
  - note = this is a rather common element
- A simple Hydrogen, with:
  - ID = 2
  - registering_user = h.canvenids@chemicalize.com
  - note = I bet this is more common, how could you miss it?
- A simple Helium, with:
  - ID = 3
  - registering_user = pjc_janssen@chemicalize.com
  - note = This is boring il ne reagit pas avec quoi que ce soit!

But the user can specify molecule during import which header to use. For example this file:

```
id,CHEMICAL_DATA,name
1,c1ccccc1CC(N)C,amphetamin
2,c1ccccc1,benzene
```

Can be imported with the following settings:

```
csv:strucCHEMICAL_DATA
```

With this MolImporter recognise that CHEMICAL_DATA filed holds the structure.

Import options

Headers

Automatically recognized molecule headers

Molecule can have any ChemAxon supported formats, but they must be written in one line. The recognized molecule headers are:

- mol
- molecule
- structure
- struc
- smiles
- cxsmiles
- smarts
- cxsmarts
- inchi

User defined header

User can define which header to use as identifier of the molecule column when importing structure. This can be done with the "**struc**" parameter.

For example this file:

```
id,CHEMICAL_DATA,name
1,c1ccccc1CC(N)C,amphetamin
2,c1ccccc1,benzene
```

Can be imported with the following settings:

```
csv:strucCHEMICAL_DATA
```

With this MolImporter recognise that CHEMICAL_DATA filed holds the structure.

Headless import

User can import CSV molecules without header, in this case csv importer must be informed that all rows are data (for this use "**headless**" keyword), and the which colum has the chemical structure. This can be done by defining the zero-based index of the structure column. For example the following file

```
7,12,4,ccCCcc,rt,gh,jk
23,1,56,COO,rf,gg,kk
```

Can be imported as:

```
csv:headless,struc3
```

This would import the following structure:

- ccCCcc (as smiles) with the following properties:
  - column_0 = 7
  - column_1 = 12
  - column_2 = 4
  - column_3 = rt
  - column_4 = gh
  - column_5 = jk
- COO (as smiles) with the following properties:
  - column_0 = 23
  - column_1 = 1
  - column_2 = 56
  - column_3 = rf
  - column_4 = gg
  - column_5 = kk

Override column names

During import user can dynamically ovverride column names. For this he has to set the names in order. (Every definition starts with an "**f**" and spearated by coma".) For example this file:

```
result,hour
S.[He],11:15:00
[He],11:10:00
```

can be imported as:

- S.[He]
  - TIME = 11:15:00
- [He]
  - TIME = 11:10:00

With the following params:

```
csv:fMOL,fTIME
```

In the above example the renamed headers contained an autoreconizable header name, so we did not have to specifiy molecule colum. But this can be than as it is described in Header section with the "**struc**" keyword.?

Molecule format

User can specify what is the format of the molecules in the molecule comulmn with the "**input**" keyword. For example fro names use:

```
csv:inputname
```

Export options

Define Molecule column name:

User can set the name of the molecule column with "**struc**" keyword, like:

```
csv:strucMY_MOL_COLUMN
```

Define headless export

User can export molecules without headers with the "**headless**" keyword, like:

```
csv:headless
```

Define export format

User can define which format to use when export molecule with the "**format**" keyword, like:

```
csv:formatsmarts
```

Define exported column header names

It is possible to define the name of the exported clumns every name must start with an "**s**" like:

```
csv:sname,smol,suser
```

`a, +a, +a_gen`	General aromatization	`XXX:a`
`a_loose`	Loose aromatization	`XXX:a_loose`
`a_ambig`	Ambiguous aromatization	`XXX:a_ambig`
`-a, -a_gen`	General Dearomatization	`XXX:-a`
`-a_huckel`	Huckel dearomatization	`XXX:-a_huckel`
`-a_huckel_ex`	Huckel dearomatization, throwing exception in case of failure	`XXX:-a_huckel_ex`
`H, +H`	Add explicit Hydrogen atoms	`XXX:H`
`-H`	Remove explicit Hydrogen atoms	`XXX:-H`
`+numbering`	assigns atom numberings corresponding to the IUPAC name	`XXX:+numbering`
`H_off`	Do not show implicit Hydrogen labels.	`XXX:H_off`
`H_hetero`	Implicit Hydrogen labels on heteroatoms only.	`XXX:H_hetero`

`H_heteroterm`	Implicit Hydrogen labels on hetero- and terminal atoms (default).	`XXX: H_heteroterm`
`H_all`	Implicit Hydrogen labels on all atoms.	`XXX:H_all`
`chiral_off`	Switch off chirality support, do not show R/S labels (default).	`XXX:chiral_off`
`chiral_selected`	Show R/S if the chiral flag is set for the molecule.	`XXX: chiral_selected`
`chiral_all`	Show R/S for any molecule.	`XXX:chiral_all`
`MP_LABEL_VISIBLE`	Show M/P for any molecule.	`XXX:mp`
`noRGroups`	Do not show R-groups.	`XXX:noRgroups`
`noRLogic`	Do not show R-logic.	`XXX:noRLogic`

`w...` `h...`	Image width and height in pixels. If only one from *w* and *h* is specified, then the other will have the same value. If none of them is specified, then their values are calculated from scale. If scale is not specified, then the default size is 200x200.	`XXX:w200,h200`
`scale...`	Magnification. 1.54Å (C-C bond length) is *scale* pixels.	
`maxscale...`	Maximizes the magnification to prevent overscaling of small molecules. It is usually set to 28, which is the scale factor for 100% magnification.	

`atsiz...`	Atom label font size in C-C bond length units. Default: 0.4 Note: *atsiz**1.54 Å = *atsiz***scale*** points	

`atomFont...`	Atom label font type and size in pt.	`atomFont: SansSerif-ITALIC-10`
		
		`atomFont:Times New Roman-PLAIN-10`

`bondl...`	Bond length in pt. Default: 28	`bondl42.0`
`bondw...`	Bond spacing in C-C bond length units. Default: 0.18 Note: *bondw**1.54 Å = *bondw***scale*** pixels	
`boldbondw...`	Width of bold bond in pt. Default: 6	
`bondHashSpacing...`	The spacing of the hash in hashed bonds in C-C bond length units.	

`wireThickness...`	Bond thickness in wireframe mode. Default: 0.064	
`stickThickness...`	The stick diameter for ball and stick mode. Default: 0.1	
`ballRadius...`	Ball radius for ball and stick mode. Default: 0.5	
`#rrggbb`	Background color. It also determines the brightness of the CPK palette (for atoms and bonds); lighter colors are choosen automatically for dark background and conversely. Default: "#ffffff"	

`#aarrggbb`	Background color with alpha value. Use alpha=0 for transparent background, e.g. "#00ffffff". Note that the alpha channel is not supported by all image formats. Default: "#ffffffff"	
`transbg`	Sets the image background to transparent.	
`mono`	Black & white.	
`cpk`	Use CPK colors (default).	
`shapely`	Use the shapely color scheme.	
`group`	Use coloring based on residue sequence numbers.	

`setcolors:...`	Use atom/bond set colors. Colors can be specified as a colon separated list of values. Use "a***k***:#rrggbb" for atom set *k*, "b***k***:#rrggbb" for bond set *k*. The hashmark "#" can be omitted. Human-readable color names like "red", "green", "blue" can also be used.	
`wireframe`	Wireframe rendering style (default for 2D).	
`wireknobs`	Wireframe with knobs - used til version 17.9. Later versions fall back to `wireframe`	
`ballstick`	"Ball & stick" rendering style (default for 3D).	
`spacefill`	Spacefill rendering style.	
`noantialias`	Switch off antialiasing.	
`amap`	Displays atom mapping.	

`anum`	Displays atom numbers.	
`atomNumberingType...`	Sets the type of atom numbering. Implies anum parameter. Possible values: • 1 (Atom numbers) • 2 (IUPAC numbering)	
`lp`	Displays lone pairs.	
`lpexpl`	Display the explicit lone pairs instead of the implicit lone pairs if lone pair displaying is switched on. See the `lp` parameter.	
`lonePairsAsLine`	Display lone pairs as a line instead of the default two dots. This parameter has effect only if the `lp` parameter is also specified.	

`downwedge_mdl`	Down wedge orientation points downward (MDL). (default)	
`downwedge_daylight`	Down wedge orientation points upward (Daylight).	
`anybond_auto`	Draw any bonds with dashed lines in most cases. If all bonds are generated from atom coordinates, any bonds are displayed with solid lines. (default)	
`anybond_dashed`	Draw any bonds with dashed lines.	
`anybond_solid`	Draw any bond with solid lines.	
`noatsym`	Hide atom symbols in 3D mode.	
`valprop`	Show valence property on atoms that have the valence property explicitly set.	

`ez`	Show E/Z labels.	
`cv_on`	Always show the atom labels of carbon atoms.	
`cv_off`	Never show the atom labels of carbon atoms.	
`cv_inChain`	Show the atom labels of carbon atoms at straight angles and at implicit Hydrogens.	
`bondLengthVisible`	Display the length of bonds in Angstroms.	
`valenceErrorVisible`	Display valence errors.	
`absLabelVisible`	Set the Absolute label visibility to true.	
`ligandOrderVisibility_withDef`	Active by default. Show ligand order on images only when the R-group definition is present.	
`ligandOrderVisibility_on`	Show all ligand order on images for R-groups.	

`ligandOrderVisibility_off`	Never show ligand order on images for R-groups.	
`aprop`	Show explicitly set properties on atoms.	
`liganderr`	Show ligand errors on R-groups.	
`coordBondStyle_solid`	Display coordinate bond as a single bond.	
`coordBondStyle_arrow`	Display coordinate bond as an arrow.	
`coordBondStyleAtMulticenter_hashed`	Display coordinate bond as a dashed bond when it connects to a multicenter atom.	
`coordBondStyleAtMulticenter_solid`	Display coordinate bond as a single bond when it connects to a multicenter atom.	
`chargeWithCircle`	Display charge symbols in a circle.	

`oneLetterPeptideDisplay`	Display peptides with their one letter abbreviation instead of the three letter abbreviation which is the default.	
`disableAminoAcidBondColoring`	Disable the amino acid bond coloring.	
`fogFactor...`	Set the fog factor scale value (integer). Default value: 0, range: 0..100.	
`marginSize...`	Set the margin width in pt. Default: 10	

2D defaults: `H_heteroterm,w200,h200,#ffffffff,cpk,wireframe`

3D defaults: `H_heteroterm,w200,h200,#ff000000,cpk,ballstick`

Examples:

`jpeg`	Default settings: 200x200 pixels, white background (or black in 3D).
`jpeg:w100,#ffff00`	100x100 JPEG with yellow background.
`jpeg:w100,h150`	100x150 JPEG with default background.
`png:aprop -s "C1-C10 alkyl" -o alkyl.png`	PNG showing "C1-C10 alkyl".

Examples

1. Printing the SMILES string of a molecule in a molfile

```
molconvert smiles caffeine.mol
```

2. Dearomatizing an aromatic molecule:

```
molconvert smiles:-a -s "c1ccccc1"
```

3. Aromatizing a molecule:

```
molconvert smiles:a -s "C1=CC=CC=C1"
```

   (The default general aromatization is used.)
4. Aromatizing a molecule using the basic algorithm:

```
molconvert smiles:a_bas -s "CN1C=NC2=C1C(=O)N(C)C(=O)N2C"
```

5. Converting a SMILES file to MDL Molfile

```
molconvert mol caffeine.smiles -o caffeine.mol
```

6. Making an SDF from molfiles:

```
molconvert sdf *.mol -o molecules.sdf
```

7. Printing the encodings of SDfiles in the working directory:

```
molconvert query-encoding *.sdf
```

8. SMILES to Molfile with optimized 2D coordinate calculation, converting double bonds with unspecified cis/trans to "either"

```
molconvert -2:2e mol caffeine.smiles -o caffeine.mol
```

9. 2D coordinate calculation with optimization and fixed atom coordinates for atoms 1, 5, 6:

```
molconvert -2:2:F1,5,6 mol caffeine.mol
```

10. Import a file as XYZ, do not try to recognize the file format:

```
molconvert smiles "foo.xyz{xyz:}"
```

    Note: This is just an example. XYZ and other formats known by Marvin are always recognized (send us a bug report otherwise), so the specification of the input format is usually not needed. It is only relevant if a user-defined import module is used.
11. Import a file as XYZ, with bond-length cut-off = 1.4, and max. number of Carbon connections = 4, export to SMILES:

```
molconvert smiles "foo.xyz{f1.4C4}"
```

12. Import a file as Gzipped XYZ, with the same import options as in the previous example:

```
molconvert smiles "foo.xyz.gz{gzip:xyz:f1.4C4}"
```

13. Like the previous example but merge the molecules into one molecule that contains multiple atom sets. MDL molfile is exported.

```
molconvert mol "foo.xyz.gz{gzip:xyz:MULTISET,f1.4C4}"
```

14. Import an SDF and export a table containing selected molecules with columns: SMILES, ID, and logP:

```
molconvert smiles -c "ID<=1000&logP>=-2&logP<=4" -T ID:logP foo.sdf
```

15. Fuse R2 definition from file, filter fragments with 1 attachment point:

```
molconvert mrv in.mrv -R2:1 rdef.mrv
```

16. Fuse fragments from file (note, that the input molecule, which the fragments are fused to, should also be specified):

```
molconvert mrv in.mrv -R frags.mrv
```

17. Generate all common names for a structure:

```
molconvert "name:common,all" -s tylenol
```

18. Generate the most popular common name for a structure (It fails if none is known.):

```
molconvert name:common -s viagra
```

19. Generate SMILES from those molecules that names are mentioned in a file foo.html:

```
molconvert smiles foo.html
```