Maestro Command Reference Manual

Version 9.3, September 2012

For inquiries about Maestro ™ contact:

Schrödinger
101 SW Main Street, Suite 1300
Portland, OR 97204–3212
503–299–1150
503–299–4532 fax
Email: help@schrodinger.com

Maestro Version 93515
September 2012

1 The Maestro Command Language

This manual contains a listing of all supported Maestro commands, their functions, and their usage options. Use Maestro commands by entering them in the Command Input Area of the Main Application Window.

The Maestro command language syntax is:

keyword [*options*] [operand]

For example:

entryimport *wsreplace*=true *all*=true *format*=maestro ligands.mae

In the example given, "entryimport" is the keyword. As this example illustrates, the first item entered must be a keyword, and this keyword must match a known keyword completely, i.e., to all characters. However, Maestro does have an alias facility, which allows users to map keywords to shorter names, for example, "entryimport" to simply "import". See the "alias" command for details.

Command options are used to set values that generally correspond to states within the program. As the name "options" suggests, inclusion of these items in a Maestro command is not mandatory. However, any number of option *name*=⟨ value ⟩ pairs can be specified. Options can be of type: Boolean (yes/no, y/n, t/f, and true/false are valid values), string, integer, or real.

Once an option is set, its value will persist for all subsequent commands until it is specified again. A command consisting only of a keyword and options will set the states corresponding to those option values, but will not perform any other action. For example, the command:

entryimport *format*=maestro *all*=true

will not import structures from any file. It will only set the format for subsequent file reading and specify that all structures contained in the subsequently specified file be imported. If the following command is then issued:

entryimport ligands.mae

all structures from the ligands.mae Maestro file will be imported.

The operand is the remaining element of a Maestro command. The type of operand required is dependent on the specific command. For "entryimport", the operand must be a file name. For other commands, it may be one or more atom numbers, or a string describing arbitrary atom sets, specified using Maestro's Atom Specification Language (ASL). For more information about ASL, see Chapter 3 [ASL], page 5.

2 Conventions Used in this Manual

This manual uses a number of typographical conventions to describe the command language.

The names of keywords are printed in boldface:

entryimport

The names of options are printed in italic, and literal choices for that option are in normal roman:

wsreplace=true

Values which the user is to replace with appropriate values are represented in angle brackets:

entryimport ⟨ file_name ⟩

Where there is a choice of a number of values, these are shown separated by vertical bars:

entryimport *format*=maestro|mmod|sd|pdb|mol2

If a value is optional then it is enclosed in square brackets:

print [⟨ file_name ⟩]

The following conventions are used to describe acceptable values for options:

⟨ n ⟩	means an integer
⟨ x ⟩	means a real number
yes|no	means a boolean value (could also be true|false, y|n, on|off)
⟨ text ⟩	represents a string value. If this contains spaces, it must be enclosed in double quotation marks

The following are some conventions used for operands:

⟨ atom_number ⟩	represents an atom number
⟨ atom1 ⟩	represents the first atom number in the operations, ⟨ atom2 ⟩ the second, and so on
⟨ ASL ⟩	represents a valid string in the Atom Specification Language
⟨ ESL ⟩	represents a valid string in the Entry Specification Language

3 The Atom Specification Language

This chapter documents the syntax of Maestro's Atom Specification Language (ASL).

3.1 Why an Atom Specification Language?

- To provide a flexible way to define sets of atoms in complex macromolecular systems
- To serve as the basis of a "Sets" facility in Maestro
- To allow atom specification from textual input which some users find faster then picking atoms from the main structure window.

3.2 The ASL Hierarchy

`entry > molecule > chain > residue > atom`

There are five classes which make up the atom specification language. Each is listed below.

In this section the minimum acceptable abbreviation is shown outside the square brackets. So for example `m[olecule]` means that the minimal acceptable abbreviation for `molecule` is `m`.

`e[ntry]` This is the top level class in the language. An entry is all atoms in the workspace associated with a single entry in the currently open project.

`m[olecule]`
The term molecule is used in the normal chemical sense meaning all atoms which are connected by a single covalent path.

`c[hain]` This corresponds to a chain as specified in the PDB file format. Note that this chain may be a subset of a molecule, e.g., when chains a linked by disulphide linkages.

`r[esidue]`
An arbitrary collection of one or more covalently bound atoms within a molecule, such as the monomer units in a polymer.

`a[tom]` A single atom.

Each class is optional. If absent all entities of that type are matched.

3.3 Atom Specification

A complete specification is a class name and some property specified by a property name and property list. The syntax is:

class.property ⟨ propertylist ⟩

Items in a property list may be separated by comma, whitespace or both. Ranges (lower-upper) may be used where appropriate. Unterminated ranges are taken to include all available numbers. For example, if there are four molecules in the system then the specifications:

- `mol. 2, 3, 4`
- `mol. >=2`
- `mol. 2-4`
- `mol. >1`

are equivalent.

In a similar manner,

- `mol. 1, 2, 3`
- `mol. <=3`
- `mol. 1-3`
- `mol. <4`

are equivalent.

All names of properties and characters in property lists are treated in a case insensitive manner.

Wildcards are supported for atom and set names. A '`*`' will match zero or more characters and a '`?`' will match any single character. You can include comments in a specification by placing a '`#`' character before the text you wish to hide.

In this section the minimum acceptable abbreviation is shown outside the square brackets. So, for example, `a[tom].pt[ype]` means that the minimal acceptable abbreviation for `atom.ptype` is `a.pt`.

`e[ntry].`

`[name]` Because entry names are the only entry properties specifiable using ASL expressions, the word `name` can be completely omitted, i.e., `entry.name ename` and `entry. ename` are equivalent. Note, however, that the '`.`' is still required even if the property name is not included. A valid property list for the 'entry name' property is a list of entry names. Wildcard characters are permitted. For example:

```
entry. e1
entry.name recep, lig*
```

`m[olecule].`

`[number]` Because numbers are the only molecular properties specifiable using ASL expressions, the word `number` can be completely omitted, i.e., `mol.number 1` and `mol. 1` are equivalent. Note, however, that the '`.`' is still required even if the property name is not included. A valid property list for the 'molecule number' property is a set of numbers or a range. For example:

```
mol. 1-4
mol. 1,2,3,4
```

`m[odulo]` a property which can be used to select every *n*th molecule. For example: `mol.mod 10 1` will select molecule 1, 11, 21, etc.

`e[ntrynum]`
a property which can be used to select molecules based on their entry-relative numbering. For example: `mol.entrynum 1` will select the first molecule in each entry.

`a[toms]` a property which can be used to select molecules based on the number of atoms they contain. For example: `mol.atoms 200` will select molecules that contain exactly 300 atoms. Other examples are:

```
mol.atoms 200-300
mol.a > 200
```

`w[eight]` a property which can be used to select molecules based on their molecular weight. For example: `mol.weight 200.12` will select the atoms that have a molecular weight of exactly 200.12. Other examples are:

```
mol.weight <=300.0
mol.weight > 200.0
```

`c[hain]` This class designation allows you to specify atoms using chain attributes. Combine with the 'name' property.

`[name]` Because names are the only chain properties that are specifiable by ASL, the word `name` can be omitted. For example, the specifications `chain.name A` and `chain. A` are equivalent. Note, however, that the '`.`' is required even if the property name is not included. A valid property list for the 'chain name' property is a single character representing a PDB

chain name. Some examples of equivalent acceptable 'chain name' expressions are:

```
chain.name A
chain. A
c. A
```

`r[esidue]`

This ASL class designation allows you to specify atoms based on residue properties. Combine with one of the following property specifications.

`[name]` or `[number]`

Either residue names or numbers (but not both) can be used in property lists which do not specify a property name For example, the following are valid ASL expressions:

```
res. ala val leu
res. 1 2 3
```

and will return atoms which are either in alanine, valine, or leucine residues, or atoms in residues 1, 2 or 3, respectively. Residue numbers can be specified with a range, e.g., `residue. 1-4` and may include negative values or zero.

`pt[ype]` The three-letter PDB code for the residue. This is the default for non-numeric characters in the property list, so the expression `res. arg` and `res.ptype arg` are equivalent. A valid property list for 'ptype' is comprised of three-character tokens. For example:

```
res.ptype gly,val,ala
res. gly val ala
```

`m[type]` The one letter residue codes as used in Maestro. A valid property list for 'mtype' is comprised of one-character tokens. For example:

```
res.mtype g,v,a
res.m g,v,a
```

`po[larity]`

The polarity of the residue. The property list must consist only of the following descriptor types:

`h[ydrophobic]`
: returns atoms in hydrophobic residues

`pol[ar]` returns atoms in polar residues

`pos[itive]`
: returns atoms in residues with positive formal charges

`n[egative]`
: returns atoms in residues with negative formal charges

For example,

```
residue.polarity hydrophobic
residue.pol pos,neg
res.pol h pos neg
```

`sec[ondary_structure]`
: The secondary structure of the residue. The property list must consist only of the following descriptor types:

  `h[elix]` returns atoms in helical regions

  `s[trand]` returns atoms in strand regions

  `l[oop]` returns atoms in loop regions

  For example,

  ```
  residue.sec helix
  residue.sec hel, str
  res.sec l, s
  ```

`pos[ition]`
: The fractional position of the residue. The property list must include two real numbers representing a fractional range of residue numbers. For example, if there are 100 residues numbered from 1-100, the specification:

  ```
  residue.pos 0.0 0.1
  ```

  will return residues 1 to 10.

`i[nscode]`
: The insertion code of the residue. A property list should include one-character tokens representing insertion codes. For example:

  ```
  residue.inscode a
  ```

  will get all residues with insertion code 'a', while:

  ```
  res. 25 and res.inscode b
  ```

  will get residue 25b.

`a[tom]` Atom ptypes and numbers may be mixed in property lists where no explicit property is specified. For example, the following is valid:

```
atom. 1,2,3,CA
```

and returns atoms 1,2 and 3 and any alpha carbons.

`pt[ype]` The PDB atom names. A valid property list for 'ptype' consists of acceptable PDB names. This is the default property for non-numeric components of property lists and, as such, the word 'ptype' may be omitted. Note that the '`.`' is required. The following specifications are equivalent and return the "backbone" atoms in a structure:

```
atom.ptype N,CA,C,O
atom. N,CA,C,O
a. n,ca,c,o
```

Note: see below for a discussion of how PDB atom names are specified and matched.

Wildcards as described above can be applied to ptypes.

`na[me]` The atom names. The property list must contain valid atom names. A valid atom name could be a string of any length which:

1. must have at least one non-digit character
2. must not contain any control characters (ASCII value < 0x20)
3. must not contain spaces or equal signs, unless the name is quoted

Wildcards as described above can be applied to atom names. Examples:

`atom.name the_36th_carbon`
`atom.na C15, O:66, H-77`
`atom.na C*` (returns atoms with name starting with C)
`atom.nam ??O*` (returns atoms whose name's 3rd character is '`O`')

`n[umber]` The atom numbering. The property list must be a list or a range of numbers. This is the default property for numeric components of property lists. The following expressions are equivalent and return the atoms numbered 1, 2, 3, and 4.

```
atom.num 1,2,3,4
a. 1 2 3 4
atom. 1-4
```

`mo[lnum]` The atom numbering in the "by molecule" scheme. The property list must be a list or a range of numbers. For example, the expression:

```
atom.molnum 1
```

returns the first atom in each molecule, while the specification:

```
atom.molnum 1-10
```

returns the first 10 atoms in each molecule.

`en[trynum]`
The atom numbering in the "by entry" scheme. The property list must be a list or a range of numbers. For example, the expression:

```
atom.entrynum 1
```

returns the first atom in each entry, while the specification:

```
atom.entrynum 1-10
```

returns the first 10 atoms in each entry.

`m[type]` The Maestro atom type. A valid property list for the 'mtype' property consists of Maestro atom types. The following expression is valid and specifies sp2 carbons and oxygens.

```
atom.mtype C2,O2
```

`e[lement]`
The element symbol for the atom. A valid property list for the element is a list of standard periodic table symbols. To define all carbons and oxygens:

```
atom.ele C,O
```

`att[achments]`
The number of bonds the atom has to it. The property list must be a number in the range 0-6, but greater than '`>`', less than '`<`', and equals '`=`' signs may also be used. The expression:

```
atom.att 1
```

returns all terminal atoms. The specification:

```
atom.att <=2
```

returns all terminal atoms and other atoms with 2 or fewer bonds. The specification:

```
atom.att 0
```

returns all isolated atoms.

`ato[micnumber]`
: The atomic number. The property list must contain numbers only. Ranges of integers and greater than '`>`', less than '`>`', and equals '`=`' signs may also be used. The expression:

```
atom.atomicnum 1
```

returns all hydrogen atoms. The specification:

```
atom.ato 1-6
```

returns all atoms in the range H to C.

`c[harge]`
: The partial charge on the atom. A valid property list contains a value or range of floating point values. The expression:

```
atom.charge 0.400
```

returns atoms with partial charges of 0.400. The expression:

```
atom.charge -0.6--0.4
```

returns atoms with partial charges -0.6 to -0.4,

```
atom.charge <0.0
```

returns atoms with negative partial charges, and

```
atom.charge >=0.5
```

returns atoms with charges of 0.5 or greater.

`f[ormalcharge]`
: The formal charge on the atom. A valid property list contains a value or range of integer values. The expression:

```
atom.formal 0
```

returns atoms with formal charges of 0. The expression:

```
atom.formal -2 -- 1
```

returns atoms with formal charges -2 to -1,

```
atom.formal <0
```

returns atoms with negative formal charges, and

```
atom.formal >=1
```

returns atoms with formal charges of 1 or greater.

`d[isplayed]`
: Whether or not the atom is currently displayed in Maestro. No property list is used. For example, the expressions:

  `atom.displayed` returns the set of all displayed atoms
  `not atom.disp` returns all atoms not currently displayed
  `sidechain and fillres atom.disp` returns the side chains of residues where at least one atom is displayed.

`s[elected]`
: Whether or not the atom is currently selected in Maestro. No property list is used. For example, the expression:

  `atom.selected` returns the set of all atoms selected in the Workspace

Using generalized atom properties. Some structures may have additional properties available. These are referenced directly by their data names appended onto the "atom." class. These properties are either of integer, real, boolean or string type and the datanames are encoded as beginning with i_, r_, b_ or s_ respectively. It is possible to use these atom properties in conjunction with any other ASL expression. Any atoms which don't have these properties associated with them will never match. Some examples of using the ASL to address these properties are:

atom.i_my_integer_prop 1-4 atom.b_my_boolean_prop atom.r_my_real_prop < 4.0 atom.s_my_string_prop LIG_

3.4 Operators

A number of operators are supported:

3.4.1 The Boolean `and` Operator

Boolean `AND` (set intersection). The syntax for this operation is:

```
spec1 and spec2
```

where `spec1` and `spec2` are valid atom specifications. This operation will return the set of atms which meets the specifications `spec1` and `spec2`. For example, the expression:

```
mol. 1 and atom. CA
```

will return the set of all the alpha carbons of molecule 1. The specification:

```
res.num 1-100 and res. ala
```

returns all alanines in residues with numbers in the range 1-100.

3.4.2 The Boolean `or` Operator

The Boolean `OR` operator. The syntax for this operations is:

```
spec1 or spec2
```

where `spec1` and `spec2` are valid atom specifications. This returns the set of atoms which meet either specification `spec1` or `spec2`. For example, the expression:

```
mol. 1 or atom.ptype CA
```

returns the set of all atoms that are in molecule number 1, or are alpha carbons. The specification:

```
res.num 1-100 or res.ptype ala
```

returns all residues either with numbers in the range 1-100, or any alanines.

3.4.3 The Boolean `not` Operator

The Boolean `NOT` operator. The syntax for this operation is:

```
not spec1
```

where `spec1` is a valid atom specification. This returns the set of atoms that are not part of those defined by `spec1`. For example, the expression:

```
not atom. CA,C,N,O
```

will return a set containing all side chain atoms.

3.4.4 The `fillres` and `fillmol` Operators

Two special operations, `fillres` and `fillmol`, can be used to "fill out" the atoms defined by a atom specification to complete residue or molecule boundaries. For example:

```
fillres atom.num 1,100,40
```

will return all the atoms in residues of which atoms 1,100 and 40 are members. In a similar way:

```
fillmol atom.num 1,100,40
```

will return all the atoms in molecules of which atoms 1,100 and 40 are members.

3.4.5 The `within` and `beyond` Operators

The operators `within` and `beyond` can be used to define sets of atoms based on their distance from a set of atoms defined defined by an atom specification. The syntax for these operators is:

`within distance spec1`

which returns the atoms within, i.e, less than or equal to the distance in Angstroms of the set defined by spec1, and

`beyond distance spec2`

which returns the atoms which are further than the distance in Angstroms from the set defined by spec2.

For example, the expression:

`within 5.0 mol. 1`

returns the set of all atoms that are within 5 Å of molecule 1. The expression:

`beyond 5.0 mol. 2`

returns all atoms that are farther than 5 Å from molecule 2.

The combination of `fillres` and `within` or `beyond` is especially powerful.

`fillres within 5.0 mol. 1`

will produce a set containing the atoms of all complete residues that have atoms within 5 Å of molecule 1. Note that the `within` operator will also return the reference set of atoms:

`within 5.0 mol. 1`

returns the reference set of all atoms that are within 5 Å of molecule 1 and those that are part of molecule 1.

The `and` operator, when used with `within` and `beyond`, can be used to allow more specificity:

`mol. 2 and within 5.0 mol. 1`

returns the set of all atoms of molecule 2 that are within 5 Å of molecule 1.

3.4.6 The `withinbonds` Operator

A special type of `within`, one that finds all atoms within a certain number of bonds of the reference set:

`withinbonds` ⟨ num_bonds ⟩ ⟨ spec ⟩

For example:

`withinbonds 4 atom. 1`

will find all the atoms that are within four bonds of atom 1.

3.4.7 The `beyondbonds` Operator

A special type of `within`, one that finds all atoms beyond a certain number of bonds of the reference set:

`beyondbonds` ⟨ num_bonds ⟩ ⟨ spec ⟩

For example:

```
beyondbonds 4 atom. 1
```

will find all the atoms that are in the same molecule as atom 1 but beyond four bonds of atom 1.

3.5 Operator Priority

The order of priority of operators is (in decreasing order):

- `not`/`fillres`/`fillmol`
- `and`/`or`
- `within`/`beyond`

At equal levels of priority the expression will be evaluated left to right.

Examples:

```
within 5.0 mol. 1 or mol. 2
```

returns the set of all atoms that are within 5.0 Å of either molecule 1 or molecule 2 (`or` has higher priority).

```
not atom.ptype CA,C,O,N and mol. 1
```

returns the side chain atoms of molecule 1.

```
atom.ptype CA or mol. 1 and not res.pol polar
```

returns all alpha carbons and atoms in hydrophobic residues of molecule 1.

Parentheses can be used to override the order of evaluation:

```
not (atom.ptype CA,C,O,N or mol. 1)
```

produces all atoms either not in the backbone or not in molecule 1.

3.6 Implicit Operators

When no operator is specified, the following operations are assumed:

- Specifications within a property list have a Boolean `or` relationship. So the following specification:

  ```
  atom.ptype CA,CB
  ```

  matches any atom which has a PDB name of either CA or CB.
- When the classes are in order of decreasing priority in the class hierarchy, the `and` operator is assumed:

```
mol. 1 chain. A atom.ptype CA
```

is equivalent to:

```
mol. 1 and chain. A and atom.ptype CA
```

and both return the alpha carbons in chain A of molecule 1.

3.7 Creating New Sets from Existing Ones

Maestro supports a mechanism for defining and naming atom sets via its "Sets" panel. The names of existing sets may be used in expressions if they are prefixed with the word `set`. For example, if there are two sets defined as:

set S1 `mol. 1` (molecule number 1)
set S2 `atom.ptype C,O,N,CA` (all backbone atoms)

the following are then valid atom specifications:

- `set S1 and set S2`, all backbone atoms in molecule 1
- `set S1 or set S2`, all backbone atoms or atoms in molecule 1
- `within 5.0 set S1`, all atoms within 5.0 Å of molecule 1.

3.8 Special Specifications

- The specification `all` matches everything. You can use this in any Maestro command which expects an ASL operand in order to apply the command to all the atoms. For example to color all atoms green:

  ```
  coloratom color=green all
  ```

  Note that the syntax for this is just `all`, not `atom. all` or `molecule. all`.

- Anything enclosed in `/ /` is treated as a string in the linear substructure notation - see the BatchMin Reference Manual for a complete description of this. Some examples are:

  `/C3(-H1)(-H1)(-H1)/` specifies all methyl groups
  `/C2(=O2)-N2-H2/` specifies all amide groups
  `/C0-N0/` specifies all pairs bound C-N atoms, and
  `/O0-S0/` specifies any atom bound to a sulfur.

  Specifications made in this way can be treated in the same way as any other specification in the ASL, e.g. they can have operators applied to them.

3.9 Matching PDB Atom Names

Because PDB atom names are four characters wide we need to employ the following strategy in order to conveniently specify PDB atom names:

- Unquoted names which begin with a non-numeric character have a blank character inserted in front of them and are padded with blanks from the right to make up four characters before matching. Examples:
  - Property List: CA,C
    Actually Matched: " CA "," C "
  - Property List: CG1,CG2
    Actually Matched: " CG1 "," CG2"
- Unquoted names which begin with a number do not have an initial blank character inserted but are right to padded four characters. Example:
  - Property List: CA,1HB,2HB
    Actually Matched: " CA ","1HB ","2HB "
- Quoted names (double or single quotes are acceptable) are treated as literals but will be right padded with blanks to make up four characters. Examples:
  - Property List: CA,"CA"
    Actually Matched: " CA ","CA " (alpha carbons or calciums)
  - Property List: " N A"
    Actually Matched: " N A" (heme atoms)

3.10 Miscellaneous

3.10.1 Atoms not yet present in a structure

If a structure in the Workspace has only 100 atoms when the ASL definition: `atom.num 1,8,44,101,103` is issued, Maestro will simply match the atoms numbered 1,8 and 44. If additional atoms are subsequently added to the structure, the atoms bearing the numbers 101 and 103 will be added to the previously defined set.

3.10.2 Aliasing

Maestro allows you to define your own aliases, using either the Command Input Area or the Command Aliases panel. Maestro converts all aliases into their corresponding commands before performing operations involving the aliased commands. Users must ensure that aliases produce sensible results. Some aliases are supplied with the distribution. They are:

Operator: `and`
Aliases: intersection, INTERSECTION, &

Operator: `or`
Aliases: UNION, union, `|`

Operator: `not`
Aliases: !

Class Designator: `mol.`
Aliases: MOL, mol

Class Designator: `atom.`
Aliases: ATOM, atom

Class Designator: `res.`
Aliases: RES, res

Class Designator: `chain.`
Aliases: CHAIN, chain

ASL Definition: `atom. ca,c,n,h,o`
Aliases: BACKBONE, backbone

ASL Definition: `not (atom.pt ca,c,n,h,o)`
Aliases: SIDECHAIN, sidechain

ASL Definition: `"/H2-O3-H2/ or atom.mtype OW"`
Aliases: WATER, water

3.11 Useful Hints when using ASL with the Project Facility

Maestro 4.1 introduced its Project Facility. Entries can be included into and excluded from the Workspace. The order in which this is performed affects the molecule numbers. For example, if you have two entries in your Project Table called "A" and "B" and you include into an empty Workspace first A and then B, the molecule numbers will be 1 for A and 2 for B; however, if you first include B and then A, the molecule numbers will be 1 for B and 2 for A.

This means that an expression such as `mol. 1` will match different atoms in each of the above cases. In earlier versions of Maestro, it often made sense to use the `mol.` notation because there was no Project Facility. But with the Project Facility, Maestro is entry-centric and in most cases it makes more sense to use entry names.

For example, if you have an inhibitor and a receptor that are in different entries and wish to have a ribbon appear on only the receptor, use the entry

name in the ASL expression, not the molecule number. This will ensure that when the receptor is included that it, and only it, will be used to generate the ribbon(s). A different inclusion order of entries in the Workspace will then result in the same matching atoms. So for ribbons with a receptor called "receptor" it is more useful to use `entry.name receptor` as the ASL definition.

3.12 ASL Examples

This section gives some examples of the use of the ASL in real-life situations. Note that while these examples all use lower-case, the ASL expressions themselves are not case sensitive.

i) Defining a set to refer to a ligand and/or receptor.

The exact command will depend on the nature of your system. If the ligand and the receptor are separate entries then it will suffice to use

```
set ligand entry.name <ligand_name>
```

where <ligand_name> is the name of the entry that contains the ligand. Similarly

```
set receptor entry.name <receptor_name>
```

for the receptor with entry called <receptor_name>.

In order to define sets that will work with multiple ligands it's also possible to define the ligand as everything that is not part of the receptor. A definition of:

```
set ligand not set receptor
```

will identify the ligand as anything that's not part of the receptor.

If the ligand and the receptor are part of the same entry then molecule numbers are the best way to define the ligand and the receptor. Assuming the receptor is molecule 1 and the ligand molecule 2:

```
set ligand mol.num 2
set receptor mol.num 1
```

Note however that the use of molecule numbers in set definitions should be avoided where possible as these depend on the order in which the project entries are included into the Workspace. If it is possible to use entry names, then these should be used.

The subsequent examples assume that sets for the receptor and the ligand have been defined using one of the methods defined above.

ii) The set of atoms within a given distance of the ligand.

One common task is to do something with the set of atoms within a given distance of the ligand. For example to only display those atoms or to include them in a substructure region for a MacroModel calculation. These examples

will use the "displayonlyatom" command but the ASL which follows can be used with any other command that uses ASL.

To only display atoms within 5.0 Angstroms of the ligand:

```
displayonlyatom within 5.0 set ligand
```

A common variation is to display complete residues which have any of their atoms within a given distance of the ligand:

```
displayonlyatom fillres within 5.0 set ligand
```

It's also possible to restrict the expression so that it only applies to receptor atoms within a given distance of the ligand. Here the Boolean "and" operator is used to restrict the displayed atoms to the receptor only:

```
displayonlyatom set receptor and fillres within 5.0 set
ligand
```

Because this is a lengthy expression it's often convenient to make this into a set itself:

```
set active_site set receptor and fillres within 5.0 set
ligand
```

An equivalent form of this is:

```
set active_site (! set ligand) & fillres within 5.0 set
ligand
```

Note that "!" is a standard alias for "not" and similarly "&" for "and".

iii) Sidechain and backbone.

The ASL has standard aliases for the definition of sidechain and backbone atoms in proteins. For example to only display the atoms of the backbone:

```
displayonlyatom backbone
```

These aliases can be used with operators to build up more complicated expressions. For example to only display the sidechain of the receptor:

```
displayonlyatom sidechain and set receptor
```

To display only the sidechains of the atoms within 5.0 Angstroms of the ligand:

```
displayonlyatom sidechain and set receptor and fillres
within 5.0 set ligand
```

iv) Atoms of a given type.

There are a variety of ways to specify atoms of a given type. For example to specify all carbons, nitrogens and oxygens the following is used:

```
atom.ele C,N,O
```

To specify non-hydrogen atoms:

```
not atom.ele H
```

To specify the alpha carbons in a protein:

```
atom.ptype CA
```

To specify all sp2 carbons there are two choices. The first relies on knowing that the MacroModel atom type for such an atom is "C2" and using:

```
atom.mtype C2
```

The other (assuming no formally charged or radical carbons are present) uses the number of attachments to the atom:

```
atom.ele C and atom.att 3
```

To specify polar hydrogens:

```
atom.ele H and not /C0-H0/
```

or

```
atom.ele H and not atom.mtype H1
```

v) Water molecules.

The ASL has a standard alias "water". For example to delete all water molecules the Maestro command is:

```
delete atom water
```

vii) Restricting an operation to the atoms that are currently displayed in the workspace.

Often a user will be working with only a subset of the atoms in the workspace displayed. If an operation is to be performed only on the atoms that are displayed then the "atom.displayed" property can be used. For example to change the color to green of all the atoms currently displayed in the Workspace and to leave alone the undisplayed Workspace atoms:

```
coloratom color=green atom.disp
```

To only do it for the atoms that are displayed and in the receptor:

```
coloratom color=green atom.disp and receptor
```

viii) Specifying molecules.

All molecules with between 30 and 100 atoms:

```
mol.atoms 30-100
```

All molecules with over 100 atoms:

```
mol.atoms >100
```

All molecules with a molecular weight over 300:

```
mol.weight > 300.0
```

All molecules which contain a halogen:

```
fillmol atom.ele F,Cl,Br,I
```

ix) Specifying atoms based on a linear-substructure notation.

The ASL supports the use of a SMILES-like linear substructure notation to specify atoms with a particular bonding arrangement. The atoms are referred to by MacroModel atom types, but there are wildcard types that can be used to allow the expression to apply to any atoms of a given element type.

Some examples:

Any five-membered ring:

```
/00-00-00-00-00-1/
```

Aromatic six-membered carbon rings(C2 is sp2 carbon)

```
/C2-C2*C2-C2*C2-C2*1/
```

Amide groups:

```
/C2(*O2)-N2/
```

Methyl groups:

```
/C3(-H1)(-H1)(-H1)/
```

Water:

```
/H2-O3-H2/
```

Guanadinium group:

```
/N2(-H3)-C2(*N4(-H4)(-H4))-N2(-H3)(-H3)/
```

x) Using wildcard characters.

Most string-type property values can use wildcard characters. Some examples:

All PDB atom names beginning with C

```
atom.ptype C*
```

All forms of the histidine residue:

```
res.ptype HI*
```

All entries that begin with "lig":

```
entry.name lig*
```

xi) Addressing residues with negative residue numbers.

From version 6.1 of Maestro it is possible to use residue numbers that are negative or zero.

```
res. -8
```

will match residues with residue number -8.

xii) Use SMARTS expressions.

From version 7.0 of Maestro it is possible to use SMARTS expressions as part of an ASL expression.

Some examples:

```
smarts. CCC
```

will match all three-carbon subsequences

```
smarts. [R] and atom.ele N
```

will match all ring nitrogens

```
smarts. C1CCCCCC1
```

will match all six-membered carbon rings

4 The Entry Specification Language

This chapter documents the syntax of Maestro's Entry Specification Language (ESL).

4.1 Why an Entry Selection Language?

The Entry Selection Language (ESL) is used for selecting entries in a Maestro project based on the properties of those entries. ESL expressions can be of arbitrary complexity and can use parentheses and logical operators. While the ESL is similar in appearance to the Atom Specification Language (ASL), they are used for quite different purposes - the ASL for specifying sets of atoms in the workspace, the ESL for selecting entries in the project. A typical application for the ESL would be for filtering. An ESL expression can be used to select only those entries that meet a particular criteria, usually based on entry properties. Once selected, those entries can be displayed in the workspace or exported to an external file.

4.2 Entry Properties

There are a number of properties that can be used in ESL expressions:

`entry` The name of the entry is the most useful property at present. The syntax is:

> `entry` ⟨ name ⟩
> `entry_re` ⟨ name_expression ⟩

Entry names that contain spaces need to be enclosed in double quotes. The following wildcard characters can be used for the `entry_re` expression:

> `*` matches any number of any characters
> `?` matches any single character
> `#` matches any single digit

Examples:

```
entry anentry
entry crambin-1
entry_re cramb*
entry_re ligand##
```

Note: `name` can be used as an alias for `entry` and `name_re` can be used as an alias for `entry_re`.

`selected` This allows the selection of entries based on whether or not they are already selected in the project. Examples:

```
entry_re lig* and not selected
not selected
```

`included` This allows the selection of entries based on whether or not they are already included for display in the workspace.

```
entry_re lig* and not included
not included
```

`all` This is a quick way to select all entries.

4.3 Logical Operators

The following logical operators are supported (in order of decreasing priority)

- `not`
- `and` `or`

At equal levels of priority expression will be evaluated from left to right. Parentheses can be used to override the default order of priority.

4.4 Entry Property Comparisons

ESL expressions may also contain comparisons involving entry properties. Properties may be defined by their data names or their user names. The data names are the names that appear in the Maestro files. An example is "i_qp_n_stars". User names are the names that appear in the first row of Maestro's Project Table. An example of how this would appear in Maestro's Project Table Header is `n stars`. User names are specified in commands as:

`user(`⟨ propertyname ⟩`)`

For example:

`user(volume)`

and so,

`user(n stars)` and
`i_qp_n_stars`

are equivalent names for properties.

For integer and real type properties the following comparison operators are supported: `==`, `!=`, `<`, `>`, `<=`, `>=`.

For boolean (logical) properties, the name of the property can just be specified and will match any entries where that property is true. It is also possible to use `==` and `!=` in explicit comparisons with the values `true` and `false`.

Comparisons involving string properties have the form:

⟨ property name ⟩ ⟨ string literal ⟩

For example:

```
s_m_string_prop left
s_m_string_prop left*
```

String literals may include the `*`, `?` and `#` wildcard characters available for entry names as described above.

4.5 Examples

Some examples of the use of the ESL are:

```
entry anentry
entry_re ligand*
entry_re lig* and selected
(entry_re lig# and selected ) and not included
user(volume) < 100.0 and selected
included and user(n stars) == 2 and user(n amine) == 0
user(log s) > 0.5 and user(log S) < 1.0
```

5 Commands

1ddataset

Controls the appearance of the a single dataset on a plot of a dihedral drive of a single dihedral.

Syntax:

`1ddataset` *ccolor*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *curve*=solid | dashed | noline *cwidth*=⟨ n ⟩ *scolor*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *ssize*=⟨ n ⟩ *symbol*=filled_rectangle | hollow_rectangle | filled_circle | hollow_circle | cross | point | filled_diamond | hollow_diamond | no_symbol ⟨ data_set_name ⟩ [⟨ grd_file_name ⟩]

Options:

ccolor The color of the curve used on the graph for this dataset.

Valid values: black, red, green, blue, purple, orange, blue_green, light_green, red_purple, yellow, cyan

Default value: **black**

curve The type of curve which will join the points on the graph.

Valid values: solid, dashed, noline

Default value: **solid**

cwidth The width of the line used on the graph for this dataset.

Valid values: integers

Default value: **1**

Minimum:	1
Maximum:	5

scolor The color of the symbols used on the graph for this dataset.

Valid values:	black red green blue purple orange blue_green light_green red_purple yellow cyan
Default value:	**black**

ssize The size of the symbol used on this plot for this dataset.

Valid values:	integers
Default value:	**3**
Minimum:	2
Maximum:	5

symbol The type of symbol used for this data set on the graph.

Valid values:	filled_rectangle hollow_rectangle filled_circle hollow_circle cross point filled_diamond hollow_diamond no_symbol
Default value:	**filled_rectangle**

Operands:

⟨ data_set_name ⟩ [⟨ grd_file_name ⟩]

The first operand is the name of the dataset which is to be created or to have its properties changed. If a second operand is present then that is treated as the name of a file from which the data is to be read. The full name of the file, including the suffix, should be included.

1dplot

Controls the appearance of the plot the energy as a function of a drive of a single dihedral angle.

Syntax:

`1dplot` *amax*=⟨ x ⟩ *amin*=⟨ x ⟩ *emax*=⟨ x ⟩ *emin*=⟨ x ⟩
escale=absolute | relative *orientation*=landscape | portrait
papersize=letter | a4 *showlegend*=yes | no
squareplot=yes | no *title*=⟨ text ⟩ *units*=kj | kcal
xaxislabel=⟨ text ⟩ *xdecimal*=⟨ n ⟩ *yaxislabel*=⟨ text ⟩
ydecimal=⟨ n ⟩ [⟨ postscript_file_name ⟩]

Options:

amax The maximum angle value displayed on the graph
Valid values: reals
Default value: **0**

amin The minimum angle value displayed on the graph
Valid values: reals
Default value: **0**

emax The maximum energy value displayed on the graph
Valid values: reals
Default value: **0**

emin The minimum energy value displayed on the graph
Valid values: reals
Default value: **0**

escale The energy scale (absolute/relative) for the graph
Valid values: absolute
relative
Default value: **absolute**

orientation
The paper orientation for the postscript output
Valid values: landscape
portrait
Default value: **landscape**

papersize The paper size for the postscript output
Valid values: letter
a4
Default value: **letter**

showlegend
Whether the legend is displayed for the 1D plot
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

squareplot A boolean which controls whether the plot will be constrained to be square.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

title The plot title
Valid values: text strings
Default value:

units The energy units for the graph
Valid values: kj
kcal
Default value: **kj**

xaxislabel The label for the X-axis.
Valid values: text strings
Default value: **Coordinate**

xdecimal The number of decimal places for the X axis.
Valid values: integers
Default value: **0**
Minimum: 0
Maximum: 5

yaxislabel The label for the Y-axis.
Valid values: text strings
Default value: **Energy**

ydecimal The number of decimal places for the Y axis.
Valid values: integers
Default value: **1**
Minimum: 0
Maximum: 5

Operands:
[⟨ postscript_file_name ⟩]
If an operand is given then this will be treated as the name of a file to which a postscript representation of the plot will be written. The full name of the file, including any suffix, must be included.

1drescale

Rescales the 1D plot axes back to be able to view the entire data range.

Syntax:

`1drescale`

1dtable

Write out a table of the data from the 1D plot to an external file.

Syntax:

`1dtable`

2ddataset

Controls the appearance of the dataset plotted on the 2D grid contour plot.

Syntax:

`2ddataset` *colormap*=⟨ text ⟩ *cwidth*=⟨ n ⟩ *emax*=⟨ x ⟩ *emin*=⟨ x ⟩
fillcontours=yes | no *labelcontours*=yes | no
negativedashed=yes | no *numcontours*=⟨ n ⟩
showlegend=yes | no *title*=⟨ text ⟩ *xaxislabel*=⟨ text ⟩
yaxislabel=⟨ text ⟩

Options:

colormap The colormap to be used for displaying the contour plot

Valid values:	text strings
Default value:	**jet**

cwidth The width of the contours drawn on the plot

Valid values:	integers
Default value:	**1**
Minimum:	1
Maximum:	5

emax The maximum energy value displayed on the graph

Valid values:	reals
Default value:	**0**

emin The minimum energy value displayed on the graph

Valid values: reals
Default value: **0**

fillcontours

Whether contours are filled or drawn as lines

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

labelcontours

Whether contours are labelled directly on the plot.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

negativedashed

A boolean which controls whether contours corresponding to negative values will be drawn with a dashed line.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numcontours

The number of contours used on the graph

Valid values: integers
Default value: **10**
Minimum: 2
Maximum: 20

showlegend

Whether the legend is displayed for the contour plot

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

title The plot title

Valid values: text strings
Default value:

xaxislabel The label for the X-axis.

Valid values: text strings
Default value: **Coordinate 1**

yaxislabel The label for the Y-axis.

Valid values: text strings
Default value: **Coordinate 2**

2dplot

Controls the appearance of a plot of the energy as a function of two dihedral angles.

Syntax:

`2dplot` *contdecimal*=⟨ n ⟩ *escale*=absolute | relative *orientation*=landscape | portrait *papersize*=letter | a4 *squareplot*=yes | no *units*=kj | kcal *xdecimal*=⟨ n ⟩ *ydecimal*=⟨ n ⟩ [⟨ postscript_file_name ⟩]

Options:

contdecimal
The number of decimal places for the contour labels

Valid values:	integers
Default value:	**0**
Minimum:	0
Maximum:	5

escale The energy scale (absolute/relative) for the graph

Valid values:	absolute relative
Default value:	**absolute**

orientation
The paper orientation for the postscript output

Valid values:	landscape portrait
Default value:	**landscape**

papersize The paper size for the postscript output

Valid values:	letter a4
Default value:	**letter**

squareplot A boolean which controls whether the plot will be constrained to be square.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

units The energy units for the graph

Valid values:	kj kcal
Default value:	**kj**

xdecimal The number of decimal places for the X axis.

Valid values:	integers
Default value:	**0**
Minimum:	0
Maximum:	5

ydecimal The number of decimal places for the Y axis.

Valid values:	integers
Default value:	**0**
Minimum:	0
Maximum:	5

Operands:

[⟨ postscript_file_name ⟩]

If an operand is given, this is treated as the name of a file to which a postscript representation of the file is to be written.

2drescale

Rescales the plot axes back to be able to view the entire data range.

Syntax:

`2drescale`

addatomproperty

Creates a new atom-level property for the atoms which are selected in the Workspace.

Syntax:

`addatomproperty` *value*=⟨ text ⟩ ⟨ property ⟩

Options:

value An optional initial value for the property.

Valid values:	text strings
Default value:	

Operands:

⟨ property ⟩

The M2IO data name of the new property.

addfromhold

This command retrieves the structure from the hold set and adds it to the current on-screen structure. The added hold structure is placed at the periphery of the on-screen structure and then a scale-to-screen is done.

Syntax:

addfromhold ⟨ hold_name ⟩

Operands:

⟨ hold_name ⟩

The name of the hold. This must be the name which was specified when the hold was created using the "hold" command.

adjustangle

Adjust the bond angle specified by the 3 atoms to the given value.

Syntax:

adjustangle *angle*=⟨ x ⟩ *move*=attached | terminal | single
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

Options:

angle Value to which to set the angle

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	180.0

move This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values:	attached terminal single
Default value:	**attached**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩
Three atoms which are used to adjust an angle. Note that direction does matter. So a-b-c is different from c-b-a. The third atom 'points' to the part of the structure that will be adjusted.

adjustchirality

Adjustl chiral center(s) in the specified set of atoms.

Syntax:

`adjustchirality` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
An ASL specification of the atoms which are to have their chiral centers inverted. Because of the way this works, this set of atom should involve at least whole molecules.

adjustdihedral

Adjust the dihedral specified by 4 atoms to the given value.

Syntax:

`adjustdihedral` *dihedral*=⟨ x ⟩ *dihedralterm*=⟨ x ⟩ *move*=attached | terminal | single ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

dihedral Value to which to set the torsion

Valid values:	reals
Default value:	**0**
Minimum:	-180.1
Maximum:	180.1

dihedralterm
: Value to which to terminally attached to the dihedral

  Valid values: reals
  Default value: **0**
  Minimum: -180.1
  Maximum: 180.1

move
: This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

  Valid values: attached
  terminal
  single
  Default value: **attached**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Four atoms which are used to adjust a torsion. Note that direction does matter. So a-b-c-d is different from d-c-b-a. The fourth atom 'points' to the part of the structure that will be adjusted.

adjustdistance

Adjust the distance specified by 2 atoms to the given value.

Syntax:

`adjustdistance` *distance*=⟨ x ⟩ *move*=attached | terminal | single
⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

distance
: Value to which to set the torsion

  Valid values: reals
  Default value: **0**
  Minimum: 0.0

move
: This option sets how to move other atoms that attached to the selected moving atom. There are three options: (1) move all attached atoms, (2) move all attached terminal atoms, and (3) move only single atom. Default option is (1).

Valid values:	attached terminal single
Default value:	**attached**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
Two atoms which are used to adjust a distance. Note that direction does matter. So a-b is different from b-a. The second atom 'points' to the part of the structure that will be adjusted.

alias

Define an alias for a command. This allows an abreviated symbol to be defined for any command.

Syntax:

`alias` ⟨ alias_name ⟩ ⟨ definition ⟩

Operands:
⟨ alias_name ⟩ ⟨ definition ⟩
The first operand is the name of the alias. If this contains embedded spaces then it must be enclosed in double quotes. The remaining operands are the definition of the alias. When command processing takes place, all occurences of the alias name (the first operand) will be replaced by the definition of the alias.

alignaxis

Specifies a pair of atoms of a molecule to align the molecule to one of X, Y, or Z axis.

Syntax:

`alignaxis` *axis*=x | y | z ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

axis	This option determines to which axis will the molecule be aligned: x=1, y=2, z=3.

Valid values:	x
	y
	z
Default value:	**x**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

The first atom will be put at the origine and the second atom will be at whatever the distance is on the chosen axis.

alignplane

Define a plane of all atoms in the set described by the ASL operand and align the plane to the XY plane.

Syntax:

alignplane *plane*=xy | yz | zx ⟨ ASL ⟩

Options:

plane This option determines to which plane will the molecule be aligned: xy=1, yz=2, zx=3.

Valid values:	xy
	yz
	zx
Default value:	**xy**

Operands:

⟨ ASL ⟩

A string in the atom specification language which describes the set of atoms which are used to define a plane to be aligned.

angle

Specifies a triplet of atoms to have their bond angle measured and displayed.

Syntax:

`angle` *xoffset*=⟨ x ⟩ *yoffset*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

Options:

xoffset Specifies the X offset in Angstroms. Any angle created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created angles.

Valid values: reals
Default value: **0**

yoffset Specifies the Y offset in Angstroms. Any angle created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created angles.

Valid values: reals
Default value: **0**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

The three atoms between which the angle is to be measured. Note that the specifying a-b-c is the same as specifying c-b-a.

appendribbons

Generate a ribbon for the parts that are not currently in ribbon.

Syntax:

```
appendribbons
```

application

Determines which backend application is currently active and which application-specific menu is displayed. For example the command application Impact will result in the Impact menu being displayed in the main menu bar.

Syntax:

`application` ⟨ application_name ⟩

Operands:
⟨ application_name ⟩
The name of the backend application which Maestro is currently interacting with. Current options are "none", "macromodel" or "impact".

assigncharges

This keyword is used to assign partial charges to all atoms in the workspace. The forcefield used is OPLS2005.

Syntax:

`assigncharges`

atom

Create a new atom in space with the current type and at the position specified by the operands.

Syntax:

`atom` *by*=element | type *element*=⟨ text ⟩ *type*=⟨ n ⟩ ⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩

Options:

by — This options determines whether atoms are to be described by element symbols (the element= option) or by atom types (type= option).
Valid values: element, type
Default value: **element**

element — This option sets the current element of all atoms to be created (or retyped) if the by option == element
Valid values: text strings
Default value: **C**

type — This option sets the current type of all atoms to be created (or retyped) if the by option == type
Valid values: integers
Default value: **3**

Operands:
⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩
The operands are three real numbers which specify the x, y and z coordinates for the new atom.

atomname

Set the PDB atom name for all atoms which match the ASL specification.

Syntax:

`atomname` ⟨ PDBNAME ⟩ ⟨ ASL ⟩

Operands:
⟨ PDBNAME ⟩ ⟨ ASL ⟩
The first operand is the PDB atom name which will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their atom names changed.

atomproplabel

Turns the display of additional (non-standard) atom properties on or off. These are the properties read from the structure file at run-time as opposed to the standard properties controlled by the 'labelatom' command.

Syntax:

`atomproplabel` *display*=yes | no ⟨ property display name ⟩

Options:

display — A boolean option which determines if the atom property is to be displayed or not in the current label definition.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ property display name ⟩
The name of the atom property. It's not an error if this property does not exist. Note this can be the 'Display Name' as it appears in the atom labels panel or the 'Data Name' as it appears in the Maestro format file.

attach

Attach the currently selected fragment to the specified atom. The bond to this atom (and the atom itself) will be replaced by a bond from the incoming fragment.

Syntax:

`attach` ⟨ atom_num ⟩

Operands:
⟨ atom_num ⟩
The atom number defines the bond to which the fragment be attached. The atom specifed must be a terminally attached atom.

attachmentmarkerdump

Print out the current option values of the attachment marker command.

Syntax:

`attachmentmarkerdump`

attachmentmarkersettings

Set graphical data of attachment markers.

Syntax:

`attachmentmarkersettings` *ambient*=⟨ x ⟩ *blue*=⟨ x ⟩
cornradius=⟨ x ⟩ *cylinderheight*=⟨ x ⟩ *cylinderradius*=⟨ x ⟩
diffuse=⟨ x ⟩ *drawstyle*=solid | line *emission*=⟨ x ⟩ *green*=⟨ x ⟩
linewidth=⟨ n ⟩ *reagentradius*=⟨ x ⟩ *red*=⟨ x ⟩ *selectblue*=⟨ x ⟩
selectgreen=⟨ x ⟩ *selectred*=⟨ x ⟩ *shininess*=⟨ x ⟩ *sliceline*=⟨ n ⟩
slicesolid=⟨ n ⟩ *specular*=⟨ x ⟩ *stackline*=⟨ n ⟩ *stacksolid*=⟨ n ⟩
transparency=⟨ x ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.
Valid values: reals

Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

blue The blue component of attachment markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

cornradius

The radius of corn of attachment markers.

Valid values: reals
Default value: **0.55**
Minimum: 0.0

cylinderheight

The cylinder height ratio of attachment markers.

Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

cylinderradius

The radius of cylinder of attachment markers.

Valid values: reals
Default value: **0.26**
Minimum: 0.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

drawstyle The styles of rendering attachment markers, they are: 1 - solid, and 2 - lines. Default is solid.

Valid values: solid
line
Default value: **solid**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 1.0

green The green component of attachment markers.
Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

linewidth Set the width of lines in drawing attachment.
Valid values: integers
Default value: **2**
Minimum: 1

reagentradius
The radius of sphere for attachment markers having associated reagents.
Valid values: reals
Default value: **0.8**
Minimum: 0.0

red The red component of attachment markers.
Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

selectblue The blue component of selected attachment markers.
Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

selectgreen
The green component of selected attachment markers.
Valid values: reals
Default value: **0.9**
Minimum: 0.0
Maximum: 1.0

selectred The red component of selected attachment markers.
Valid values: reals
Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

shininess Set material property - shininess, for front face.
Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

sliceline Set the slices of drawing line attachment.

Valid values:	integers
Default value:	**10**
Minimum:	2

slicesolid Set the slices of drawing solid attachment.

Valid values:	integers
Default value:	**36**
Minimum:	2

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.2**
Minimum:	0.0
Maximum:	1.0

stackline Set the stacks of drawing line attachment.

Valid values:	integers
Default value:	**8**
Minimum:	2

stacksolid Set the stacks of drawing solid attachment.

Valid values:	integers
Default value:	**18**
Minimum:	2

transparency
The transparency of rendering attachment markers.

Valid values:	reals
Default value:	**20**
Minimum:	0.0
Maximum:	100.0

autosetup

Performs the automatic setup for the conformational search. The setup is only performed on those atoms which match the ASL definition given in the operand.

Syntax:

`autosetup` *chiralatoms*=yes | no *compatoms*=yes | no
rings=yes | no *torsionchecks*=yes | no ⟨ ASL ⟩

Options:

chiralatoms
A boolean option which determines whether the the automatic setup will include finding chiral atoms.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

compatoms
A boolean option which determines whether the the automatic setup will include finding comparison atoms.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rings A boolean option which determines whether the the automatic setup will include finding ring closures.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

torsionchecks
A boolean option which determines whether the the automatic setup will include finding torsion angles to be checked.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ ASL ⟩
A string in the atom specification language. The automatic setup will be peformed for only that atoms which match this ASL specification.

beginundoblock

Begin a new undoable command block.

Syntax:

`beginundoblock`

bmincomfile

Write a MacroModel command file with the current energy settings.

Syntax:

`bmincomfile`

bond

Create a new bond between two atoms or increments the bond order between two existing atoms.

Syntax:

`bond` ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The operands are the numbers of the two atoms which are to be connected by a bond or have the bond order of an existing bond incremented.

bondorder

Increment, decrement or set the bond order for the specified bond.

Syntax:

`bondorder` increment | decrement | ⟨ bond_order ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:
increment | decrement | ⟨ bond_order ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩
The first operand is either the word "increment", the word "decrement", or an integer value representing the bond order to be applied to the specifed bond. Note that "0" is a valid bond order. The second and third operands are the atoms which define the bond which is to have its bond order changed.

bondoutline

Control features related to bond outlines

Syntax:

bondoutline *autoscale*=yes | no *fixedwidth*=⟨ x ⟩ *largewidth*=⟨ x ⟩
mediumwidth=⟨ x ⟩ *smallwidth*=⟨ x ⟩ *technique*=2pass | 3pass
usewhenmoving=never | automatic | always
visibleatomcutoff=⟨ n ⟩ *whenmovingcutoff*=⟨ n ⟩

Options:

autoscale The width of the outline border is automatically scaled when this is enabled. If this if off, then the value of width is used at all scales.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fixedwidth Fixed width bond outline. Ignore if repall bondoutline is off or when bondoutline is on but autoscale is on.

Valid values: reals
Default value: **2**
Minimum: 0.01
Maximum: 20.0

largewidth The border width to use when the scale is large, meaning you are zoomed in and things appear large.

Valid values: reals
Default value: **4**
Minimum: .5
Maximum: 20.0

mediumwidth

The border width to use when the scale is medium, meaning you are partially zoomed out.

Valid values: reals
Default value: **3**
Minimum: .5
Maximum: 20.0

smallwidth

The border width to use when the scale is small, meaning you are zoomed out and things appear small.

Valid values: reals
Default value: **1**
Minimum: .5
Maximum: 20.0

technique Use a 2-pass + stenciling technique. The results are slightly different. Currently, only the 2-pass is used. 3-pass is not currently used and should do the same as using 2-pass. When bond

outlines are enabled you may not notice any performance degredation depending on you graphics system and what you're viewing. It's possible, however, that you may see a significant performance hit (roughly 5x).

Valid values: 2pass
3pass
Default value: **2pass**

usewhenmoving

When set to always bond outlines will be drawn when the structure is being rotated. If never, then no bond outlines will be drawn when the structure is rotated. If set to automatic, then bond outlines are drawn but only when the number of atoms is less then the value of 'whenmovingcutoff'.

Valid values: never
automatic
always
Default value: **never**

visibleatomcutoff

If the number of visible atoms is larger than this amount, then do not display bond borders.

Valid values: integers
Default value: **1500**
Minimum: 1

whenmovingcutoff

If usewhenmoving is set to automatic, then see if the number of visible atoms is less than this amount. If so, then display bond borders. If it is greater than this amount, then do not display bond borders when the user is roating or translating the structure.

Valid values: integers
Default value: **1500**

bondtonew

Create a new atom and bond it with a single bond to an existing atom.

Syntax:

bondtonew ⟨ atom_num ⟩ ⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩

Operands:
⟨ atom_num ⟩ ⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩
The first operand is the number of an existing atom from which the bond is to be drawn. The remaining three operands are the x, y and z coordinates at which the new atom is to be placed.

buildoptions

Sets options associated with structure building. These are persistent and will be retained between Maestro sessions.

Syntax:

buildoptions *adjustbondlengths*=yes | no
adjustnumhydrogens=yes | no *useunitedatoms*=yes | no

Options:

adjustbondlengths
When an atom type is changed, bond lengths around that atom will be set to ideal values.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

adjustnumhydrogens
During bond order, drawing or formal charge change operations, rectify the number of hydrogens to be consistent with the new bonding or formal charge.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

useunitedatoms
Will allow the generation of united atom types while structure building.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

calcenergy

Used to set options associated with a single point energy calculation.

Syntax:

`calcenergy` *listing*=none | minimal | complete

Options:

listing This option determines the extent of the listing of energy components.

Valid values:	none minimal complete
Default value:	**none**

Aliases:

ecalc (see [ecalc], page 127)

canonicalname

Sets atom names to one generated from atom properties (such as element or PDB names) for all the atoms matched by the ASL operand.

Syntax:

`canonicalname` ⟨ separator ⟩ ⟨ option_flag ⟩ ⟨ ASL ⟩

Operands:

⟨ separator ⟩ ⟨ option_flag ⟩ ⟨ ASL ⟩

The first operand is the separator symbol to be used in generating the canonical names of all atoms. The second operand is the option_flag, which indicates which two atom properties will be used to construct the canonical names. Possible values are "3" - from element and atom number and "12" PDB chain, residue and atom name. The final operand is the ASL expression to indicate which atoms are to have a canonical name set.

caption

Create a Workspace caption and select it

Syntax:

`caption`

captionbackground

Caption backgroud settings

Syntax:

`captionbackground` *type*=automatic | auto | none | user

Options:

type	Background caption type	
	Valid values:	automatic auto none user
	Default value:	**automatic**

captioncolor

Color of the caption text. Default color is yellow

Syntax:

`captioncolor` *alpha*=⟨ x ⟩ *blue*=⟨ x ⟩ *green*=⟨ x ⟩ *red*=⟨ x ⟩

Options:

alpha	Alpha component. 0 is invisible. 1 is opaque.	
	Valid values:	reals
	Default value:	**0.67**
	Minimum:	0.0
	Maximum:	1.0
blue	Blue component.	
	Valid values:	reals
	Default value:	**0**
	Minimum:	0.0
	Maximum:	1.0
green	Green component.	
	Valid values:	reals

	Default value:	**1**
	Minimum:	0.0
	Maximum:	1.0
red	Red component.	
	Valid values:	reals
	Default value:	**1**
	Minimum:	0.0
	Maximum:	1.0

captiondelete

Delete the selected captions.

Syntax:

`captiondelete`

captionfont

Font settings for the caption

Syntax:

`captionfont` *name*=⟨ text ⟩ *size*=⟨ x ⟩ *style*=regular | bold | italic | italicbold

Options:

name	Font name for the caption (helvetica, etc.)	
	Valid values:	text strings
	Default value:	**helvetica**
size	Font size for the caption	
	Valid values:	reals
	Default value:	**14**
	Minimum:	3
	Maximum:	96
style	Font style for the caption (regular, bold, etc.)	
	Valid values:	regular bold italic italicbold

Default value: **regular**

captionhide

Hide the caption

Syntax:

`captionhide`

captionposition

Position of the caption.

Syntax:

`captionposition` *x*=⟨ x ⟩ *y*=⟨ x ⟩

Options:

x Left-side X position of the caption. A fraction from zero to one. X origin is left-hand side of the window.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	0.98

y Bottom left Y position of the caption. A fraction from zero to one. Y origin is bottom of window.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0
Maximum:	0.98

captionselect

Select only the named caption

Syntax:

`captionselect`

captionselectadd

Add the named caption to the selected captions

Syntax:

`captionselectadd`

captionshow

Show the caption

Syntax:

`captionshow`

captionsoff

Hide all captions but leave their visible state alone.

Syntax:

`captionsoff`

captionson

Display all captions which are visible

Syntax:

`captionson`

captiontext

Text for the caption to be displayed in the Workspace

Syntax:

```
captiontext
```

captionunselect

Unselect the specifiecd caption

Syntax:

```
captionunselect
```

captionunselectall

Unselect all captions

Syntax:

```
captionunselectall
```

cascadepanels

Cascade visible panels

Syntax:

```
cascadepanels
```

cd

This is a standard alias for **changedirectory** (see [changedirectory], page 68).

cellsmarkerdump

Print out the current option values of the Glide hydrophobic Cells markers command.

Syntax:

`cellsmarkerdump`

cellsmarkersettings

Set graphical data of Glide hydrophobic cells markers.

Syntax:

`cellsmarkersettings` *activeblue*=⟨ x ⟩ *activegreen*=⟨ x ⟩ *activered*=⟨ x ⟩ *ambient*=⟨ x ⟩ *blue*=⟨ x ⟩ *diffuse*=⟨ x ⟩ *emission*=⟨ x ⟩ *green*=⟨ x ⟩ *highlightblue*=⟨ x ⟩ *highlightgreen*=⟨ x ⟩ *highlightred*=⟨ x ⟩ *labelregions*=yes | no *red*=⟨ x ⟩ *regionblue*=⟨ x ⟩ *regiongreen*=⟨ x ⟩ *regionred*=⟨ x ⟩ *roundingeffect*=⟨ x ⟩ *shininess*=⟨ x ⟩ *specular*=⟨ x ⟩ *step*=⟨ n ⟩ *transparency*=⟨ x ⟩

Options:

activeblue The blue color component of cell active frame and region label.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

activegreen

The green color component of cell active frame and region label.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

activered The red color component of cell active frame and region label.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

blue The blue color component of cell markers.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

green The green color component of cell markers.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

highlightblue

The blue color component of cell markers if the cell is of a highlight region.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

highlightgreen

The green color component of cell markers if the cell is of a highlight region.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

highlightred

The red color component of cell markers if the cell is of a highlight region.

Valid values: reals

Default value: **1**
Minimum: 0.0
Maximum: 1.0

labelregions

This option determines whether a region should be labeled (true) or not (false).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

red The red color component of cell markers.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

regionblue The blue color component of cell markers if the cell is of a region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

regiongreen

The green color component of cell markers if the cell is of a region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

regionred The red color component of cell markers if the cell is of a region.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

roundingeffect

This determines the rounding effect of edges of a cell, for front face.

Valid values: reals
Default value: **8**
Minimum: 0.0

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0
Maximum:	1.0

step The step domain tolerance of cells.

Valid values:	integers
Default value:	**3**
Minimum:	1

transparency
The transparency of phobic cell markers.

Valid values:	reals
Default value:	**30**
Minimum:	0.0
Maximum:	100.0

centeratom

Set global center of rotation to the given atom

Syntax:

`centeratom` *atom*=⟨ n ⟩

Options:

atom Atom to center transformations on

Valid values:	integers
Default value:	**1**
Minimum:	1

centerbond

Set global center of rotation to the given bond

Syntax:

`centerbond` *at1*=⟨ n ⟩ *at2*=⟨ n ⟩

Options:

at1 Atom1 of bond to be center of transformation

Valid values:	integers
Default value:	**1**
Minimum:	1

at2 Atom 2 of bond to be center of transformation

Valid values:	integers
Default value:	**1**
Minimum:	1

centercoordinates

Set global center of rotation to the given coordinate

Syntax:

`centercoordinates` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

x X center

Valid values:	reals
Default value:	**0**

y Y center

Valid values:	reals
Default value:	**0**

z Z center

Valid values:	reals
Default value:	**0**

centroid

This command takes the previously defined centroid atoms, averages the x, y, z co-ordinates and then creates a new atom of type 61 at the average position.

Syntax:

`centroid`

centroidatom

Define a single atom which will be used to define the centroid. Associated with this atom is a single atom marker.

Syntax:

`centroidatom` ⟨ atom_num ⟩

Operands:

⟨ atom_num ⟩

The number of the atom to be used to define the centroid.

centroidatomset

Specifies a set of atoms to be added to the current centroid definition.

Syntax:

`centroidatomset` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language which describes the set of atoms which are to be added to the current centroid definition.

cglidedockconstraintposition

Specifies a constraint position in the receptor for a Glide calculation.

Syntax:

`cglidedockconstraintposition` *feature*=⟨ n ⟩ *index*=⟨ n ⟩ *radius*=⟨ x ⟩ *radiusnoe*=⟨ x ⟩ *type*=⟨ n ⟩ *use1*=yes | no *use2*=yes | no *use3*=yes | no *use4*=yes | no *usenoe*=yes | no *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

feature The constraint feature of position in docking.
Valid values: integers
Default value: **-1**

index The index of position in docking.
Valid values: integers
Default value: **0**

radius The radius of a Glide constraint position.
Valid values: reals
Default value: **1**
Minimum: 0.0001

radiusnoe The minimum NOE distance of a Glide constraint position.
Valid values: reals
Default value: **0**
Minimum: 0.0

type The constraint type of position in docking.
Valid values: integers
Default value: **0**

use1 The flag indicates if this position will be used in docking for group 1.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use2 The flag indicates if this position will be used in docking for group 2.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use3 The flag indicates if this position will be used in docking for group 3.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use4 The flag indicates if this position will be used in docking for group 4.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

usenoe Use the NOE constraint or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

x The X coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

y The Y coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

z The Z coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

cglidedockconstraintregion

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

`cglidedockconstraintregion` *atoms*=⟨ n ⟩ *feature*=⟨ n ⟩
index=⟨ n ⟩ *type*=⟨ n ⟩ *use1*=yes | no *use2*=yes | no
use3=yes | no *use4*=yes | no ⟨ region_name ⟩

Options:

atoms The number of required ligand atoms of constraint region in docking.
Valid values: integers
Default value: **1**

feature The constraint feature of constraint region in docking.
Valid values: integers
Default value: **-1**

index The index of constraint region in docking.
Valid values: integers
Default value: **0**

type The constraint type of constraint region in docking.
Valid values: integers
Default value: **0**

use1 The flag indicates if this constraint region will be used in docking for group 1.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use2 The flag indicates if this constraint region will be used in docking for group 2.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use3 The flag indicates if this constraint region will be used in docking for group 3.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use4 The flag indicates if this constraint region will be used in docking for group 4.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ region_name ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

chainname

Set the chain name for all atoms which match the ASL specification.

Syntax:

`chainname` ⟨ CHAINNAME ⟩ ⟨ ASL ⟩

Operands:
⟨ CHAINNAME ⟩ ⟨ ASL ⟩
The first operand is the PDB chain name (a single character) that will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their residue names changed.

changedirectory

Change the current directory to that given by the operand of this command.

Syntax:

`changedirectory` ⟨ new_directory ⟩

Operands:

⟨ new_directory ⟩

The name of the directory to be made the current directory.

Aliases:

cd (see [cd], page 59), **chdir** (see [chdir], page 69)

chdir

This is a standard alias for **changedirectory** (see [changedirectory], page 68).

chiralatom

Specifies an atom which is to be marked as "chiral" in the conformational search

Syntax:

`chiralatom` ⟨ atom_number ⟩

Operands:

⟨ atom_number ⟩

The number of an atom which is to be added to the list of atoms which are to be considered as chiral in the conformational search.

clip

Adjust the clipping planes

Syntax:

`clip` *back*=⟨ x ⟩ *backselect*=⟨ x ⟩ *backsurface*=⟨ x ⟩ *boxoffset*=⟨ x ⟩ *front*=⟨ x ⟩ *frontselect*=⟨ x ⟩ *frontsurface*=⟨ x ⟩ *leftslopesurface*=⟨ x ⟩ *leftsurface*=⟨ x ⟩ *objects*=all | surfaces | surfacesselected | selected *rightslopesurface*=⟨ x ⟩ *rightsurface*=⟨ x ⟩

Options:

back Position at which to set the back clipping plane
Valid values: reals
Default value: **0**

backselect Position at which to set the back clipping plane for selected atoms
Valid values: reals
Default value: **0**

backsurface
Position at which to set the back clipping plane for surfaces
Valid values: reals
Default value: **0**

boxoffset The offset of surface clipping box for clipping surfaces to selected atoms.
Valid values: reals
Default value: **0**

front Position at which to set the front clipping plane
Valid values: reals
Default value: **0**

frontselect Position at which to set the front clipping plane for selected atoms
Valid values: reals
Default value: **0**

frontsurface
Position at which to set the front clipping plane for surfaces
Valid values: reals
Default value: **0**

leftslopesurface
Position at which to set the left slope clipping plane for surfaces
Valid values: reals
Default value: **0**

leftsurface Position at which to set the left clipping plane for surfaces
Valid values: reals
Default value: **0**

objects What type of objects is to be clipped.

Valid values: all
surfaces
surfacesselected
selected

Default value: **all**

rightslopesurface
Position at which to set the right slope clipping plane for surfaces

Valid values: reals
Default value: **0**

rightsurface
Position at which to set the right clipping plane for surfaces

Valid values: reals
Default value: **0**

clipatomset

Syntax:

`clipatomset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. This set defines clipping atoms for clipping atoms.

clipsurfaceset

Syntax:

`clipsurfaceset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. This set defines clipping atoms for clipping surfaces.

cliptoallatoms

Move clipping planes to all atoms.

Syntax:

`cliptoallatoms`

cliptoligand

Move clipping planes to ligand atoms.

Syntax:

`cliptoligand`

cliptoselectedatoms

Move clipping planes to selected atoms.

Syntax:

`cliptoselectedatoms`

clusteratom

Defines a single comparison atom to be used for XCluster jobs where clustering is based on Atomic RMS differences.

Syntax:

`clusteratom` ⟨ atom ⟩

Operands:
⟨ atom ⟩
The number of an atom which is to added to the list of comparison atoms for a clustering calculation.

clusterheavyatoms

Specifies the atoms to be used in comparisons of conformers in XCluster jobs where clustering is based on Atomic RMS differences. The atoms specified are all of the heavy atoms and optionally hydrogens on oxygen and sulfur.

Syntax:

`clusterheavyatoms` *osh*=yes | no

Options:

osh A boolean which controls whether hydrogens on oxygen and sulfur are included with heavy atoms.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

clusterset

Specifies a set of atoms to be used in comparisons of conformers in XCluster jobs where clustering is based on Atomic RMS differences.

Syntax:

`clusterset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. All atoms which match this description will be added to the list of XCluster comparison atoms.

clustertorsion

Specifies four atoms which define a torsion to be compared in XCluster jobs where clustering is based on Torsional RMS differences.

Syntax:

`clustertorsion` ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

The numbers of four atoms which define a torsion angle to be compared in XCluster jobs. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

color

Specifies a color or set color options.

Syntax:

`color` *alpha*=⟨ n ⟩ *blue*=⟨ n ⟩ *green*=⟨ n ⟩ *red*=⟨ n ⟩

Options:

alpha Alpha color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

blue Blue color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

green Green color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

red Red color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

coloratom

Set the color for a group of atoms defined by the ASL operand.

Syntax:

`coloratom` *cindex*=⟨ n ⟩ *color*=⟨ text ⟩ ⟨ ASL ⟩

Options:

cindex An integer which indicates color index which is to be used for the atoms

Valid values:	integers
Default value:	**2**
Minimum:	1
Maximum:	256

color A string which is the color name for atom coloring. Valid color names are described in the file $SCHRODINGER/maestro-vX.X/data/res/colors.res

Valid values:	text strings
Default value:	

Operands:
⟨ ASL ⟩
A string in the atom specification language. All atoms which match this specification will have their color changed to the current color.

coloratombonds

Set the color of all bonds, specified in ASL Change the color used to draw an on-screen bond.

Syntax:

`coloratombonds` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language.

coloratomrgb

Set the color for a group of atoms defined by the ASL operand.

Syntax:

`coloratomrgb` *blue*=⟨ n ⟩ *green*=⟨ n ⟩ *red*=⟨ n ⟩ ⟨ ASL ⟩

Options:

blue Blue color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

green Green color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

red Red color component.

Valid values:	integers
Default value:	**128**
Minimum:	0
Maximum:	255

Operands:
⟨ ASL ⟩
A string in the atom specification language. All atoms which match this specification will have their color changed to the current color.

colorbond

Set the color of a bond, specified by two atoms Change the color used to draw an on-screen bond.

Syntax:

`colorbond` *cindex*=⟨ n ⟩ *color*=⟨ text ⟩ *reset*=yes | no ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

cindex An integer which indicates color index which is to be used for bond coloring.

Valid values:	integers

Default value: **2**
Minimum: 1
Maximum: 256

color A string which is the color name for bond coloring. Valid color names are described in the file $SCHRODINGER/maestro-vX.X/data/res/colors.res

Valid values: text strings
Default value:

reset Resets the bond color to use atom colors.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The operands represent the numbers of the two atoms which define the bond to have its color changed.

colorscheme

Apply a predefined color scheme to the group of atoms defined by the ASL spec.

Syntax:

`colorscheme` *scheme*=⟨ text ⟩ ⟨ ASL ⟩

Options:

scheme The name of the current color scheme.

Valid values: text strings
Default value: **atype**

Operands:
⟨ ASL ⟩
A string in the atom specification language. All atoms which match this specification will have their color changed to match the currently selected color scheme.

combilibenum

Start the job of Combinatorial Library Enumeration.

Syntax:

`combilibenum`

combilibenumaddattachment

Adds an attachment to the core molecule using the given atoms.

Syntax:

`combilibenumaddattachment` *atom1*=⟨ n ⟩ *atom2*=⟨ n ⟩
⟨ attachment name ⟩

Options:

atom1 The atom number of the atom in the original core to set as an attachment point. This is the atom which will be kept.

Valid values:	integers
Default value:	**1**
Minimum:	1

atom2 The atom number of the atom in the original core to set as an attachment point. This is the atom which will be removed.

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:

⟨ attachment name ⟩

The name of the attachment.

combilibenumclearreagentfile

Clears the reagent file for the selected rows.

Syntax:

`combilibenumclearreagentfile`

combilibenumdeleteattachment

Deletes all of the selected attachments.

Syntax:

`combilibenumdeleteattachment`

combilibenumexportdefinition

Stores the current core molecule and attachments in a file.

Syntax:

`combilibenumexportdefinition` ⟨file name⟩

Operands:
⟨file name⟩
The name of the file to store the core definition in.

combilibenumimportdefinition

Reads a core molecule and attachments from the given file.

Syntax:

`combilibenumimportdefinition` ⟨file name⟩

Operands:
⟨file name⟩
The name of the file to read the core definition from.

combilibenumoptions

This command holds general options for Combinatorial Library Enumeration.

Syntax:

`combilibenumoptions` *untangle*=yes | no

Options:

untangle An option which allows post-combgen minimization (for library enumeration).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

combilibenumrefreshstructure

This function refreshes the structure in the Workspace from the current core structure in CombiGlide.

Syntax:

`combilibenumrefreshstructure` *viewcappedcore*=yes | no

Options:

viewcappedcore

An option which allows viewing of the minimally capped core, rather than the original core, in the Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

combilibenumrenameattachment

Renames the attachment in CombiLibEnum to the new name.

Syntax:

`combilibenumrenameattachment` *row*=⟨ n ⟩ ⟨ new name ⟩

Options:

row The row to rename.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:

⟨ new name ⟩

The new name for the attachment.

combilibenumselectextendtablerow

Extends the selection to this row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

`combilibenumselectextendtablerow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to extend the select to.

combilibenumselectonlytablerow

Selects only this row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

`combilibenumselectonlytablerow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to select only in the table row.

combilibenumselecttablerow

Selects the given row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

`combilibenumselecttablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the table.

combilibenumsetmolecule

Sets the core molecule for the CombiLibEnum to the molecule containing the given atom.

Syntax:

`combilibenumsetmolecule` *title*=⟨ text ⟩ ⟨ atom number ⟩

Options:

title	This option sets the title for the core molecule.	
	Valid values:	text strings
	Default value:	**core**

Operands:
⟨ atom number ⟩
The atom number of the molecule.

combilibenumsetreagentfile

Sets the reagent file for the selected rows.

Syntax:

`combilibenumsetreagentfile` ⟨ reagent name ⟩

Operands:
⟨ reagent name ⟩
The name of the reagent file.

combilibenumunselecttablerow

Unselects the given row in the attachments table in the CombiGlide Library Enumeration.

Syntax:

`combilibenumunselecttablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the table.

compareatom

Defines a single comparison atom to be used during a multiple minimization or conformational search. Comparison atoms are used to make comparisons in the process of determining if conformers are unique.

Syntax:

`compareatom` ⟨ atom ⟩

Operands:
⟨ atom ⟩
The number of an atom which is to added to the list of comparison atoms during a conformational search.

compareset

Specifies a set of atoms to be used in comparisons of conformers during a multiple minimization or conformational search.

Syntax:

`compareset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. All atoms which match this description will be added to the list of comparison atoms.

confelim

This keyword is used to set various options associated with starting Redundant Conformer Elimination jobs from Maestro.

Syntax:

`confelim` *compare_in_place*=yes | no *energy_source*=none | jaguar | mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs | oplsaa | opls2005 | opls2008 *incorporate*=append | replace | ignore | appendungrouped | workspace

Options:

compare_in_place
: A boolean which controls whether ConfElim will compare structures without first doing a superposition.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

energy_source
: Source of energy for the comparisons

  Valid values: none
  jaguar
  mm2*
  mm3*
  amber*
  opls*
  amber94
  mmff
  mmffs
  oplsaa
  opls2005
  opls2008
  Default value: **opls2005**

incorporate
: How the results are to be incorporated into the project. This can be done with replacement of the existing entries or by appending as new entries to the project or by ignoring the final results.

  Valid values: append
  replace
  ignore
  appendungrouped
  workspace
  Default value: **append**

confelimstart

Start a Redundant Conformer Elimination job with the current settings.

Syntax:

`confelimstart`

confelimwrite

Write a Redundant Conformer Elimination input file with the current settings.

Syntax:

`confelimwrite`

confgenltsearch

Defines settings for ConfGen conformational searching in MacroModel.

Syntax:

`confgenltsearch` *amidebonds*=vary | retain | trans
compareatoms=none | heavy | heavy_polar_h
distinguishenantiomers=yes | no *eliminate*=atom_deviation | rmsd *limitsave*=yes | no *max_rmsd*=⟨ x ⟩ *maxdist*=⟨ x ⟩ *maxringconf*=⟨ n ⟩ *maxtorsdiff*=⟨ x ⟩ *numsave*=⟨ n ⟩ *numsteps*=⟨ n ⟩ *samplerings*=yes | no *searchmode*=standard | rapid | complete | thorough *usenumsteps*=⟨ n ⟩ *usesearchmoves*=yes | no *window*=⟨ x ⟩

Options:

amidebonds
This determines whether to vary amide bond conformation, retain original amide bond conformation, or set amide bond conformation to trans.

Valid values:	vary retain trans
Default value:	**vary**

compareatoms
This determines which kinds of comparison atoms are automatically identified for judging redundant conformers.

Valid values: none
heavy
heavy_polar_h
Default value: **heavy_polar_h**

distinguishenantiomers
A boolean for whether to retain different enantiomers. If true, an additional opcode, NANT, is written to the .com file.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

eliminate The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.

Valid values: atom_deviation
rmsd
Default value: **rmsd**

limitsave A boolean which controls whether number of structures saved is limited by numsave option (if true) or by numsteps (if false).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

max_rmsd Maximum RMSD for considering two structures equal.

Valid values: reals
Default value: **1**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **1.5**
Minimum: 0.0

maxringconf
The maximum number of ring conformations generated by the search.

Valid values: integers
Default value: **16**
Minimum: 1

maxtorsdiff
Maximum torsional angle difference between polar hydrogens in equal structures.

Valid values: reals
Default value: **60**

Minimum: 0.0
Maximum: 180.0

numsave The number of structures that will be saved at the end of each search.

Valid values: integers
Default value: **1000**
Minimum: 0

numsteps An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values: integers
Default value: **1000**
Minimum: 0

samplerings

A boolean which controls whether to sample rings.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

searchmode

This determines whether rapid (standard) or thorough (complete) search will be used.

Valid values: standard
rapid
complete
thorough
Default value: **standard**

usenumsteps

An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values: integers
Default value: **5**
Minimum: 1

usesearchmoves

A boolean which controls whether number of search moves is limited by maxsearch moves option (if true).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

window The energy window (in kJ/mol) within which structures will be saved.

Valid values: reals
Default value: **100**
Minimum: 0.0

confgenmini

Used to set values associated with a MacroModel energy minimization for Ligand Torsional Search (ConfGen)

Syntax:

`confgenmini` *converge*=nothing | energy | gradient | movement
method=sd | prcg | osvm | fmnr | tncg | lbfgs | optimal
postmaxiter=⟨ n ⟩ *premaxiter*=⟨ n ⟩ *threshold*=⟨ x ⟩

Options:

converge This option determines which convergence criterion will be used during an energy minimization.

Valid values:	nothing energy gradient movement
Default value:	**gradient**

method This option determines which minimization method will be used.

Valid values:	sd prcg osvm fmnr tncg lbfgs optimal
Default value:	**tncg**

postmaxiter

This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values:	integers
Default value:	**0**
Minimum:	0
Maximum:	9999999

premaxiter

This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values:	integers
Default value:	**100**
Minimum:	0
Maximum:	9999999

threshold This option determines what the convergence threshold will be.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0

confgenpotential

Set various options associated with the definition of the potential energy to be used in a MacroModel job.

Syntax:

`confgenpotential` *cele*=⟨ x ⟩ *charges*=force_field | structure_file *chnd*=⟨ x ⟩ *cutoff*=normal | extended | user_defined | none *cvdw*=⟨ x ⟩ *dielectric*=⟨ x ⟩ *electrostatics*=field_field | constant | distance_dependant *field*=mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs | oplsaa | opls2005 | opls2008 *solvent*=none | water | chcl3 | octanol *suppresshbond*=yes | no

Options:

cele This option determines what cutoff will be used for the electrostatic part of the energy calculation.

Valid values:	reals
Default value:	**12**
Minimum:	0.0
Maximum:	99999.0

charges This option determines where the charges to be used in the energy calculation will come from.

Valid values:	force_field structure_file
Default value:	**force_field**

chnd This option determines what cutoff will be used for the hydrogen bond part of the energy calculation.

Valid values:	reals
Default value:	**4**
Minimum:	0.0
Maximum:	99999.0

cutoff This option determines what type of non-bonded cutoff will be used in the energy calculation.

Valid values: normal
extended
user_defined
none
Default value: **normal**

cvdw This option determines what VDW cutoff will be used in the energy calculation.

Valid values: reals
Default value: **7**
Minimum: 0.0
Maximum: 99999.0

dielectric The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values: reals
Default value: **4**
Minimum: 0.9999999999

electrostatics
The electrostatic treatment to be used in the energy calculation.

Valid values: field_field
constant
distance_dependant
Default value: **distance_dependant**

field The force field to be used for the energy calculation.

Valid values: mm2*
mm3*
amber*
opls*
amber94
mmff
mmffs
oplsaa
opls2005
opls2008
Default value: **opls2005**

solvent The solvent model to be used for the energy calculation

Valid values: none
water
chcl3
octanol
Default value: **none**

suppresshbond
: A boolean which controls whether to suppress hydrogen bonding electrostatics.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

confgenreadpotential

Read potential settings from a command file.

Syntax:

`confgenreadpotential` ⟨file_name⟩

Operands:

⟨file_name⟩

The name of the file from which the potential settings will be read. If no name is specified, the default settings will be used.

confgenstart

Start a Conformer Generation (Ligand Torsion Search) job with the current settings.

Syntax:

`confgenstart`

confgenstd

Launches a Standard ConfGen job.

Syntax:

`confgenstd`

confgenstdsettings

Used to set values associated with a Standard ConfGen job.

Syntax:

`confgenstdsettings` *minimize_input*=yes | no
minimize_output=yes | no *strategy*=very_fast | fast | intermediate | comprehensive | fast_cf | phase_fast | phase_quality *structure_file*=⟨ text ⟩
structure_source=selected_entries | workspace | file

Options:

minimize_input
Indicates whether or not the input structures will be minimized before the search.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

minimize_output
Indicates whether or not the output conformers will be minimized.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

strategy This option determines which search strategy will be used.

Valid values: very_fast
fast
intermediate
comprehensive
fast_cf
phase_fast
phase_quality
Default value: **fast**

structure_file
If the source of structures is set to file, then this option contains the file to use.

Valid values: text strings
Default value:

structure_source
Whether to use the selected entries in the current project, or a specified file with multiple structures as structure input for the job.

Valid values:	selected_entries workspace file
Default value:	**selected_entries**

confgenstdwrite

Write out the Standard ConfGen job files.

Syntax:

```
confgenstdwrite
```

confgenwrite

Write a Conformer Generation (Ligand Torsion Search) input file with the current settings.

Syntax:

```
confgenwrite
```

confsearch

Defines settings for conformational searching in MacroModel.

Syntax:

`confsearch` *distinguishenatiomers*=yes | no
eliminate=atom_deviation | rmsd *enableauto*=yes | no
max_distance_lmcs=⟨ x ⟩ *max_rmsd*=⟨ x ⟩ *maxdist*=⟨ x ⟩
method=mcmm | serial_mcmm | summ | mixed_lmcs |
serial_mcmm_lmcs | pure_lmcs | serial_lmcs | large_lmcs |
mixed_large_lmcs *min_distance_lmcs*=⟨ x ⟩
multiligand=yes | no *numsteps*=⟨ n ⟩ *numstructures*=⟨ n ⟩
probability_tors_lmcs=⟨ x ⟩ *searchvariable*=ringclose | torsrot |
moltrans | compatom | chiralatom | distcheck | torscheck |
ligbond *torsionsampling*=restricted | intermediate | enhanced
| extended *usenumsteps*=⟨ n ⟩ *usesteps*=yes | no *window*=⟨ x ⟩

Options:

distinguishenatiomers
This option determines to conside conformers more than once

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

eliminate The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.

Valid values:	atom_deviation rmsd
Default value:	**atom_deviation**

enableauto Toggle whether or not to use AUTO setup for processing multiple structures.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

max_distance_lmcs
The maximum possible distance for the fastest moving atom in each LMCS move.

Valid values:	reals
Default value:	**6**
Minimum:	0.0

max_rmsd Maximum RMSD for considering two structures equal.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0

maxdist Maximum distance between atoms in equal structures.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0

method This determines which method will be used to perform the conformational search.

Valid values:	mcmm serial_mcmm summ mixed_lmcs serial_mcmm_lmcs pure_lmcs serial_lmcs large_lmcs mixed_large_lmcs
Default value:	**serial_mcmm_lmcs**

min_distance_lmcs
The minimum possible distance for the fastest moving atom in each LMCS move.

Valid values:	reals
Default value:	**3**
Minimum:	0.0

multiligand
An option determines to use multiple ligands or not.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

numsteps An option which sets the number of steps which will be performed during the conformational search.

Valid values:	integers
Default value:	**1000**
Minimum:	0

numstructures
The number of structures that will be saved at the end of each search.

Valid values:	integers
Default value:	**0**
Minimum:	0

probability_tors_lmcs
The probability that a TORS/MOLS move will be made during an LMCS search.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

searchvariable
Controls which search variable will be used.

Valid values:	ringclose torsrot moltrans compatom chiralatom distcheck torscheck ligbond
Default value:	**ringclose**

torsionsampling
Controls how sampling of torsions is done during the automatic setup for torsional sampling methods.

Valid values:	restricted intermediate enhanced extended
Default value:	**intermediate**

usenumsteps
An option which sets the number of steps which will be performed during the conformational search.

Valid values:	integers
Default value:	**100**
Minimum:	1

usesteps An option which sets the number of steps which will be performed during the conformational search.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

window The energy window (in kJ/mol) within which structures will be saved.

Valid values:	reals
Default value:	**21**
Minimum:	0.0

connect

Connect atom pairs.

Syntax:

`connect`

connectfuseatom

Define an atom pair for which will be connected in a subsequent connect operation.

Syntax:

`connectfuseatom` ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
Two atom numbers which represent an atom pair to be connected by a subsequent connect or fuse command. The two atoms must be from different molecules and all connectfuseatom commands must specify molecules in the same order.

constrainedangle

Specifies a angle between three atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

`constrainedangle` *angle*=⟨ x ⟩ *constant*=⟨ x ⟩ *select*=yes | no
width=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

Options:

angle The angle at which the atoms are to be constrained.
Valid values: reals
Default value: **-1**

constant The force constant for the harmonic constraint to be applied to the angle.
Valid values: reals
Default value: **100**
Minimum: 0.0

select Selection state of the model.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

width The half-width for a flat-bottomed constraint.
Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩
Three atoms which are to have the angle between them constrained by a harmonic constraint. The three atoms are not necessarily connected. Note that specifying a-b-c will be treated by the program as the same as specifying c-b-a.

constrainedatom

Specifies a single atom to be constrained at its current positions during a MacroModel energy calculation. The atom will be constrained by the use of a harmonic constraint.

Syntax:

`constrainedatom` *constant*=⟨ x ⟩ *frozen*=yes | no *select*=yes | no *width*=⟨ x ⟩ ⟨ atom_number ⟩

Options:

constant The force constant (in kJ/mol/Angs) for the harmonic constraint.
Valid values: reals
Default value: **0**

frozen The constraint atom is frozen or not
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

select Selection state of the model.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

width The half-width of a flat bottomed restraint.
Valid values: reals
Default value: **0**
Minimum: 0

Operands:
⟨ atom_number ⟩
The number of the atom to which a constraint is to be applied.

constraineddist

Specifies a distance between two atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

constraineddist *constant*=⟨ x ⟩ *distance*=⟨ x ⟩
remove_nb=yes | no *select*=yes | no *width*=⟨ x ⟩ ⟨ atom1 ⟩
⟨ atom2 ⟩

Options:

constant The force constant for the harmonic constraint to be applied to the distance.

Valid values:	reals
Default value:	**100**
Minimum:	0.0

distance The distance at which the atoms are to be constrained.

Valid values:	reals
Default value:	**-1**

remove_nb A boolean option which controls whether the non-bonded interaction (if any) between the two specified atoms will be removed.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

select Selection state of the model.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

width The half-width for a flat-bottomed constraint.

Valid values:	reals
Default value:	**0**
Minimum:	0.0

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩

Two atoms which are to have the distance between them constrained by a harmonic constraint Note that specifying a-b will be treated by the program as the same as specifying b-a.

constrainedset

Specifies a set of atoms to be constrained at their current positions. The atoms will be constrained by the use of a harmonic constraint.

Syntax:

`constrainedset` *constant*=⟨ x ⟩ *width*=⟨ x ⟩ ⟨ ASL ⟩

Options:

constant The force constant (in kJ/mol/Angs) for the harmonic constraint.

Valid values:	reals
Default value:	**100**
Minimum:	0.0

width The half-width of a flat bottomed restraint

Valid values:	reals
Default value:	**0**
Minimum:	0

Operands:

⟨ ASL ⟩

The operand must be a valid string in the atom specification language. Any atoms which match will have constraints generated for them.

constrainedtorsion

Specifies a torsion between four atoms which is to be constrained by a harmonic constraint during a MacroModel calculation.

Syntax:

`constrainedtorsion` *constant*=⟨ x ⟩ *select*=yes | no *torsion*=⟨ x ⟩
width=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

constant The force constant for the harmonic constraint to be applied to the torsion.

Valid values: reals
Default value: **100**
Minimum: 0.0

select Selection state of the model.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

torsion The torsion at which the atoms are to be constrained.

Valid values: reals
Default value: **500**

width The half-width for a flat-bottomed constraint.

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Four atoms which are to have the angle between then constrained by a harmonic constraint. Note that specifying a-b-c-d is treated by the program the same as specifying d-c-b-a.

contactcriteria

Specify the criteria for calculating good, bad, ugly contacts. There is no upper-bound to C. C = {dist between the two atoms} divided by {vdW radius first atom + vdW radius of second atom}.

Syntax:

`contactcriteria` *bad*=⟨ x ⟩ *display*=yes | no *displaybad*=yes | no *displaygood*=yes | no *displayugly*=yes | no *exclude14interactions*=yes | no *excludehbond*=yes | no *good*=⟨ x ⟩ *ugly*=⟨ x ⟩

Options:

bad Criteria for bad contact

Valid values: reals
Default value: **0.89**
Minimum: 0.0

display This option determines whether contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displaybad If contact markers are shown, this option determines whether the bad contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displaygood
If contact markers are shown, this option determines whether the good contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayugly
If contact markers are shown, this option determines whether the ugly contact markers will be displayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

exclude14interactions
If this option is true, atoms that have 1,4 interactions should not be considered to have bad or ugly contacts.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

excludehbond
If this option is true, atoms that are H-bonded should not be considered to have bad or ugly contacts.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

good	Criteria for good contact	
	Valid values:	reals
	Default value:	**1.3**
	Minimum:	0.0
ugly	Criteria for ugly contact	
	Valid values:	reals
	Default value:	**0.75**
	Minimum:	0.0

contactset1

Specify the first set of atoms used in determining good, bad, and ugly contacts.

Syntax:

`contactset1` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. Typical usage is to define contactset1 and contactset2. This set, contactset1, defines the "from" atoms. The contactset2 atoms define the "to" atoms. Contacts are calculated between these two sets. That is, the contacts are inter-set contacts. No intra-set contacts are calculated. If contactset2's ASL string is empty, then contacts are calculated for all atoms in contactset1.

contactset2

Specify the second set of atoms used in determining good, bad, and ugly contacts.

Syntax:

`contactset2` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. Typical usage is to define contactset1 and contactset2. This set, contactset2, defines the "to" atoms. The

contactset1 atoms define the "from" atoms. Contacts are calculated between these two sets. That is, the contacts are inter-set contacts. No intra-set contacts are calculated. If contactset2's ASL string is empty, then contacts are calculated for all atoms in contactset1.

coordinatescan

Used to set a distance or angle or dihedral coordinate to be scanned. The four operands are the atom numbers defining the dihedral to be driven. A maximum of two angles can be driven. The four operands are the atom numbers defining the coordinate. Depending on the type of scan coordiante set in MM_SCAN_MODE: For distance the third and fourth atom numbers will be 0 and for angle the fourth one will be 0. In case of dihedral, all the four atom numbers are used. A maximum of two coordiantes can be driven.

Syntax:

`coordinatescan` *finishangle*=⟨ x ⟩ *finishdihedral*=⟨ x ⟩
finishdist=⟨ x ⟩ *incrementangle*=⟨ x ⟩ *incrementdihedral*=⟨ x ⟩
incrementdist=⟨ x ⟩ *startangle*=⟨ x ⟩ *startdihedral*=⟨ x ⟩
startdist=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

finishangle
Specifies the finishing angle for the angle scan.
Valid values: reals
Default value: **360**

finishdihedral
Specifies the finishing angle for the dihedral scan.
Valid values: reals
Default value: **360**

finishdist Specifies the finishing distance for the distance scan.
Valid values: reals
Default value: **10**

incrementangle
Specifies the angle increment for the angle scan.
Valid values: reals
Default value: **30**

incrementdihedral
Specifies the angle increment for the dihedral scan.

Valid values: reals
Default value: **30**

incrementdist
Specifies the distance increment for the distance scan.

Valid values: reals
Default value: **1**

startangle Specifies the starting angle for the angle scan.

Valid values: reals
Default value: **0**

startdihedral
Specifies the starting angle for the dihedral scan.

Valid values: reals
Default value: **0**

startdist Specifies the starting distance for the distance scan.

Valid values: reals
Default value: **1**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩
The four operands are treated as atom numbers which define the distance/angle/dihedral to be driven. The MM_SCAN_MODE option will determine the type of scan.

coupling

Specifies a pair of atoms to have their coupling measured and displayed.

Syntax:

`coupling` *xoffset*=⟨ x ⟩ *yoffset*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

xoffset Specifies the X offset in Angstroms. Any coupling created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created couplings.

Valid values: reals
Default value: **0**

yoffset Specifies the Y offset in Angstroms. Any coupling created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created couplings.

Valid values: reals
Default value: **0**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The two hydrogen atoms between which the coupling is to be measured. Note that the specifying a-b is the same as specifying b-a.

createcustompropertygroup

This command creates a custom property group for the given space separated property names.

Syntax:

`createcustompropertygroup` ⟨ propertygroupname ⟩
⟨ propertynames ⟩

Operands:
⟨ propertygroupname ⟩ ⟨ propertynames ⟩
The name of the custom property group. The space separated property names use to create the custom property group.

createlibrary

Start the creation of the library in the Analyse Library step of CombiGlide.

Syntax:

`createlibrary`

createpropsubset

This command creates a property subset for the given table using the given property names.

Syntax:

`createpropsubset` ⟨table⟩ ⟨propertynames⟩

Operands:
⟨table⟩ ⟨propertynames⟩
The number of the table to create the subset for. If the table operand is missing, no resize will be done. The names of the properties to use to create the subset.

createsubset

Creates a subset in the project table using the currently selected entries. This function also switches to subset view.

Syntax:

`createsubset`

defaultfc

Requires a single operand which is the default force constant to be used for the constrainedatom command. The default value is used only when no value has been specified by the constant= option of the "constrainedatom" command.

Syntax:

`defaultfc` ⟨default_force_constant_value⟩

Operands:
⟨default_force_constant_value⟩
The value of the default force constant in kJ/mol/Angstrom

delete

Delete a named object. The object type is the same as the command which is used to create that type of object. For example to delete a set named "set1" use: delete set set1. All instances of any type of object can be deleted with

the "all" name. There are three special objects: "atom" and "bond" for deleting atoms and bonds respectively
For example: delete set set1
delete constrainedatom 1
delete set all delete res. 1-5 delete bond 1 2

Syntax:

`delete` *includeterminal*=yes | no ⟨object_type⟩
⟨object_name⟩|all|⟨ASL⟩|⟨bond⟩

Options:

includeterminal

This option is used to control the behavior when deleting atoms and bonds. If this is "true" then all terminally attached atoms will also be deleted when deleting bonds and atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨object_type⟩ ⟨object_name⟩|all|⟨ASL⟩|⟨bond⟩
The first operand determines what type of object is to be deleted. This can be the name of any of the named objects in Maestro (sets, filters, holds etc.) or can be "atom" or "bond". The subject operands depend on the first operand. If the first operand names an object then the second operand must be the name which was specified when that object was created. If the first operand is "atom" then the second operand must be a valid string in the atom specification language and all atoms which match that string will be deleted. Finally if the first operand is "bond" then the second and third operands are expected to be the numbers of the two atoms which define that bond.

deletecustompropertygroup

This command deletes a custom property group.

Syntax:

`deletecustompropertygroup` ⟨propertygroupname⟩

Operands:
⟨propertygroupname⟩

The name of the custom property group.

deleteemptyentrygroups

Deletes all the empty entry groups.

Syntax:

`deleteemptyentrygroups`

deleteproperty

This is a standard alias for **propertydelete** (see [propertydelete], page 604).

deleteselectedproperties

Deletes the selected properties from the atom-property table and the Workspace.

Syntax:

`deleteselectedproperties`

deletezeroorderbonds

Delete zero order bonds connected to atoms in the specified set of atoms. The zero order bond will be deleted if it connects any atom within the set.

Syntax:

`deletezeroorderbonds` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The set of atoms which will have zero order bonds deleted.

densityblobupdate

Update electron density blobs for current density map.

Syntax:

`densityblobupdate` *minvol*=⟨ x ⟩

Options:

minvol Threshold for displaying density blobs. Only blobs whose volumes are greater than or equal to this number will be displayed.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0

dialogoptions

Sets options for dialog boxes.

Syntax:

`dialogoptions` *show*=yes | no

Options:

show If this option is set to false, then dialog boxes will not be displayed.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

dihedral

Specifies a quartet of atoms to have their dihedral angle measured and displayed.

Syntax:

`dihedral` *xoffset*=⟨ x ⟩ *yoffset*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

xoffset Specifies the X offset in Angstroms. Any dihedral created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created dihedrals.

Valid values: reals
Default value: **0**

yoffset Specifies the Y offset in Angstroms. Any dihedral created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created dihedrals.

Valid values: reals
Default value: **0**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩
The four atoms between which the dihedral angle is to be measured. Note that the specifying a-b-c-d is the same as specifying d-c-b-a.

dihedraldrive

Used to set a dihedral angle to be driven. The four operands are the atom numbers defining the dihedral to be driven. A maximum of two angles can be driven.

Syntax:

`dihedraldrive` *finish*=⟨ x ⟩ *increment*=⟨ x ⟩ *start*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

finish Specifies the finishing angle for the dihedral drive.

Valid values: reals
Default value: **360**

increment Specifies the angle increment for the dihedral drive.

Valid values: reals
Default value: **30**

start Specifies the starting angle for the dihedral drive.
Valid values: reals
Default value: **0**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩
The four operands are treated as atom numbers which define the dihedral to be driven.

dipolemoment

Set dipole moment options.

Syntax:

`dipolemoment` *ambient*=⟨ x ⟩ *arrowwidth*=⟨ x ⟩ *chargeblue*=⟨ x ⟩ *chargegreen*=⟨ x ⟩ *chargered*=⟨ x ⟩ *diffuse*=⟨ x ⟩ *diskblue*=⟨ x ⟩ *diskgreen*=⟨ x ⟩ *diskoffset*=⟨ x ⟩ *diskred*=⟨ x ⟩ *diskwidth*=⟨ x ⟩ *display*=yes | no *emission*=⟨ x ⟩ *frompositive*=yes | no *jagusrblue*=⟨ x ⟩ *jagusrgreen*=⟨ x ⟩ *jagusrred*=⟨ x ⟩ *negativeblue*=⟨ x ⟩ *negativegreen*=⟨ x ⟩ *negativered*=⟨ x ⟩ *positiveblue*=⟨ x ⟩ *positivegreen*=⟨ x ⟩ *positivered*=⟨ x ⟩ *ringblue*=⟨ x ⟩ *ringgreen*=⟨ x ⟩ *ringred*=⟨ x ⟩ *ringustep*=⟨ n ⟩ *ringvstep*=⟨ n ⟩ *ringwidth*=⟨ x ⟩ *scale*=⟨ x ⟩ *shaftwidth*=⟨ x ⟩ *shininess*=⟨ x ⟩ *showdisk*=yes | no *showpositive*=yes | no *specular*=⟨ x ⟩ *transparency*=⟨ x ⟩ *ustep*=⟨ n ⟩ *vstep*=⟨ n ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.
Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

arrowwidth
The width of dipole moment arrow.
Valid values: reals
Default value: **0.2**
Minimum: 0.0001

chargeblue The blue color component for dipole moments defined by partial atomic charges.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

chargegreen

The green color component for dipole moments defined by partial atomic charges.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

chargered The red color component for dipole moments defined by partial atomic charges.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

diskblue The blue color component of disk.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

diskgreen The green color component of disk.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

diskoffset The offset (percentage) of distance between the dipole moment disk and the positive end.

Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 1.0

diskred The red color component of disk.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

diskwidth The width of dipole moment disk.

Valid values: reals
Default value: **0.3**
Minimum: 0.0001

display Whether or not display dipole moment in workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.15**
Minimum: 0.0
Maximum: 1.0

frompositive

Set to true if the arrow is drawn from positive to negative.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

jagusrblue The blue color component for dipole moments defined by Jaguar properties.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

jagusrgreen

The green color component for dipole moments defined by Jaguar properties.

Valid values: reals
Default value: **0.7**
Minimum: 0.0
Maximum: 1.0

jagusrred The red color component for dipole moments defined by Jaguar properties.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

negativeblue

The blue color component of dipole moment markers for negative region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

negativegreen

The green color component of dipole moment markers for negative region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

negativered

The red color component of dipole moment markers for negative region.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positiveblue

The blue color component of dipole moment markers for positive region.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positivegreen

The green color component of dipole moment markers for positive region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

positivered

The red color component of dipole moment markers for positive region.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

ringblue The blue color component of the ring.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

ringgreen The green color component of the ring.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

ringred The red color component of the ring.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

ringustep The U step domain tolerance for the ring.

Valid values: integers
Default value: **5**
Minimum: 2

ringvstep The V step domain tolerance for the ring.

Valid values: integers
Default value: **3**
Minimum: 2

ringwidth The width of the ring around the disk.

Valid values: reals
Default value: **0.05**
Minimum: 0.0001

scale The scale to convert unit from Debye to Angstrong for the length of dipole moment arrow.

Valid values: reals
Default value: **1**
Minimum: 0.0001

shaftwidth The width of dipole moment shaft.

Valid values: reals
Default value: **0.1**
Minimum: 0.0001

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**

Minimum: 0.0
Maximum: 128.0

showdisk Set to true if the disk for positive sign is shown.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showpositive
Set to true if positive and negative colors are used.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.7**
Minimum: 0.0
Maximum: 1.0

transparency
The transparency of dipole moment markers.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

ustep The U step domain tolerance for the cylinder.

Valid values: integers
Default value: **3**
Minimum: 2

vstep The V step domain tolerance for the cylinder.

Valid values: integers
Default value: **5**
Minimum: 2

displayatom

Display all atoms in the set described by the ASL operand.

Syntax:

`displayatom` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be displayed.

displayonlyatom

Display only the set described by the ASL operand.

Syntax:

`displayonlyatom` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be displayed.

displayopt

Set properties for the display, such as the background color.

Syntax:

`displayopt` *adjustclip*=yes | no *aligndepth*=⟨ x ⟩ *angledependenttransparency*=yes | no *bgcindex*=⟨ n ⟩ *bgcolor*=⟨ text ⟩ *eye_sepf*=⟨ x ⟩ *fog*=auto | on | true | yes | off | false | no *fogatomlabels*=yes | no *fogcutoff*=⟨ n ⟩ *fogdensity*=⟨ x ⟩ *fogendz*=⟨ x ⟩ *fogstartz*=⟨ x ⟩ *fogtype*=linear | exp | exp2 | automatic *fsaa*=yes | no *invertstereo*=yes | no *maximize*=yes | no *opengl_level*=⟨ text ⟩ *perspective*=yes | no *perspectivescale*=⟨ x ⟩ *rubber_banding*=xor | main *sepf*=⟨ x ⟩ *sizef*=⟨ x ⟩ *start_str*=⟨ text ⟩ *stereo*=yes | no *stereomethod*=hardware | crosseyed | walleyed | fullscreen | interlaced *stop_str*=⟨ text ⟩

Options:

adjustclip A boolean option which determines whether to adjust clipping to preserve stereo depth.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

aligndepth Relative z-depth (1 for back clipping plane, 0 for front clipping plane) at which stereo images appear to converge. Moves region inside clipping planes in or out of the screen.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

angledependenttransparency
A bool which determines whether transparent objects should use angle dependent transparency or not

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

bgcindex An integer which indicates color index that may be used to set the background color. Since the background color can now be an arbitrary RGB color (via MM_OPTION_BACKGROUND_COLOR), and this MM_OPTION_BACKGROUND_CINDEX value is only the nearest indexed color, not the actual color, this option is deprecated.

Valid values: integers
Default value: **1**
Minimum: 1
Maximum: 256

bgcolor A string which is the color name or 6-hexadecimal-digit RGB string for background color. Valid color names are described in the file $SCHRODINGER/mmshare-vX.X/data/colors.res

Valid values: text strings
Default value: **000000**

eye_sepf Eye separation for stereo as fraction of normal. Actual separation may be made smaller to limit maximum stereo displarity to about 1 inch at clipping planes.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 2.0

fog Set fog to be displayed, to not be displayed, or to toggle automatically based on the number of atoms in the workspace.

Valid values: auto
on
true
yes
off
false
no
Default value: **auto**

fogatomlabels
A Boolean option which determines whether to display fog on atom labels (only displayed if fog is already on in general).
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fogcutoff Specifies the minimum number of atoms in the workspace for which fog will be displayed, when fogging is set to auto.
Valid values: integers
Default value: **40**
Minimum: 0

fogdensity Factor controlling optical density of fog.
Valid values: reals
Default value: **1.5**
Minimum: 0.0
Maximum: 4.0

fogendz Factor controlling end of linear fog ramp.
Valid values: reals
Default value: **1.25**
Minimum: 1.0
Maximum: 2.0

fogstartz Factor controlling start of linear fog ramp.
Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 0.999

fogtype Specifies method used for calculating fog.
Valid values: linear
exp
exp2
automatic
Default value: **automatic**

fsaa This option is used to toggle full scene anti aliasing on the systems that support it.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

invertstereo

A boolean option which determines whether to swap stereo images

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

maximize A boolean option which determines whether to maximize the display window (on) or restore to the original window size (off)

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

opengl_level

Sets an OpenGL level.

Valid values: text strings
Default value:

perspective

A Boolean option which when on enables perspective and when off uses orthogonal projection in the Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

perspectivescale

Ratio between the actual size and the projected size of an object placed 1 foot behind display screen. This has a corresponding viewing distance, view_dist = (1 ft) / (perspectivescale - 1). For a known viewing distance, perspectivescale = 1 + (1 ft / view_dist). Therefore, for a normal viewing distance of 2 feet, perspectivescale = 1 + (1 ft / 2 ft) = 1.5.

Valid values: reals
Default value: **1.35**
Minimum: 1.0
Maximum: 4.0

rubber_banding

Specifies method used for drawing temporary graphics to track cursor motion (such as lasso rectangle or rubber band line). Valid values are "xor" or "main". These draw to the front buffer in XOR mode, or draw to the back buffer at the end of main window drawing (before the background and foreground GL buffers are swapped).

Valid values: xor
main
Default value: **xor**

sepf Image separation as fraction of maximum horizontal separation (window width minus image width)

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

sizef Image size as fraction of available width

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

start_str The system command to start hardware stereo.

Valid values:	text strings
Default value:	

stereo A boolean option which determines whether to display in stereo

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

stereomethod
Specifies method used for stereo viewing

Valid values:	hardware crosseyed walleyed fullscreen interlaced
Default value:	**hardware**

stop_str The system command to stop hardware stereo.

Valid values:	text strings
Default value:	

distance

Specifies a pair of atoms to have their distance measured and displayed.

Syntax:

`distance` *surfacename1*=⟨ text ⟩ *surfacename2*=⟨ text ⟩
surfacex1=⟨ x ⟩ *surfacex2*=⟨ x ⟩ *surfacey1*=⟨ x ⟩ *surfacey2*=⟨ x ⟩
surfacez1=⟨ x ⟩ *surfacez2*=⟨ x ⟩ *xoffset*=⟨ x ⟩ *yoffset*=⟨ x ⟩
⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

surfacename1
Specifies the name of surface on which atom1 sits.
Valid values: text strings
Default value:

surfacename2
Specifies the name of surface on which atom1 sits.
Valid values: text strings
Default value:

surfacex1 Specifies the X coordinate for atom1 on surface.
Valid values: reals
Default value: **0**

surfacex2 Specifies the X coordinate for atom2 on surface.
Valid values: reals
Default value: **0**

surfacey1 Specifies the Y coordinate for atom1 on surface.
Valid values: reals
Default value: **0**

surfacey2 Specifies the Y coordinate for atom2 on surface.
Valid values: reals
Default value: **0**

surfacez1 Specifies the Z coordinate for atom1 on surface.
Valid values: reals
Default value: **0**

surfacez2 Specifies the Z coordinate for atom2 on surface.
Valid values: reals
Default value: **0**

xoffset Specifies the X offset in Angstroms. Any distance created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created distances.
Valid values: reals
Default value: **0**

yoffset Specifies the Y offset in Angstroms. Any distance created after this value is set will use this new value. Can be applied to an existing measurement by respecifying the measurement. Does not affect any already created distances.

Valid values: reals
Default value: **0**

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

The two atoms between which the distance is to be measured. Note that the specifying a-b is the same as specifying b-a.

distancecheck

Specifies two atoms which define a distance to be checked during a conformational search.

Syntax:

`distancecheck` *allowed*=⟨ x ⟩ *maximum*=⟨ x ⟩ *minimum*=⟨ x ⟩ *value*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

allowed The maximum allowed variation from the "target" value during the distance check.

Valid values: reals
Default value: **0**
Minimum: 0.0

maximum The maximum distance for the distance check (Angstroms)

Valid values: reals
Default value: **0**
Minimum: 0.0

minimum The minimum distance for the distance check (Angstroms)

Valid values: reals
Default value: **0**
Minimum: 0.0

value The target vaue for the distance check (Angstroms)

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The numbers of two atoms which are to be considered are to have their distance checked during a conformational search. Note that specifying a-b is the same as specifying b-a.

dockgridbox

Settings for the grid boxes for docking.

Syntax:

`dockgridbox` *innersizex*=⟨ x ⟩ *innersizey*=⟨ x ⟩ *innersizez*=⟨ x ⟩
outersizex=⟨ x ⟩ *outersizey*=⟨ x ⟩ *outersizez*=⟨ x ⟩
show=yes | no *xcent*=⟨ x ⟩ *ycent*=⟨ x ⟩ *zcent*=⟨ x ⟩

Options:

innersizex The size of the inner box in X-coordinate.
Valid values: reals
Default value: **5**
Minimum: 0.0

innersizey The size of the inner box in Y-coordinate.
Valid values: reals
Default value: **5**
Minimum: 0.0

innersizez The size of the inner box in Z-coordinate.
Valid values: reals
Default value: **5**
Minimum: 0.0

outersizex The size of the outer box in X-coordinate.
Valid values: reals
Default value: **10**
Minimum: 0.0

outersizey The size of the outer box in Y-coordinate.
Valid values: reals
Default value: **10**
Minimum: 0.0

outersizez The size of the outer box in Z-coordinate.
Valid values: reals

Default value: **10**
Minimum: 0.0

show An option which determines whether to show grid boxes or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

xcent The X-coordinate of the box center.
Valid values: reals
Default value: **0**

ycent The Y-coordinate of the box center.
Valid values: reals
Default value: **0**

zcent The Z-coordinate of the box center.
Valid values: reals
Default value: **0**

doplanealignment

This command actually does the plane alignment.

Syntax:

`doplanealignment`

dynamics

Used to set options associated with MacroModel molecular or stochastic dynamics procedures.

Syntax:

`dynamics` *equilibration*=⟨ x ⟩ *method*=molecular | stochastic *minimize_sampled*=yes | no *sample*=⟨ n ⟩ *shake*=nothing | hydrogens | all *simulation*=⟨ x ⟩ *temperature*=⟨ x ⟩ *timestep*=⟨ x ⟩

Options:

equilibration
The time used for equilibration for the equilibration part of the simulation in ps.

Valid values: reals
Default value: **1**
Minimum: 0.0

method This option determines which dynamics method will be used.

Valid values: molecular
stochastic
Default value: **stochastic**

minimize_sampled
Minimize the sampled structure.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sample The number of structures to be sampled during the simulation.

Valid values: integers
Default value: **0**
Minimum: 0

shake This option determines how the SHAKE algorithm will be applied during the simulation.

Valid values: nothing
hydrogens
all
Default value: **nothing**

simulation
The total time to be used in the simulation.

Valid values: reals
Default value: **10**
Minimum: 0.0

temperature
The temperature at which the simulation is to be run in Kelvin.

Valid values: reals
Default value: **300**
Minimum: 0.0

timestep The timesetp which is to be used for the simulation in fs.

Valid values: reals
Default value: **1.5**
Minimum: 0.0

ecalc

This is a standard alias for **calcenergy** (see [calcenergy], page 53).

editatomproperty

Sets the given property for the given atom in the atom-property table to the new value.

Syntax:

`editatomproperty` *column*=⟨ n ⟩ *row*=⟨ n ⟩ ⟨ new value ⟩

Options:

column The column number in the atom-property table.

Valid values:	integers
Default value:	**2**
Minimum:	2

row The row number in the atom-property table. If set to -1, it means all selected rows.

Valid values:	integers
Default value:	**1**
Minimum:	-1

Operands:

⟨ new value ⟩

The operand is the new value for the atom property.

ejob

This keyword is used to set various options associated with running MacroModel jobs.

Syntax:

`ejob` *host*=⟨ text ⟩ *incorporate*=append | replace | ignore | appendungrouped | workspace *input_file*=⟨ text ⟩ *job*=⟨ text ⟩ *login*=⟨ text ⟩ *rce_structure_source*=selected_entries | workspace | file *structure_source*=selected_entries | workspace | file

Options:

host The name of the host for the MacroModel job.

Valid values: text strings
Default value:

incorporate

How the results are to be incorporated into the project. This can be done with replacement of the existing entries or by appending as new entries to the project or by ignoring the final results.

Valid values: append
replace
ignore
appendungrouped
workspace
Default value: **append**

input_file The name of the structure input file.

Valid values: text strings
Default value:

job The name for the MacroModel job.

Valid values: text strings
Default value: **mmodtmp**

login The login name under which a MacroModel will be run.

Valid values: text strings
Default value:

rce_structure_source

Whether to use the selected entries in the current project, or a specified file with multiple structures as structure input for the job.

Valid values: selected_entries
workspace
file
Default value: **selected_entries**

structure_source

Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values:	selected_entries workspace file
Default value:	**workspace**

elementlabels

Deprecated and has no effect. It used to set the element label options. Now it is always off.

Syntax:

`elementlabels` *labelscheme*=heterohydro | hetero | off

Options:

labelscheme

Deprecated and always set to off. We used to supported three options for displaying element labels: 1) on all non_carbon atoms, including hydrogen. 2) on all non-carbon atoms, except hydrogen (i.e., not on hydrogen) 3) off (no element labels displayed).

Valid values:	heterohydro hetero off
Default value:	**off**

embrace

Used to control the eBMrAcE feature of MacroModel.

Syntax:

`embrace` *energymode*=interaction | energydiff *inputfile*=⟨ text ⟩ *ligandsmode*=inputfile | selectedentries *ouput*=complexes | ligands | all *receptor*=⟨ text ⟩ *receptormode*=firstitem | entry

Options:

energymode

This option determines which energy mode will be used by eBMrAcE.

Valid values: interaction
energydiff
Default value: **energydiff**

inputfile The name of the Embrace ligands input file.

Valid values: text strings
Default value:

ligandsmode
This option determines which source of ligands will be used by eBMrAcE.

Valid values: inputfile
selectedentries
Default value: **inputfile**

ouput This option determines what structural output will be produced.

Valid values: complexes
ligands
all
Default value: **complexes**

receptor This option specifies which entry in the project is to be treated as the receptor.

Valid values: text strings
Default value:

receptormode
This option determines which kind of receptor will be used by eBMrAcE.

Valid values: firstitem
entry
Default value: **firstitem**

embracecsearch

Defines settings for Embrace conformational searching in MacroModel.

Syntax:

`embracecsearch` *eliminate*=atom_deviation | rmsd
max_distance_lmcs=⟨ x ⟩ *max_rmsd*=⟨ x ⟩ *maxdist*=⟨ x ⟩
method=embrace_mcmm | embrace_lmcs | embrace_mixed_lmcs
min_distance_lmcs=⟨ x ⟩ *numconfs*=⟨ n ⟩ *numsteps*=⟨ n ⟩
numstructures=⟨ n ⟩ *probability_tors_lmcs*=⟨ x ⟩ *window*=⟨ x ⟩

Options:

eliminate The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.

Valid values: atom_deviation
rmsd
Default value: **atom_deviation**

max_distance_lmcs
The maximum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals
Default value: **6**
Minimum: 0.0

max_rmsd Maximum RMSD for considering two structures equal.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **0.25**
Minimum: 0.0

method This determines which method will be used to perform the Embrace conformational search.

Valid values: embrace_mcmm
embrace_lmcs
embrace_mixed_lmcs
Default value: **embrace_mcmm**

min_distance_lmcs
The minimum possible distance for the fastest moving atom in each LMCS move.

Valid values: reals
Default value: **3**
Minimum: 0.0

numconfs The number of input conformations available to seed each search.

Valid values: integers

Default value: **1**
Minimum: 0

numsteps An option which sets the number of steps which will be performed during the Embrace conformational search.

Valid values: integers
Default value: **100**
Minimum: 0

numstructures
The number of structures that will be saved at the end of each search.

Valid values: integers
Default value: **1**
Minimum: 0

probability_tors_lmcs
The probability that a TORS/MOLS move will be made during an LMCS search.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

window The energy window (in kJ/mol) within which structures will be saved.

Valid values: reals
Default value: **500**
Minimum: 0.0

emptyviewreset

Resets the viewing transform to empty workspace.

Syntax:

```
emptyviewreset
```

endundoblock

End the current undoable command block.

Syntax:

`endundoblock`

energykill

Specifies a running job to be killed. This will force the program running the job to be stopped.

Syntax:

`energykill` ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the job which is to be put to killed. This must be the ID or name of a job which is currently running or sleeping.
Aliases:
kill (see [kill], page 310)

energymonitor

Specifies a job to monitor.

Syntax:

`energymonitor` ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩
⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID or name of the job which is to be monitored. If no operand is specified then the job which was most recently monitored will be monitored.
Aliases:
monitor (see [monitor], page 346)

energysleep

Specifies a job to be put to sleep

Syntax:

energysleep ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:

⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

The ID or name of the job which is to be put to sleep. This must be the ID or name of a job which is currently running.

Aliases:

sleep (see [sleep], page 707)

energystart

Start a MacroModel job with the current energy settings.

Syntax:

energystart *onscreen*=yes | no ⟨job_name⟩

Options:

onscreen Determines whether the on-screen structure will be written and used as the .dat file for the MacroModel job.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨job_name⟩

The name of the job which is to be started. This operand is not the name of a file and should not contain any suffix. From this name the files "job_name".com and "job_name".mae will be created.

energystop

Specifies a running job to be stopped. The program running the job will be requested to stop but may take some time to respond to the command.

Syntax:

`energystop` ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:

⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

The ID or name of the job which is to be stopped. This must be the ID or name of a job which is currently running or sleeping.

Aliases:

stop (see [stop], page 709)

energytask

Determines which energy task is currently being set up.

Syntax:

`energytask` ecalc | mini | moldyn | drive | mult | csearch | mcsd | minta | embrace | embracecsearch

Operands:

ecalc | mini | moldyn | drive | mult | csearch | mcsd | minta | embrace | embracecsearch

The type of energy task which is to be set up. The operand cannot be abreviated and must be given in full.

energyupdate

Specifies a running job to be updated.

Syntax:

`energyupdate` ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:

⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

The ID or name of the job which is to be put to updated. This is not the name of a file and no suffix is required. At present only MacroModel conformational searches will pay any attention to a request to update, however it is no error for an update to be issued for any type of job.

Aliases:
update (see [update], page 775)

energyupdatejobstatus

update the job status of all the jobs in the monitor panel.

Syntax:

`energyupdatejobstatus`

energywake

Specifies a sleeping job to be woken up.

Syntax:

`energywake` ⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩

Operands:
⟨job_id⟩ | ⟨job_name⟩ | ⟨job_name⟩ ⟨project_name⟩
The ID of the job which is to be put to woken up. This must be the ID of a job which is currently sleeping.
Aliases:
wake (see [wake], page 818)

enhance3d

Fit clipping planes to displayed structure. If display stereo is on, this may help prevent conflicts between the clipping range and the stereo angle that lead to clipping of the displayed structure or flatter appearance of the structure in stereo, especially when zooming in. If display perspective is on, this may help prevent conflicts between the front clipping plane and the viewing distance (which corresponds to the perspective scale) that lead to clipping of the displayed structure or reduced perspective scaling, especially when zooming in. If display fog is on, this helps maximize the shading contrast between near and far atoms and helps make the front part of the displayed structure more visible.

Syntax:

`enhance3d` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

If present,we do enhance3d on ASL, if not present we do enhance3d on all atoms.

enterposeviewermode

This command enters into the pose viewer mode.

Syntax:

`enterposeviewermode`

entryadddisplay

Adds the given type of data to be displayed.

Syntax:

`entryadddisplay` ⟨ display properties ⟩

Operands:

⟨ display properties ⟩

A bitwise combination of flags for data to be displayed.

entrycolorbyproperty

Colors the entry property values in Project Table based on the options.

Syntax:

entrycolorbyproperty *abovevalue*=⟨ x ⟩ *belowvalue*=⟨ x ⟩
colorbetween=solid | interpolate *exactvalue*=⟨ x ⟩
interpolatesteps=⟨ n ⟩ *useabove*=yes | no *usebelow*=yes | no
usebetween=yes | no *useexact*=yes | no ⟨ property_name ⟩

Options:

abovevalue
Valid values: reals
Default value: **10**

belowvalue
Valid values: reals
Default value: **1**

colorbetween
Valid values: solid
interpolate
Default value: **interpolate**

exactvalue
Valid values: reals
Default value: **10**

interpolatesteps
Valid values: integers
Default value: **10**

useabove
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

usebelow
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

usebetween
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

useexact
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ property_name ⟩
Indicates the property name for which color applied to the property value in Project Table.

entrycombine

Combine selected project entries into a single entry.

Syntax:

entrycombine *order*=entryid | row *replace*=yes | no ⟨new_name⟩ ⟨ESL⟩

Options:

order This determines whether the entries should be combined based on entryid order or table row order in the project table.

Valid values: entryid
row
Default value: **entryid**

replace This determines whether the combined entry is allowed to replace an existing entry having the name <new_name>. If the name of the combined entry matches the name of an entry in the workspace, the entry in the workspace is also replaced. If the replace option is off, the user will be asked to accept a unique name based upon <new_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨new_name⟩ ⟨ESL⟩

The name to be given to the single combined project entry, followed by a valid ESL expression. If no name is specified, no rename will be done. The ESL expression specifies which entries are to be combined.

entrycopyprop

Set property values for selected project entries from another property. The to property value will be copied from the from property for the entries selected by the ESL expression. If the to property does not already exist, it will be created. When the data type for the two properties is not the same, a reasonable attempt is made to convert between them. If a selected entry has no value for the from property, or the conversion from a string value fails, the value is generally cleared for the to value for that entry. Entry names are never cleared.

Syntax:

`entrycopyprop` *from*=⟨ text ⟩ *to*=⟨ text ⟩ ⟨ ESL ⟩

Options:

from The name of the property which provides the values to be copied. This can be either the user name or the m2io data name for the property.

Valid values: text strings
Default value:

to The name of the property to be modified. This can be either the user name or the m2io data name for the property, if the to property exists. If the to property does not exist, it will be created using the data type of the from property, with user as the author. If the property is the entry name (e.g. Entry Name or s_m_entry_name), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b_m_entry_in_workspace), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values: text strings
Default value:

Operands:
⟨ ESL ⟩
The ESL expression specifies for which entries the property values are to be copied.

entrycreatesmoothed

Create new smoothed entries using the entries specified in the ESL. The entries from the ESL are duplicated and the intermediates are placed in between these duplicates. All these new entries are selected.

Syntax:

`entrycreatesmoothed` *intermediates*=⟨ n ⟩ ⟨ ESL ⟩

Options:

intermediates
Specifies the number of intermediate entries to create between each entry pair. The first half of the intermediates are duplicated

from the current key entry and the second half from the next key entry. If this number is odd, then the second half gets the additional entry.

Valid values: integers
Default value: **48**
Minimum: 1

Operands:
⟨ ESL ⟩
A valid ESL specification.

entrydelete

Delete each selected project entry.

Syntax:

`entrydelete` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Delete those entries which match the ESL description.

entrydisassociate

For each selected project entry, create an entry for each molecule except water, and one for all waters in the entry. Each of these new entries will be given a unique name, based upon the name of its originating entry. Any of the originating entries that are in the workspace are replaced by their disassociated molecule entries.

Syntax:

`entrydisassociate` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Disassociate those entries which match the ESL description.

entrydisassociatebychain

Create a new entry for each chain in each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry. Any of the originating entries that are in the workspace are replaced by their disassociated chain entries.

Syntax:

`entrydisassociatebychain` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Disassociate those entries which match the ESL description.

entrydisassociatebyligand

Create a new entry for each chain in each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry. Any of the originating entries that are in the workspace are replaced by their disassociated chain entries. For each selected project entry, create an entry for each ligand-sized molecule, an entry for all waters in the selected entry, and an entry for all other atoms in the selected entry. Each of these new entries will be given a unique name, based upon the name of its originating entry. Any of the originating entries that are in the workspace are replaced by their disassociated molecule entries.

Syntax:

`entrydisassociatebyligand` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Disassociate those entries which match the ESL description.

entrydisplaysurfaces

Display surfaces for the entry.

Syntax:

`entrydisplaysurfaces` ⟨ entry_index ⟩

Operands:
⟨ entry_index ⟩
The index of entry.

entrydragselection

Move current entry selection to the specified table row.

Syntax:

`entrydragselection` ⟨ row ⟩ ⟨ table ⟩

Operands:
⟨ row ⟩ ⟨ table ⟩
The destination row number. In place of the row number, "top" may be used to specify the first row of the table (row 1) and "bottom" may be used to specify placement after the last unselected row (last_row + 1). The selected entries will be moved as a block, placing the first selected entry below the first unselected entry which is above the destination row. If there is no unselected entry above the destination row, the selected entries will be moved to the top of the table. The name of the table to use as the source for the row number. If the table operand is missing, the current or default table (1) will be used.

entryduplicate

Create a new duplicate entry for each selected project entry. Each of these new entries will be given a unique name, based upon the name of its originating entry.

Syntax:

`entryduplicate` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩

A valid ESL specification. Duplicate those entries which match the ESL description.

entryexport

Export selected entries from the current project, or workspace entries, to a file or a number of files. If the export command is issued with options, but without a filename, then nothing is written but the options are updated. An export command with both options and a file name specified will result in the export being performed with the new options to the specified file(s). When exporting each entry to an individual file, the given file name will be the base file name which is used to name those individual files by adding their associated postfixes to the base file name.

Syntax:

`entryexport` *append*=yes | no *displayedonly*=yes | no *files*=single | individual *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any | reagentprep | mcproinput | cms | maegz | sdcompressed | glideposeviewer | smiles | smilesgz | common *graphical*=yes | no *names*=withentry | withnumber | entry | withentryid | title *properties*=all | subset *reorderbyresnum*=yes | no *sdfv3000*=yes | no *seqheaders*=yes | no *source*=workspace | selected *writedupconectrecs*=yes | no [⟨ filename ⟩]

Options:

append This option determines whether to append to the file which is going to be written. It will be grayed out when PDB format is selected.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayedonly
If this option is set to true, then only the displayed atoms in the Workspace will be exported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

files This option sets the number of export files. The two options are: export entries to a single file (0 default) or export each entry to an individual file.

Valid values:	single individual
Default value:	**single**

format This option sets the format of the file to be written. Valid values are “mmod”, “pdb”, “mol2”, “maestro” or “sd”.

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
reagentprep
mcproinput
cms
maegz
sdcompressed
glideposeviewer
smiles
smilesgz
common

Default value: **maestro**

graphical This option determines whether to export graphical information when an entry is being saved to disk.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

names This option sets output file names when a number of files are exported. There are three options: file name + entry name, file name + number, just entry name. This option will be grayed out if only one file is exported.

Valid values: withentry
withnumber
entry
withentryid
title
Default value: **withentry**

properties This option sets the properties to be exported. The value can either be all or subset .

Valid values: all
subset
Default value: **all**

reorderbyresnum
This option determines whether to reorder the records by resideu number while writing pdb files to disk

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sdfv3000 If set, then SDF files will be written in v3000 format.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

seqheaders This option determines whether to export secondary structure information and seqres headers when an entry is being written in pdb format to disk

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

source Whether to export selected entries in the current project table or to export what is in the workspace, including scratch entry.

Valid values: workspace
selected
Default value: **selected**

writedupconectrecs
This option determines whether to write duplicate CONECT record information while writing pdb files to disk.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
[⟨ filename ⟩]
The name of the file to which entries will be written. If no name is specified, then no export will be done.

entryexportcanvas

Export selected entries from the current project to a canvas project.

Syntax:

entryexportcanvas *canvasimport*=⟨ text ⟩ ⟨ canvas_project ⟩

Options:

canvasimport
: This sets import option for canvas, if the canvas project already exists. The option can be append or replace . If blank, this defaults to append .
Valid values: text strings
Default value:

Operands:
⟨ canvas_project ⟩
The path of the canvas project to which selected entries will be exported. If no path is specified, then no export will be done.

entryexportspreadsheet

Export selected entries from the current project to a file for use in a spreadsheet. This can be comma-separated value (.csv) format or tab-delimited format.

Syntax:

entryexportspreadsheet *columns*=all | subset *delimiter*=⟨ text ⟩
rows=all | subset | selected | included ⟨ filename ⟩

Options:

columns
: This option sets the columns to be output. The columns can either be all or subset .

Valid values: all
subset
Default value: **all**

delimiter This option sets the delimiter to use to separate columns.

Valid values: text strings
Default value: **,**

rows This option sets the rows to be written out. The rows can be all , subset , selected , or included .

Valid values: all
subset
selected
included
Default value: **all**

Operands:

⟨ filename ⟩

The name of the file to which entries will be written. If no name is specified, then no export will be done.

entryextendselection

Extend current entry selection to encompass the specified entry. This is slower than using entryextendselectrow, because of the need to locate the entry row, given the entry name.

Syntax:

`entryextendselection` ⟨ entry_name ⟩

Operands:

⟨ entry_name ⟩

The name of an entry in the current project. A range selection will be done between currently selected entries and the specified entry.

entryextendselectrow

Extend current entry selection to encompass the specified entry table row.

Syntax:

`entryextendselectrow` ⟨ row ⟩

Operands:
⟨ row ⟩
A row number between 1 and the total number of rows in the project. A range selection will be done for the entries between currently selected entry rows and the specified entry row number.

entryextendwsinclude

Extends the included entries to encompass the selected project entry.

Syntax:

`entryextendwsinclude` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Extends included entries to encompass the entries which match the ESL description into the workspace.

entrygroupbyproperty

Create new groups, if required; and re-group the entries according to their property values.

Syntax:

`entrygroupbyproperty` *alignon*=⟨ x ⟩ *basename*=⟨ text ⟩
maximum=⟨ x ⟩ *minimum*=⟨ x ⟩ *stepsize*=⟨ x ⟩
usealign=yes | no *usemaximum*=yes | no
useminimum=yes | no ⟨ property_name ⟩ ⟨ ESL ⟩

Options:

alignon

Valid values:	reals
Default value:	**0**
Minimum:	0.0

basename

Valid values: text strings
Default value: **group**

maximum

Valid values: reals
Default value: **0**

minimum

Valid values: reals
Default value: **0**

stepsize

Valid values: reals
Default value: **1**
Minimum: 0.0

usealign

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

usemaximum

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

useminimum

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ property_name ⟩ ⟨ ESL ⟩

entrygroupcollapse

Collapses the given group.

Syntax:

`entrygroupcollapse` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ group_name ⟩

Group name. Collapses the given group and hides all the entries in that group.

entrygroupcountcontacts

Record contact counts for entries in a group, using the first group entry as the reference structure (receptor).

Syntax:

`entrygroupcountcontacts` ⟨ group_name ⟩

Operands:

⟨ group_name ⟩

An existing group name. The first entry in the group is treated as the reference, for measuring contacts. Its contact count property values are deleted, and the counts for all other entries in the group are set.

entrygroupcounthbonds

Record H-bond counts for entries in a group, using the first group entry as the reference structure (receptor).

Syntax:

`entrygroupcounthbonds` ⟨ group_name ⟩

Operands:

⟨ group_name ⟩

An existing group name. The first entry in the group is treated as the reference, for measuring H-bonds. Its H-bond count property value is deleted, and the count for all other entries in the group is set.

entrygroupcreate

Creates a group for given entries.

Syntax:

`entrygroupcreate` ⟨group_name⟩ ⟨ESL⟩

Operands:

⟨group_name⟩ ⟨ESL⟩

The new name to be given to the group of entries that match given ESL expression. The name should be unique. The new group name should be followed by a valid ESL expression to specify which entries are to be grouped. If no entry matches the ESL expression, no new group will be created. No empty group can exists.

entrygroupdelete

Deletes the given group and all the entries in it.

Syntax:

`entrygroupdelete` *isexp*=yes | no ⟨group_name⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨group_name⟩

Name of the group to be deleted. Deletes the given group and all the entries in that group.

entrygroupdragselection

Moves the selected entry groups to the specified table row.

Syntax:

`entrygroupdragselection` ⟨ row ⟩

Operands:

⟨ row ⟩

the destination row number. In place of the row number, "top" may be used to specify the first row of the table (row 1) and "bottom" may be used to specify placement after the last row (last_row + 1). If the target row corresponds to an ungrouped entry, the group will be placed just after that last ungrouped entry section. If the target row corresponds to a grouped entry, the groupe will be placed just before the group to which the targer entry belongs to. All the selected groups will be moved as a block.

entrygroupduplicate

Creates a new duplicate entry for each entry of the source group. Each of these new entries will be given a unique name, based upon the name of its originating entry. All these newly created entries will be grouped under a new group; and that new group will be given a unique name, based upon the name of source group.

Syntax:

`entrygroupduplicate` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ group_name ⟩

The name of the group to be duplicated. Duplicates the given group along with all its entries.

entrygroupexpand

Expands the given group.

Syntax:

`entrygroupexpand` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ group_name ⟩
Group name. Expands the given group and displays all the entries in that group in PT.

entrygroupexpandonly

Expands the given group and collaples all other groups

Syntax:

`entrygroupexpandonly` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ group_name ⟩
Group name. Expands the given group, display all the entries in that group in PT and collaples all the other groups.

entrygroupextendselection

Extend current entry selection to encompass all the entries in the specified group.

Syntax:

`entrygroupextendselection` ⟨ group_name ⟩

Operands:
⟨ group_name ⟩
The name of a entry group in the current project. A range selection will be done between currently selected entries and the specified group; all the entries in that group will also be selected.

entrygroupinvertselection

Invert the selection of entries of an existing group.

Syntax:

`entrygroupinvertselection` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ group_name ⟩
An existing group name. Inverts the selection state of entries of the given group.

entrygroupmove

Moves the given entry group to the specified table row.

Syntax:

`entrygroupmove` ⟨ group_name ⟩ ⟨ row ⟩

Operands:
⟨ group_name ⟩ ⟨ row ⟩

Name of the group to be moved. Group name and the destination row number. In place of the row number, "top" may be used to specify the first row of the table (row 1) and "bottom" may be used to specify placement after the last row (last_row + 1). If the target row corresponds to an ungrouped entry, the group will be placed just after that last ungrouped entry section. If the target row corresponds to a grouped entry, the groupe will be placed just before the group to which the targer entry belongs to. All the entries of the group will be moved as a block.

entrygrouprename

Rename the given group.

Syntax:

`entrygrouprename` ⟨ old_name ⟩ ⟨ new_name ⟩

Operands:
⟨ old_name ⟩ ⟨ new_name ⟩
The name of the group which we want to rename followed by the new name for the group.

entrygroupselect

Select all the entries of given group.

Syntax:

`entrygroupselect` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ group_name ⟩
Group name. Selects all the entries of given group does not change the selection state of the other entries.

entrygroupselectonly

Selects the all entries of the given group and unselects all other entries.

Syntax:

`entrygroupselectonly` ⟨group_name⟩

Operands:

⟨group_name⟩

Group name. Selects all the entries of given group and unselects all other entries.

entrygroupsettitle

Set the title for the given group.

Syntax:

`entrygroupsettitle` ⟨group_name⟩ ⟨title⟩

Operands:

⟨group_name⟩ ⟨title⟩

The name of the group whose title we want to set followed by the title.

entrygroupungroup

Ungroups the entries in the given group.

Syntax:

`entrygroupungroup` *isexp*=yes | no ⟨group_name⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ group_name ⟩
<group_name> Remove the group and move all the entries in that group to the end of ungrouped section. In project table, all the ungrouped entries (if any) will be present at the top i.e. before all the groups.

entrygroupunselect

Unselects all the entries of given group.

Syntax:

`entrygroupunselect` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ group_name ⟩
Group name. Unselectis all the entries of given group and does not change the selection state of the other entries.

entrygroupviewposes

Prepare to view group poses (using eplayer), using the first group entry as the reference structure (receptor).

Syntax:

`entrygroupviewposes` *contactdisplay*=yes | no
hbonddisplay=yes | no ⟨ group_name ⟩

Options:

contactdisplay
This option determines whether Contact markers will be displayed for entry group view poses.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

hbonddisplay
This option determines whether H-bond markers will be displayed for entry group view poses.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ group_name ⟩

An existing group name. The first entry in the group is treated as the reference, for viewing H-bonds and contacts, and is fixed in the Workspace so that it remains while stepping through other group entries with the eplayer.

entrygroupwsexclude

Exclude all the entries of given group from the workspace.

Syntax:

`entrygroupwsexclude` *isexp*=yes | no ⟨ group_name ⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ group_name ⟩

A valid group name. Excludes those entries which belong to the given group from the workspace.

entrygroupwsinclude

Include all entries of given group into the workspace.

Syntax:

`entrygroupwsinclude` *isexp*=yes | no ⟨group_name⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨group_name⟩

A valid group name. Include all the entries which belong to the given group.

entrygroupwsincludeonly

Include only the entries of given group into the workspace. Exclude all others.

Syntax:

`entrygroupwsincludeonly` *isexp*=yes | no ⟨group_name⟩

Options:

isexp Specifies whether the <group_name> is actual group name or it is expression such as 'selected'.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨group_name⟩

A valid group name. Include only those entries which belong to the given group and exclude all other entries.

entryimport

Import structures into the current project.

Syntax:

entryimport *all*=yes | no *allfiles*=yes | no *creategroups*=multiple | all | none *end*=yes | no *extradata*=yes | no *firstonly*=yes | no *fittoscreen*=yes | no *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any | reagentprep | mcproinput | cms | maegz | sdcompressed | glideposeviewer | smiles | smilesgz | common *glideposeviewer*=yes | no *graphical*=yes | no *hidewarnings*=yes | no *openprepwizard*=yes | no *readalternate*=yes | no *sdtitle*=moleculename | propertyname *sdtitlepropertyname*=⟨ text ⟩ *start*=⟨ n ⟩ *total*=⟨ n ⟩ *wsinclude*=none | first | all *wsreplace*=yes | no [⟨ filename ⟩]

Options:

all This determines whether all structures will be imported, or just a specified range.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

allfiles This determines whether all files in the directory given in the operand will be imported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

creategroups
This option determines which of the imported structures are to be grouped. Valid values are "multiple", "all" or "none".

Valid values: multiple
all
none
Default value: **multiple**

end This determines if all structures in the file are to be imported starting from the structure specified by start.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

extradata This option determines whether to import data files associated with imported structures. The additional data file associations

are expected to be listed in .smap files having the same base name as the structure files.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

firstonly This option imports only the first structure incase the file format is of Desmond (.cms) type.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fittoscreen This determines whether the newly imported and included structures should be fit to Workspace or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

format This option sets the format of the file to be read. Valid values are “maestro”, “mmod”, “pdb”, “mol2”, or “sd”.

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
reagentprep
mcproinput
cms
maegz
sdcompressed
glideposeviewer
smiles
smilesgz
common

Default value: **common**

glideposeviewer

This option determines whether to setup pose viewer files for viewing poses with the ePlayer.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

graphical This determines whether the file will be read with just the geometric information or if the visual information in the file will be read and used for displaying

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

hidewarnings

This flag turns off the Maestro warnings about invalid structures on import.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

openprepwizard

This option determines whether to open protein prepwizard panel after importing PDB files.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

readalternate

This option determines whether to read alternate locations when importing PDB files.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sdtitle The source of titles from a SD format file. Valid values are "moleculename" or "propertyname" .

Valid values: moleculename
propertyname
Default value: **moleculename**

sdtitlepropertyname

The property to be used to construct titles from a SD format file, if sdtitle option is "propertyname".

Valid values: text strings
Default value:

start This option sets the number of the first structure to be imported, if not importing all.

Valid values: integers
Default value: **1**
Minimum: 1

total The total number of structures to be imported from the file, if not importing all structures.

Valid values: integers
Default value: **1**
Minimum: 1

wsinclude This option determines which of the imported structures are to be included in the workspace. Valid values are "none", "first", or "all".

Valid values: none
first
all
Default value: **first**

wsreplace This determines whether the structures currently in the workspace will be replaced by the included imported structures, or whether they will be be included in the workspace also.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:

[⟨ filename ⟩]
The name of the file from which structures will be imported. If no name is specified, then no import will be done.

entryimportspreadsheet

Import entries from a file into the project table.

Syntax:

`entryimportspreadsheet` *applyto*=selected | all *delimiter*=comma | tab | userdefined *userdelimiter*=⟨ text ⟩ ⟨ filename ⟩ ⟨ import_key ⟩ ⟨ proj_key ⟩

Options:

applyto This option sets whether the data needs to be imported to all or selected entries.

Valid values: selected
all
Default value: **selected**

delimiter This option sets the delimiter to use to separate columns.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
: This option sets the delimiter defined by the user to separate columns.

  Valid values: text strings
  Default value:

Operands:
⟨ filename ⟩ ⟨ import_key ⟩ ⟨ proj_key ⟩
The name of the file from which entries will be read. The key to be considered as the reference from file. Property in project that maps to import_key.

entryimportvibration

Import vibration data for the given entry

Syntax:

`entryimportvibration` *entry*=⟨ text ⟩ ⟨ filename ⟩

Options:

entry
: This is the entry which the vibration data will be attached to.

  Valid values: text strings
  Default value:

Operands:
⟨ filename ⟩
The name of the file from which vibration data will be imported.

entryinvertselection

Invert the selection state of all project entries.

Syntax:

`entryinvertselection`

entrymergeprop

Merge properties from a destination, appending to a source property. The from property values will be appended to the to property for the entries selected by the ESL expression. If the to property does not already exist, it will be created. The to property must be a string type. The separator value will be placed between the to and each from property during the merge operation.

Syntax:

`entrymergeprop` *from*=⟨ text ⟩ *separator*=⟨ text ⟩ *to*=⟨ text ⟩ ⟨ ESL ⟩

Options:

from The name of the property(s) which provides the values to be merged. This can be either the user names or the m2io data names for the property.

Valid values: text strings
Default value:

separator The string that will be placed between each value during the merge operation.

Valid values: text strings
Default value: **:**

to The name of the property to be modified. This can be either the user name or the m2io data name for the property, if the to property exists. If the to property does not exist, it will be created using the data type of the from property, with user as the author. The to property must be a string type.

Valid values: text strings
Default value:

Operands:

⟨ ESL ⟩

The ESL expression specifies for which entries the property values are to be merged.

entrymovetogroup

Move the given entries to an existing group.

Syntax:

entrymovetogroup ⟨ group_name ⟩ ⟨ ESL ⟩

Operands:
⟨ group_name ⟩ ⟨ ESL ⟩
An existing group name and a valid ESL expression. Moves the entries that match the ESL description to the given group. The entries are removed from the previous groups.

entryremovedisplay

Removes the given type of data to be displayed.

Syntax:

entryremovedisplay ⟨ display properties ⟩

Operands:
⟨ display properties ⟩
A bitwise combination of flags for data to be displayed.

entryrename

Rename selected project entries.

Syntax:

entryrename *replace*=yes | no ⟨ new_name ⟩ ⟨ ESL ⟩

Options:

replace This determines whether a single selected entry is allowed to replace an existing entry having the name <new_name>. Otherwise, the user will be asked to accept a unique name based upon <new_name>.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ new_name ⟩ ⟨ ESL ⟩

The new name to be given to a single selected project entry, or the basename for multiple selected project entries from which unique names will be derived for each. The new name should be followed by a valid ESL expression to specify which entries are to be renamed. If no name is specified, no rename will be done.

entryresetcolor

Reset background color of selected project entries to default.

Syntax:

`entryresetcolor` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Reset the background color of those entries which match the ESL description to default.

entryselect

Select specified entries in current project.

Syntax:

`entryselect` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Selects those entries which match the ESL description, adding to the currently selected entries.

entryselectall

Select all project entries.

Syntax:

`entryselectall`

entryselectonly

Select only the specified entries in current project.

Syntax:

`entryselectonly` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Selects only those entries which match the ESL description, unselecting all other entries.

entryselectonlyrow

Select only the entry corresponding to the given row in the current project.

Syntax:

`entryselectonlyrow` ⟨ row ⟩

Operands:
⟨ row ⟩
A row number between 1 and the total number of rows. Selects only the entry corresponding to the given row, unselecting all other entries.

entryselectrandom

Selects a random set of entries.

Syntax:

`entryselectrandom` *entries*=selected | all *numentries*=⟨ n ⟩
percentage=⟨ x ⟩ *selectoption*=nentries | npercentofentries None

Options:

entries This determines whether the random entries will be chosen from the selected entries or from all entries.

Valid values:	selected all
Default value:	**selected**

numentries

This is the number of entries to select.

Valid values:	integers
Default value:	**1**
Minimum:	0

percentage This is the percentage of entries to select.

Valid values:	reals
Default value:	**50**
Minimum:	0.0
Maximum:	100.0

selectoption

This determines whether the percentage of entries has to be selected or the number of entries has to be selected.

Valid values:	nentries npercentofentries
Default value:	**npercentofentries**

Operands:

None

None

entryselectrow

Select the entry corresponding to the given row in the current project.

Syntax:

`entryselectrow` ⟨ row ⟩

Operands:

⟨ row ⟩

A row number between 1 and the total number of rows. Selects the entry corresponding to the given row.

entrysetcolor

Set background color for selected project entries.

Syntax:

`entrysetcolor` *blue*=⟨ n ⟩ *green*=⟨ n ⟩ *red*=⟨ n ⟩ ⟨ ESL ⟩

Options:

blue	Blue value.	
	Valid values:	integers
	Default value:	**0**
	Minimum:	0
	Maximum:	255
green	Green value.	
	Valid values:	integers
	Default value:	**0**
	Minimum:	0
	Maximum:	255
red	Red value.	
	Valid values:	integers
	Default value:	**0**
	Minimum:	0
	Maximum:	255

Operands:

⟨ ESL ⟩

A valid ESL specification. Set the background color of those entries which match the ESL description with given RGB value.

entrysetdeletable

Allow deletion of selected project entries.

Syntax:

`entrysetdeletable` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Allow deletion of entries that match the ESL description.

entrysetnondeletable

Disallow deletion of selected project entries.

Syntax:

`entrysetnondeletable` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Disallow deletion of entries that match the ESL description.

entrysetprop

Set property value for selected project entries.

Syntax:

`entrysetprop` *property*=⟨ text ⟩ *value*=⟨ text ⟩ ⟨ ESL ⟩

Options:

property The name of the property to be modified. This can be either the user name or the m2io data name for the property. If the property is the entry name (e.g. Entry Name or s_m_entry_name), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b_m_entry_in_workspace), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values: text strings
Default value:

value The value to be set for the selected entries. For Boolean properties, legal values are yes, no, true, false, on, off, and 1, 0.

Valid values: text strings
Default value:

Operands:

⟨ ESL ⟩

The specified property value will be assigned to the entries selected by the ESL expression.

entrysetreadonly

Disallow modification of selected project entries.

Syntax:

`entrysetreadonly` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

A valid ESL specification. Disallow modification to entries that match the ESL description.

entrysettitle

Set entry title.

Syntax:

`entrysettitle` ⟨ title ⟩ ⟨ ESL ⟩

Operands:

⟨ title ⟩ ⟨ ESL ⟩

New entry title and a valid ESL specification. Set title for those entries which match the ESL description from the workspace.

entrysetwritable

Allow modification of selected project entries.

Syntax:

`entrysetwritable` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Allow modification to entries that match the ESL description.

entryshowall

Creates a entry subset in the project table consisting of all the entries in project.

Syntax:

`entryshowall`

entrystars

Sets the number of stars for the given entries

Syntax:

`entrystars` *stars*=⟨ n ⟩ ⟨ ESL ⟩

Options:

stars	The number of stars to set for the given entries.	
	Valid values:	integers
	Default value:	**1**
	Minimum:	0

Operands:
⟨ ESL ⟩
Which entries to set stars for Sets the number of stars for the given entries.

entrytable

This keyword is used to set various options associated with the project entry table (Project Table).

Syntax:

`entrytable` *swapnametitle*=yes | no

Options:

swapnametitle

This determines whether the table is shown and otherwise treated as if the entry name and title columns are swapped with respect to their positions in the project table.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryundisplaysurfaces

Undisplay surfaces for the entry.

Syntax:

`entryundisplaysurfaces` ⟨ entry_index ⟩

Operands:
⟨ entry_index ⟩
The index of entry.

entryunselect

Unselect specified entries in current project.

Syntax:

`entryunselect` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Unselect those entries which match the ESL description.

entryunselectall

Unselect all project entries.

Syntax:

`entryunselectall`

entryunselectrow

Unselect the entry corresponding to the given row.

Syntax:

`entryunselectrow` ⟨ row ⟩

Operands:

⟨ row ⟩

A row number between 1 and the total number of rows in the project. Unselects the entry corresponding to the given row number.

entrywatermapexamineresults

Attempt to view WaterMap results for the specified entry, if it has WaterMap files.

Syntax:

`entrywatermapexamineresults` ⟨ entry_name ⟩

Operands:

⟨ entry_name ⟩

The name (entry ID) of the entry, whose WaterMap data is to be viewed.

entrywscreate

Create project entry from atoms in the workspace.

Syntax:

`entrywscreate` *replace*=yes | no ⟨ new_name ⟩ ⟨ ASL ⟩

Options:

replace This option is ignored starting with version 60105 of Maestro.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ new_name ⟩ ⟨ ASL ⟩
The name to be given to the created project entry, followed by a valid ASL expression to specify which atoms constitute the new project entry. If no name is specified, no operation will be done.

entrywsexclude

Exclude selected project entries from the workspace.

Syntax:

`entrywsexclude` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Excludes those entries which match the ESL description from the workspace.

entrywsexcludenotfixed

Exclude all unfixed entries from the workspace.

Syntax:

`entrywsexcludenotfixed`

entrywsinclude

Include selected project entries into the workspace.

Syntax:

`entrywsinclude` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Include those entries which match the ESL description into the workspace.

entrywsincludelock

Locks the given entries into the workspace.

Syntax:

`entrywsincludelock` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Locks the given entries into the workspace.

entrywsincludeonly

Include only selected project entries in the workspace. Exclude all others.

Syntax:

`entrywsincludeonly` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Include only those entries which match the ESL description in the workspace. Exclude all others.

entrywsincludeunlock

Unlocks the given entries in the Workspace.

Syntax:

`entrywsincludeunlock` ⟨ ESL ⟩

Operands:
⟨ ESL ⟩
A valid ESL specification. Unlocks the given entries in the Workspace.

epik

This keyword is used to set various options associated with running epik jobs.

Syntax:

`epik` *analysis_mode*=query | predict | scan *gen_tautomers*=yes | no *input_file*=⟨ text ⟩ *max_output_struct*=⟨ n ⟩ *metal_binding*=yes | no *metal_params*=⟨ text ⟩ *original_ion_state*=yes | no *original_tautomer*=yes | no *ph*=⟨ x ⟩ *ph_tolerance*=⟨ x ⟩ *pka_file*=⟨ text ⟩ *solvent*=h2o | dmso *structure_source*=selected_entries | workspace | file *tautomer_file*=⟨ text ⟩

Options:

analysis_mode
In predict mode it generates the new structure and calculates the pka values for the ionizable atoms in these new structures. In Querry mode it just calculates pka values for certain ionizable atoms in the structure.

Valid values:	query predict scan
Default value:	**predict**

gen_tautomers
Generate tautomers

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

input_file The name of the structure input file.

Valid values:	text strings
Default value:	

max_output_struct
Maximum output structure to be generated.
Valid values: integers
Default value: **16**
Minimum: 0

metal_binding
Set this to true to turn on generation of metal-binding states.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

metal_params
This is the file name of parameters to use for generating metal binding states.
Valid values: text strings
Default value:

original_ion_state
Include original ionization state
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

original_tautomer
Include original Tautomer.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ph Ionization pH
Valid values: reals
Default value: **5**
Minimum: 0.0
Maximum: 14.0

ph_tolerance
Ionization pH tolerance
Valid values: reals
Default value: **2**
Minimum: 0.0
Maximum: 7.0

pka_file The name of the custom pKa parameter file.
Valid values: text strings
Default value:

solvent Specify the solvent used for calculating pka values for the structure.
Valid values: h2o
dmso

Default value: **h2o**

structure_source
Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

tautomer_file
The name of the custom tautomer parameter file.

Valid values: text strings
Default value:

epikreadinput

Read the given Epik options file and set the Epik panel options.

Syntax:

`epikreadinput` ⟨ filename ⟩

Operands:
⟨ filename ⟩
The name of the epik options file to read.

epikstart

Start a epik job with the current settings.

Syntax:

`epikstart`

epikwriteinput

Write the Epik options to the given file.

Syntax:

`epikwriteinput` ⟨ filename ⟩

Operands:
⟨ filename ⟩
The name of the epik file to write epik options.

eplayergoto

Go to the specified entry in the ordered sequence of selected project entries and pause there. The table specified in the eplayersettings command determines the sequence order.

Syntax:

`eplayergoto` ⟨ entry_name ⟩

Operands:
⟨ entry_name ⟩
The name of the entry, within the ordered sequence of selected project entries, which is to be included in the workspace. This frame is recorded as an option to the eplayersettings command.

eplayergotofirst

Go to the first entry in the ordered sequence of selected project entries. The table specified in the eplayersettings command determines the sequence order.

Syntax:

`eplayergotofirst`

eplayergotolast

Go to the last entry in the ordered sequence of selected project entries. The table specified in the eplayersettings command determines the sequence order.

Syntax:

```
eplayergotolast
```

eplayernextgroup

Go to the first selected entry in the next group that has any selected entries, if there is one, after the group for the current eplayersettings frame entry. If the current frame is an ungrouped entry, go to the first selected entry that is in an entry group. The table specified in the eplayersettings command determines the sequence order.

Syntax:

```
eplayernextgroup
```

eplayerpreviousgroup

Go to the first selected entry in the previous group that has any selected entries, if there is one, before the group for the current eplayersettings frame entry. This will never go to the ungrouped entries at the top of the table. The table specified in the eplayersettings command determines the sequence order.

Syntax:

```
eplayerpreviousgroup
```

eplayersettings

Set eplayer state variables.

Syntax:

`eplayersettings` *frame*=⟨ text ⟩ *frameduration*=⟨ x ⟩
intermediates=⟨ n ⟩ *playmode*=loop | reverse | once
playsync=yes | no *referentry*=⟨ text ⟩ *script*=noaction |
current | file *scriptfile*=⟨ text ⟩ *superimpose*=none | previous |
reference *table*=⟨ text ⟩ *title*=⟨ text ⟩
useintermediates=yes | no

Options:

frame This option sets the name of the entry being displayed.

Valid values: text strings
Default value:

frameduration
This option determines the minimum duration, in seconds, of each displayed (entry) frame during continuous play. The actual frame duration may be longer than the specified value, due to time required for drawing and screen update.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 5.0

intermediates
Set the number of interpolated intermediates to generate between each real entry. This does not create entries. Recording movies has a similar option to generate smoother movies.

Valid values: integers
Default value: **20**

playmode This option sets the mode for continuous play. Valid values are "loop", "reverse", or "once". These cause play to continue, change direction, or stop, respectively, when reaching either end of the current entry selection.

Valid values: loop
reverse
once
Default value: **once**

playsync If true, save Workspace changes during continuous play.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

referentry This option sets the name of the reference entry used when the superimpose option is set to "reference". The reference entry must exist in the current project in order to use this option.

Valid values: text strings

Default value:

script This option sets the action for each step in the ePlayer. Valid values are "noaction", "current", or "file". At each step, ePlayer may take no action, run current command script, or run command script from a script file.

Valid values: noaction
current
file

Default value: **noaction**

scriptfile This option sets the name of the script file used when the script option is set to "Execute Command Script From File".

Valid values: text strings
Default value:

superimpose

This option sets the type of superposition done for entries displayed by the eplayer. Valid values are "none", "previous", or "reference". These cause the incoming entry to be displayed with its current coordinates, with its atoms superimposed upon those of the outgoing entry, or with its atoms superimposed upon those of the reference structure for registration.

Valid values: none
previous
reference

Default value: **none**

table This option identifies the project table which is used to define to sequence order of the selected entries in the project.

Valid values: text strings
Default value:

title This option sets the title of the entry being displayed.

Valid values: text strings
Default value:

useintermediates

If true, temporarily generate and visualize 'intermediate' number of additional frames between each real entry when playing forward. Between any two real entries are n intermediates. The first half are duplicates of the first entry, the second half are duplicates of the second entry. All have their coordinates interpolated between the 2 entries. Note that this only works if the entries have the same number of atoms though they should really be conformers. If false, then we don't generate any intermediates and simply eplay forward the actual entries. Note

that eplaying backwards does not display intermediates even if tihs option is on. These are temporary intermediates that are for display only in the Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

eplayerstepahead

Go to the next entry in the ordered sequence of selected project entries, if there is one after the frame specified in the eplayersettings command. The table specified in the eplayersettings command determines the sequence order.

Syntax:

`eplayerstepahead`

eplayerstepback

Go to the previous entry in the ordered sequence of selected project entries, if there is one before the frame specified in the eplayersettings command. The table specified in the eplayersettings command determines the sequence order.

Syntax:

`eplayerstepback`

errorcheck

Specifies optional error checking to be performed.

Syntax:

`errorcheck` *ct*=yes | no

Options:

ct An option which determines whether mmct error checking is enabled during certain (primarily graphical) operations.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

excludedvolumesmarkersettings

Set graphical data of Phase excluded volume markers.

Syntax:

`excludedvolumesmarkersettings` *ambient*=⟨ x ⟩ *diffuse*=⟨ x ⟩ *emission*=⟨ x ⟩ *radius*=⟨ x ⟩ *shininess*=⟨ x ⟩ *slices*=⟨ n ⟩ *specular*=⟨ x ⟩ *stacks*=⟨ n ⟩ *transparency*=⟨ x ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

radius The radius of QSAR markers.

Valid values: reals
Default value: **1**
Minimum: 0.1

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**

Minimum: 0.0
Maximum: 128.0

slices Set the slices of drawing sphere.

Valid values: integers
Default value: **18**
Minimum: 2

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

stacks Set the stacks of drawing sphere.

Valid values: integers
Default value: **9**
Minimum: 2

transparency
The transparency of QSAR markers.

Valid values: reals
Default value: **20**
Minimum: 0.0
Maximum: 100.0

exportrotatablegroups

Exports the receptor plus optional poses, setting the position of the receptor's movable atoms from the current pose.

Syntax:

`exportrotatablegroups` *poses*=none | current | matching
⟨ file name ⟩

Options:

poses Determine what to export.

Valid values: none
current
matching
Default value: **none**

Operands:
⟨ file name ⟩
The file name to export the receptor and poses to.

extendselecteditatom

Extends the selection to the given row in the edit atom-property table.

Syntax:

`extendselecteditatom` ⟨ row number ⟩

Operands:
⟨ row number ⟩
The operand is the row number to use to extend the selection.

fieldqsarmarkersettings

Set graphical data of Phase field-based QSAR markers.

Syntax:

`fieldqsarmarkersettings` *ambient*=⟨ x ⟩ *diffuse*=⟨ x ⟩
emission=⟨ x ⟩ *field*=⟨ text ⟩ *negative_color*=⟨ text ⟩
negativecoefficient=⟨ x ⟩ *negativesaturation*=⟨ x ⟩
numberpls=⟨ n ⟩ *positive_color*=⟨ text ⟩ *positivecoefficient*=⟨ x ⟩
positivesaturation=⟨ x ⟩ *radius*=⟨ x ⟩ *shininess*=⟨ x ⟩ *slice*=⟨ n ⟩
specular=⟨ x ⟩ *stack*=⟨ n ⟩ *transparency*=⟨ x ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	1.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values:	reals

Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

field The field to set coefficients and saturation for.

Valid values: text strings
Default value: **ff_e**

negative_color
Negative coefficient color

Valid values: text strings
Default value: **red**

negativecoefficient
Set the QSAR visualization option of negative coefficient threshold.

Valid values: reals
Default value: **-0.044**
Maximum: 0.0

negativesaturation
Set the QSAR visualization option of negative saturation threshold.

Valid values: reals
Default value: **-0.044**
Maximum: 0.0

numberpls Set the QSAR visualization option of number of PLS factors.

Valid values: integers
Default value: **1**
Minimum: 1

positive_color
Positive coefficient color

Valid values: text strings
Default value: **blue**

positivecoefficient
Set the QSAR visualization option of positive coefficient threshold.

Valid values: reals
Default value: **0.044**
Minimum: 0.0

positivesaturation
Set the QSAR visualization option of positive saturation threshold.

Valid values: reals
Default value: **0.044**
Minimum: 0.0

radius Radius in angstroms of the spheres used for visualizing intensities.

Valid values: reals
Default value: **0.1**
Minimum: 0.1

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

slice Number of slices to use for visualizing intensities.

Valid values: integers
Default value: **20**
Minimum: 1

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

stack Number of stacks to use for visualizing intensities.

Valid values: integers
Default value: **20**
Minimum: 1

transparency
The transparency of QSAR markers.

Valid values: reals
Default value: **50**
Minimum: 0.0
Maximum: 100.0

fileread

Read a structure file. If a filename is given then structures will be read from that file. If an explicit "start=" option is not included then starting structure is incremented at each read.
For example:
fileread start=1 fileread mmodtmp.dat fileread start=1 test.dat

Syntax:

`fileread` *delete*=yes | no *format*=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any | reagentprep | mcproinput | cms | maegz | sdcompressed | glideposeviewer | smiles | smilesgz | common *ginfo*=yes | no *start*=⟨ n ⟩ *tile*=yes | no *total*=⟨ n ⟩ [⟨ file_name ⟩]

Options:

delete This determines whether the currently displayed structure will be deleted before the new structure is read in.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

format This option sets the format of the file to be read. Valid values are "maestro", "mmod", "pdb", "mol2", or "sd".

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
reagentprep
mcproinput
cms
maegz
sdcompressed
glideposeviewer
smiles
smilesgz
common

Default value: **maestro**

ginfo This determines whether the file will be read with just the geometric information or if the visual information in the file will be read and used for displaying the structures.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

start This option sets the number of the first structure to be read

Valid values: integers
Default value: **1**
Minimum: 1

tile This determines whether the structures displayed on screen will be "tiled".

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

total The total number of structures to be read from the file

Valid values: integers
Default value: **1**
Minimum: 1

Operands:

[⟨file_name⟩]

The name of the file from which the structure will be read. If no name is specified, then no file read will be done.

Aliases:

read (see [read], page 666)

filewrite

If the write command is issued with options, but without a filename then nothing is written but the options are updated. If the write command alone is issued then writing is performed to the currently open file with the current options. A write command with both options and a file name specified will result in the write being performed with the new options to the specified file.

Syntax:

`filewrite` *append*=yes | no *displayed_atoms*=yes | no
format=mmod | pdb | mol2 | maestro | sd | jaguarinput | gaussinput | biograf | xyz | jaguaroutput | gaussian92 | gaussian94 | gamess | mopaccartesian | mopacinternal | mopacoutput | babelpdb | mdl | babelmol | babelmol2 | spartan | spartansemi | spartanmm | gamessinput | gausscartesian | gaussianz | jaguarzmatrix | jaguarcartesian | any | reagentprep | mcproinput | cms | maegz | sdcompressed | glideposeviewer | smiles | smilesgz | common
graphical=yes | no *separate*=yes | no *title*=⟨ text ⟩
⟨ file_name ⟩

Options:

append This option determines whether to append to the file which is going to be written

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayed_atoms
This option determines whether the displayed atoms will be saved. This option is not currently supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

format This option sets the format of the file to be read. Valid values are “mmod”, “pdb” or “mol2”.

Valid values: mmod
pdb
mol2
maestro
sd
jaguarinput
gaussinput
biograf
xyz
jaguaroutput
gaussian92
gaussian94
gamess
mopaccartesian
mopacinternal
mopacoutput
babelpdb
mdl
babelmol
babelmol2
spartan
spartansemi
spartanmm
gamessinput
gausscartesian
gaussianz
jaguarzmatrix
jaguarcartesian
any
reagentprep
mcproinput
cms
maegz
sdcompressed
glideposeviewer
smiles
smilesgz
common

Default value: **maestro**

graphical This option determines whether to write out graphical information when a structure is being saved to disk.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

separate This option determines whether to write out the CT as one structure or separate structures. This option is not currently supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

title The title for the output structure.

Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩

The name of the file to which the structure will be written.

Aliases:

write (see [write], page 824)

filter

Creates a new entry filter. The filter name must be a single token (or "quoted" if multiple tokens). A filter can be redefined by specifying a new definition.

Syntax:

`filter` ⟨ filter_name ⟩ ⟨ ESL-definition ⟩

Operands:

⟨ filter_name ⟩ ⟨ ESL-definition ⟩

The name which will be applied to the filter. If the name contains embedded spaces then it must be enclosed in double quotation marks.

find

Find atom(s), residue(s), chain, or molecule described by the ASL operand.

Syntax:

`find` *atombyname*=⟨ text ⟩ *atombynum*=⟨ n ⟩
atomnumberstring=⟨ text ⟩ *byasl*=⟨ text ⟩ *center*=yes | no
chainname=⟨ text ⟩ *elementstring*=⟨ text ⟩
findmethod=bynumber | byname *findtype*=atom | residue |
chain | molecule | asl *fit*=yes | no *inscode*=⟨ text ⟩
label=yes | no *markall*=yes | no *matchesnum*=⟨ text ⟩
mode=resnum | atomnum | smarts | restype | secstruct |
element | pdbatom | userdef *molnum*=⟨ n ⟩
pdbatomnamestring=⟨ text ⟩ *residuenumberstring*=⟨ text ⟩
residuetypestring=⟨ text ⟩ *resnum*=⟨ n ⟩
secondarystructure=helix | loop | strand
showmarkers=yes | no *smartsstring*=⟨ text ⟩
userdefinedstring=⟨ text ⟩ *zoom*=yes | no ⟨ ASL ⟩

Options:

atombyname
This option determines name of the atom to be found.
Valid values: text strings
Default value:

atombynum
This option determines value of the atom number to be found
Valid values: integers
Default value: **1**
Minimum: 1

atomnumberstring
The string of Atom number mode.
Valid values: text strings
Default value:

byasl This option determines the ASL specification fo the objects to be found (atom, residue, chain ot molecule).
Valid values: text strings
Default value:

center This option determines whether the structure will center on the found atom or on the centroid of the atoms [of residue, chain or molecule].
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

chainname
This option determines the chain name to be found, or the chain name in which to find a specified residue.
Valid values: text strings

Default value:

elementstring
The string of Element mode.
Valid values: text strings
Default value:

findmethod
This option determines the method used to specify an atom to be found, including By Number and By Name .
Valid values: bynumber
byname
Default value: **bynumber**

findtype This option determines the find type, including five options of atom, residue, chain, molecule and asl.
Valid values: atom
residue
chain
molecule
asl
Default value: **atom**

fit This option determines whether to zoom and center the found item.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

inscode This option determines value of the insertion code of the residue to be found.
Valid values: text strings
Default value:

label This option determines whether the found atoms will be labeled.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

markall This option determines whether all the found atom(s) or atom(s) of residue, chain or molecule are marked (TRUE), or only the principal atom(s) are marked (FALSE).
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

matchesnum
This option shows the number of matched atoms over the total number of atoms in the structure.
Valid values: text strings
Default value: /

mode This option determines the find mode.

Valid values: resnum
atomnum
smarts
restype
secstruct
element
pdbatom
userdef

Default value: **resnum**

molnum This option determines the number of the molecule to be found

Valid values: integers
Default value: **1**
Minimum: 1

pdbatomnamestring

The string of PDB atom name mode.

Valid values: text strings
Default value:

residuenumberstring

The string of Residue number mode.

Valid values: text strings
Default value:

residuetypestring

The string of residue type.

Valid values: text strings
Default value:

resnum This option determines value of the residue number to be found.

Valid values: integers
Default value: **1**

secondarystructure

This option determines the secondary structure type.

Valid values: helix
loop
strand

Default value: **helix**

showmarkers

This option determines whether the markers for the principal atoms should be shown (TRUE) or not (FALSE).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

smartsstring
The string of SMARTS mode.

Valid values: text strings
Default value:

userdefinedstring
The string of User defined mode.

Valid values: text strings
Default value:

zoom This option determines whether or not the structure will be zoomed to fit the found atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be found.

findnext

Select the next item of found items.

Syntax:

```
findnext
```

findprevious

Select the previous item of found items.

Syntax:

```
findprevious
```

fit

Fit structures to the Workspace.

Syntax:

`fit` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
If present,we do fit on the ASL, if not present we do fit on all atoms.

fitbox

Fit the box to the Workspace.

Syntax:

`fitbox` ⟨ xmin xmax ymin ymax zmin zmax ⟩

Operands:
⟨ xmin xmax ymin ymax zmin zmax ⟩
Fit the box to the Workspace.

fitligand

Fit a ligand to the Workspace.

Syntax:

`fitligand`

fitnextligand

Fit next ligand to the Workspace.

Syntax:

`fitnextligand`

fitoptions

Options for the fit command

Syntax:

`fitoptions` *move_clip_planes*=yes | no *zoom_topview*=yes | no

Options:

move_clip_planes
: If set to true, then the clipping planes will be moved when we do a fit to Workspace. If it is set to false, we won't.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

zoom_topview
: If set to true, then the clipping planes view will be zoomed when we fit to Workspace.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

fitpreviousligand

Fit previous ligand to the Workspace.

Syntax:

`fitpreviousligand`

flexibleligandalignment

Align ligands in a flexible manner against a specified template. This always operates on selected entries in the project but allows the template to be specified either as the first selected entry, the included entry or a specific entry.

Syntax:

`flexibleligandalignment` *template*=first_selected | included | entry [templat_entry_id]

Options:

template This option determines how the template for alignment is specifed. It's either the first selected entry, the included entry or a specified entry.

Valid values: first_selected, included, entry
Default value: **first_selected**

Operands:

[templat_entry_id]

If the template is being specified by a entry_id, the operand is used as the id of the template entry.

forcefield

Used to display the force field in the force field viewer.

Syntax:

`forcefield`

forcefieldbend

A keyword which controls the display of the bend interactions in the FF viewer.

Syntax:

`forcefieldbend` *select*=⟨ n ⟩ *show*=all | high | medium | low *sortbyenergy*=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low

Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldele

A keyword which controls the display of the electrostatic interactions in the FF viewer.

Syntax:

`forcefieldele` *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low

Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldgbsolv

A keyword which controls the display of the GBSOLV interactions in the FF viewer.

Syntax:

`forcefieldgbsolv` *select*=⟨ n ⟩ *show*=all | high | medium | low *sortbyenergy*=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldimproper

A keyword which controls the display of the improper torsion interactions in the FF viewer.

Syntax:

`forcefieldimproper` *select*=⟨ n ⟩ *show*=all | high | medium | low *sortbyenergy*=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**

Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low

Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldsasolv

A keyword which controls the display of the SASOLV interactions in the FF viewer.

Syntax:

`forcefieldsasolv` *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low

Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldstretch

A keyword which controls the display of the stretch interactions in the FF viewer.

Syntax:

`forcefieldstretch` *select*=⟨ n ⟩ *show*=all | high | medium | low *sortbyenergy*=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low
Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldtorsion

A keyword which controls the display of the torsion interactions in the FF viewer.

Syntax:

`forcefieldtorsion` *select*=⟨ n ⟩ *show*=all | high | medium | low *sortbyenergy*=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**

Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low

Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldvdw

A keyword which controls the display of the VDW interactions in the FF viewer.

Syntax:

`forcefieldvdw` *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values: integers
Default value: **0**
Minimum: 0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values: all
high
medium
low

Default value: **all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

forcefieldview

Used to specify a .mmo file which contains the interactions to be displayed in the FF viewer.

Syntax:

`forcefieldview` ⟨ mmo_file_name ⟩

Operands:

⟨ mmo_file_name ⟩

The name of the .mmo file which is to be opened and displayed. The full filename, including the .mmo suffix must be specified.

forcefieldwilson

A keyword which controls the display of the wilson interactions in the FF viewer.

Syntax:

`forcefieldwilson` *select*=⟨ n ⟩ *show*=all | high | medium | low
sortbyenergy=yes | no

Options:

select Set the interaction which is currently selected

Valid values:	integers
Default value:	**0**
Minimum:	0

show Sets a filter which determines which interactions are shown depending on the quality of their parameters.

Valid values:	all high medium low
Default value:	**all**

sortbyenergy
Determines if the interactions are sorted by energy

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

formalcharge

Increment, decrement or set the formal charge for atoms which match the ASL specification.

Syntax:

`formalcharge` increment | decrement | ⟨ formal_charge ⟩ ⟨ ASL ⟩

Operands:

increment | decrement | ⟨ formal_charge ⟩ ⟨ ASL ⟩

The first operand is either the word "increment", the word "decrement", or an integer value represening the formal charge to be used for all atoms which match the specification. "increment" and "decrement" change the atom's current formal charge by +1 or -1 respectively. The second operand is a valid ASL string which defines the set of atoms which are to have their atom names changed. An error will be issued for the first atom in the set which cannot have its formal charge set (usually because no suitable charged type exists) and no further atoms will have their charge changed.

fragment

This command sets the fragment mode and the current fragment used for growing, placing and replacing.

Syntax:

`fragment` ⟨ fragment_mode ⟩ [⟨ current_fragment ⟩]

Operands:

⟨ fragment_mode ⟩ [⟨ current_fragment ⟩]

The first operand is the name of the fragment mode, this must be a valid name from the file $SCHRODINGER/mmshare-vX.X/data/res/mmfrag.ini. The optional second argument is the name of a fragment within that mode which is to be made the current fragment.

fragmentfuse

Fuse the currently selected fragment on to the atoms given in the operand. If the atoms are not bound then it's an error. It is also an error if the currently

selected fragment is not one for which fusion is defined (mostly just small rings in Organic mode).

Syntax:

`fragmentfuse` at1 at2

Operands:

at1 at2

The two atoms which define the bond where fusion of the currently selected fragment should take place.

fragmenttype

This command sets the fragment type for the current fragment mode.

Syntax:

`fragmenttype` ⟨ fragment_type ⟩

Operands:

⟨ fragment_type ⟩

The first operand is the a qualifier for the fragment which some modes (such as furanose and pyranose) have. For example alpha or beta for pyranose or furanose sugars. The fragment_type must be a valid type for the current fragment mode.

frozenatom

Specifies a single atom to be frozen at its current position in a MacroModel calculation. Other atoms in the molecule will continue to interact with this atom. The frozen atom however will not "feel" any other atoms in the molecule.

Syntax:

`frozenatom` ⟨ atom_number ⟩

Operands:

⟨ atom_number ⟩

The number of an atom which is to be treated as frozen during a MacroModel calculation.

frozenset

Specifies a set of atoms to be frozen at their current positions in a MacroModel calculation. Other atoms in the molecule will continue to interact with these atoms. The frozen atoms however will not "feel" any other atoms in the molecule.

Syntax:

`frozenset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be treated as frozen in a MacroModel calculation.

fuse

Fuse atom list pairs

Syntax:

`fuse`

geometrycleanup

Cleans up the geometry

Syntax:

`geometrycleanup`

glideactivesiteres

Defines a residue to be included in the active site for a Glide grid calculation.

Syntax:

`glideactivesiteres` ⟨chain⟩:⟨molnum⟩:⟨resnum⟩:⟨insertioncode⟩

Operands:

⟨chain⟩:⟨molnum⟩:⟨resnum⟩:⟨insertioncode⟩

A colon separated list giving the PDB chain name, the residue number and insertion code for a residue to be included in the list of active site residues.

glideactivesiteset

Defines a set from atoms, from which all residues will be added as part of the active site for Glide calculations.

Syntax:

`glideactivesiteset` ⟨ASL⟩

Operands:

⟨ASL⟩

A valid ASL specification. All residues which have any atoms in this set will be added to the list of residues that define the active site.

glidecalcboxfromligand

Uses the information specified in the glideligandgrid command to calculate the grid box for a Glide calculation.

Syntax:

`glidecalcboxfromligand`

glideconstraintatomlabel

Specifies a constraint atom label in the receptor for a Glide calculation.

Syntax:

`glideconstraintatomlabel` ⟨atom-number⟩ ⟨label⟩

Operands:
⟨atom-number⟩ ⟨label⟩
The name which will be applied to the atom.

glideconstraintposition

Specifies a constraint position in the receptor for a Glide calculation.

Syntax:

`glideconstraintposition` *radius*=⟨x⟩ *radiusnoe*=⟨x⟩
usenoe=yes | no *x*=⟨x⟩ *y*=⟨x⟩ *z*=⟨x⟩

Options:

radius The radius of a Glide constraint position.
Valid values: reals
Default value: **1**
Minimum: 0.0001

radiusnoe The minimum NOE distance of a Glide constraint position.
Valid values: reals
Default value: **0**
Minimum: 0.0

usenoe Use the NOE constraint or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

x The X coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

y The Y coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

z The Z coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

glideconstraintregion

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

glideconstraintregion ⟨ region_name ⟩ ⟨ cell ⟩

Operands:
⟨ region_name ⟩ ⟨ cell ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregionactivecell

Set an active cell for the region.

Syntax:

glideconstraintregionactivecell ⟨ region_name ⟩ ⟨ active_cell ⟩

Operands:
⟨ region_name ⟩ ⟨ active_cell ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregiongrow

Grow cells for a region.

Syntax:

glideconstraintregiongrow ⟨ region_name ⟩

Operands:
⟨ region_name ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregioninvisible

Set a constraint region as invisible in workspace.

Syntax:

`glideconstraintregioninvisible` ⟨ region_name ⟩

Operands:
⟨ region_name ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregionshrink

Shrink cells for a region.

Syntax:

`glideconstraintregionshrink` ⟨ region_name ⟩

Operands:
⟨ region_name ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideconstraintregionvisible

Set a constraint region as visible in workspace.

Syntax:

`glideconstraintregionvisible` ⟨ region_name ⟩

Operands:
⟨ region_name ⟩
The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidecoremolecule

Defines a core molecule to be treated as the reference ligand for a Glide docking.

Syntax:

`glidecoremolecule` ⟨ molecule_number ⟩

Operands:
⟨ molecule_number ⟩
The number of a molecule to be included as the ligand.

glidecorermsdsubset

Sets the subset for RMSD calculation in Glide docking.

Syntax:

`glidecorermsdsubset` ⟨ atom number ⟩

Operands:
⟨ atom number ⟩
The atom number of the RMSD subset.

glidedisplayreceptor

Displays the receptor in the Workspace.

Syntax:

`glidedisplayreceptor`

glidedockconstraintposition

Specifies a constraint position in the receptor for a Glide calculation.

Syntax:

glidedockconstraintposition *feature*=⟨ n ⟩ *index*=⟨ n ⟩
radius=⟨ x ⟩ *radiusnoe*=⟨ x ⟩ *type*=⟨ n ⟩ *use1*=yes | no
use2=yes | no *use3*=yes | no *use4*=yes | no
usenoe=yes | no *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

feature The constraint feature of position in docking.
Valid values: integers
Default value: **-1**

index The index of position in docking.
Valid values: integers
Default value: **0**

radius The radius of a Glide constraint position.
Valid values: reals
Default value: **1**
Minimum: 0.0001

radiusnoe The minimum NOE distance of a Glide constraint position.
Valid values: reals
Default value: **0**
Minimum: 0.0

type The constraint type of position in docking.
Valid values: integers
Default value: **0**

use1 The flag indicates if this position will be used in docking for group 1.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use2 The flag indicates if this position will be used in docking for group 2.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use3 The flag indicates if this position will be used in docking for group 3.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use4 The flag indicates if this position will be used in docking for group 4.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

usenoe Use the NOE constraint or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

x The X coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

y The Y coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

z The Z coordinate of a Glide constraint position.
Valid values: reals
Default value: **0**

glidedockconstraintregion

Specifies a constraint region in the receptor for a Glide calculation.

Syntax:

`glidedockconstraintregion` *atoms*=⟨ n ⟩ *feature*=⟨ n ⟩
index=⟨ n ⟩ *type*=⟨ n ⟩ *use1*=yes | no *use2*=yes | no
use3=yes | no *use4*=yes | no ⟨ region_name ⟩

Options:

atoms The number of required ligand atoms of constraint region in docking.
Valid values: integers
Default value: **1**

feature The constraint feature of constraint region in docking.
Valid values: integers
Default value: **-1**

index The index of constraint region in docking.
Valid values: integers
Default value: **0**

type The constraint type of constraint region in docking.
Valid values: integers
Default value: **0**

use1 The flag indicates if this constraint region will be used in docking for group 1.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use2 The flag indicates if this constraint region will be used in docking for group 2.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use3 The flag indicates if this constraint region will be used in docking for group 3.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use4 The flag indicates if this constraint region will be used in docking for group 4.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ region_name ⟩

The name which will be applied to the region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidedockexcludedvolume

Specifies an excluded volume in the receptor for a Glide docking calculation.

Syntax:

`glidedockexcludedvolume` *radius*=⟨ x ⟩ *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

radius The radius of a Glide excluded volume.

Valid values: reals
Default value: **3**
Minimum: 0.0001

x The X coordinate of a Glide excluded volume.

Valid values: reals
Default value: **0**

y The Y coordinate of a Glide excluded volume.

Valid values: reals
Default value: **0**

z The Z coordinate of a Glide excluded volume.

Valid values: reals
Default value: **0**

glidedockingreset

Resets all the options related to glide ligand docking to there defaults.

Syntax:

`glidedockingreset`

glidedockregionnumatoms

Specifies the required ligand atoms for this constraint region.

Syntax:

`glidedockregionnumatoms` ⟨region_name⟩ ⟨num_atoms⟩

Operands:

⟨region_name⟩ ⟨num_atoms⟩

The name of the region and the minimum number of atoms permitted for this region. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidedockregionselect

Selects a hydrophobic constraint region to be used as a constraint in the docking job.

Syntax:

`glidedockregionselect` *group*=⟨ n ⟩ ⟨ region_name ⟩

Options:

group The constraint group for region.

Valid values:	integers
Default value:	**1**
Minimum:	0
Maximum:	3

Operands:

⟨ region_name ⟩

The name of the region to be selected as a constraint in the docking job. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glidedockregionunselect

Unselects a hydrophobic constraint region so it will not be used as a constraint in the docking job.

Syntax:

`glidedockregionunselect` *group*=⟨ n ⟩ ⟨ region_name ⟩

Options:

group The constraint group for region.

Valid values:	integers
Default value:	**1**
Minimum:	0
Maximum:	3

Operands:

⟨ region_name ⟩

The name of the region to be unselected as a constraint in the docking job. If the name contains embedded spaces then it must be enclosed in double quotation marks.

glideexcludedvolume

Specifies an excluded volume in the receptor for a Glide calculation.

Syntax:

glideexcludedvolume *radius*=⟨ x ⟩ *select*=yes | no *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

radius The radius of a Glide excluded volume.
Valid values: reals
Default value: **3**
Minimum: 0.0001

select Selection state of the excludeed volume.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

x The X coordinate of a Glide excluded volume.
Valid values: reals
Default value: **0**

y The Y coordinate of a Glide excluded volume.
Valid values: reals
Default value: **0**

z The Z coordinate of a Glide excluded volume.
Valid values: reals
Default value: **0**

glidegridhydrophobic

Used to set all the options associated with the hydrophobic constraints in Glide grid generation.

Syntax:

glidegridhydrophobic *num_vertices*=⟨ n ⟩ *threshold*=⟨ x ⟩

Options:

num_vertices
At least this number of cell vertices must meet the threshold in order for the cell to be displayed.

Valid values: integers
Default value: **5**
Minimum: 1
Maximum: 8

threshold This is cutoff for which grid cells sre considered hydrophobic.

Valid values: reals
Default value: **-0.5**
Minimum: -0.65
Maximum: -0.01

glidegridreset

Resets all the options related to glide grid generation to there defaults.

Syntax:

`glidegridreset`

glideligand

Specifies settings about the ligand(s) to be used in a Glide job.

Syntax:

`glideligand` *amidebondrotations*=yes | no *bondrotation*=⟨ text ⟩
bondrotationtype=all | amides | none
conformations=usesupplied | generate | inplace | mininplace
definereference=yes | no *dockdisplayed*=yes | no
dockfromfile=yes | no *dockrange*=all | specified *econfcut*=⟨ x ⟩
endlig=⟨ n ⟩ *format*=maestro | sd | mol2 | pdb
inputring=yes | no *lig_ccut*=⟨ x ⟩ *lig_vscale*=⟨ x ⟩
ligandsfile=⟨ text ⟩ *ligandsource*=neither | entries | extfile
ninvert=yes | no *reflig*=yes | no *ringconf*=yes | no
startlig=⟨ n ⟩ *useinputcharges*=yes | no

Options:

amidebondrotations
An option which allows amide bond rotations.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

bondrotation
The option of amide bond rotation.

Valid values: text strings
Default value: **penal**

bondrotationtype
The type of bond rotation.

Valid values: all
amides
none
Default value: **amides**

conformations
Whether to use only supplied conformations, to generate them with confgen or to treat them "in-place".

Valid values: usesupplied
generate
inplace
mininplace
Default value: **generate**

definereference
An option which determines if a reference ligand is to be defined.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dockdisplayed
An option which determines if the currently displayed ligand will be docked.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dockfromfile
[NOTE: This option is no longer used.] An option which determines ligands from external files are to be docked:

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

dockrange The range of structures in the file to be docked - whether to dock all from the file or a specified range.

Valid values: all
specified
Default value: **all**

econfcut The energy window for keeping conformations.

Valid values: reals

Default value: **12**
Minimum: 0.0

endlig The final structure from the file to be docked.
Valid values: integers
Default value: **1000**
Minimum: 0

format The format of the ligands.
Valid values: maestro
sd
mol2
pdb
Default value: **maestro**

inputring An option which allows input ring conformation
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

lig_ccut The partial atomic charge below which ligand atoms are considered to be non-polar and will have their VDW radii scaled.
Valid values: reals
Default value: **0.15**
Minimum: 0.00000001

lig_vscale The scaling factor for the VDW radii of non-polar ligand atoms.
Valid values: reals
Default value: **0.8**
Minimum: 0.00000001

ligandsfile The file containing one or more ligands.
Valid values: text strings
Default value:

ligandsource
The source of the ligands to be docked
Valid values: neither
entries
extfile
Default value: **extfile**

ninvert An option which allows nitrogen inversions
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

reflig An option which determines if a reference ligand is to be used in this Glide job.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ringconf An option which allows ring flips

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

startlig The first structure from the file to be docked.

Valid values: integers
Default value: **1**
Minimum: 1

useinputcharges
An option which determines if Impact uses Maestro charges (true) or force field derived ones (false).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

glideligandfile

Specifies a file containing one or more ligands to be used for a Glide docking run.

Syntax:

`glideligandfile` ⟨ligand_file_name⟩

Operands:
⟨ligand_file_name⟩
The name of a file containing one or more ligands.

glideligandgrid

Defines the ligand to be used to calculate the grid for the glide calculation.

Syntax:

`glideligandgrid` *ligandentry*=⟨text⟩ *ligandformat*=maestro | sd | mol2 | pdb *ligandindex*=⟨n⟩ *ligandsource*=displayed | entryname | extfile | none *usedisplayed*=yes | no ⟨ligand file name⟩

Options:

ligandentry
: The name of the entry that will be used to define the grid for the Glide calculation.

  Valid values: text strings
  Default value:

ligandformat
: The format of the file that will be used to calculate the grid box.

  Valid values: maestro
  sd
  mol2
  pdb
  Default value: **maestro**

ligandindex
: The index of the structure that is to be read as a ligand.

  Valid values: integers
  Default value: **1**
  Minimum: 1

ligandsource
: The source of the ligand to be used to define the Grid for the Glide calculation.

  Valid values: displayed
  entryname
  extfile
  none
  Default value: **displayed**

usedisplayed
: [NOTE: This option is no longer used.] An option which determines if the currently displayed ligand will be used to calculate the grid for the Glide calculation

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

Operands:
⟨ ligand file name ⟩
The name of the file containing the ligand to be used to define the grid.

glidelocatehydrophobiccells

Runs hppmap and displays hydrophobic cells.

Syntax:

`glidelocatehydrophobiccells`

glideoutput

Controls a number of options for how the poses are output during a Glide calculation.

Syntax:

`glideoutput` *applystrain*=yes | no *delpose*=⟨ x ⟩
interactionradius=⟨ x ⟩ *maxperlig*=⟨ n ⟩ *nreport*=⟨ n ⟩
numposes=⟨ n ⟩ *outputdisp*=poseviewer | ligandsonly
outputformat=maestro | sd *posedist*=⟨ x ⟩ *postdock*=yes | no
residueinteraction=range | picking *rmsdtoinputgeom*=yes | no
strainscale=⟨ x ⟩ *strainthreshold*=⟨ x ⟩ *tolerance*=⟨ x ⟩
writeinteraction=yes | no *writereport*=yes | no

Options:

applystrain
An option which determines if the Glide job will apply strain correction terms.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

delpose The maximum displacement for discarding duplicate poses.
Valid values: reals
Default value: **1.3**
Minimum: 0.0

interactionradius
The radius of grid center for per-residue interaction scores.
Valid values: reals
Default value: **12**
Minimum: 0.0

maxperlig The maximum number of poses per ligand to be written.
Valid values: integers
Default value: **1**
Minimum: 0

nreport The number of coordinate sets to write to disk.
Valid values: integers

Default value: **10000**
Minimum: 1

numposes The number of poses per ligand to be included.

Valid values: integers
Default value: **5**
Minimum: 0

outputdisp Whether the output structure file will be intended to be viewed with the Maestro pose viewer (and include the receptor structure) or is just to be the docked ligands.

Valid values: poseviewer
ligandsonly
Default value: **poseviewer**

outputformat
Whether the output structure file format will be the Maestro format or SD format.

Valid values: maestro
sd
Default value: **maestro**

posedist The distance criterion for distinct poses.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

postdock An option which determines if the Glide job will do post-docking minimization.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

residueinteraction
Whether writing er-residue interaction scores for 1) residues within a distance of grid center or 2) picking residues to include.

Valid values: range
picking
Default value: **range**

rmsdtoinputgeom
An option which determines if the Glide job will calculate the RMSD against the user supplied input geometry.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

strainscale The amount to scale ligand strain by.

Valid values: reals

Default value: **0.25**
Minimum: 0.0
Maximum: 1.0

strainthreshold
Threshold for correcting ligand strain.

Valid values: reals
Default value: **4**
Minimum: 0.0

tolerance For XP docking, a tolerance value that controls the window size where a post-docking minimized structure is kept, even though its GlideScore is less favorable then the original pose.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 10000.0

writeinteraction
An option which determines if the Glide job will write per-residue interaction scores for residues.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

writereport
Turn this on to generate a .rept or .scor file

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

glidereceptorligand

Defines a on-screen molecule or entry to be treated as the ligand for a Glide calculation.

Syntax:

`glidereceptorligand` *ligand*=molecule | entry
⟨ molecule_number | entry_id ⟩

Options:

ligand Determine the ligand is 1) a molecule (operand is molecule number), or 2) an entry (operand is entry ID).

Valid values: molecule
entry

Default value: **molecule**

Operands:
⟨ molecule_number | entry_id ⟩
The molecule number or entry ID to be included as the ligand.

glidereferenceligand

Specifies a file containing the reference ligand for a glide docking run.

Syntax:

`glidereferenceligand` *index*=⟨ n ⟩ *ligandentry*=⟨ text ⟩
ligandfile=⟨ text ⟩ *ligandformat*=maestro | sd | mol2 | pdb
ligandsource=displayed | entryname | extfile | none
use_displayed=yes | no

Options:

index The index of the reference ligand in the reference ligand file.
Valid values: integers
Default value: **1**
Minimum: 1

ligandentry
The name of the entry that will be used as the reference ligand.
Valid values: text strings
Default value:

ligandfile The name of the file containing the ligand structure that will be used as the reference ligand.
Valid values: text strings
Default value:

ligandformat
The format of the file that will be used to calculate the grid box.
Valid values: maestro
sd
mol2
pdb
Default value: **maestro**

ligandsource
The source of the ligand to be used as the reference ligand.

Valid values: displayed
entryname
extfile
none
Default value: **none**

use_displayed
[NOTE: This option is no longer used.] An option which determines if the reference ligand is that which is already displayed.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

glideringsampling

Controls a number of options for how rings are sampled.

Syntax:

`glideringsampling` *energywindow*=⟨ x ⟩

Options:

energywindow
Set the cutoff, in kcal/mol, for ring-conformation energies in Glide.
Valid values: reals
Default value: **2.5**
Minimum: 0.0

gliderotatablegroups

Settings for rotatable groups in a grid generation job.

Syntax:

`gliderotatablegroups` *allowrotation*=yes | no

Options:

allowrotation
An option which controls whether or not we allow rotation of receptor hydroxyl groups.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

gliderotatablegroupsextenduserow

Extend the used groups to include the given row

Syntax:

`gliderotatablegroupsextenduserow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row to extend to in the rotatable groups table.

gliderotatablegroupstoggleatom

Toggles the row containing the given atom.

Syntax:

`gliderotatablegroupstoggleatom` ⟨ row ⟩

Operands:
⟨ row ⟩
The atom to toggle in the rotatable groups table.

gliderotatablegroupstogglerow

Toggles the given row

Syntax:

`gliderotatablegroupstogglerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row to toggle in the rotatable groups table.

gliderotatablegroupsuseexisting

Select only the groups which have existing definitions in the Workspace.

Syntax:

```
gliderotatablegroupsuseexisting
```

glidescoring

Controls a number of options for how the poses are scored.

Syntax:

glidescoring *cvcutoff*=⟨ x ⟩ *dielectric*=⟨ x ⟩
expandedsampling=yes | no *hbcutoff*=⟨ x ⟩ *itmax*=⟨ n ⟩
maxkeep=⟨ n ⟩ *maxref*=⟨ n ⟩ *mlcutoff*=⟨ x ⟩ *posedist*=⟨ x ⟩
scorecut=⟨ x ⟩ *short_distance*=anneal | softcore

Options:

cvcutoff Reject poses if the Coulomb/VDW energy exceeds this value
Valid values: reals
Default value: **0**

dielectric The dielectric constant used in the grid energy calculation.
Valid values: reals
Default value: **2**
Minimum: 0.9999999999

expandedsampling
Whether or not to use expanded sampling.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

hbcutoff Reject poses if the H-bond energy exceeds this value
Valid values: reals
Default value: **0**
Maximum: 0.00000001

itmax The maximum number of conjugate gradient steps
Valid values: integers
Default value: **100**
Minimum: 0

maxkeep The number of ligand poses on the coarse grid.

Valid values: integers
Default value: **5000**
Minimum: 1

maxref The maximum number of poses to keep after rough score refinement.

Valid values: integers
Default value: **400**
Minimum: 1

mlcutoff Reject poses if the metal-ligand interaction score exceeds this value.

Valid values: reals
Default value: **10**

posedist The distance criterion for distinct poses.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

scorecut The window of rough scores for keeping poses.

Valid values: reals
Default value: **100**
Minimum: 0.0

short_distance
The short distance behavior of the potentials.

Valid values: anneal
softcore
Default value: **anneal**

glidesettings

Allows the settings of some values that determine how the overall Glide job runs.

Syntax:

`glidesettings` *enhancedplanarity*=yes | no
epikstatepenalties=yes | no *function*=run | setup
griddirectory=⟨ text ⟩ *gridfilename*=⟨ text ⟩ *gridfiletype*=⟨ text ⟩
maxatom=⟨ n ⟩ *maxreceptoratom*=⟨ n ⟩ *maxrotbonds*=⟨ n ⟩
mode=normal | throughput | accurate *numjobs*=⟨ n ⟩
numprocs=⟨ n ⟩ *numreqgroup1*=⟨ n ⟩ *numreqgroup2*=⟨ n ⟩
numreqgroup3=⟨ n ⟩ *numreqgroup4*=⟨ n ⟩ *penaltylevel*=large | medium | small *receptor*=alone | withligand | ligandonly
recyclesteps=⟨ n ⟩ *reqmodegroup1*=all | atleast
reqmodegroup2=all | atleast *reqmodegroup3*=all | atleast
reqmodegroup4=all | atleast *showvolumes*=yes | no
source=structure | file *splitligands*=yes | no
testsatisfaction=yes | no *volumepenalty*=yes | no
writegrid=yes | no *writekeyvalue*=yes | no
writexpviz=yes | no

Options:

enhancedplanarity
An option which determines if Glide will use the enhanced planarity force field parameters.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

epikstatepenalties
An option which determines if Glide will use the state penalties calculated for the input ligands by Epik as part of the Glide docking score.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

function Determines whether the Glide job is actually to dock some ligands or just calculate a grid for a receptor.
Valid values: run
setup
Default value: **setup**

griddirectory
The directory where the grid files are located.
Valid values: text strings
Default value:

gridfilename
The base name for the file which the receptor grid is to be written to or read from.

Valid values: text strings
Default value:

gridfiletype

The type for the file which the receptor grid is to be written to or read from. The type is either zip or grd .

Valid values: text strings
Default value: **zip**

maxatom Any ligands in the input with more than this number of atoms will be skipped.

Valid values: integers
Default value: **300**
Minimum: 1
Maximum: 300

maxreceptoratom

Receptor in the input with more than this number of atoms will be stopped.

Valid values: integers
Default value: **80000**
Minimum: 1
Maximum: 80000

maxrotbonds

Any ligands in the input with more than this number of rotatable bonds will be skipped.

Valid values: integers
Default value: **50**
Minimum: 1
Maximum: 50

mode Determines how a number of other values are set to ensure the job runs either faster than normal or more accurate than normal.

Valid values: normal
throughput
accurate
Default value: **normal**

numjobs Number of jobs to split this docking job into

Valid values: integers
Default value: **1**
Minimum: 1

numprocs Number of processors to use for each job.

Valid values: integers
Default value: **1**
Minimum: 1

numreqgroup1
Number of constraints to be required for group 1 in docking.

Valid values:	integers
Default value:	**1**
Minimum:	0
Maximum:	4

numreqgroup2
Number of constraints to be required for group 2 in docking.

Valid values:	integers
Default value:	**1**
Minimum:	0
Maximum:	4

numreqgroup3
Number of constraints to be required for group 3 in docking.

Valid values:	integers
Default value:	**1**
Minimum:	0
Maximum:	4

numreqgroup4
Number of constraints to be required for group 4 in docking.

Valid values:	integers
Default value:	**1**
Minimum:	0
Maximum:	4

penaltylevel
Determine the penalty specification for Glide excluded volumes: large, medium, or small.

Valid values:	large medium small
Default value:	**large**

receptor Determines whether the structure displayed is just the receptor or a receptor + ligand.

Valid values:	alone withligand ligandonly
Default value:	**alone**

recyclesteps
Specifies the number of ligand recycling iterations in high accuracy docking jobs (glidesettings mode=accurate).

Valid values: integers
Default value: **5**
Minimum: 1

reqmodegroup1
The mode determines how to set the number of required constraints for group 1 in docking.
Valid values: all
atleast
Default value: **atleast**

reqmodegroup2
The mode determines how to set the number of required constraints for group 2 in docking.
Valid values: all
atleast
Default value: **atleast**

reqmodegroup3
The mode determines how to set the number of required constraints for group 3 in docking.
Valid values: all
atleast
Default value: **atleast**

reqmodegroup4
The mode determines how to set the number of required constraints for group 4 in docking.
Valid values: all
atleast
Default value: **atleast**

showvolumes
An option which determines if showing excluded volumes or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

source Where the grid files come from: calculated from the structure or from previously calculated files.
Valid values: structure
file
Default value: **structure**

splitligands
Whether to split the ligand structures into separate files per job
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

testsatisfaction
An option which determines if Glide tests constraint satisfaction after docking or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

volumepenalty
An option which determines if applying excluded volumes penalties or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

writegrid An option which determines if the grid will be written following the job.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

writekeyvalue
An option which determines whether to write the input file in traditional format or key value format.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

writexpviz An option which determines if the Glide job will write out a special data file for Glide XP visualizer.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

glidesite

Settings for the site in a glide job.

Syntax:

`glidesite` *boxcenter*=ligand | residues | supplied *boxsize*=⟨ n ⟩
boxsizex=⟨ n ⟩ *boxsizey*=⟨ n ⟩ *boxsizez*=⟨ n ⟩
display_center_box=yes | no *display_enclosing_box*=yes | no
enclosingbox=fitlargest | fitdisplayed | fitlength
maxliglength=⟨ x ⟩ *recep_ccut*=⟨ x ⟩ *recep_cscale_per_atom*=⟨ x ⟩
recep_scale_per_atom=none | file | asl *recep_vscale*=⟨ x ⟩
recep_vscale_per_atom=⟨ x ⟩ *useinputcharges*=yes | no
xcent=⟨ x ⟩ *ycent*=⟨ x ⟩ *zcent*=⟨ x ⟩

Options:

boxcenter Specifies how the box center is defined - by specifiying residues in the active site, by specifying a ligand or by using the values as input by the user.

Valid values:	ligand residues supplied
Default value:	**ligand**

boxsize The size of the enclosing box.

Valid values:	integers
Default value:	**10**
Minimum:	6
Maximum:	40

boxsizex The length in X of the inner box.

Valid values:	integers
Default value:	**10**
Minimum:	6
Maximum:	40

boxsizey The length in Y of the inner box.

Valid values:	integers
Default value:	**10**
Minimum:	6
Maximum:	40

boxsizez The length in Z of the inner box.

Valid values:	integers
Default value:	**10**
Minimum:	6
Maximum:	40

display_center_box
An option which determines if the box which contains the ligand center of motion will be displayed.

Valid values:	boolean (true|false; yes|no; y|n; on|off)

Default value: **true**

display_enclosing_box
An option which determines if the box which encloses the ligand center of motion box will be displayed.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

enclosingbox
Specifies how the enclosing box is sized - based on the largest ligand, the displayed ligand or ligand length
Valid values: fitlargest
fitdisplayed
fitlength
Default value: **fitdisplayed**

maxliglength
The maximum ligand length expected.
Valid values: reals
Default value: **20**
Minimum: 0.0000000001

recep_ccut The partial atomic charge below which receptor atoms are considered to be non-polar and will have their VDW radii scaled.
Valid values: reals
Default value: **0.25**
Minimum: 0.00000001

recep_cscale_per_atom
The scaling factor for the Charge of receptor per atoms.
Valid values: reals
Default value: **1**
Minimum: 0.0

recep_scale_per_atom
An option which specifies the source from which backend will pick the VdW Radius scale factor and Charge scaling factor.
Valid values: none
file
asl
Default value: **none**

recep_vscale
The scaling factor for the VDW radii of non-polar receptor atoms.
Valid values: reals
Default value: **1**
Minimum: 0.00000001

recep_vscale_per_atom
The scaling factor for the VDW radii of receptor per atoms.

Valid values: reals
Default value: **1**
Minimum: 0.0

useinputcharges
An option which determines if Impact uses Maestro charges (true) or force field derived ones (false).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

xcent The X-coordinate of the enclosing box.

Valid values: reals
Default value: **1e-07**

ycent The Y-coordinate of the enclosing box.

Valid values: reals
Default value: **1e-07**

zcent The Z-coordinate of the enclosing box.

Valid values: reals
Default value: **1e-07**

glidetorsionalconstraint

Specifies overall options for torsional constraints used in Glide ligand docking.

Syntax:

`glidetorsionalconstraint` *apply*=yes | no
displaystructure=yes | no

Options:

apply The flag indicates whether torsional constraints should be applied for Glide ligand docking.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displaystructure
The flag indicates whether structure for selected pattern should be displayed in Workspace, when pattern has an associated structure and torsional constraints are applied.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

glidetorsionalconstraintallbonds

Specify whether all bonds in the selected SMARTS pattern should be constrained to their current torsion angle values.

Syntax:

`glidetorsionalconstraintallbonds` true|false

Operands:

true|false

true to constrain all bonds in the selected SMARTS pattern, false to constrain only the torsions defined in the torsions table for the selected SMARTS pattern.

glidetorsionalconstraintpatternadd

Add a new SMARTS pattern, if pattern is valid. This adds a new row to the patterns table, with 0 torsions defined, and selects it. No check is made for duplication of an existing SMARTS pattern.

Syntax:

`glidetorsionalconstraintpatternadd` ⟨ pattern ⟩

Operands:

⟨ pattern ⟩

The new SMARTS pattern.

glidetorsionalconstraintpatterndelete

Delete the currently selected pattern row (including its table of torsions).

Syntax:

`glidetorsionalconstraintpatterndelete`

glidetorsionalconstraintpatterndeleteall

Delete all torsion constraint pattern rows (and all torsion tables associated with them).

Syntax:

`glidetorsionalconstraintpatterndeleteall`

glidetorsionalconstraintpatterndrag

Move the selected pattern row to the specified row.

Syntax:

`glidetorsionalconstraintpatterndrag` ⟨ row ⟩

Operands:

⟨ row ⟩

The new row number for the selected SMARTS pattern.

glidetorsionalconstraintpatternreplace

Replace existing pattern for currently selected pattern row, if new SMARTS pattern is valid.

Syntax:

`glidetorsionalconstraintpatternreplace` ⟨ pattern ⟩

Operands:

⟨ pattern ⟩

The new SMARTS pattern.

glidetorsionalconstraintpatternselect

Select a SMARTS pattern, specified by row index.

Syntax:

`glidetorsionalconstraintpatternselect` ⟨index⟩

Operands:
⟨index⟩
The 1-based table row index for the pattern.

glidetorsionalconstrainttorsionadd

Add and select a new torsion row, in the torsion table for the currently selected SMARTS pattern, if the four specified atoms define a proper torsion that does not conflict with any previously constrained torsion.

Syntax:

`glidetorsionalconstrainttorsionadd` ⟨atom1⟩ ⟨atom2⟩
⟨atom3⟩ ⟨atom4⟩

Operands:
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩
Atom numbers, relative to the SMARTS pattern, for four atoms to define the torsion to be constrained.

glidetorsionalconstrainttorsiondelete

Delete the currently selected torsion row of the torsion table for the currently selected pattern.

Syntax:

`glidetorsionalconstrainttorsiondelete`

glidetorsionalconstrainttorsiondeleteall

Delete all rows from the torsion table for the currently selected pattern.

Syntax:

`glidetorsionalconstrainttorsiondeleteall`

glidetorsionalconstrainttorsiondeleteinvalid

Delete all rows from the torsion table having Status that is not OK, but keep the lowest-numbered torsion for any group of torsions that conflict with each other.

Syntax:

`glidetorsionalconstrainttorsiondeleteinvalid`

glidetorsionalconstrainttorsionreplace

Replace atoms for currently selected torsion row, if new atoms define a proper torsion.

Syntax:

`glidetorsionalconstrainttorsionreplace` ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Operands:
⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩
Atom numbers, relative to the SMARTS pattern, for four atoms to define the torsion to be constrained.

glidetorsionalconstrainttorsionselect

Select a torsion, specified by row index.

Syntax:

`glidetorsionalconstrainttorsionselect` ⟨index⟩

Operands:
⟨index⟩
The 1-based table row index for the torsion.

glidetorsionalconstrainttorsionsetangle

Specify whether the currently selected torsion should be set to a specified angle, or constrained to its current value.

Syntax:

`glidetorsionalconstrainttorsionsetangle` true|false [⟨ angle ⟩]

Operands:

true|false [⟨ angle ⟩]

true to set the angle to a specified value, false to constrain the torsion angle to its current value. This argument can be followed by an angle, in degrees, to specify the value to which the torsion angle should be set.

glideundisplayreceptor

Undisplays the receptor in the Workspace.

Syntax:

`glideundisplayreceptor`

googleprofiling

Turn on or off google cpu profiling. This uses the google perftools mechanism and Maestro needs to have been compiled with this feature to work. Otherwise, enabling profiling does nothing. You can specify an output file as the operand. If no operand is specified and you enable profiling, then the file name is set to maestro.perf . If profiling is enabled and you change the profile file name, then the current experiment is terminated, the output flushed to the current file and a new experiment started with the output going to the new file name. The environment variable CPUPROFILE is not used.

Syntax:

`googleprofiling` *enable*=yes | no

Options:

enable Enable or disable google profiling.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

graphicsdebugdisable

Do not allow graphics debug messages to be printed for specfied item.

Syntax:

`graphicsdebugdisable` ⟨ item ⟩

Operands:

⟨ item ⟩

If present, we disable debug messages for the specified item. If not present, or item is all , disable debug messages for all items. If all debug messages are enabled, then disabling individual items will have no effect until all is disabled.

graphicsdebugenable

Allow graphics debug messages to be printed for specfied item.

Syntax:

`graphicsdebugenable` ⟨ item ⟩

Operands:

⟨ item ⟩

If present, we enable debug messages for the specified item (without disabling previously enabled items). If not present, or item is all , enable debug messages for all items.

grow

Grows the currently selected fragment onto the currently selected grow bond.

Syntax:

`grow`

growbond

Specifies a grow bond for growing. A subsequent grow command will grow the currently selected fragment onto the this grow bond.

Syntax:

`growbond` ⟨ from_atom ⟩ ⟨ to_atom ⟩

Operands:

⟨ from_atom ⟩ ⟨ to_atom ⟩

The two atoms which define the grow bond. In the growing procedure the "to" atom will be replaced by a suitable atom from the incoming fragment.

growbond2

Specifies a second grow bond for growing. A subsequent grow command will grow the currently selected fragment onto the this grow bond.

Syntax:

`growbond2` ⟨ from_atom ⟩ ⟨ to_atom ⟩

Operands:

⟨ from_atom ⟩ ⟨ to_atom ⟩

The two atoms which define the grow bond. In the growing procedure the "to" atom will be replaced by a suitable atom from the incoming fragment.

growdirection

Sets the grow direction in the current fragment mode. The direction name must be one of the valid directions for the current fragment mode.

Syntax:

`growdirection` ⟨direction_name⟩

Operands:

⟨direction_name⟩

The name of the grow direction which is to be made current. This must be a valid grow direction name within the current fragment.

growname

Set the growname to that specified for all atoms which match the ASL specification.

Syntax:

`growname` ⟨grow_name⟩ ⟨ASL⟩

Operands:

⟨grow_name⟩ ⟨ASL⟩

The first operand is the grow name which is to be applied to the atom. Only the first four characters of the growname will be used. The second operand is the ASL specification for all the atoms which are to have the growname applied.

happly

This is a standard alias for **hydrogenapply** (see [hydrogenapply], page 283).

hbondcriteria

Specify the criteria for calculating H-bonds in the workspace. The acceptor atom (A), H-bonded to the donor Hydrogen (H), is Oxygen, Nitrogen, Sulfur, or Fluorine.

Syntax:

`hbondcriteria` *acceptorangle*=⟨ x ⟩ *display*=yes | no *distance*=⟨ x ⟩ *donorangle*=⟨ x ⟩

Options:

acceptorangle
Minimum H — A-R angle, in degrees.

Valid values:	reals
Default value:	**90**
Minimum:	0.0
Maximum:	180.0

display This option determines whether H-bond markers will be displayed.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

distance Maximum separation between donor hydrogen atom and acceptor atom (H — A distance, in Angstroms).

Valid values:	reals
Default value:	**2.5**
Minimum:	0.0

donorangle
Minimum D-H — A angle, in degrees.

Valid values:	reals
Default value:	**120**
Minimum:	0.0
Maximum:	180.0

hbondset1

Specify the first set of atoms used in finding H-bonds in the 3D workspace.

Syntax:

`hbondset1` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. Typical usage is to define hbondset1 and hbondset2. This set, hbondset1, defines the "from" atoms. The hbondset2 atoms define the "to" atoms. H-bonds are calculated between these two sets. That is, the H-bonds are inter-set H-bonds. No intra-set

H-bonds are calculated. If hbondset2's ASL string is empty, then H-bonds are calculated for all atoms in hbondset1.

hbondset2

Specify the second set of atoms used in finding H-bonds in the 3D workspace.

Syntax:

`hbondset2` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. Typical usage is to define hbondset1 and hbondset2. This set, hbondset2, defines the "to" atoms. The hbondset1 atoms define the "from" atoms. H-bonds are calculated between these two sets. That is, the H-bonds are inter-set H-bonds. No intra-set H-bonds are calculated. If hbondset2's ASL string is empty, then H-bonds are calculated for all atoms in hbondset1.

help

This is a standard alias for **helpsearch** (see [helpsearch], page 259).

helpauto

Turns on or off the automatic popup help.

Syntax:

`helpauto` *delay*=⟨ n ⟩ on|off

Options:

delay The delay (in seconds) before the automatic help is popped up after the mouse enters the widget.

Valid values: integers
Default value: **1**

Operands:

on|off

If the operand is "on" then the popup help will be turned on. If off then it will be turned off.

helpcategory

Changes the current help category

Syntax:

helpcategory ⟨category_name⟩

Operands:

⟨category_name⟩

The name of the category for which the help topics are to be displayed.

helpsearch

Searches the help file for the specified search string. The search can be performed on all text or just the titles.

Syntax:

helpsearch *casesensitive*=yes | no *regularexpression*=yes | no *titleonly*=yes | no ⟨search_string⟩

Options:

casesensitive

A boolean option which determines if the search is performed in a case sensitive manner

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

regularexpression

A boolean option which determines if the search string is to be treated as a regular expression

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

titleonly A boolean option which determines if the search applies to the title of the help topics or the entire text

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ search_string ⟩

The text which the help file is to be searched for.

Aliases:

help (see [help], page 258)

helptopic

Loads the topic given by the operand

Syntax:

`helptopic` ⟨ topic_name ⟩

Operands:

⟨ topic_name ⟩

The name of the help topic to be displayed in the help panel. This must match the name of a topic exactly.

hidedockablepanels

Hide all visible dockable panels

Syntax:

`hidedockablepanels`

hideentries

Hide the selected entried in the project table and create a subset of unselected entries.This function also switches to the subset view.

Syntax:

`hideentries`

hidemarkers

This command hides the given marker.

Syntax:

`hidemarkers` ⟨ marker_name ⟩

Operands:
⟨ marker_name ⟩
The name of the marker.

hidepanel

Hide the panel whose name is given by the operands.

Syntax:

`hidepanel` ⟨ panel_name ⟩

Operands:
⟨ panel_name ⟩
The name of the panel which is to be hidden. The names available for use in the "hidepanel" command are displayed in parentheses after each item in the main menu bar.

hidepanels

Hide all visible panels

Syntax:

`hidepanels`

hideproperty

This command hides the given property by creating a subset of all remaining properties in show state.This function also switch to the property subset view.

Syntax:

`hideproperty` ⟨ propertyname ⟩

Operands:
⟨ propertyname ⟩
The name of the property to hide.

hidetoolbar

Hide the toolbar of given id displayed under given panel.

Syntax:

`hidetoolbar` ⟨ panel_name:toolbar_id ⟩

Operands:
⟨ panel_name:toolbar_id ⟩
The first part is the name of the panel under which toolbar would be marked as hidden. The name must match to all characters and it is a case sensitive. The second part is the name of a toolbar id which needs to be marked as hidden. Id must match to all characters and it is a case sensitive. Any toolbar can not be displayed under any panel. There is a fix set of toolbars which can be displayed under given panel.

highlight

Create a Workspace highlight

Syntax:

`highlight`

highlightarrowatoms

Specify the ASL for the named highlight arrow. The ASL specifies the atoms in the Workspace to which the arrow is associated. This can change as the ASL is re-applied when composition of the Workspace changes.

Syntax:

`highlightarrowatoms`

highlightarrowcolor

Color of the arrow highlight. Default color is yellow

Syntax:

`highlightarrowcolor` *blue*=⟨ x ⟩ *green*=⟨ x ⟩ *red*=⟨ x ⟩

Options:

blue Blue component.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

green Green component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

red Red component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

highlightarrowsettings

Global settings that affect all highlight arrows

Syntax:

`highlightarrowsettings` *headdirection*=pointing | flat

Options:

headdirection

How the arrow head should be oriented. Currently this is either 'pointing' which means it points in the direction the tail points or 'flat' which means the arrow head is flattened into the screen so it is maximally visible no matter the orientation.

Valid values:	pointing
	flat
Default value:	**flat**

highlightatoms

Specify the ASL for the named highlight. selected can be used to specify the highlight currently selected in the highlights table. The ASL specifies the atoms in the Workspace to which the highlight is associated. This can change as the ASL is re-applied when composition of the Workspace changes

Syntax:

`highlightatoms`

highlightbackground

Highlight backgroud settings

Syntax:

`highlightbackground` *type*=automatic | auto | none

Options:

type Background highlight type

Valid values:	automatic
	auto
	none
Default value:	**automatic**

highlightcolor

Color of the silhouette highlight.

Syntax:

`highlightcolor` *alpha*=⟨x⟩ *blue*=⟨x⟩ *green*=⟨x⟩ *red*=⟨x⟩

Options:

alpha Alpha component. 0 is invisible. 1 is opaque.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

blue Blue component.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

green Green component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

red Red component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

highlightdelete

Delete the named highlight

Syntax:

`highlightdelete`

highlightdirection

Direction in which the highlight points. Only applies to those highlights that have direction.

Syntax:

`highlightdirection` x=⟨x⟩ y=⟨x⟩ z=⟨x⟩

Options:

x	X component of highlight direction vector	
	Valid values:	reals
	Default value:	**0**
y	Y component of highlight direction vector	
	Valid values:	reals
	Default value:	**0**
z	Z direction of the highlight.	
	Valid values:	reals
	Default value:	**0**

highlightduplicate

Duplicate the named highlight.

Syntax:

`highlightduplicate`

highlighthide

Hide the highlight

Syntax:

`highlighthide`

highlightmethod

Create or modify a Workspace highlight.

Syntax:

`highlightmethod` *type*=silhouette | contrast | saturation

Options:

type Sets the type of highlight

Valid values:	silhouette contrast saturation
Default value:	**silhouette**

highlightpreference

Modify various highlight preferences

Syntax:

`highlightpreference` *contrast*=⟨ x ⟩ *saturation*=⟨ x ⟩

Options:

contrast Increases or decreases the contrast of the atoms and bonds that are not in any contrast highlight. Atoms and bonds in a contrast highlight definition are left alone.

Valid values:	reals
Default value:	**0.55**
Minimum:	0.0
Maximum:	1.0

saturation Increases or decreases the saturation of the atoms and bonds that are not in any saturation highlight. Atoms and bonds in a saturation highlight definition are left alone.

Valid values:	reals
Default value:	**0.3**
Minimum:	0.0
Maximum:	1.0

highlightramp

Control over a highlight's intensity ramp. Only applies to highlights that use a ramp.

Syntax:

`highlightramp` *ramp*=constant | linear | exponential *step*=⟨ x ⟩

Options:

ramp Ramp type to use for those highlights that change over time. Those that don't ignore this option (effectively setting it to constant).

Valid values:	constant linear exponential
Default value:	**linear**

step Step size for linear ramp. 0 causes the highlight to not cycle and to use its initial color values.

Valid values:	reals
Default value:	**0.025**
Minimum:	0.0

highlightrename

Rename the specified highlight

Syntax:

`highlightrename`

highlightselect

Select only the named highlight

Syntax:

`highlightselect`

highlightshow

Show the highlight

Syntax:

`highlightshow`

highlightsoff

Hide all ighlights

Syntax:

`highlightsoff`

highlightson

Display all highlights which are visible

Syntax:

`highlightson`

highlighttext

Optional text for the highlight to be displayed in the Workspace

Syntax:

`highlighttext`

highlighttextcolor

Color of the highlight text text. Default color is yellow

Syntax:

`highlighttextcolor` *blue*=⟨ x ⟩ *green*=⟨ x ⟩ *red*=⟨ x ⟩

Options:

blue	Blue component.	
	Valid values:	reals

	Default value:	**0**
	Minimum:	0.0
	Maximum:	1.0

green Green component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

red Red component.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

highlighttextfont

Font settings for the highlight text

Syntax:

`highlighttextfont` *name*=⟨ text ⟩ *size*=⟨ x ⟩ *style*=regular | bold | italic | italicbold

Options:

name Font name for the highlight text (helvetica, etc.)

Valid values:	text strings
Default value:	**helvetica**

size Font size for the highlight text

Valid values:	reals
Default value:	**14**
Minimum:	3
Maximum:	96

style Font style for the highlight text (regular, bold, etc.)

Valid values:	regular bold italic italicbold
Default value:	**regular**

highlighttextposition

Position of the optional text for a highlight. If there is text an associated, an arrow is rendered. Its tail is always attached to the text. The head position is controlled by the headx, heady, headz option values. These are automatically set and updated based on the ASL definition of the highlight. If the text is empty, no arrow is drawn.

Syntax:

`highlighttextposition` *headx*=⟨ x ⟩ *heady*=⟨ x ⟩ *headz*=⟨ x ⟩
x=⟨ x ⟩ *y*=⟨ x ⟩

Options:

headx 3D X position of the arrow head

Valid values: reals
Default value: **0**

heady 3D Y position of the arrow head

Valid values: reals
Default value: **0**

headz 3D Z position of the arrow head

Valid values: reals
Default value: **0**

x Fractional X window position of the optional highlight text.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

y Fractional Y window position of the optional highlight text.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

highlighttextselect

Select only the named highlight text

Syntax:

`highlighttextselect`

highlighttextselectadd

Add the named highlight text to the selected highlight texts

Syntax:

`highlighttextselectadd`

highlighttextsettings

Modify settings for the named highlight

Syntax:

`highlighttextsettings` *whenhighlightedatoms*=yes | no

Options:

whenhighlightedatoms

When false always display any text. When true only display text if there are highlighted atoms for the named highlight. This means that there have to be atoms that matched the highlight ASL.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

highlighttextunselect

Unselect the specifiecd highlight text

Syntax:

`highlighttextunselect`

highlighttextunselectall

Unselect all highlight texts

Syntax:

`highlighttextunselectall`

highlightwidth

Width of the border around the bond's line or tube.

Syntax:

`highlightwidth` *line*=⟨ x ⟩ *sphere*=⟨ x ⟩ *tube*=⟨ x ⟩

Options:

line Border width around the bond when it is in line representation

Valid values:	reals
Default value:	**0.15**
Minimum:	0.01
Maximum:	0.5

sphere Border width around an atom sphere (when it is in CPK or Ball representation)

Valid values:	reals
Default value:	**0.15**
Minimum:	0.01
Maximum:	0.5

tube Border width around the bond when it is in tube representation

Valid values:	reals
Default value:	**0.15**
Minimum:	0.01
Maximum:	0.5

historyvisible

Determines which types of commands will be visible in the command history list. Issuing a "historyvisible" alone will cause the list to be rebuilt with the current settings. A historyvisible all will make all commands in the list visible. A historyvisible readwrite=n will make all the read,write,sread and swrite commands hidden. Note that this only affects the visibility of the commands in the command history list of the script editor, not which commands are logged to a file.

Syntax:

`historyvisible` *1dplot*=yes | no *2dplot*=yes | no *adjust*=yes | no
alias=yes | no *application*=yes | no *atomprop*=yes | no
beginundoblock=yes | no *bondprop*=yes | no *build*=yes | no
clip=yes | no *coloratom*=yes | no *debug*=yes | no
delete=yes | no *displayatom*=yes | no *displayopt*=yes | no
entryexport=yes | no *entryimport*=yes | no
entrywscreate=yes | no *errorcheck*=yes | no *ffview*=yes | no
find=yes | no *fit*=yes | no *glide*=yes | no *helpauto*=yes | no
helpcategory=yes | no *helpsearch*=yes | no *helptopic*=yes | no
historyvisible=yes | no *hold*=yes | no *htreat*=yes | no
impact=yes | no *labelatom*=yes | no *liaison*=yes | no
ligrep=yes | no *macromodel*=yes | no *measurements*=yes | no
monitor=yes | no *pause*=yes | no *pausecommands*=yes | no
prefer=yes | no *print*=yes | no *project*=yes | no
qikprop=yes | no *qsite*=yes | no *quit*=yes | no
rename=yes | no *rep*=yes | no *ribbons*=yes | no
rotate=yes | no *saveimage*=yes | no *savelayout*=yes | no
saverestoreview=yes | no *script*=yes | no *sets*=yes | no
showhide=yes | no *spotcenter*=yes | no *superimpose*=yes | no
system=yes | no *table*=yes | no *tile*=yes | no
transformation=yes | no *translate*=yes | no
undoredo=yes | no *zoom*=yes | no [all]

Options:

1dplot

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

2dplot

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

adjust

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

alias

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

application

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

atomprop

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

beginundoblock

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

bondprop

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

build

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

clip

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

coloratom

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

debug

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

delete

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displayatom

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displayopt

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryexport

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryimport

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entrywscreate
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

errorcheck
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ffview
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

find
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fit
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

glide
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helpauto
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helpcategory
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helpsearch
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helptopic
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

historyvisible
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

hold
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

htreat

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impact

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

labelatom

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

liaison

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ligrep

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

macromodel

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

measurements

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

monitor

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pause

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pausecommands

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

prefer

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

print

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

project

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

qikprop

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

qsite

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

quit

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rename

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rep

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ribbons

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rotate

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

saveimage

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

savelayout

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

saverestoreview

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

script

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sets

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showhide

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

spotcenter

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

superimpose

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

system

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

table

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

tile

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

transformation

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

translate

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

undoredo

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

zoom

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

[all]

If "all" is specified then all commands will be visible in the command history list.

hold

The hold command permits holding the current structures off screen.

Syntax:

`hold` ⟨ hold_name ⟩

Operands:
⟨ hold_name ⟩
The name which will be applied to the hold set. This name will reference the hold in subsequent "addfromhold" and "replacefromhold" commands. If the name contains embedded spaces then it must be enclosed in quotation marks.

hppmap

Sets the parameters for hppmap

Syntax:

`hppmap` *boxmargin*=⟨ x ⟩ *boxtype*=box | ligand *cutoff*=⟨ x ⟩
cutoffscheme=atom | neutral *gridspacing*=standard | high
incorporate=append | replace | ignore | appendungrouped |
workspace *method*=h2o | dipole *source*=selected_entries |
workspace | file

Options:

boxmargin This value is used to increase the size of the box.

Valid values:	reals
Default value:	**6**
Minimum:	0.

boxtype This option indicates whether to treat the box ASL as defining a box or as defining a ligand.

Valid values:	box ligand
Default value:	**box**

cutoff This specifies the cutoff value to use for the hppmap calculation, in angstroms.

Valid values:	reals
Default value:	**20**
Minimum:	0.

cutoffscheme
This option controls the cutoff scheme.

Valid values:	atom neutral
Default value:	**atom**

gridspacing
This option indicates what type of grid spacing (standard or high) to use for the hppmap.

Valid values:	standard high
Default value:	**standard**

incorporate
This option controls the incorporation of the results.

Valid values:	append replace ignore appendungrouped workspace
Default value:	**replace**

method This option controls the method.

Valid values:	h2o dipole
Default value:	**dipole**

source This is the source for the structure.

Valid values:	selected_entries workspace file
Default value:	**workspace**

hppmapbox

Sets the box parameters for hppmap

Syntax:

`hppmapbox` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩

The ASL defines the box to use for the hppmap calculation.

hppmapset

This command is used to specify the atoms for which the hppmap calculation will be performed.

Syntax:

`hppmapset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL defines the set of atoms for which the hppmap calculation is to be performed.

hppmapstart

Starts a hppmap job with the current parameters.

Syntax:

`hppmapstart`

hppmapwrite

Writes the hppmap input file with the current parameters.

Syntax:

`hppmapwrite` ⟨ file name ⟩

Operands:
⟨ file name ⟩
Name of the file to write to. If the operand is blank, then use the default file name.

htreat

This is a standard alias for **hydrogentreat** (see [hydrogentreat], page 283).

hydrogenapply

Add or remove hydrogens and lone pairs to the atoms defined by the ASL operand according to the currently set treatment.

Syntax:

`hydrogenapply` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have hydrogens added or removed to become consistent with the current treatment.

Aliases:

happly (see [happly], page 256)

hydrogentreat

Choose a treatment for hydrogen atom addition/deletion.

Syntax:

`hydrogentreat` ⟨ treatment_name ⟩

Operands:

⟨ treatment_name ⟩

The operand is the name of a treatment which is to be made current. Note: treatment names are defined in $SCHRODINGER/mmshare-vX.X/data/mmhtreat.ini. There are both long and short names for each treatments and either can be used.

Aliases:

htreat (see [htreat], page 282)

impactbufferedatom

Specifies an atom to be buffered in an Impact calculation.

Syntax:

`impactbufferedatom` ⟨ atom_number ⟩

Operands:

⟨ atom_number ⟩

The number of an atom which is to be treated as buffered during an Impact calculation.

impactbufferedset

Specifies a set of atoms to be buffered in an Impact calculation.

Syntax:

`impactbufferedset` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The operand must be a valid string in the atom specification language.

impactconstraints

Used to set all the options associated with constraints in impact

Syntax:

`impactconstraints` *bonds*=yes | no *buffer_force*=⟨ x ⟩
hmcbonds=yes | no *hmcsolvent*=yes | no *shake_tolerance*=⟨ x ⟩
solvent=yes | no

Options:

bonds If true, then all bonds will be constrained during the Impact calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

buffer_force
The buffered atom force constant

Valid values: reals
Default value: **25**

hmcbonds If true, then all bonds will be constrained during the HMC calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

hmcsolvent
If true, then all solvent molecules will be held rigid during the HMC calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

shake_tolerance
The tolerance for the SHAKE (or RATTLE) algorithms.

Valid values: reals
Default value: **1e-07**
Minimum: 0.0

solvent [NOTE: This option is no longer used.] If true, then all solvent molecules will be held rigid during the Impact calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impactcontinuumsolvent

Used to set all the options associated with the contiunuum solvent in Impact.

Syntax:

`impactcontinuumsolvent` *pbfcutoff*=⟨ x ⟩ *qsitepbfcutoff*=⟨ x ⟩ *qsiteresolution*=low | medium | high *qsitesgbcutoff*=⟨ x ⟩ *qsitetype*=sgb | pbf | agbnp *resolution*=low | medium | high *sgbcutoff*=⟨ x ⟩ *type*=sgb | pbf | agbnp

Options:

pbfcutoff The type displacement threshold for the PBF calculation.

Valid values: reals
Default value: **0.1**
Minimum: 0.0

qsitepbfcutoff
The type displacement threshold for the PBF calculation. shake_tolerance=d

Valid values: reals
Default value: **0.1**
Minimum: 0.0

qsiteresolution
The resolution for the PBF solvation calculation.

Valid values: low
medium
high
Default value: **low**

qsitesgbcutoff
The type displacement threshold for the SGB calculation.

Valid values: reals
Default value: **0.1**
Minimum: 0.0

qsitetype The type of continuum solvent which is to be used in the calculation.

Valid values: sgb
pbf
agbnp
Default value: **sgb**

resolution The resolution for the PBF solvation calculation.

Valid values: low
medium
high
Default value: **low**

sgbcutoff The type displacement threshold for the SGB calculation.

Valid values: reals
Default value: **0.1**
Minimum: 0.0

type The type of continuum solvent which is to be used in the calculation.

Valid values: sgb
pbf
agbnp
Default value: **sgb**

impactdynamics

Settings associated with molecular dynamics simulations in Impact.

Syntax:

`impactdynamics` *effectivedensity*=⟨ x ⟩ *ensemble*=nvt | nve | npt *inittempgauss*=⟨ x ⟩ *initvelo*=yes | no *isothercomp*=⟨ x ⟩ *numbermdsteps*=⟨ n ⟩ *targetpress*=⟨ x ⟩ *targettemp*=⟨ x ⟩ *tautemp*=⟨ x ⟩ *tauvol*=⟨ x ⟩ *timestep*=⟨ x ⟩ *volumescaling*=centerofmass | atom

Options:

effectivedensity
: The effective density used during an NPT Impact dynamics simulation.

  Valid values: reals
  Default value: **1**
  Minimum: 0.0000000000001

ensemble
: The target ensemble to be achieved during an Impact dynamics simulation.

  Valid values: nvt, nve, npt
  Default value: **nvt**

inittempgauss
: The temperature used to initialize the velocities from a Gaussian distribution during an NPT Impact dynamics simulation.

  Valid values: reals
  Default value: **298.15**
  Minimum: 0.0000000000001

initvelo
: An option which determines if the velocities are to be initialized from a Gaussian distribution during an Impact dynamics simulation.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

isothercomp
: The solvent isothermal compressibility to be used during an NPT Impact dynamics simulation.

  Valid values: reals
  Default value: **4.96e-05**
  Minimum: 0.0000000000001

numbermdsteps
The total number of time steps to be performed during an Impact dynamics simulation.

Valid values: integers
Default value: **100**
Minimum: 1

targetpress
The target pressure to be used during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **1**

targettemp
The target temperature to be used during a NVT or NPT Impact dynamics simulation.

Valid values: reals
Default value: **298.15**
Minimum: 0.0000000001

tautemp The temperature relaxation time to be used during a NVT or NPT Impact dynamics simulation.

Valid values: reals
Default value: **0.01**
Minimum: 0.0000000001

tauvol The volume relaxation time to be used during an NPT Impact dynamics simulation.

Valid values: reals
Default value: **0.01**
Minimum: 0.0000000001

timestep The time step to be used (in ps) during an Impact dynamics simulation.

Valid values: reals
Default value: **0.001**
Minimum: 0.0000001

volumescaling
The method of volume scaling to be used during an NPT Impact Dynamics simulation.

Valid values: centerofmass
atom
Default value: **centerofmass**

impactfastmultipole

Used to set all the options associated with the fast multipole method in Impact.

Syntax:

`impactfastmultipole` *level*=⟨ n ⟩ *maximum*=⟨ n ⟩
smoothing=yes | no

Options:

level The level for the fast multipole method.

Valid values:	integers
Default value:	**2**
Minimum:	1

maximum The maximum for the fast multipole method.

Valid values:	integers
Default value:	**7**
Minimum:	4
Maximum:	20

smoothing Whether or not to use smoothing with the fast multipole method.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

impactfrozenatom

Specifies a single atom to be frozen at its current position in an Impact calculation.

Syntax:

`impactfrozenatom` ⟨ atom_number ⟩

Operands:
⟨ atom_number ⟩
The number of an atom which is to be treated as frozen during an Impact calculation.

impactfrozenset

Specifies a set of atoms to be frozen at their current positions in an Impact calculation.

Syntax:

`impactfrozenset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be treated as frozen in an Impact calculation.

impacthybridmc

Settings associated with hybrid Monte Carlo simulations in Impact.

Syntax:

`impacthybridmc` *ncycles*=⟨ n ⟩ *nmdmc*=⟨ n ⟩ *timestep*=⟨ x ⟩

Options:

ncycles The total number of HMC cycles to be performed during an Impact hybrid Monte Carlo simulation.

Valid values:	integers
Default value:	**100**
Minimum:	1

nmdmc The number of MD steps per HMC cycle to be performed during an Impact hybrid Monte Carlo simulation.

Valid values:	integers
Default value:	**4**
Minimum:	1

timestep The time step to be used (in ps) during an Impact hybrid Monte Carlo simulation

Valid values:	reals
Default value:	**0.001**
Minimum:	0.0000001

impactjob

This keyword is used to set various options associated with running Impact jobs.

Syntax:

`impactjob` *host*=⟨ text ⟩ *incorporate*=append | replace | ignore | appendungrouped | workspace *job*=⟨ text ⟩ *login*=⟨ text ⟩ *structure_source*=selected_entries | workspace | file

Options:

host The name of the host for the Impact job.

Valid values: text strings
Default value:

incorporate
How the results are to be incorporated into the project. This can be done with replacement of the existing entries, by appending as new entries to the project or by ignoring the final results.

Valid values: append
replace
ignore
appendungrouped
workspace
Default value: **append**

job The name for the Impact job.

Valid values: text strings
Default value: **impacttmp**

login The login name under which a Impact will be run.

Valid values: text strings
Default value:

structure_source
Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

impactmdparams

Used to set parameters associated with molecular dynamics simulations in Impact.

Syntax:

impactmdparams *every*=⟨ n ⟩ *fastfreq*=⟨ n ⟩ *hmcintegrator*=verlet | rrespa *integrator*=verlet | rrespa *mdstatistics*=yes | no *mediumfreq*=⟨ n ⟩ *nprint*=⟨ n ⟩ *slowfreq*=⟨ n ⟩ *stoprot*=yes | no *traj*=yes | no *trajfile*=⟨ text ⟩ *trajvelocities*=yes | no

Options:

every An option which determines how many MD steps between saving a frame of the trajectory for an Impact dynamics simulation.

Valid values: integers
Default value: **5**
Minimum: 1

fastfreq RRESPA frequency for the fast forces.

Valid values: integers
Default value: **4**
Minimum: 1

hmcintegrator
Which integration technique is to be used during the Hybrid Monte Carlo simulation

Valid values: verlet
rrespa
Default value: **rrespa**

integrator Which integration technique is to be used during the dynamics simulation

Valid values: verlet
rrespa
Default value: **verlet**

mdstatistics
An option which determines if statistics are to be gathered during an Impact dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

mediumfreq
RRESPA frequency for the medium forces.

Valid values: integers

Default value: **2**
Minimum: 1

nprint The number of steps at which output will be written during an Impact dynamics simulation.

Valid values: integers
Default value: **5**
Minimum: 0

slowfreq RRESPA frequency for the slow forces.

Valid values: integers
Default value: **1**
Minimum: 1

stoprot An option which determines if overall rotation and translation is stopped during the dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

traj An option which determines if the trajectory is to be saved during an Impact dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

trajfile The name of the trajectory file.

Valid values: text strings
Default value: **trajectory.trj**

trajvelocities
An option which determines if the velocities are to be saved during an Impact dynamics simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impactminimization

Used to set up a minimization in Impact.

Syntax:

`impactminimization` *algorithm*=newton | conjugate | steepest
convergence=eadng | energy | gradient *cutoff_1*=⟨ x ⟩
cutoff_2=⟨ x ⟩ *energy_change*=⟨ x ⟩ *gradient*=⟨ x ⟩
initial_step_size=⟨ x ⟩ *maximum_step_size*=⟨ x ⟩ *ncycles*=⟨ n ⟩
nfull_1=⟨ n ⟩ *nfull_2*=⟨ n ⟩ *qsite_energychange*=⟨ x ⟩
qsite_gradient=⟨ x ⟩ *qsite_ncycles*=⟨ n ⟩
qsiteminialgorithm=newton | conjugate | steepest

Options:

algorithm Which algorithm is to be used for the minimization

Valid values:	newton conjugate steepest
Default value:	**newton**

convergence

How convergence is to be established during the minimization.

Valid values:	eadng energy gradient
Default value:	**eadng**

cutoff_1 Minimization parameter for Truncated Newton. Long range force cutoff.

Valid values:	reals
Default value:	**10**
Minimum:	0.0

cutoff_2 Minimization parameter for Truncated Newton. Long range force cutoff.

Valid values:	reals
Default value:	**10**
Minimum:	0.0

energy_change

The energy change convergence criterion.

Valid values:	reals
Default value:	**1e-07**
Minimum:	0.0

gradient The gradient convergence criterion.

Valid values:	reals
Default value:	**0.01**
Minimum:	0.0

initial_step_size

The initial steps size for the minimization

Valid values: reals
Default value: **0.05**
Minimum: 0.0

maximum_step_size
The maximum step size for the minimization.
Valid values: reals
Default value: **1**
Minimum: 0.0

ncycles The maximum number of minimization cycles to be performed in non-QSite jobs.
Valid values: integers
Default value: **100**
Minimum: 0

nfull_1 Minimization parameter for Truncated Newton. Update long range forces every X steps.
Valid values: integers
Default value: **10**
Minimum: 0

nfull_2 Minimization parameter for Truncated Newton. Update long range forces every X steps.
Valid values: integers
Default value: **10**
Minimum: 0

qsite_energychange
The energy change convergence criterion for QSite.
Valid values: reals
Default value: **0.1**
Minimum: 0.0

qsite_gradient
The gradient convergence criterion for QSite.
Valid values: reals
Default value: **0.01**
Minimum: 0.0

qsite_ncycles
The maximum number of minimization cycles to be performed for QSite jobs.
Valid values: integers
Default value: **1000**

qsiteminialgorithm
Which algorithm is to be used for the minimization for qsite jobs.

Valid values:	newton conjugate steepest
Default value:	**newton**

impactperiodicboundary

Used to set all the options associated with the periodic boundary conditions in Impact.

Syntax:

`impactperiodicboundary` *alpha*=⟨ x ⟩ *ewald*=yes | no
kvectormax=⟨ n ⟩ *xsize*=⟨ x ⟩ *ysize*=⟨ x ⟩ *zsize*=⟨ x ⟩

Options:

alpha The alpha factor for the Ewald long-range correction.

Valid values:	reals
Default value:	**0.25**
Minimum:	0.0

ewald Whether or not to use the Ewald long-range correction with the fast multipole method.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

kvectormax
The maximum length of the k-space vectors in the Ewald long-range correction.

Valid values:	integers
Default value:	**5**
Minimum:	1

xsize The size of the box in the X-dimension.

Valid values:	reals
Default value:	**18.65**
Minimum:	18.65

ysize The size of the box in the Y-dimension.

Valid values:	reals
Default value:	**18.65**
Minimum:	18.65

zsize The size of the box in the Z-dimension.

Valid values: reals
Default value: **18.65**
Minimum: 18.65

impactpotential

Set various options associated with the definition of the potential energy to be used in a Impact job.

Syntax:

`impactpotential` *contiuum_solvent*=yes | no *dielectric*=⟨ x ⟩ *electrostatics*=constant | distance_dependant *fast_multipole*=yes | no *field*=oplsaa | opls1999 | opls2001 | opls2005 | opls2008 *force_field_checks*=yes | no *paramfile*=⟨ text ⟩ *partial_charges*=yes | no *periodic_boundary*=yes | no *qsitecontiuum_solvent*=yes | no *qsitedielectric*=⟨ x ⟩ *qsiteelectrostatics*=constant | distance_dependant *qsitefield*=oplsaa | opls1999 | opls2001 | opls2005 | opls2008 *qsitetruncate*=yes | no *truncate*=yes | no

Options:

contiuum_solvent
Whether or not to use the continuum solvent
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dielectric The dielectric constant to be used in the electrostatic part of the energy calculation.
Valid values: reals
Default value: **1**
Minimum: 0.9999999999

electrostatics
The electrostatic treatment to be used in the Impact calculation.
Valid values: constant
distance_dependant
Default value: **constant**

fast_multipole
Whether or not to use the fast multipole method for electrostatic interactions.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

field The force field to be used for the Impact calculation.

Valid values: oplsaa
opls1999
opls2001
opls2005
opls2008
Default value: **opls2005**

force_field_checks
Whether or not to skip force field checks.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

paramfile The name of the parameter file.

Valid values: text strings
Default value: **paramstd.dat**

partial_charges
Whether or not to use atomic partial charges in structure file.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

periodic_boundary
Whether or not to use the periodic boundary conditions

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

qsitecontiuum_solvent
Whether or not to use the continuum solvent

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

qsitedielectric
The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values: reals
Default value: **1**
Minimum: 0.9999999999

qsiteelectrostatics
The electrostatic treatment to be used in the QSite calculation.

Valid values: constant
distance_dependant
Default value: **constant**

qsitefield The force field to be used for the QSite calculation.

Valid values: oplsaa
opls1999
opls2001
opls2005
opls2008
Default value: **opls2005**

qsitetruncate
Whether or not to truncate the non-bonded interactions.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

truncate Whether or not to truncate the non-bonded interactions.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

impactread

Read an Impact input file and set corresponding panels.

Syntax:

`impactread` ⟨ filename ⟩

Operands:
⟨ filename ⟩
The name of an impact input file.

impactrepexch

Settings associated with Replica Exchange in Impact

Syntax:

`impactrepexch` *dorxmd*=yes | no *nexch*=⟨ n ⟩ *nrepl*=⟨ n ⟩
reftempid=⟨ n ⟩ *restart*=yes | no *restartfile*=⟨ text ⟩
targettemp=⟨ text ⟩

Options:

dorxmd Set the mode to Replica Exchange
Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

nexch The number of MD steps between Exchanges
Valid values: integers
Default value: **250**
Minimum: 1

nrepl The number of Replicas
Valid values: integers
Default value: **2**
Minimum: 2
Maximum: 100

reftempid The index of temperature of interest
Valid values: integers
Default value: **1**
Minimum: 1

restart Set the mode to restart a previous Replica Exchange job.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

restartfile Restart input file name.
Valid values: text strings
Default value:

targettemp
List of target temperatures separated by commas
Valid values: text strings
Default value: **300.00, 350.00**

impactselectextendtablerow

Extends the selection to this row in the given orbital table in the Surfaces Property for Impact.

Syntax:

`impactselectextendtablerow` *table*=alpha | beta ⟨row⟩

Options:

table Indicates whether to select a row in the alpha or beta orbital table.
Valid values: alpha
beta

Default value: **alpha**

Operands:
⟨ row ⟩
The row number to extend the select to.

impactselectonlytablerow

Selects only this row in the given orbital table in the Surface Property for Impact.

Syntax:

`impactselectonlytablerow` *table*=alpha | beta ⟨ row ⟩

Options:

table Indicates whether to select a row in the alpha or beta orbital table.
Valid values: alpha
beta
Default value: **alpha**

Operands:
⟨ row ⟩
The row number to select only in the table row.

impactstart

Start an Impact input file with the current settings.

Syntax:

`impactstart`

impacttask

Determines which impact task is currently being set up.

Syntax:

`impacttask` soak | mini | dynamics | hmc | qsite | glide | liaison

Operands:

soak | mini | dynamics | hmc | qsite | glide | liaison

The type of impact task which is to be set up. The operand cannot be abreviated and must be given in full.

impacttruncation

Used to set all the options associated with the truncation of non-bonded interactions in Impact.

Syntax:

`impacttruncation` *distance*=⟨ x ⟩ *qsitedistance*=⟨ x ⟩
qsiteupdatefrequency=⟨ n ⟩ *updatefrequency*=⟨ n ⟩

Options:

distance The truncation distance for residue-based cutoffs.

Valid values:	reals
Default value:	**12**
Minimum:	0.0

qsitedistance

The truncation distance for residue-based cutoffs.

Valid values:	reals
Default value:	**12**
Minimum:	0.0

qsiteupdatefrequency

The number of steps between update of the neighbor list.

Valid values:	integers
Default value:	**10**
Minimum:	1

updatefrequency

The number of steps between update of the neighbor list.

Valid values:	integers
Default value:	**10**
Minimum:	1

impactwrite

Write an Impact input file with the current settings.

Syntax:

`impactwrite`

invert

Inverts the chirality around a chiral atom.

Syntax:

`invert` ⟨ chiral_atom ⟩ ⟨ non_moving_atom1 ⟩ ⟨ non_moving_atom2 ⟩

Operands:

⟨ chiral_atom ⟩ ⟨ non_moving_atom1 ⟩ ⟨ non_moving_atom2 ⟩

Three atom numbers. The first is the atom around which the chirality is to be inverted. The second and third are atoms which are attached to the chiral atom but are not to be moved in the inversion process.

invertset

Inverts all chiral centers in the specified set of atoms.

Syntax:

`invertset` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

An ASL specification of the atoms which are to have their chiral centers inverted. Because of the way this works, this set of atom should involve at least whole molecules.

jaguarassignatomnames

Sets the atom names of the matching atoms as per the jaguar standards i.e, element name + atom number. Incase of any duplicates, integer part of the name is incremented till it is unique in the Workspace

Syntax:

`jaguarassignatomnames` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression describing the set of atoms which are to have their atom names made unique.

jaguarimportgeometry

Import structures into the current project from a Jaguar input file.

Syntax:

`jaguarimportgeometry` *all*=yes | no *end*=yes | no *start*=⟨ n ⟩ *total*=⟨ n ⟩ *wsinclude*=none | first | all *wsreplace*=yes | no [⟨ filename ⟩]

Options:

all This determines whether all structures will be imported, or just a specified range.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

end This determines if all structures in the file are to be imported starting from the structure specified by start.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

start This option sets the number of the first structure to be imported, if not importing all.
Valid values: integers
Default value: **1**
Minimum: 1

total The total number of structures to be imported from the file, if not importing all structures.

Valid values: integers
Default value: **1**
Minimum: 1

wsinclude This option determines which of the imported structures are to be included in the workspace. Valid values are "none", "first", or "all".

Valid values: none
first
all
Default value: **none**

wsreplace This determines whether the structures currently in the workspace will be replaced by the included imported structures, or whether they will be be included in the workspace also.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:

[⟨ filename ⟩]
The name of the file from which structures will be imported. If no name is specified, then no import will be done.

jaguarinputfilesjob

This keyword is used to set various options associated with running Jaguar input files jobs.

Syntax:

`jaguarinputfilesjob` *input_files*=⟨ text ⟩

Options:

input_files Specifies existing structure file to be used as input files for running the job

Valid values: text strings
Default value:

jaguarjob

This keyword is used to set various options associated with running Jaguar jobs.

Syntax:

`jaguarjob` *structure_files*=⟨ text ⟩ *structure_source*=selected_entries | workspace | file

Options:

structure_files

If structure_source is Selected structure files , specifies existing structure file to be used as input files for running the job.

Valid values: text strings
Default value:

structure_source

Whether to use the selected entries in the current project or what is in the workspace as input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

jaguarselectextendtablerow

Extends the selection to this row in the given orbital table in the Surfaces Property for Jaguar.

Syntax:

`jaguarselectextendtablerow` *table*=alpha | beta ⟨ row ⟩

Options:

table Indicates whether to select a row in the alpha or beta orbital table.

Valid values: alpha
beta
Default value: **alpha**

Operands:
⟨ row ⟩
The row number to extend the select to.

jaguarselectonlytablerow

Selects only this row in the given orbital table in the Surface Property for Jaguar.

Syntax:

`jaguarselectonlytablerow` *table*=alpha | beta ⟨ row ⟩

Options:

table Indicates whether to select a row in the alpha or beta orbital table.
Valid values: alpha, beta
Default value: **alpha**

Operands:
⟨ row ⟩
The row number to select only in the table row.

jobcleanup

Specifies a job to cleanup.

Syntax:

`jobcleanup` *files*=jobandmonitor | all ⟨ job_id ⟩

Options:

files This option determines which files are removed when the job is cleaned up. Either just the job record and monitoring files or all files associated with the job.
Valid values: jobandmonitor, all
Default value: **jobandmonitor**

Operands:
⟨ job_id ⟩
The ID of the job which is to be cleaned up.

jobsettings

This keyword is used to set various options associated with running a backend job.

Syntax:

`jobsettings` *compress*=yes | no *directory*=⟨ text ⟩
distributesubjobs=maxprocessors | specifiedprocessors
gpuindices=⟨ text ⟩ *host*=⟨ text ⟩ *hostlist*=⟨ text ⟩
hostsubjobs=⟨ n ⟩ *incorporate*=appendentries | replaceentries | ignoreentries | ingoreentries | appendentriesungrouped | workspace *jobname*=⟨ text ⟩ *login*=⟨ text ⟩ *maxstructures*=⟨ n ⟩
numcpus=⟨ n ⟩ *numsubjobs*=⟨ n ⟩ *subjobsprocessors*=⟨ n ⟩
title=⟨ text ⟩ *tmpdir*=⟨ text ⟩ *writedirectory*=⟨ text ⟩
⟨ model_name ⟩

Options:

compress An option which determines if output files are generated in an mmzip archive.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

directory The directory to write the output to.
Valid values: text strings
Default value:

distributesubjobs
How the subjobs are to be distributed over the available processors
Valid values: maxprocessors
specifiedprocessors
Default value: **maxprocessors**

gpuindices The GPU indices for ri=unning jobs.
Valid values: text strings
Default value:

host The name of the host for the backend job.

Valid values: text strings
Default value: **host**

hostlist The string of hosts and number of CPUs which will be passed to the job.

Valid values: text strings
Default value:

hostsubjobs

The number of subjobs this current job is to be run as.

Valid values: integers
Default value: **1**
Minimum: 1

incorporate

How the results are to be incorporated into the project. This can be done by appending as new entries to the project, replacing entries, or by ignoring the final results.

Valid values: appendentries
replaceentries
ignoreentries
ingoreentries
appendentriesungrouped
workspace
Default value: **appendentries**

jobname The name for the backend job.

Valid values: text strings
Default value:

login The login name under which the job will be run.

Valid values: text strings
Default value: **user**

maxstructures

The maximum number of structures to be incorporated into the project.

Valid values: integers
Default value: **100**
Minimum: 0

numcpus The number of CPUs to be used for jobs that support multiple CPUs.

Valid values: integers
Default value: **1**
Minimum: 1

numsubjobs
The number of subjobs this current job is to be run as.

Valid values: integers
Default value: **1**
Minimum: 1

subjobsprocessors
The number of processors to distribute the subjobs over

Valid values: integers
Default value: **1**
Minimum: 1

title Set the input entry's title.

Valid values: text strings
Default value:

tmpdir The path name of the scratch directory.

Valid values: text strings
Default value:

writedirectory
The directory to write job input files to.

Valid values: text strings
Default value:

Operands:

⟨ model_name ⟩

The name of a model which has job settings like host, login, and so on.

kill

This is a standard alias for **energykill** (see [energykill], page 134).

labelatom

Displays labels for the atoms specified by the ASL operand.

Syntax:

`labelatom` *1charge*=yes | no *1chargeformatstring*=⟨ text ⟩
2charge=yes | no *2chargeformatstring*=⟨ text ⟩
acolor=yes | no *anum*=yes | no *anumformatstring*=⟨ text ⟩
atomicnumber=yes | no *atomname*=yes | no
atomnameformatstring=⟨ text ⟩ *atype*=yes | no
atypeformatstring=⟨ text ⟩ *chain*=yes | no
chainformatstring=⟨ text ⟩ *chirality*=yes | no
chiralityformatstring=⟨ text ⟩ *cindex*=⟨ n ⟩ *color*=⟨ text ⟩
compositionfields=⟨ text ⟩ *dmsopka*=yes | no
dmsopkaformatstring=⟨ text ⟩ *element*=yes | no
elementformatstring=⟨ text ⟩ *entryname*=yes | no
entrynameformatstring=⟨ text ⟩
entrypropertyformatstring=⟨ text ⟩ *entrypropertynames*=⟨ text ⟩
font=⟨ text ⟩ *font_size*=⟨ n ⟩ *font_style*=normal | italic | bold | bolditalic *formalcharge*=yes | no
formalchargeformatstring=⟨ text ⟩ *growname*=yes | no
grownameformatstring=⟨ text ⟩ *h2opka*=yes | no
h2opkaformatstring=⟨ text ⟩ *headings*=yes | no
inscode=yes | no *keeplabels*=on_top | with_atom
mode=replace | append | clear *molnum*=yes | no
molnumentry=yes | no *molnumentryformatstring*=⟨ text ⟩
molnumformatstring=⟨ text ⟩ *numentry*=yes | no
numentryformatstring=⟨ text ⟩ *nummol*=yes | no
nummolformatstring=⟨ text ⟩ *occupancy*=yes | no
occupancyformatstring=⟨ text ⟩ *oneletter*=yes | no
pdbbfactor=yes | no *pdbbfactorformatstring*=⟨ text ⟩
pdbname=yes | no *pdbnameformatstring*=⟨ text ⟩
reapply=yes | no *reapplylabels*=yes | no *resname*=yes | no
resnameformatstring=⟨ text ⟩ *resnum*=yes | no
resnumformatstring=⟨ text ⟩ *separator*=⟨ text ⟩
showlabel=yes | no *stereochemistry*=yes | no
stereochemistryformatstring=⟨ text ⟩ *title*=yes | no
titleformatstring=⟨ text ⟩ *user*=yes | no
useratompropertyformatstring=⟨ text ⟩ *utext*=⟨ text ⟩
xoffset=⟨ x ⟩ *xyz*=yes | no *xyzformatstring*=⟨ text ⟩
yoffset=⟨ x ⟩ ⟨ ASL ⟩

Options:

1charge A boolean option which determines if the charge 1 value will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

1chargeformatstring
A string which determines which format is to be used for the charge 1 value labels

Valid values: text strings
Default value: **%C1.3**

2charge A boolean option which determines if the charge 2 value will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

2chargeformatstring
A string which determines which format is to be used for the charge 2 value labels

Valid values: text strings
Default value: **%C2.3**

acolor A boolean option which determines if labels will be colored the same as the atoms they label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

anum A boolean option which determines if the atom numbers will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

anumformatstring
A string which determines which format is to be used for the atom number labels

Valid values: text strings
Default value: **%NU**

atomicnumber
A boolean option which determines if the atomic number will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

atomname A boolean option which determines if the atom name will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

atomnameformatstring
A string which determines which format is to be used for the atom name labels

Valid values: text strings
Default value: **%AT**

atype A boolean option which determines if the MacroModel atom types will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

atypeformatstring
A string which determines which format is to be used for the MacroModel atom type labels

Valid values: text strings
Default value: **%TY**

chain A boolean option which determines if the chain name will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

chainformatstring
A string which determines which format is to be used for chain name labels

Valid values: text strings
Default value: **%CH**

chirality A boolean option which determines if the atom chirality (R or S) will be included in the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

chiralityformatstring
A string which determines which format is to be used for the atom chirality labels

Valid values: text strings
Default value: **%CY**

cindex An integer which indicates color index which is to be used for the labels. This will be ignored unless the acolor option is off

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 256

color A string which is the color name for atom labels. This will be ignored unless the acolor option is off

Valid values: text strings
Default value:

compositionfields
This option sets the names of the properties (semicolon separated list) to be used to compose the labels for selected atoms.

Valid values: text strings
Default value:

dmsopka A boolean option which determines if the pka value in DMSO solvent will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dmsopkaformatstring
A string which determines which format is to be used for the labels of pka value in DMSO solvent

Valid values: text strings
Default value: **%DP**

element A boolean option which determines if the element symbol will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

elementformatstring
A string which determines which format is to be used for the element symbol labels

Valid values: text strings
Default value: **%EL**

entryname
A boolean option which determines if the entry name will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

entrynameformatstring
A string which determines which format is to be used for the entry name labels

Valid values: text strings
Default value: **%EN**

entrypropertyformatstring
A string which determines which format is to be used for the ct property labels

Valid values: text strings
Default value: **%EP**

entrypropertynames
This option sets the names of the entry properties (semicolon separated list) to be used to compose the labels for selected atoms.
Valid values: text strings
Default value:

font A string which determines which font is to be used for the labels
Valid values: text strings
Default value:

font_size An integer determines which font size to be used for the labels
Valid values: integers
Default value: **14**
Minimum: 0

font_style An integer determines which font style to be used for the labels
Valid values: normal
italic
bold
bolditalic
Default value: **bold**

formalcharge
A boolean option which determines if the formal charge will be included in the atom label
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

formalchargeformatstring
A string which determines which format is to be used for the formal charge labels
Valid values: text strings
Default value: **%FC**

growname A boolean option which determines if the grow name will be included in the atom label
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

grownameformatstring
A string which determines which format is to be used for the grow name labels
Valid values: text strings
Default value: **%GN**

h2opka A boolean option which determines if the pka value in H2O solvent will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

h2opkaformatstring
A string which determines which format is to be used for the labels of pka value in H2O solvent

Valid values: text strings
Default value: **%HP**

headings A boolean option which determines if the label fields will have headers in the labels (anum=1, atype=C2 and so on).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

inscode A boolean option which determines if the pdb residue insertion code will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

keeplabels This option controls whether labels are kept on top of atoms or whether they are kept at the atom level. Keeping the labels at atom level means labels will be obscured by other atoms and objects in the Workspace.

Valid values: on_top
with_atom
Default value: **on_top**

mode Specifies one of two modes: 1) replace: replace the existing label string with the new one. 2) append: append to the existing label string

Valid values: replace
append
clear
Default value: **replace**

molnum A boolean option which determines if the molecule number will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

molnumentry
A boolean option which determines if the molecule number by entry will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

molnumentryformatstring
A string which determines which format is to be used for molecule number by entry labels

Valid values: text strings
Default value: **%ME**

molnumformatstring
A string which determines which format is to be used for the molecule number labels

Valid values: text strings
Default value: **%MN**

numentry A boolean option which determines if the atom number by entry will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numentryformatstring
A string which determines which format is to be used for the atom number by entry labels

Valid values: text strings
Default value: **%NE**

nummol A boolean option which determines if the atom number by molecule will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

nummolformatstring
A string which determines which format is to be used for the atom number by molecule labels

Valid values: text strings
Default value: **%NM**

occupancy A boolean option which determines if the partial occupancy data will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

occupancyformatstring
A string which determines which format is to be used for the partial occupancy data labels

Valid values: text strings
Default value: **%OC**

oneletter A boolean option which determines whether single letter pdb residue name is displayed or three letters.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pdbbfactor A boolean option which determines if the pdb bfactor will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pdbbfactorformatstring
A string which determines which format is to be used for the pdb bfactor labels

Valid values: text strings
Default value: **%BF**

pdbname A boolean option which determines if the pdb atom name will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pdbnameformatstring
A string which determines which format is to be used for pdb name labels

Valid values: text strings
Default value: **%PA**

reapply A boolean option which determines if reapplying current atom labels when Workspace changes, by using last used option in the main toolbar Label atoms button menu

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

reapplylabels
A boolean option which determines if reapplying current atom labels when Workspace changes, by using the options specified in the Atom Labels panel

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

resname A boolean option which determines if the pdb residue name will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

resnameformatstring
A string which determines which format is to be used for pdb residue name labels

Valid values: text strings
Default value: **%RT**

resnum A boolean option which determines if the residue number will be included in the label string

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

resnumformatstring
A string which determines which format is to be used for the residue number labels

Valid values: text strings
Default value: **%RN**

separator A string option which is the inserted between each field in the label string.

Valid values: text strings
Default value:

showlabel A boolean option which determines if an atom label is displayed when the atom itself is undisplayed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

stereochemistry
A boolean option which determines if the double bond strerochemistry (E or Z) will be included in the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

stereochemistryformatstring
A string which determines which format is to be used for the stereochemistry labels

Valid values: text strings
Default value: **%ST**

title A boolean option which determines if the entry title will be included in the atom label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

titleformatstring
A string which determines which format is to be used for the entry title labels

Valid values: text strings
Default value: **%ET**

user A boolean option which determines if the user-defined text will be included in the label

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

useratompropertyformatstring
A string which determines which format is to be used for the user specified atom property labels

Valid values: text strings
Default value: **%UA**

utext This option sets the user defined text for atom labels

Valid values: text strings
Default value:

xoffset A double option which sets the x-offset from the atom center at which the labels are to appear.

Valid values: reals
Default value: **0.05**

xyz A boolean option which determines if the xyz location of the atom will be included in the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

xyzformatstring
A string which determines which format is to be used for the xyz location labels

Valid values: text strings
Default value: **%XY**

yoffset A double option which sets the y-offset from the atom center at which the labels are to appear.

Valid values: reals
Default value: **-0.15**

Operands:
⟨ ASL ⟩
A string in the atom specification language. All atoms which match this specification will be labeled with the current label settings.

labelatomfieldsnone

Allows user to turn off all the atom label fields in one simple command instead of having to issue a labelatom command with all of the options set to false: anum=false etc.

Syntax:

`labelatomfieldsnone`

labelatomoffset

Set the offset for the specified atoms. If not specified, then Maestro uses the offset specified in preferences (i.e. last value specified to the labelatom command).

Syntax:

`labelatomoffset` *x*=⟨ x ⟩ *y*=⟨ x ⟩ ⟨ ASL ⟩

Options:

x	Specifies x offset	
	Valid values:	reals
	Default value:	**0**
y	Specifies y offset	
	Valid values:	reals
	Default value:	**0**

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have their offset modified.

labelatomoffsetreset

Remove any customized label offsets for the specified atoms. Note: any entry included after this command that has customized label offsets will display with those customized offsets. This command only removes offsets for those labels (structures) in the Workspace.

Syntax:

`labelatomoffsetreset`

labelclear

Clears labels from the atoms specified by the ASL operand.

Syntax:

`labelclear` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have their labels removed.

labeldisplayusertext

Displays user text labels for the atoms specified by the ASL operand.

Syntax:

`labeldisplayusertext` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have their previously defined user text label shown in the workspace.

labelformat

Sets the label format string which will be used in subsequent labeling operations.

Syntax:

`labelformat` ⟨ format_string ⟩

Operands:

⟨ format_string ⟩

A string which defines the current labeling format. See on-line help for details.

labelupdate

Apply the current labelatom format to all the atoms that have a non-empty atom label format. This does not apply to atoms with a heteroatom element label format that is generated automatically from the elementlabels scheme (i.e., to Hetatom Labels). The update is based on the atom label format, regardless of whether any non-blank label is displayed for the atom in the Workspace.

Syntax:

`labelupdate` [NON_EL]

Operands:

[NON_EL]

An optional string, to indicate that only atom labels, not elementlabels, are relabelled. This argument is now ignored.

labelusertextatomupdate

Update the user text for the specified atoms

Syntax:

`labelusertextatomupdate` *utext*=⟨ text ⟩ ⟨ ASL ⟩

Options:

utext This option sets the user defined text for atom relabeling

Valid values: text strings
Default value:

Operands:

⟨ ASL ⟩

ASL of atoms to relabel

liaisonanalysis

Used to do the analysis

Syntax:

`liaisonanalysis` *liaisonalpha*=⟨ x ⟩ *liaisonanalysistype*=fit | predict *liaisonbefile*=⟨ text ⟩ *liaisonbeta*=⟨ x ⟩ *liaisonenerymodeltype*=equation | glidescore *liaisonfitname*=⟨ text ⟩ *liaisonfixalpha*=yes | no *liaisonfixbeta*=yes | no *liaisonfixgamma*=yes | no *liaisonfixintercept*=yes | no *liaisonfixslope*=yes | no *liaisongamma*=⟨ x ⟩ *liaisonintercept*=⟨ x ⟩ *liaisonlistfile*=⟨ text ⟩ *liaisonslope*=⟨ x ⟩ *liaisonspecligs*=readligs | enterligs

Options:

liaisonalpha

This option sets the van der Waals (Alpha) coefficient.

Valid values: reals
Default value: **0**

liaisonanalysistype

This option determines the analysis type, including two options of Predict (predict) and Fit (fit).

Valid values: fit
predict
Default value: **fit**

liaisonbefile

This option determines the name of the ligand binding energy file

Valid values: text strings
Default value:

liaisonbeta

This option sets the electrostatic (Beta) coefficient.

Valid values: reals
Default value: **0**

liaisonenerymodeltype

This option determines which binding enery model is used, LIA equation or Glidescore

Valid values: equation
glidescore
Default value: **equation**

liaisonfitname

This option determines the names of ligands to be predicted

Valid values: text strings
Default value:

liaisonfixalpha

Allows the alpha value to be used as a constraint.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixbeta

Allows the beta value to be used as a constraint.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixgamma

Allows the gamma value to be used as a constraint

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixintercept

Allows the Glidescore intercept to be used as a constraint

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisonfixslope

Allows the glidescore slope to be used as a constraint.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

liaisongamma

This option sets cavity (Gamma) coefficient.

Valid values: reals
Default value: **0**

liaisonintercept

This option sets the Glidescore intercept

Valid values: reals
Default value: **0**

liaisonlistfile

This option determines the name of file that contains names of ligands to be predicted

Valid values: text strings
Default value:

liaisonslope

This option sets the Glidescore slope

Valid values: reals
Default value: **0**

liaisonspecligs

This option determines the methods used to specify the ligands either by reading ligand names from a text file (readligs), or entering a comma separated list of ligand names (enterligs).

Valid values: readligs
enterligs
Default value: **readligs**

liaisonparameters

Used to specify simulation parameters in a Liasion (Linear Interaction Approximation).

Syntax:

`liaisonparameters` *algorithm*=newton | conjugate | steepest
liaisonboundgcut=⟨ x ⟩ *liaisonboundheat*=⟨ x ⟩
liaisonboundint=⟨ n ⟩ *liaisonboundmini*=⟨ n ⟩
liaisonboundpremini=⟨ n ⟩ *liaisonboundprod*=⟨ x ⟩
liaisonfreeint=⟨ n ⟩ *liaisonliggcut*=⟨ x ⟩ *liaisonligheat*=⟨ x ⟩
liaisonligmini=⟨ n ⟩ *liaisonligpremini*=⟨ n ⟩ *liaisonligprod*=⟨ x ⟩
liaisonmethod=mini | hmc | dyn *liaisonreltime*=⟨ x ⟩
liaisonrescut=⟨ x ⟩ *liaisontemp*=⟨ x ⟩ *minibound*=yes | no
minilig=yes | no

Options:

algorithm Which algorithm is to be used for the minimization

Valid values: newton
conjugate
steepest
Default value: **newton**

liaisonboundgcut

This option determines the RMS gradient for convergence (in kcal/mol/A)

Valid values: reals
Default value: **0.05**

liaisonboundheat

This option determines the time for heating/ equilibration (ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonboundint

This option determines the number of steps for collecting Liaison statistics

Valid values: integers
Default value: **10**
Minimum: 0

liaisonboundmini

This option determines the maximum number of minimization steps

Valid values: integers
Default value: **500**
Minimum: 0

liaisonboundpremini

This option determines the maximum number of minimization steps

Valid values: integers
Default value: **500**
Minimum: 0

liaisonboundprod

This option determines the time for data collection (in ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonfreeint

This option determines the number of steps for collecting Liasion statistics

Valid values: integers
Default value: **10**
Minimum: 0

liaisonliggcut

This option determines the RMS gradient for convergence

Valid values: reals
Default value: **0.01**
Minimum: 0.0

liaisonligheat

This option determines the time for heating/ equilibration (ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonligmini

This option determines the maximum number of minimization steps

Valid values: integers
Default value: **500**
Minimum: 0

liaisonligpremini

This option determines maximum number of minimization steps

Valid values: integers
Default value: **1000**
Minimum: 0

liaisonligprod

This option determines the time for data collection (in ps)

Valid values: reals
Default value: **5**
Minimum: 0.0

liaisonmethod

This option determines which sampling method will be used during a Liaison analysis. The 3 options are minimization (mini), Hybrid Monte Carlo (hmc) and Molecular dynamics (dyn).

Valid values: mini
hmc
dyn
Default value: **mini**

liaisonreltime

This option determines the temperature relaxation time

Valid values: reals
Default value: **0.01**
Minimum: 0.0

liaisonrescut

This option determines the residue-based cutoff for nonbonded interactions

Valid values: reals
Default value: **15**
Minimum: 0.0

liaisontemp

This option determines the simulation temperature

Valid values: reals
Default value: **300**
Minimum: 0.0

minibound
This option determines whether the ligand- receptor complex is minimized before running simulation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

minilig This option determines whether the ligand is minimized before running the simulation

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

liaisonselectlig

Defines a on-screen molecule to be treated as the ligand for a Liaison calculation

Syntax:

`liaisonselectlig` ⟨ molecule_number ⟩

Operands:
⟨ molecule_number ⟩
The number of a molecule to be included as the ligand.

liaisonsettings

Used to set values associated with the Linear Interaction Approximation (Liaison)

Syntax:

`liaisonsettings` *liaisonsubdir*=⟨ text ⟩ *liaisontype*=sim | analyze *numproc*=⟨ n ⟩

Options:

liaisonsubdir
This option determines the subdirectory to use for Liaison jobs.

Valid values: text strings
Default value: **liaison**

liaisontype
This option determines the job type for Liaison analysis. There are two types: (1) Simulate (sim); (2) Analyze results of earlier simulations (analyze)

Valid values: sim
analyze
Default value: **sim**

numproc This option determines the number of processors the job is to be run on.

Valid values: integers
Default value: **1**
Minimum: 1

liaisonsystem

Used to specify receptor and ligand(s) to be simulated in the Linear Interaction Approximation (Liaison)

Syntax:

`liaisonsystem` *haslialig*=yes | no *liaisonligfmt*=m2io | sd | mol2 | pdb *liaisonligfname*=⟨ text ⟩ *liaisonsimtype*=mult | single *liaisonsname*=⟨ text ⟩ *ligftype*=ligstruct | liglist *usepartialcharges*=yes | no

Options:

haslialig This option determines whether the displayed structure includes the ligands

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

liaisonligfmt
This option for selecting the format of the ligand file. The four possible formats are MAESTRO (m2io), MDL SD (sd), MOL2 (mol2) and PDB (pdb)

Valid values: m2io
sd
mol2
pdb
Default value: **m2io**

liaisonligfname
This option determines the name of the ligand file

Valid values: text strings
Default value:

liaisonsimtype
This option determines the type of ligand(s)-receptor structures to be simulated, having two options: (1) Multiple ligands, single receptor; (2) Single ligand, single receptor.

Valid values: mult
single
Default value: **mult**

liaisonsname
This option determines the name to be used for this ligand

Valid values: text strings
Default value: **Lig**

ligftype This option detemines the source of ligand structure(s), having two options: (1) File containing a single ligand structure (ligstruct); and (2) File containing a list of ligand names and structure files (liglist).

Valid values: ligstruct
liglist
Default value: **ligstruct**

usepartialcharges
This option determines whether the to use partial charges from the Maesro file

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ligandbond

Specifies two atoms which define a bond to be treated as a ligand bond during a conformational search.

Syntax:

`ligandbond` ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The numbers of two atoms which are to be treated as a ligand bond during a conformational search. Note that specifying a-b is the same as specifying b-a.

light

Specifies a light or set light options.

Syntax:

`light` *ambient*=⟨ n ⟩ *diffuse*=⟨ n ⟩ *specular*=⟨ n ⟩ *use*=yes | no
x=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

ambient Ambient lighting component.

Valid values:	integers
Default value:	**20**
Minimum:	0
Maximum:	100

diffuse Diffuse lighting component.

Valid values:	integers
Default value:	**70**
Minimum:	0
Maximum:	100

specular Specular lighting component.

Valid values:	integers
Default value:	**30**
Minimum:	0
Maximum:	100

use Use the light or not (switch it ON or OFF).

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

x The X coordinate of light position.

Valid values:	reals
Default value:	**-1**

y The Y coordinate of light position.

Valid values:	reals
Default value:	**1**

z The Z coordinate of light position.

Valid values:	reals
Default value:	**3**

lightambient

Sets the ambient intensity of the light.

Syntax:

`lightambient` *intensity*=⟨ x ⟩

Options:

intensity	Ambient lighting component.	
	Valid values:	reals
	Default value:	**0.2**
	Minimum:	0.0
	Maximum:	1.0

lightdiffuse

Sets the diffuse intensity of the light.

Syntax:

`lightdiffuse` *intensity*=⟨ x ⟩

Options:

intensity	Diffuse lighting component.	
	Valid values:	reals
	Default value:	**0.7**
	Minimum:	0.0
	Maximum:	1.0

lightposition

Sets the position of the light.

Syntax:

`lightposition` ⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩

Operands:
⟨ x ⟩ ⟨ y ⟩ ⟨ z ⟩

The position of the light.

lightspecular

Sets the specular intensity of the light.

Syntax:

`lightspecular` *intensity*=⟨ x ⟩

Options:

intensity Specular lighting component.

Valid values:	reals
Default value:	**0.3**
Minimum:	0.0
Maximum:	1.0

ligprep

This keyword is used to set various options associated with running LigPrep jobs.

Syntax:

`ligprep` *desalt*=yes | no *epikmetalbinding*=yes | no *filter_file*=⟨ text ⟩ *forcefield*=mmffs | opls2005 | opls2008 *gen_conform*=⟨ n ⟩ *gen_stereo*=⟨ n ⟩ *gen_tautomers*=yes | no *includeoriginalstate*=yes | no *input_file*=⟨ text ⟩ *ionization*=generate | neutralize | retain *ionizationmethod*=ionizer | epik *output_format*=maestro | sdf *ph*=⟨ x ⟩ *ph_tolerance*=⟨ x ⟩ *stereoisomers*=retain | determine | generate *structure_source*=selected_entries | workspace | file

Options:

desalt Desalt

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

epikmetalbinding
Add metal binding states

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

filter_file The name of the filter criteria file.
Valid values: text strings
Default value:

forcefield Force field to use for minimization and filtering
Valid values: mmffs
opls2005
opls2008
Default value: **opls2005**

gen_conform
Generate low energy ring confirmations
Valid values: integers
Default value: **1**
Minimum: 1

gen_stereo Generate stereoisomers (maximum)
Valid values: integers
Default value: **32**
Minimum: 1

gen_tautomers
Generate tautomers
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

includeoriginalstate
Includes original state.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

input_file The name of the structure input file.
Valid values: text strings
Default value:

ionization How to generate the ionization states
Valid values: generate
neutralize
retain
Default value: **generate**

ionizationmethod
Which ionization computation method to use (default or Epik)
Valid values: ionizer
epik
Default value: **ionizer**

output_format
Whether to output the file in Maestro or SDF format.

Valid values:	maestro sdf
Default value:	**maestro**

ph Ionization pH

Valid values:	reals
Default value:	**7**
Minimum:	0.0
Maximum:	14.0

ph_tolerance
Ionization pH tolerance

Valid values:	reals
Default value:	**2**
Minimum:	0.0
Maximum:	7.0

stereoisomers
How to generate the stereoisomers

Valid values:	retain determine generate
Default value:	**retain**

structure_source
Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values:	selected_entries workspace file
Default value:	**file**

ligprepread

Read the given Ligprep options file and set the Ligprep panel options.

Syntax:

`ligprepread` ⟨ filename ⟩

Operands:
⟨ filename ⟩
The name of the ligprep options file to read.

ligprepstart

Start a LigPrep job with the current settings.

Syntax:

`ligprepstart`

ligprepwrite

Write a Ligprep input file with the current settings.

Syntax:

`ligprepwrite`

localatoms

Specify the group of local (transformation) atoms

Syntax:

`localatoms` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
For specifying which atoms are to be locally transformed. A list of atoms is created at the time the command is issued. Use the transform command to specify local scope, then use the rotate and/or translate commands to perform local transformations. To get back to global scope use the transform command to set the scope back to global.

localbitset

Specify whether local bitsets (atoms and center) are updated whenever contents of Workspace change. Unless the lock is enabled, these bitsets are re-evaluated each time atoms are changed, including changes in atom coordinates due to local adjustment.

Syntax:

`localbitset` *lock*=yes | no

Options:

lock Enable/disable lock on local bitsets for as long as ASL expressions and number of atoms in Workspace remain unchanged.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

localcenter

Set local transformation center to given centroid

Syntax:

`localcenter`

logfile

This is a standard alias for **scriptlogfile** (see [scriptlogfile], page 698).

logp

Support calculation of LogP of MacroModel.

Syntax:

`logp` *enable*=yes | no *secsolvent*=none | water | chcl3 | octanol

Options:

enable Determines whether logp will be used with this multiple minimization.

	Valid values:	boolean (true|false; yes|no; y|n; on|off)
	Default value:	**false**
secsolvent	This option determines the second solvent for logp.	
	Valid values:	none water chcl3 octanol
	Default value:	**water**

macrodefine

Define a macro. Definition is a semicolon separated string of maestro commands.

Syntax:

`macrodefine` ⟨ macro_name ⟩ ⟨ definition ⟩

Operands:
⟨ macro_name ⟩ ⟨ definition ⟩
The first operand is the name of the macro. If this contains embedded spaces then it must be enclosed in double quotes. The second operand is the definition. If this contains embedded spaces then it must be enclosed in double quotes. To run the macro, execute macrorun <macro_name>

macrorun

Invokes an already defined macro

Syntax:

`macrorun` ⟨ macro_name ⟩

Operands:
⟨ macro_name ⟩
The operand is the name of the macro. If this contains embedded spaces then it must be enclosed in double quotes. As a shortcut macros can also be run without preceding them with macrorun . Using macrorun has the advantage that if the macro doesn't exist the program will tell you the macro doesn't exist. If you omit macrorun and Maestro cannot find the macro, then

Maestro will report that the command does not exist. This will be misleading and possibly confusing because it's really the macro that will not exist.

makedirectory

Create new directory.

Syntax:

`makedirectory` ⟨ directory_name ⟩

Operands:
⟨ directory_name ⟩
The name of the directory to created.

markers

The markers command creates a new marker.

Syntax:

`markers` *blue*=⟨ x ⟩ *green*=⟨ x ⟩ *red*=⟨ x ⟩ ⟨ marker_name ⟩

Options:

blue The blue component of the color for the new marker.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

green The green component of the color for the new marker.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

red The red component of the color for the new marker.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

Operands:
⟨ marker_name ⟩
The name is the name to use for subsequent showmarkername and hidemarkername commands.

material

Specifies a material or set material options.

Syntax:

`material` *ambient*=⟨ n ⟩ *changed*=yes | no *diffuse*=⟨ n ⟩
emission=⟨ n ⟩ *minimize*=yes | no *shininess*=⟨ n ⟩
specular=⟨ n ⟩

Options:

ambient Ambient material component.

Valid values:	integers
Default value:	**14**
Minimum:	0
Maximum:	100

changed If any material option changed or not.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

diffuse Diffuse material component.

Valid values:	integers
Default value:	**100**
Minimum:	0
Maximum:	100

emission Emission material component.

Valid values:	integers
Default value:	**40**
Minimum:	0
Maximum:	100

minimize Use glColorMaterial to minimize performance cost or not.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

shininess Shininess material component.

Valid values:	integers

Default value:	**55**
Minimum:	0
Maximum:	100

specular Specular material component.

Valid values:	integers
Default value:	**50**
Minimum:	0
Maximum:	100

materialdefault

A command which sets default material settings.

Syntax:

`materialdefault`

materialmolecular

Syntax:

`materialmolecular` *shininess*=⟨ x ⟩ *specular*=⟨ x ⟩

Options:

shininess Shininess component for material of molecules

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

specular Specular component for material of molecules

Valid values:	reals
Default value:	**0.7**
Minimum:	0.0
Maximum:	1.0

mcsd

A command which defines settings for the MC/SD (Monte Carlo/Stochastic Dynamics) simulation.

Syntax:

`mcsd` *maxdof*=⟨ n ⟩ *mindof*=⟨ n ⟩ *ratio*=⟨ n ⟩ *temperature*=⟨ x ⟩

Options:

maxdof The maximum number of torsions which will will be varied at each MC trial.

Valid values: integers
Default value: **1**
Minimum: 1

mindof The minimum number of torsions which will will be varied at each MC trial.

Valid values: integers
Default value: **1**
Minimum: 1

ratio The ratio of Monte Carlo to Stochastic Dynamics steps in the simulation.

Valid values: integers
Default value: **1**
Minimum: 1

temperature
The temperature at which the MCSD simulation is to be performed. [NOTE: This option is no longer used.]

Valid values: reals
Default value: **300**
Minimum: 0.0

mini

This is a standard alias for **minienergy** (see [minienergy], page 343).

minienergy

Used to set values associated with a MacroModel energy minimization

Syntax:

minienergy *converge*=nothing | energy | gradient | movement *maxiter*=⟨ n ⟩ *method*=sd | prcg | osvm | fmnr | tncg | lbfgs | optimal *threshold*=⟨ x ⟩

Options:

converge This option determines which convergence criterion will be used during an energy minimization.

Valid values:	nothing energy gradient movement
Default value:	**gradient**

maxiter This option determines the maximum number of iterations which will be performed during an energy minimization.

Valid values:	integers
Default value:	**500**
Minimum:	0
Maximum:	9999999

method This option determines which minimization method will be used.

Valid values:	sd prcg osvm fmnr tncg lbfgs optimal
Default value:	**prcg**

threshold This option determines what the convergence threshold will be.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0

Aliases:
mini (see [mini], page 343)

minimize

Minimizes the structure in the Workspace, or just the selected atoms if there's a selection.

Syntax:

`minimize`

minta

Used to set up options associated with BatchMin molecular energy calculation.

Syntax:

`minta` *hardlimit*=⟨ x ⟩ *numenergy*=⟨ n ⟩ *numint*=⟨ n ⟩ *softlimit*=⟨ n ⟩ *temperature*=⟨ x ⟩ ⟨ input_file_name ⟩

Options:

hardlimit The hard limit for sampling along normal modes.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	3.0

numenergy

This option determines the number of energy evaluations per MINTA integration.

Valid values:	integers
Default value:	**2000**
Minimum:	1

numint This option determines the number of MINTA iteration.

Valid values:	integers
Default value:	**5**
Minimum:	1

softlimit The soft limit for sampling along normal modes.

Valid values:	integers
Default value:	**3**
Minimum:	1
Maximum:	3

temperature

The temperature for MINTA calculation.

Valid values:	reals
Default value:	**300**
Minimum:	0.0

Operands:
⟨ input_file_name ⟩
The name of the input file. This name must be given in full, including any suffix.

monitor

This is a standard alias for **energymonitor** (see [energymonitor], page 134).

monitorangle

Specifies a triplet of atoms to have their bond angle monitored during a MacroModel dynamics simulation.

Syntax:

`monitorangle` ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩
The atom numbers of three atoms which are to have the angle between them monitored during a dynamics simulation. Note that specifying a-b-c is the same as c-b-a

monitordistance

Specifies a pair of atoms to have their distance monitored during a MacroModel dynamics simulation.

Syntax:

`monitordistance` ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩
The atom numbers of two atoms which are to have the distance between them monitored during a dynamics simulation. Note that specifying a-b is the same as b-a

monitorhbond

Specifies a quartet of atoms which define an H-bond to be monitored during a dynamics simulation. The atoms are specified as X-H...Y-Z e.g. N2-H3...O2=C2 for a amide-amide bond.

Syntax:

`monitorhbond` *distance*=⟨ x ⟩ *hyzangle*=⟨ x ⟩ *xhyangle*=⟨ x ⟩
⟨ xatom ⟩ ⟨ hatom ⟩ ⟨ yatom ⟩ ⟨ zatom ⟩

Options:

distance Specifies the maximum H...Y distance for an acceptable hydrogen bond.

Valid values:	reals
Default value:	**2.5**
Minimum:	0.0

hyzangle Specifies the minimum H...Y-Z angle for an acceptable hydrogen bond.

Valid values:	reals
Default value:	**90**
Minimum:	0.0
Maximum:	180.0

xhyangle Specifies the minimum X-H...Y angle for an acceptable hydrogen bond.

Valid values:	reals
Default value:	**120**
Minimum:	0.0
Maximum:	180.0

Operands:

⟨ xatom ⟩ ⟨ hatom ⟩ ⟨ yatom ⟩ ⟨ zatom ⟩

The first operand is the atom number of the heavy atom (X) to which the donor hydrogen is attached. The second operand is the atom number of the donor hydrogen itself (H). The third operand is the atom number of acceptor atom (Y) and the fourth operand is the atom number of the heavy atom (Z) attached to the acceptor.

monitorsetsurf

Specifies a set of atoms to be have their surface areas monitored during a MacroModel molecular dynamics simulation.

Syntax:

monitorsetsurf ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language which describes the set of atom which are to have their surface areas monitored during a dynamics simulation.

monitorsettings

Controls settings for how the monitor panel in Maestro functions. At present it's just for whether jobs for the current project are displayed or all jobs.

Syntax:

monitorsettings *monitorinterval*=⟨ x ⟩ *monitorlines*=⟨ n ⟩
showjobs=current | active | all

Options:

monitorinterval

This option determines the minimum interval, in seconds, between monitoring updates while monitoring a job. The actual interval may be longer than the specified value, due to time spent in panel and Workspace updates.

Valid values:	reals
Default value:	**1**
Minimum:	1.0
Maximum:	300.0

monitorlines

This option determines the maximum number of lines to be displayed in the monitor panel File tab in each monitoring interval while monitoring a job.

Valid values:	integers
Default value:	**100**
Minimum:	100

showjobs This determines whether the jobs listed in the monitor panel are limited to those for the current project or all known jobs for the current user.

Valid values: current
active
all

Default value: **current**

monitorsurf

Specifies a single atom to have its surface area monitored during a MacroModel molecular dynamics simulation.

Syntax:

`monitorsurf` ⟨atom_number⟩

Operands:

⟨atom_number⟩

The number of an atom which is to have its surface areas monitored during a dynamics simulation.

monitortorsion

Specifies a quartet of atoms to have their torsion angle monitored during a MacroModel dynamics simulation.

Syntax:

`monitortorsion` ⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

Operands:

⟨atom1⟩ ⟨atom2⟩ ⟨atom3⟩ ⟨atom4⟩

The atom numbers of four atoms which are to have the torsion between them monitored during a dynamics simulation. Note that specifying a-b-c-d is the same as d-c-b-a

mouse

Specify how mouse movements are to be interpreted

Syntax:

`mouse` *rotate*=xy | x | y | z | adjust *translate*=xy | x | y | z
There are no operands for this command.

Options:

rotate How mouse movements are interpreted for rotation. Default is rotate in xy.
Valid values: xy, x, y, z, adjust
Default value: **xy**

translate How mouse movements are interpreted for translation. Default is translate in xy.
Valid values: xy, x, y, z
Default value: **xy**

Operands:
There are no operands for this command.

move

Moves the specified atoms by the offsets specified in x, y and z values

Syntax:

`move` ⟨atom_num⟩ ⟨xinc⟩ ⟨yinc⟩ ⟨zinc⟩

Operands:
⟨atom_num⟩ ⟨xinc⟩ ⟨yinc⟩ ⟨zinc⟩
The first operand is the number of the atom which is to be moved. The following three real numbers represent the offsets to be applied to the x, y and z coordinates in Angstroms.

multiplemini

Used to set up a multiple minimization. The operand is the name of a file which contains a number of structures.

Syntax:

`multiplemini` *distinguishenantiomers*=yes | no
eliminate=atom_deviation | rmsd *max_rmsd*=⟨ x ⟩
maxdist=⟨ x ⟩ *savemaximum*=⟨ n ⟩ *setsavemaximum*=yes | no
window=⟨ x ⟩ ⟨ input_file_name ⟩

Options:

distinguishenantiomers
A boolean for whether to retain different enantiomers. If true, an additional opcode, NANT, is written to the .com file.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

eliminate The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.
Valid values: atom_deviation
rmsd
Default value: **atom_deviation**

max_rmsd Maximum RMSD for considering two structures equal.
Valid values: reals
Default value: **0.5**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.
Valid values: reals
Default value: **0.5**
Minimum: 0.0

savemaximum
Maximum number of structures to save
Valid values: integers
Default value: **100**
Minimum: 1

setsavemaximum
A boolean for whether to set a maximum number of structures to save.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

window Specifies the energy window (in kJ/mol) for saving structures during a multiple minimization.

Valid values:	reals
Default value:	**21**
Minimum:	0.0

Operands:

⟨ input_file_name ⟩

The name of the input file. This name will contain all the structures which are to be minimized. This name must be given in full, including any suffix.

mutate

Replace the sidechain of any residue in set <ASL> with the sidechain of the currently selected fragment

Syntax:

`mutate` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The set of atoms which are to be mutated. Each residue represented in this set will be mutated.

nextresidue

If only a single residue is selected in the Workspace, then find the next one.

Syntax:

`nextresidue`

optimizefog

Fit clipping planes to displayed structure to maximize the shading contrast between near and far atoms.

Syntax:

`optimizefog`

partialcharge

Set the partial atomic charges for all atoms which match the ASL specification.

Syntax:

`partialcharge` ⟨ charge1 ⟩ ⟨ charge2 ⟩ ⟨ ASL ⟩

Operands:

⟨ charge1 ⟩ ⟨ charge2 ⟩ ⟨ ASL ⟩

The first operand must be a valid real number which will be used for the charge1 field of all matching atoms. The second operand must be a valid real number which will be used for the charge2 field of all matching atoms. The final operand is a valid ASL string which specifies which atoms are to have their charges changed.

pause

Pauses immediately for the length of time specified (in seconds, unless other units are specified). Setting a pause of zero (e.g. "0" or "0.00") will mean there is no pause. If no operand is specified, or a negative time is given, then Maestro will remain paused indefinitely, until it is interrupted. All pauses can be interrupted by pressing any key on the keyboard or any mouse button.

Syntax:

`pause` ⟨ pause_time ⟩ [sec | min | hour | day]

Operands:

⟨ pause_time ⟩ [sec | min | hour | day]

The length of time to pause - floating point value in seconds, accurate to within about 0.05 second.

pausecommands

Sets the time to pause after execution of every command in script (in seconds, unless other units are specified). Setting a pause of zero (e.g. "0" or "0.00") will mean there is no pause after each command. If no operand is specified, or a negative time is given, then the pause after each command will be indefinite, pausing until interrupted. All pauses can be interrupted by pressing any key on the keyboard or any mouse button.

Syntax:

`pausecommands` ⟨ pause_time ⟩ [sec | min | hour | day]

Operands:

⟨ pause_time ⟩ [sec | min | hour | day]

The time to pause after each command - floating point value in seconds, accurate to within about 0.05 second.

peptidecistrans

Perform a cis/trans interconversion of the peptide in the residue which contains the specified atom.

Syntax:

`peptidecistrans` ⟨ atom ⟩

Operands:

⟨ atom ⟩

The atom which defines the residue to have the cis/trans inter-conversion done.

peptiderotate

Rotate the plane of the peptide in the residue which contains the specified atom

Syntax:

`peptiderotate` *carbonyl_only*=yes | no *increment*=⟨ x ⟩ ⟨ atom ⟩

Options:

carbonyl_only
When this is true then only the carbonyl of the peptide is rotated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

increment Value by which to rotate the plane of the peptide from its current value.

Valid values: reals
Default value: **2**
Minimum: -180.1
Maximum: 180.1

Operands:
⟨ atom ⟩
The atom which defines the residue to have a peptide rotated.

perresiduedisplay

Specify whether or not display per-residue interactions.

Syntax:

`perresiduedisplay` *display*=yes | no

Options:

display If this option is true, display per-residue interactions.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

perresiduesubset

Set subset of per-residue interactions by the ASL operand.

Syntax:

`perresiduesubset` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language which describes the subset of per-residue interactions.

perresiduetype

Specify the type of displaying per-residue interactions.

Syntax:

`perresiduetype` *type*=eint | vdw | coulomb | hbond | distance

Options:

type Specify the type of displaying per-residue interactions, Eint, vdw, Coulimb, H-Bond energy, or Minimum distance.

Valid values: eint
vdw
coulomb
hbond
distance

Default value: **eint**

phaseaddconstraint

Adds the given constraint.

Syntax:

`phaseaddconstraint` *value*=⟨ x ⟩ ⟨ constraint ⟩

Options:

value The value to set for the given constraint.

Valid values: reals
Default value: **1**

Operands:

⟨ constraint ⟩

The constraint to add. Should be in the form: <site 1>-<site 2> and so on, up to four sites.

phaseaddcustomfeature

Adds the given custom feature.

Syntax:

`phaseaddcustomfeature` *code*=⟨ text ⟩ ⟨ name ⟩

Options:

code	The name of the custom feature to add.	
	Valid values:	text strings
	Default value:	**A**

Operands:

⟨ name ⟩

The name for the feature.

phaseaddhypotheses

Copies the selected hypotheses from Develop Common Pharmacophore panel (in either the Score Hypothesis or Build QSAR model steps) to the Project Table, and opens the Project Table.

Syntax:

`phaseaddhypotheses`

phaseaddligands

Adds the given ligands to the table.

Syntax:

phaseaddligands *activity*=⟨ text ⟩ *convert_activity*=yes | no
convert_scale=⟨ x ⟩ ⟨ ESL ⟩

Options:

activity This determines which property (if any) to use as the activity property for the ligands.

Valid values: text strings
Default value:

convert_activity
Set to true if the activity values should be converted from concentration to -log[concentration]

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

convert_scale
A scale factor for conversion.

Valid values: reals
Default value: **1**

Operands:
⟨ ESL ⟩
The entries to add as ligands.

phaseaddligandsfromfile

Adds the ligands in the given files to the current Phase run.

Syntax:

phaseaddligandsfromfile *activity*=⟨ text ⟩
convert_activity=yes | no *convert_scale*=⟨ x ⟩ ⟨ file 1 ⟩ ⟨ file 2 ⟩

Options:

activity Which property (if any) to use as the activity property for the ligands.

Valid values: text strings
Default value:

convert_activity
: Set to true if the activity values should be converted from concentration to -log[concentration]

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

convert_scale
: A scale factor for conversion

  Valid values: reals
  Default value: **1**

Operands:
⟨ file 1 ⟩ ⟨ file 2 ⟩
The files to add ligands from.

phaseaddligandsfromrun

Adds the given ligands to the table.

Syntax:

`phaseaddligandsfromrun` *runname*=⟨ text ⟩ ⟨ ligand_names ⟩

Options:

runname This determines which run to use as the reference to copy the corresponding properties of the ligands.

  Valid values: text strings
  Default value:

Operands:
⟨ ligand_names ⟩
The ligand names to be added separated by semi-colon.

phaseaddsite

Adds a site to the current freestyle hypothesis

Syntax:

`phaseaddsite` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩ ⟨ site type ⟩

Options:

x X-coordinate of the new site.
Valid values: reals
Default value: **0**

y X-coordinate of the new site.
Valid values: reals
Default value: **0**

z X-coordinate of the new site.
Valid values: reals
Default value: **0**

Operands:
⟨ site type ⟩
The single-letter feature type.

phasealignhypotheses

Sets whether or not to view non-model ligands and whether or not to have a site mask.

Syntax:

`phasealignhypotheses` *alignment_type*=sites | rmsd
must_match=⟨ n ⟩ *tolerance*=⟨ x ⟩ ⟨ hypothesis ⟩

Options:

alignment_type
Indicates how to choose the best alignment
Valid values: sites
rmsd
Default value: **sites**

must_match
The minimum number of equivalent sites that have to match to do the alignment.
Valid values: integers
Default value: **3**
Minimum: 3

tolerance The tolerance in angstroms.

Valid values:	reals
Default value:	**2**
Minimum:	0.0

Operands:
⟨ hypothesis ⟩
The name of the hypothesis to set the options for.

phasebuildqsar

Sets the QSAR options and launchs a Build QSAR job.

Syntax:

`phasebuildqsar` *color_ligands*=yes | no *eliminatetvalue*=yes | no *gridspacing*=⟨ x ⟩ *keep_training_set*=yes | no *maxplsfactor*=⟨ n ⟩ *modeltype*=atom_based | phase_based *sample_uniformly*=yes | no *tolerance_a*=⟨ x ⟩ *tolerance_d*=⟨ x ⟩ *tolerance_h*=⟨ x ⟩ *tolerance_n*=⟨ x ⟩ *tolerance_p*=⟨ x ⟩ *tolerance_r*=⟨ x ⟩ *tolerance_x*=⟨ x ⟩ *tolerance_y*=⟨ x ⟩ *tolerance_z*=⟨ x ⟩ *tvalue*=⟨ x ⟩

Options:

color_ligands
Indicates whether or not to color ligands to match the coloring in the ligands table.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

eliminatetvalue
Inidicates whether to use t-value filter before launching QSAR job.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

gridspacing
The grid spacing to be for the QSAR model grid. The valid range is 0.5 to 2.0 angstrom.

Valid values:	reals
Default value:	**1**
Minimum:	0.5
Maximum:	2.0

keep_training_set
Indicates whether or not to keep the actives and inactives in the training set.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

maxplsfactor
This option value range depends on the ligands in the training set. The valid range is 1 to N/5, where N is the number of ligands in the training set. If N is less than 5, the maximum number of factors is 1, but no QSAR model can be constructed.

Valid values: integers
Default value: **1**
Minimum: 1

modeltype The value of the model type in the Build QSAR Model step options dialog.

Valid values: atom_based
phase_based
Default value: **atom_based**

sample_uniformly
Indicates whether or not to sample uniformly over the activity coordinate when assigning to test and training sets.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

tolerance_a
The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_d
The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_h
The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1.5**
Minimum: 0.0
Maximum: 100.0

tolerance_n

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 100.0

tolerance_p

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 100.0

tolerance_r

The feature matching tolerance for the aromatic ring feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1.5**
Minimum: 0.0
Maximum: 100.0

tolerance_x

The feature matching tolerance for the custom(X) feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_y

The feature matching tolerance for the custom(Y) feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_z

The feature matching tolerance for the custom(Z) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

tvalue This option value indicates the values to be used to filter t-value.

Valid values:	reals
Default value:	**2**
Minimum:	0.01

phasechangesitetype

Changes the type of the selected sites

Syntax:

`phasechangesitetype` ⟨site type⟩

Operands:
⟨site type⟩
The new site type.

phasechooseactiveset

Sets the Active flag based on whether the activity value is larger than the cutoff value.

Syntax:

`phasechooseactiveset` ⟨cutoff⟩

Operands:
⟨cutoff⟩
The cutoff value for deciding between active and inactive.

phasecleanupstructures

Launches a Cleanup Structures job for Phase.

Syntax:

```
phasecleanupstructures
```

phaseclearcentroidatoms

Removes all atoms from the centroid list for excluded volumes.

Syntax:

```
phaseclearcentroidatoms
```

phaseclusterhypotheses

Sets the Cluster Hypotheses options and launches the job.

Syntax:

`phaseclusterhypotheses` *linkage*=complete | average | single

Options:

linkage	The linkage type.	
	Valid values:	complete average single
	Default value:	**complete**

phaseconfgen

Defines settings for Phase Generate Conformers job.

Syntax:

`phaseconfgen` *amidebonds*=vary | retain | trans
eliminate=atom_deviation | rmsd *field*=mmffs | opls2005
incorporate=append | replace | ignore | appendungrouped |
workspace *max_rmsd*=⟨ x ⟩ *maxdist*=⟨ x ⟩ *method*=default |
mixed *minimizationsteps*=⟨ n ⟩ *numrotatablesteps*=⟨ n ⟩
numsteps=⟨ n ⟩ *postmaxiter*=⟨ n ⟩ *postprocessing*=yes | no
postprocessingmethod=mini | filter | rce *premaxiter*=⟨ n ⟩
preprocessing=yes | no *sampling*=standard | rapid | complete
| thorough *solvation*=gbsa | distance_dependent *window*=⟨ x ⟩

Options:

amidebonds
This determines whether to vary amide bond conformation, retain original amide bond conformation, or set amide bond conformation to trans.

Valid values: vary
retain
trans
Default value: **vary**

eliminate The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.

Valid values: atom_deviation
rmsd
Default value: **rmsd**

field This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

Valid values: mmffs
opls2005
Default value: **opls2005**

incorporate
This option controls the incorporation of the results (replace or append).

Valid values: append
replace
ignore
appendungrouped
workspace
Default value: **replace**

max_rmsd Maximum RMSD for considering two structures equal.

Valid values: reals

Default value: **1**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **2**
Minimum: 0.0

method This determines whether MacroModel uses the ligand torsion search method (default) or the mixed MCMM/LMOD search method (mixed) to generate conformers.

Valid values: default
mixed
Default value: **default**

minimizationsteps
This option determines the maximum number of minimization steps for Mixed MCMM/LMOD generation.

Valid values: integers
Default value: **100**
Minimum: 1

numrotatablesteps
An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values: integers
Default value: **100**
Minimum: 1

numsteps An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.

Valid values: integers
Default value: **1000**
Minimum: 0

postmaxiter
This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers
Default value: **50**
Minimum: 0
Maximum: 9999999

postprocessing
Indicates whether or not to perform MacroModel postprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

postprocessingmethod
This determines which type of postprocessing method to use (minimization, filtering and redundant conformer elmination, or redundant conformer elmination only).

Valid values: mini
filter
rce
Default value: **mini**

premaxiter
This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers
Default value: **100**
Minimum: 0
Maximum: 9999999

preprocessing
Indicates whether or not to perform MacroModel preprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sampling This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values: standard
rapid
complete
thorough
Default value: **standard**

solvation This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance_dependent) solvation treatment is used.

Valid values: gbsa
distance_dependent
Default value: **distance_dependent**

window The energy window (in kcal/mol) within which structures will be saved.

Valid values: reals
Default value: **10**
Minimum: 0.0

phasecopyhypothesisfrombuilder

Copy the given hypothesis from the hypotheses table in the Edit Hypothesis panel.

Syntax:

`phasecopyhypothesisfrombuilder` ⟨ row number ⟩

Operands:
⟨ row number ⟩
The row number of the hypothesis to copy.

phasecreateevactives

Creates excluded volumes from actives and inactives.

Syntax:

`phasecreateevactives`

phasecreateevactivessettings

Settings for creating excluded volumes from actives and inactives

Syntax:

`phasecreateevactivessettings` *actives_file*=⟨ text ⟩
distance=⟨ x ⟩ *inactives_file*=⟨ text ⟩ *min_inactives*=⟨ n ⟩
radii=⟨ x ⟩

Options:

actives_file
The path to the actives file.
Valid values: text strings
Default value:

distance The minimum distance between the active surface and the excluded volumes.
Valid values: reals
Default value: **1**
Minimum: 0.0

inactives_file
The path to the inactives file.
Valid values: text strings
Default value:

min_inactives
The minimum number of inactives that must experience a clash.
Valid values: integers
Default value: **1**
Minimum: 1

radii The radii for the excluded volumes.
Valid values: reals
Default value: **1**
Minimum: 0.0

phasecreateevreceptor

Creates excluded volumes from receptor atoms.

Syntax:

`phasecreateevreceptor`

phasecreateevreceptoratoms

Specify the atoms to use to define an excluded volume shell.

Syntax:

`phasecreateevreceptoratoms` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. Defines a set of atoms to use for building an excluded volume shell.

phasecreateevreceptorsettings

Settings for creating excluded volumes from receptor atoms.

Syntax:

phasecreateevreceptorsettings *fixed_radius*=⟨ x ⟩
fixed_scaling=⟨ x ⟩ *ignore_atoms*=yes | no *ignore_distance*=⟨ x ⟩
limit_thickness=yes | no *radii_scaling*=fixed | property
radii_sizes=vdw | fixed | property *radius_property*=⟨ text ⟩
scaling_property=⟨ text ⟩ *thickness*=⟨ x ⟩ ⟨ ASL ⟩

Options:

fixed_radius
The fixed radius to use for excluded volumes.
Valid values: reals
Default value: **1**
Minimum: 0.0

fixed_scaling
The fixed scaling to apply to excluded volume radii.
Valid values: reals
Default value: **1**
Minimum: 0.0

ignore_atoms
Ignore receptor atoms which are too close to the ligand surface.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ignore_distance
The distance to consider as too close to the ligand atoms.
Valid values: reals
Default value: **1**
Minimum: 0.0

limit_thickness
Limit excluded volume shell thickness to a specified number.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

radii_scaling
Indicates how to scale the radii of the excluded volumes.
Valid values: fixed
property
Default value: **fixed**

radii_sizes Indicates how to determine the radii of the excluded volumes.
Valid values: vdw
fixed
property

Default value: **vdw**

radius_property
The atom property to use as the radius for excluded volumes.
Valid values: text strings
Default value:

scaling_property
The atom property to use to scale the excluded volume radii by.
Valid values: text strings
Default value:

thickness The thickness to limit the excluded volume shell to.
Valid values: reals
Default value: **5**
Minimum: 0.0

Operands:
⟨ ASL ⟩
An ASL to use as the receptor atoms.

phasecreateevreference

Creates excluded volumes from reference structures.

Syntax:

`phasecreateevreference`

phasecreateevreferenceatoms

Specify the atoms to use to define an excluded volume shell.

Syntax:

`phasecreateevreferenceatoms` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. Defines a set of atoms to use for building an excluded volume shell.

phasecreateevreferencesettings

Settings for creating excluded volumes from reference structure atoms.

Syntax:

`phasecreateevreferencesettings` *distance*=⟨ x ⟩ *radii*=⟨ x ⟩
source=ligand | atoms ⟨ ASL ⟩

Options:

distance The minimum distance between the ligand surface and the excluded volume shell.

Valid values:	reals
Default value:	**1**
Minimum:	0.0

radii The excluded volume sphere radii.

Valid values:	reals
Default value:	**1**
Minimum:	0.0

source Indicates the source of the reference structures.

Valid values:	ligand atoms
Default value:	**ligand**

Operands:
⟨ ASL ⟩
An ASL to use as the reference atoms.

phasecreateexcludedvolume

Creates an excluded volume from the current centroid atoms.

Syntax:

`phasecreateexcludedvolume`

phasecreatefreestylehypothesis

Sets the given entry to have the current freestyle hypothesis.

Syntax:

`phasecreatefreestylehypothesis` ⟨ entry id ⟩

Operands:
⟨ entry id ⟩
The entry to add the freestyle hypothesis to.

phasedbaddconfset

Adds the selected confset to the Phase DB

Syntax:

`phasedbaddconfset`

phasedbaddligands

Adds ligands from the specified file to the database. The ligands will go through a cleanup process.

Syntax:

`phasedbaddligands` *alreadycleaned*=yes | no
generatetautomers=yes | no *ionization*=retain | neutralize | generate *maxisomers*=⟨ n ⟩ *maxtautomers*=⟨ n ⟩
stereoisomers=retain | determine | all *target_ph*=⟨ x ⟩
⟨ file name ⟩

Options:

alreadycleaned
Indicates whether or not the structures have already been cleaned.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

generatetautomers
Indicates whether to generate tautomers or not
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ionization Sets the handling of ionization.

Valid values: retain
neutralize
generate
Default value: **retain**

maxisomers
The maximum number of stereoisomers to return for any ligand.
Valid values: integers
Default value: **10**
Minimum: 1

maxtautomers
The maximum number of stereoisomers to return for any ligand.
Valid values: integers
Default value: **8**
Minimum: 1

stereoisomers
Sets the handling of chiralities.
Valid values: retain
determine
all
Default value: **all**

target_ph The target pH for generating ionization states.
Valid values: reals
Default value: **7**

Operands:
⟨ file name ⟩
The name of the file of structure to be added. These will go through the cleanup process with the options specified below. The ligands can be in Maestro or SD format.

phasedbaddligandsfromdb

Adds ligands from the specified database

Syntax:

`phasedbaddligandsfromdb`

phasedbconfgen

Defines settings for PhaseDB Generate Conformers job.

Syntax:

phasedbconfgen *amidebonds*=vary | retain | trans
eliminate=atom_deviation | rmsd *field*=mmffs | opls2005
incorporate=append | replace | ignore | appendungrouped |
workspace *max_rmsd*=⟨ x ⟩ *maxdist*=⟨ x ⟩ *method*=default |
mixed *minimizationsteps*=⟨ n ⟩ *numrotatablesteps*=⟨ n ⟩
numsteps=⟨ n ⟩ *postmaxiter*=⟨ n ⟩ *postprocessing*=yes | no
postprocessingmethod=mini | filter | rce *premaxiter*=⟨ n ⟩
preprocessing=yes | no *sampling*=standard | rapid | complete
| thorough *solvation*=gbsa | distance_dependent *window*=⟨ x ⟩

Options:

amidebonds
This determines whether to vary amide bond conformation, retain original amide bond conformation, or set amide bond conformation to trans.

Valid values: vary
retain
trans
Default value: **vary**

eliminate The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.

Valid values: atom_deviation
rmsd
Default value: **rmsd**

field This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

Valid values: mmffs
opls2005
Default value: **opls2005**

incorporate
This option controls the incorporation of the results (replace or append).

Valid values: append
replace
ignore
appendungrouped
workspace
Default value: **replace**

max_rmsd Maximum RMSD for considering two structures equal.

Valid values: reals
Default value: **1**
Minimum: 0.0

maxdist Maximum distance between atoms in equal structures.

Valid values: reals
Default value: **2**
Minimum: 0.0

method This determines whether MacroModel uses the ligand torsion search method (default) or the mixed MCMM/LMOD search method (mixed) to generate conformers. Currently database creation always uses the default method, so it will have only one option value.

Valid values: default
mixed
Default value: **default**

minimizationsteps
This option determines the maximum number of iterations for minimization of generated structures in Mixed MCMM/LMOD.

Valid values: integers
Default value: **100**
Minimum: 1

numrotatablesteps
An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values: integers
Default value: **10**
Minimum: 1

numsteps An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.

Valid values: integers
Default value: **100**
Minimum: 0

postmaxiter
This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values:	integers
Default value:	**50**
Minimum:	0
Maximum:	9999999

postprocessing
Indicates whether or not to perform MacroModel postprocessing.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

postprocessingmethod
This determines which type of postprocessing method to use (minimization, filtering and redundant conformer elmination, or redundant conformer elmination only).

Valid values:	mini filter rce
Default value:	**rce**

premaxiter
This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values:	integers
Default value:	**100**
Minimum:	0
Maximum:	9999999

preprocessing
Indicates whether or not to perform MacroModel preprocessing.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

sampling This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values:	standard rapid complete thorough
Default value:	**standard**

solvation This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance_dependent) solvation treatment is used.

Valid values: gbsa
distance_dependent
Default value: **distance_dependent**

window The energy window (in kcal/mol) within which structures will be saved.

Valid values: reals
Default value: **10**
Minimum: 0.0

phasedbcreatesites

Launches a Create Sites job for a Phase database.

Syntax:

`phasedbcreatesites` *withligands*=all | withsites | selected

Options:

withligands
Controls which ligands have sites regenerated.

Valid values: all
withsites
selected
Default value: **withsites**

phasedbcreatesubsetfromhitfile

Create a new subset based on a Maestro structure file which is the result of a phase DB search.

Syntax:

`phasedbcreatesubsetfromhitfile` ⟨subsetname⟩ ⟨hit file⟩

Operands:
⟨subsetname⟩ ⟨hit file⟩
The name of the new subset and a Maestro structure file which is the result of a Phase database search.

phasedbcreatesubsetfromselected

Create a new subset from those ligands which are currently selected in the ligand table.

Syntax:

phasedbcreatesubsetfromselected ⟨subsetname⟩

Operands:
⟨subsetname⟩
The name of the new subset.

phasedbcreatesubsetfromtextfile

Create a new subset based on a file containing ligand titles - one per line.

Syntax:

phasedbcreatesubsetfromtextfile ⟨subset name⟩ ⟨text file⟩

Operands:
⟨subset name⟩ ⟨text file⟩
The name of the subset and a file which contains a list of ligand names - one per line.

phasedbdeleteligands

Launches a Delete Ligands job for selected ligands in a Phase DB.

Syntax:

phasedbdeleteligands *whichstructures*=selected | all

Options:

whichstructures
Determines which structures this operation will be applied to.

Valid values:	selected all
Default value:	**selected**

phasedbdeletesubset

Delete the named subset from the DB

Syntax:

`phasedbdeletesubset` ⟨subset name⟩

Operands:
⟨subset name⟩
The name of the subset to be deleted.

phasedbdisplayrange

Control the range of ligands which are displayed in the table.

Syntax:

`phasedbdisplayrange` *end*=⟨n⟩ *start*=⟨n⟩

Options:

end Sets the end point or the display range

Valid values:	integers
Default value:	**1000**
Minimum:	1

start Sets the starting point or the display range

Valid values:	integers
Default value:	**1**
Minimum:	1

phasedbexportligands

Export the ligands corresponding to selected ligands to the project table.

Syntax:

`phasedbexportligands` *confs*=firstconf | allconfs

Options:

confs Controls what is exported - all conformations or just the first of each ligand.

Valid values:	firstconf allconfs
Default value:	**firstconf**

phasedbfilter

Replace currently filtered structures with ones which match the expression

Syntax:

`phasedbfilter` ⟨ filter_expr ⟩

Operands:

⟨ filter_expr ⟩

The filter string which is to be applied to the ligand names to determine which ligands are to be displayed in the database table.

phasedbfilteradd

Add to the currently filtered structures with ones which match the expression.

Syntax:

`phasedbfilteradd` ⟨ filter_expr ⟩

Operands:

⟨ filter_expr ⟩

The filter string which is to be applied to the ligand names to determine which ligands are to be added to the database table.

phasedbgenerateconformers

Launches a Generate Conformers job for a Phase database.

Syntax:

phasedbgenerateconformers *whichstructures*=selected | all

Options:

whichstructures
: Determines which structures this operation will be applied to.

Valid values:	selected all
Default value:	**selected**

phasedbnewfromfile

Create a new Phase 3D database and open it in Maestro.

Syntax:

phasedbnewfromfile ⟨db_dir_path⟩ ⟨name⟩

Operands:

⟨db_dir_path⟩ ⟨name⟩

The path (location) of the directory to be created as the new Phase 3D database directory and the name of the DB

phasedbopen

Open an existing Phase 3D database into Maestro.

Syntax:

phasedbopen ⟨db_dir_path⟩ ⟨db_name⟩

Operands:

⟨db_dir_path⟩ ⟨db_name⟩

The path (location) and name of the Phase 3D database directory to be opened.

phasedbremoveconformers

Launches a Remove Conformers job for selected ligands in a Phase DB.

Syntax:

phasedbremoveconformers *whichstructures*=selected | all

Options:

whichstructures
Determines which structures this operation will be applied to.

Valid values:	selected all
Default value:	**selected**

phasedbselectextendtablerow

Extend the selection in the Phase Database table from the selected table row to joing up with an existing selection.

Syntax:

phasedbselectextendtablerow ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table from which the selection is to begin.

phasedbselectonlytablerow

Select a row from the phase database table.

Syntax:

phasedbselectonlytablerow ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be selected.

phasedbselectsuubsetrow

Select the row in the subset table.

Syntax:

`phasedbselectsuubsetrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number of the subset to be selected.

phasedbselecttablerow

Selects a row from the phase database table.

Syntax:

`phasedbselecttablerow` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be selected.

phasedbunselecttablerow

Unselects a row from the Phase Database table.

Syntax:

`phasedbunselecttablerow` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be unselected.

phasedeletecustomfeature

Deletes the given custom feature.

Syntax:

`phasedeletecustomfeature` ⟨ code ⟩

Operands:

⟨ code ⟩

The code for the feature to delete.

phasedeleteexcludedvolumes

Deletes the selected excluded volumes.

Syntax:

`phasedeleteexcludedvolumes`

phasedeletehypothesis

Deletes the given hypothesis from the hypotheses table in the Score Hypotheses or Build QSAR Model step.

Syntax:

`phasedeletehypothesis` ⟨ ID ⟩

Operands:

⟨ ID ⟩

The hypothesis to delete.

phasedeletehypothesisfrombuilder

Deletes the given hypothesis from the hypotheses table in the Edit Hypothesis and Find Matches panels.

Syntax:

`phasedeletehypothesisfrombuilder` ⟨row number⟩

Operands:
⟨row number⟩
The row number of the hypothesis to delete.

phasedeleteselectedconstraints

Deletes all of the selected constraints.

Syntax:

`phasedeleteselectedconstraints`

phasedeleteselectedligands

Deletes the selected ligands in the current Phase step.

Syntax:

`phasedeleteselectedligands`

phasedeletesites

Deletes the selected sites from the current freestyle hypothesis.

Syntax:

`phasedeletesites`

phasedisplayproperty

Displays the given property from the table.

Syntax:

`phasedisplayproperty` ⟨ property name ⟩

Operands:
⟨ property name ⟩
The property to display.

phaseevoptions

Options for Phase excluded volumes

Syntax:

`phaseevoptions` *radius*=⟨ x ⟩

Options:

radius	Radius for creating new excluded volumes.	
	Valid values:	reals
	Default value:	**1**
	Minimum:	0.0

phaseexcludetablerow

Excludes the given row in the first table in the step from the Workspace.

Syntax:

`phaseexcludetablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to exclude in the Workspace.

phaseexportalignmentstofile

Exports the selected alignments from the ligands table in the Score Hypotheses or Build QSAR steps.

Syntax:

`phaseexportalignmentstofile` ⟨file name⟩

Operands:
⟨file name⟩
The file name to export the selected alignments to.

phaseexportconformerstofile

Exports the selected conformers from the ligands table in the Prepare Ligands or Create Sites steps.

Syntax:

`phaseexportconformerstofile` ⟨file name⟩

Operands:
⟨file name⟩
The file name to export the selected conformers to.

phaseexportfeature

Exports the features to the given file name.

Syntax:

`phaseexportfeature` ⟨file name⟩

Operands:
⟨file name⟩
A path to a write the features file to.

phaseexporthypothesis

Exports the included hypothesis from the hypotheses table in the Score Hypotheses or Build QSAR Model step.

Syntax:

`phaseexporthypothesis` ⟨file name⟩

Operands:

⟨file name⟩

The file name to export the included hypothesis to.

phaseexporthypothesisfrombuilder

Exports the selected hypothesis from the hypotheses table in the Edit Hypotheses panel.

Syntax:

`phaseexporthypothesisfrombuilder` ⟨file name⟩

Operands:

⟨file name⟩

The file name to export the selected hypothesis to.

phaseexportselectedalignments

Exports the selected alignments from the current Phase step to the project table.

Syntax:

`phaseexportselectedalignments`

phaseexportselectedhypotheses

Exports all selected hypotheses from the hypotheses table in the Score Hypotheses or Build QSAR Model step.

Syntax:

`phaseexportselectedhypotheses` ⟨directory name⟩

Operands:
⟨directory name⟩
The directory name to export all selected hypothesis to.

phaseexportselectedhypothesesfrombuilder

Exports the selected hypotheses from the hypotheses table in the Edit Hypotheses panel.

Syntax:

`phaseexportselectedhypothesesfrombuilder` ⟨directory name⟩

Operands:
⟨directory name⟩
The directory to export the selected hypotheses to.

phaseexportselectedligands

Exports the selected ligands from the current Phase step to the project table.

Syntax:

`phaseexportselectedligands`

phaseextendsite

Extends the selection to the given site in the freestyle hypothesis

Syntax:

`phaseextendsite` ⟨site index⟩

Operands:
⟨site index⟩
The index of the site to select.

phasefieldqsar

Command for controlling Phase Field-Based QSAR Models

Syntax:

phasefieldqsar *contourstyle*=solid | mesh | dots
crossvalidation=⟨ n ⟩ *electrostaticcutoff*=⟨ x ⟩
eliminatetvalue=yes | no *extendgrid*=⟨ x ⟩
featuredefinitions=⟨ text ⟩ *fieldstyle*=ff | gauss | custom
grid_spacing=⟨ x ⟩ *ignoredistance*=⟨ x ⟩ *ignorefields*=yes | no
max_factors=⟨ n ⟩ *random*=⟨ n ⟩ *random_seed*=⟨ n ⟩
stddevcutoff=⟨ x ⟩ *stericcutoff*=⟨ x ⟩
truncateelectrostatic=yes | no *truncatesteric*=yes | no
tvaluecutoff=⟨ x ⟩ *useinputpartialcharges*=yes | no

Options:

contourstyle
What style to use for contours.

Valid values:	solid mesh dots
Default value:	**solid**

crossvalidation
Leave-n-out cross validation.

Valid values:	integers
Default value:	**1**
Minimum:	1

electrostaticcutoff
Value for truncating electrostatic fields in kcal/mol.

Valid values:	reals
Default value:	**30**
Minimum:	0.1

eliminatetvalue
Controls whether or not to eliminate variables based on t-value

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

extendgrid Indicates how much (in angstroms) to extend the grid beyond the limits of the training set ligands.

Valid values:	reals
Default value:	**3**
Minimum:	0.0

featuredefinitions
: The file name of a feature definitions file to use with a custom field style.

  Valid values: text strings
  Default value:

fieldstyle
: What type of field to use for generating the model.

  Valid values: ff
  gauss
  custom
  Default value: **gauss**

grid_spacing
: The grid spacing to be for the QSAR model grid. The valid range is 0.5 to 4.0 angstrom.

  Valid values: reals
  Default value: **1**
  Minimum: 0.5
  Maximum: 4.0

ignoredistance
: The distance with which to ignore fields.

  Valid values: reals
  Default value: **2**
  Minimum: 0.0

ignorefields
: Indicates whether or not Phase should ignore Force field fields.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

max_factors
: This option value range depends on the ligands in the training set. The valid range is 1 to N/5, where N is the number of ligands in the training set. If N is less than 5, the maximum number of factors is 1, but no QSAR model can be constructed.

  Valid values: integers
  Default value: **1**
  Minimum: 1

random
: The percentage of the ligands to set as the training set.

  Valid values: integers
  Default value: **50**
  Minimum: 0
  Maximum: 100

random_seed
: The seed used to compute the random training / test set in the Individual QSAR Model panel. Zero means to use a completely random seed–any other value is used explicitly.

  Valid values: integers
  Default value: **0**
  Minimum: 0

stddevcutoff
: Threshold for standard deviation

  Valid values: reals
  Default value: **0.01**
  Minimum: 0.001

stericcutoff
: Value for truncating steric fields in kcal/mol.

  Valid values: reals
  Default value: **30**
  Minimum: 0.1

truncateelectrostatic
: Whether or not to truncate the Force field electrostatic fields.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

truncatesteric
: Controls whether or not Phase truncates steric fields.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

tvaluecutoff
: The cutoff value for eliminating variables based on t-value.

  Valid values: reals
  Default value: **2**

useinputpartialcharges
: Whether to use the input structures' partial charges.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

phasefieldqsaraddligands

Adds the given ligands to the Individual QSAR Model panel.

Syntax:

phasefieldqsaraddligands *activity*=⟨ text ⟩
convert_activity=yes | no *convert_scale*=⟨ x ⟩
set_property=⟨ text ⟩ *test_value*=⟨ text ⟩ *training_value*=⟨ text ⟩
⟨ ESL ⟩

Options:

activity This determines which property (if any) to use as the activity property for the ligands.
Valid values: text strings
Default value:

convert_activity
Set to true if the activity values should be converted from concentration to -log[concentration]
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

convert_scale
A scale factor for conversion.
Valid values: reals
Default value: **1**

set_property
This property is used to read the training / test set information from.
Valid values: text strings
Default value:

test_value The value which is considered part of the test set.
Valid values: text strings
Default value:

training_value
The value which is considered part of the training set.
Valid values: text strings
Default value:

Operands:
⟨ ESL ⟩
The entries to add as ligands.

phasefieldqsaraddligandsfromfile

Adds the ligands in the given files to the Phase Field-Based QSAR Model panel.

Syntax:

`phasefieldqsaraddligandsfromfile` ⟨file 1⟩ ⟨file 2⟩

Operands:
⟨file 1⟩ ⟨file 2⟩
The files to add ligands from.

phasefieldqsaraddtoproject

Adds the selected ligands from the Field-Based QSAR Model panel to the Project Table.

Syntax:

`phasefieldqsaraddtoproject`

phasefieldqsarbuild

Build a QSAR model from the current ligands.

Syntax:

`phasefieldqsarbuild`

phasefieldqsardelete

Delete the selected ligands.

Syntax:

`phasefieldqsardelete`

phasefieldqsardeleteall

Deletes all of the ligands from the field-based QSAR panel.

Syntax:

`phasefieldqsardeleteall`

phasefieldqsarexport

Exports the QSAR model.

Syntax:

`phasefieldqsarexport` ⟨file name⟩

Operands:
⟨file name⟩
The name of the file to save the QSAR model to.

phasefieldqsarexportligandstofile

Exports the selected ligands from the Field-Based QSAR model.

Syntax:

`phasefieldqsarexportligandstofile` ⟨file name⟩

Operands:
⟨file name⟩
The file name to export the selected conformers to.

phasefieldqsarextendinclude

Extends the inclusion to the given ligand in the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsarextendinclude` ⟨row⟩

Operands:
⟨row⟩
The row number to include.

phasefieldqsarextendselectligandrow

Extends the selection to the given row in the ligands table in the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsarextendselectligandrow` ⟨row⟩

Operands:
⟨row⟩
The row number to extend-select.

phasefieldqsarimport

Imports a QSAR model from the given file.

Syntax:

`phasefieldqsarimport` ⟨file⟩

Operands:
⟨file⟩
The QSAR model file to import.

phasefieldqsarinclude

Includes just the given ligand in the Field-Based QSAR Model panel in the Workspace.

Syntax:

`phasefieldqsarinclude` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include.

phasefieldqsarinvertselection

Invert the selection in the ligands table.

Syntax:

`phasefieldqsarinvertselection`

phasefieldqsarplotactivity

Creates a scatter plot of predicted versus actual activities for the Field-Based QSAR Model.

Syntax:

`phasefieldqsarplotactivity` *draw_line*=yes | no
pls_factor=⟨ n ⟩ ⟨ all | selected | training | test ⟩

Options:

draw_line If set, then the plot will include a line at 45 degrees.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **true**

pls_factor This is which set of PLS factors to plot.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ all | selected | training | test ⟩
Which ligands to plot: all, selected, the training set, or the test set.

phasefieldqsarpredict

Predicts activities for the given entries using the Field-Based QSAR Model.

Syntax:

`phasefieldqsarpredict` ⟨ESL⟩

Operands:
⟨ESL⟩
The entries to predict activities for.

phasefieldqsarrandomtraining

Sets the random training set for the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsarrandomtraining`

phasefieldqsarselectall

Selects all of the ligands.

Syntax:

`phasefieldqsarselectall`

phasefieldqsarselectcustom

Selects just the given field in the Custom Field Style dialog

Syntax:

`phasefieldqsarselectcustom` ⟨field⟩

Operands:
⟨field⟩
The custom field to select

phasefieldqsarselectligandrow

Selects the given row in the ligands table on the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsarselectligandrow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to select.

phasefieldqsarsetactivity

Sets the activity for the given row in the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsarsetactivity` *activity*=⟨ x ⟩ ⟨ row ⟩

Options:

activity The activity value for the given ligand.

Valid values: reals
Default value: **1**

Operands:

⟨ row ⟩

The row number of the ligand to set the activity for.

phasefieldqsarsettrainingrows

Toggles the Training Set property on or off for the given row in the ligands table on the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsarsettrainingrows` *value*=training | test | none ⟨rows⟩

Options:

value The value of the training/test column in the ligands table in the Field-Based QSAR Model panel.

Valid values:	training
	test
	none
Default value:	**training**

Operands:

⟨rows⟩

The row numbers to toggle, two row numbers should be seprated by , or if the user wants to specify a range then it can be given like 1-5 seperated by a - . eg 1,2,5 or 1,4-7 .

phasefieldqsarsort

Sort the Field-Based QSAR Model ligands table based on the data in the specified column

Syntax:

`phasefieldqsarsort` ⟨column_name⟩

Operands:

⟨column_name⟩

The name of the column to be sorted.

phasefieldqsartest

Tests a QSAR model on the current ligand test set.

Syntax:

`phasefieldqsartest`

phasefieldqsartoggleinclude

Toggles the given ligand in the Field-Based QSAR Model panel into or out of the Workspace.

Syntax:

`phasefieldqsartoggleinclude` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle.

phasefieldqsartoggleligandrow

Toggles the selection for the given row in the ligands table in the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsartoggleligandrow` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle-select.

phasefieldqsartogglerow

Toggles the Training Set property on or off for the given row in the Field-Based QSAR Model panel.

Syntax:

`phasefieldqsartogglerow` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle.

phasefieldqsartoggleselectcustom

Toggles the given field in the Custom Field Style dialog.

Syntax:

`phasefieldqsartoggleselectcustom` ⟨ field ⟩

Operands:
⟨ field ⟩
The custom field to toggle select

phasefieldqsarvisselectcontour

Selects just the given field contour to visualize.

Syntax:

`phasefieldqsarvisselectcontour` ⟨ field ⟩

Operands:
⟨ field ⟩
The field contour to select for visualization

phasefieldqsarvisselectintensity

Selects just the given field intensity to visualize.

Syntax:

`phasefieldqsarvisselectintensity` ⟨ field ⟩

Operands:
⟨ field ⟩
The field intensity to select for visualization

phasefieldqsarvistoggleselectcontour

Toggle-selects the given field contour to visualize.

Syntax:

phasefieldqsarvistoggleselectcontour ⟨ field ⟩

Operands:
⟨ field ⟩
The field contour to toggle-select for visualization

phasefindmatches

Launches a Find Matches job for start operand. Writes Find Matches job input files for write operand.

Syntax:

phasefindmatches *align_cutoff*=⟨ x ⟩ *align_weight*=⟨ x ⟩
apply_constraints=yes | no *apply_excluded_volumes*=yes | no
apply_must_match=yes | no *apply_permitted*=yes | no
apply_tolerances=yes | no *computeincludedvolume*=yes | no
conformers=existing | generate *cpu_limit*=⟨ x ⟩
database=⟨ text ⟩ *distance_tolerance*=⟨ x ⟩ *entry_name*=⟨ text ⟩
extfile=⟨ text ⟩ *generate_sites*=yes | no *hits_molecule*=⟨ n ⟩
hits_total=⟨ n ⟩ *included_volume_cutoff*=⟨ x ⟩
included_volume_weight=⟨ x ⟩ *jobinput*=extfile | selectedentries
| 3ddatabase *matches*=new | existing *minsites*=⟨ n ⟩
prefer_big_matches=yes | no *refine*=yes | no
remote_database=yes | no *saveallmatches*=yes | no
score_in_place=yes | no *sorthits*=yes | no *subset*=⟨ text ⟩
use_align_cutoff=yes | no *use_cpu_limit*=yes | no
use_distance_tolerance=yes | no
use_included_volume_cutoff=yes | no
use_vector_cutoff=yes | no *use_volume_cutoff*=yes | no
use_volume_groups=yes | no *usedbkeys*=yes | no
useqsar=yes | no *vector_cutoff*=⟨ x ⟩ *vector_weight*=⟨ x ⟩
verbose=yes | no *volume_cutoff*=⟨ x ⟩ *volume_weight*=⟨ x ⟩
⟨ write | start ⟩

Options:

align_cutoff
The cutoff for the alignment score.

Valid values:	reals
Default value:	**1.2**
Minimum:	0.0001

align_weight
: The weight for the alignment score

  Valid values: reals
  Default value: **1**
  Minimum: 0.0

apply_constraints
: Whether or not to apply constraints.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

apply_excluded_volumes
: Indicates whether or not to use the excluded volumes information.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

apply_must_match
: Whether or not to apply site must-match values.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

apply_permitted
: Whether or not to apply site-based permitted and prohibited matching criteria.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

apply_tolerances
: Whether or not to apply site tolerances.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

computeincludedvolume
: Indicates whether or not to compute the included volume scores.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

conformers
: Whether to use existing conformers or generate them during the search

  Valid values: existing
  generate
  Default value: **generate**

cpu_limit The CPU limit for each molecule, in seconds.

  Valid values: reals

Default value: **0.1**
Minimum: 0.1

database The source database file for Phase Find Matches.
Valid values: text strings
Default value:

distance_tolerance
Valid values: reals
Default value: **2**

entry_name
The entry name to use as the hypothesis to search against.
Valid values: text strings
Default value:

extfile The source structure .sd or .mae file for Phase Find Matches.
Valid values: text strings
Default value:

generate_sites
Use hypothesis feature definitions to generate sites in memory for each database structure searched.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

hits_molecule
The maximum number of hits per molecule to return.
Valid values: integers
Default value: **1**
Minimum: 1

hits_total The maximum number of hits to return.
Valid values: integers
Default value: **1000**
Minimum: 1

included_volume_cutoff
The cutoff for the included volume score.
Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

included_volume_weight
The weight for included volume scores.
Valid values: reals
Default value: **1**
Minimum: 0.0

jobinput The source of job input of Phase Find Matches.

Valid values: extfile
selectedentries
3ddatabase
Default value: **3ddatabase**

matches Whether to search for new or existing matches.

Valid values: new
existing
Default value: **new**

minsites The minimum number of sites to match.

Valid values: integers
Default value: **5**
Minimum: 1

prefer_big_matches
If this option is true, there is a preference for partial matches that involve a larger number of sites.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

refine Whether or not to refine matches

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

remote_database
If set, then this means that we have a remote database which isn't necessarily accessible to the local machine.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

saveallmatches
If this option is true, all the matches will be written to disk, to the file matches.out.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

score_in_place
Indicates whether or not the structures get aligned to the pharmacophore before being scored.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sorthits Sort hits by decreasing fitness

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

subset The name of the subset for the Phase Find Matches.

Valid values: text strings
Default value:

use_align_cutoff
The flag for filtering hits based on align score. If this option is true, reject hits with align score greater than align_cutoff.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use_cpu_limit
If set, then use a per-molecule CPU time limit.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use_distance_tolerance
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

use_included_volume_cutoff
The flag for filtering hits based on included volume score. If this option is true, reject hits with an included volume score less than the cutoff.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use_vector_cutoff
The flag for filtering hits based on vector score. If this option is true, reject hits with vector score less than vector_cutoff.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

use_volume_cutoff
The flag for filtering hits based on volume score. If this option is true, reject hits with volume score less than volume_cutoff.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

use_volume_groups
If this option is true, volume scores will be higher for alignments that overlap atoms of the same MacroModel type.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

usedbkeys If this option is true, phase_dbsearch uses database keys to perform a single-CPU pre-screen of the database to eliminate molecules.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

useqsar The flag of using QSAR model or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

vector_cutoff
The cutoff for the vector score.

Valid values: reals
Default value: **-1**
Minimum: -1.0
Maximum: 1.0

vector_weight
The weight for the vector score.

Valid values: reals
Default value: **1**
Minimum: 0.0

verbose If set, the Find Matches job will produce more verbose output.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

volume_cutoff
The cutoff for the volume score.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

volume_weight
The weight for the volume score.

Valid values: reals
Default value: **1**
Minimum: 0.0

Operands:
⟨ write|start ⟩
This should be set to Find Matches .

phasefindpharmacophores

Sets the Find Common Pharmacophores options and launches the job.

Syntax:

`phasefindpharmacophores` *finalboxsize*=⟨ x ⟩ *maxtreedepth*=⟨ n ⟩ *minintersitedistance*=⟨ x ⟩

Options:

finalboxsize
: The final box size in angstrong.
  Valid values: reals
  Default value: **1**

maxtreedepth
: The maximum depth of the tree.
  Valid values: integers
  Default value: **5**

minintersitedistance
: The minimum distance between the sites in angstrong.
  Valid values: reals
  Default value: **2**

phasegenerateconformers

Launches a Generate Conformers job for Phase.

Syntax:

`phasegenerateconformers`

phasegroupligands

Group together ligands

Syntax:

`phasegroupligands`

phasegroupligandsbytitle

Group together ligands by title

Syntax:

```
phasegroupligandsbytitle
```

phasehookimport

Sets up the Import panel and dialog for Phase to import ligands and conformers. This command is deprecated as of Maestro 8.0. Use phaseaddligandsfromfile

Syntax:

```
phasehookimport
```

phasehypothesiscreateexcludedvolume

Creates an excluded volume from the current centroid atoms for the currently selected hypothesis.

Syntax:

```
phasehypothesiscreateexcludedvolume
```

phasehypothesisdeleteexcludedvolumes

Deletes the selected excluded volumes for the currently selected hypothesis.

Syntax:

```
phasehypothesisdeleteexcludedvolumes
```

phasehypothesisselectevrow

Selects the given row in the excluded volumes table for the currently selected hypothesis.

Syntax:

`phasehypothesisselectevrow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to select in the table.

phasehypothesisselectextendevrow

Extends the selection to this row in the excluded volumes table for the currently selected hypothesis.

Syntax:

`phasehypothesisselectextendevrow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to extend the select to.

phasehypothesisselectonlyevrow

Selects only this row in the excluded volumes table for the currently selected hypothesis.

Syntax:

`phasehypothesisselectonlyevrow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to select only in the table row.

phasehypothesisselectrow

Toggles the given hypothesis into or out of the Workspace based on the currently selected hypothesis.

Syntax:

phasehypothesisselectrow ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to toggle inclusion state for.

phasehypothesissetexcludedvolumes

Sets the value for the given cell for the currently selected hypothesis.

Syntax:

phasehypothesissetexcludedvolumes *column*=⟨ n ⟩ *row*=⟨ n ⟩ ⟨ value ⟩

Options:

column	The column number of the cell to change.	
	Valid values:	integers
	Default value:	**1**
row	The row number of the cell to change.	
	Valid values:	integers
	Default value:	**1**

Operands:
⟨ value ⟩
The value for the given cell.

phasehypothesissetid

Sets the value for the hypothesis ID for the currently selected hypothesis.

Syntax:

phasehypothesissetid *row*=⟨ n ⟩ ⟨ value ⟩

Options:

row The row number of the hypothesis ID to change.

Valid values: integers
Default value: **1**

Operands:
⟨ value ⟩
The value for the hypothesis ID.

phasehypothesissetsitemask

Sets the sitemask for the currently selected hypothesis

Syntax:

`phasehypothesissetsitemask` ⟨ value ⟩

Operands:
⟨ value ⟩
The value for the given cell.

phasehypothesissettolerances

Sets the feature matching tolerances for the currently selected hypothesis

Syntax:

`phasehypothesissettolerances` *tolerance_a*=⟨ x ⟩
tolerance_d=⟨ x ⟩ *tolerance_h*=⟨ x ⟩ *tolerance_n*=⟨ x ⟩
tolerance_p=⟨ x ⟩ *tolerance_r*=⟨ x ⟩ *tolerance_x*=⟨ x ⟩
tolerance_y=⟨ x ⟩ *tolerance_z*=⟨ x ⟩ *usetolerances*=yes | no

Options:

tolerance_a
The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_d

The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

tolerance_h

The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1.5**
Minimum:	0.0
Maximum:	100.0

tolerance_n

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0.75**
Minimum:	0.0
Maximum:	100.0

tolerance_p

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0.75**
Minimum:	0.0
Maximum:	100.0

tolerance_r

The feature matching tolerance for the aromatic ring feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1.5**
Minimum:	0.0
Maximum:	100.0

tolerance_x

The feature matching tolerance for the custom(X) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

tolerance_y

The feature matching tolerance for the custom(Y) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

tolerance_z

The feature matching tolerance for the custom(Z) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

usetolerances

When using feature matching tolerances, this option is true otherwise false.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

phasehypothesistoggleinclude

Toggles the given hypothesis into or out of the Workspace.

Syntax:

`phasehypothesistoggleinclude` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to toggle inclusion state for.

phasehypothesisunselectevrow

Unselects the given row in the excluded volumes table for the currently selected hypothesis.

Syntax:

`phasehypothesisunselectevrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the table.

phaseimportfeature

Imports the features from the given file name. This replaces the existing features.

Syntax:

`phaseimportfeature` ⟨ file name ⟩

Operands:
⟨ file name ⟩
A path to a valid features file.

phaseimporthypothesis

Imports a hypothesis from the selected file into the Find Matches and Edit Hypotheses panels.

Syntax:

`phaseimporthypothesis` *isphase*=yes | no ⟨ file name ⟩

Options:

isphase Indicates whether or not the hypothesis is a Phase hypothesis or an external hypothesis.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ file name ⟩
The file name to import a hypothesis from.

phaseincludeextendtablerow

Extends the rows included in the workspace to include this one.

Syntax:

`phaseincludeextendtablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include in the Workspace.

phaseincludeonlytablerow

Includes only the given row in the first table in the step into the Workspace.

Syntax:

`phaseincludeonlytablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include in the Workspace.

phaseincludetablerow

Includes the given row in the first table in the step into the Workspace.

Syntax:

`phaseincludetablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include in the Workspace.

phaseinverttableselection

Inverts the row selection in the first table in the step.

Syntax:

```
phaseinverttableselection
```

phaseligprep

Defines settings for Phase LigPrep job.

Syntax:

phaseligprep *gen_stereo*=⟨ n ⟩ *ionization*=generate | neutralize | retain *ph*=⟨ x ⟩ *stereoisomers*=retain | determine | generate

Options:

gen_stereo Generate stereoisomers (maximum)

Valid values:	integers
Default value:	**32**
Minimum:	1

ionization How to generate the ionization states

Valid values:	generate neutralize retain
Default value:	**retain**

ph The ionization pH

Valid values:	reals
Default value:	**7**
Minimum:	0.0

stereoisomers
How to generate the stereoisomers

Valid values:	retain determine generate
Default value:	**retain**

phaselocatesites

Launches a Locate Sites job.

Syntax:

`phaselocatesites`
phasemarkerdump

Print out the current option values of the phase marker command.

Syntax:

`phasemarkerdump`
phasemarkersettings

Set graphical data of feature markers.

Syntax:

phasemarkersettings *acceptorarrowblue*=⟨ x ⟩
acceptorarrowgreen=⟨ x ⟩ *acceptorarrowred*=⟨ x ⟩
acceptorcornradius=⟨ x ⟩ *acceptorcylinderheight*=⟨ x ⟩
acceptorcylinderradius=⟨ x ⟩ *acceptorsiteblue*=⟨ x ⟩
acceptorsitegreen=⟨ x ⟩ *acceptorsiteradius*=⟨ x ⟩
acceptorsitered=⟨ x ⟩ *ambient*=⟨ x ⟩ *aromaticringradius*=⟨ x ⟩
aromaticringradius5=⟨ x ⟩ *aromaticsiteblue*=⟨ x ⟩
aromaticsitegreen=⟨ x ⟩ *aromaticsitered*=⟨ x ⟩
aromatictuberadius=⟨ x ⟩ *aromaticustep*=⟨ n ⟩
aromaticvstep=⟨ n ⟩ *diffuse*=⟨ x ⟩ *donorarrowblue*=⟨ x ⟩
donorarrowgreen=⟨ x ⟩ *donorarrowred*=⟨ x ⟩
donorcornradius=⟨ x ⟩ *donorcylinderheight*=⟨ x ⟩
donorcylinderradius=⟨ x ⟩ *donorsiteblue*=⟨ x ⟩
donorsitegreen=⟨ x ⟩ *donorsiteradius*=⟨ x ⟩ *donorsitered*=⟨ x ⟩
drawstyle=solid | line *emission*=⟨ x ⟩ *featurexsiteblue*=⟨ x ⟩
featurexsitegreen=⟨ x ⟩ *featurexsiteradius*=⟨ x ⟩
featurexsitered=⟨ x ⟩ *featureysiteblue*=⟨ x ⟩
featureysitegreen=⟨ x ⟩ *featureysiteradius*=⟨ x ⟩
featureysitered=⟨ x ⟩ *featurezsiteblue*=⟨ x ⟩
featurezsitegreen=⟨ x ⟩ *featurezsiteradius*=⟨ x ⟩
featurezsitered=⟨ x ⟩ *hydrophobicsiteblue*=⟨ x ⟩
hydrophobicsitegreen=⟨ x ⟩ *hydrophobicsiteradius*=⟨ x ⟩
hydrophobicsitered=⟨ x ⟩ *labelblue*=⟨ x ⟩ *labelgreen*=⟨ x ⟩
labelred=⟨ x ⟩ *linewidth*=⟨ n ⟩ *negativesiteblue*=⟨ x ⟩
negativesitegreen=⟨ x ⟩ *negativesiteradius*=⟨ x ⟩
negativesitered=⟨ x ⟩ *positivesiteblue*=⟨ x ⟩
positivesitegreen=⟨ x ⟩ *positivesiteradius*=⟨ x ⟩
positivesitered=⟨ x ⟩ *projectedsiteblue*=⟨ x ⟩
projectedsitegreen=⟨ x ⟩ *projectedsiteradius*=⟨ x ⟩
projectedsitered=⟨ x ⟩ *shininess*=⟨ x ⟩ *sitedimpercentage*=⟨ x ⟩
sliceline=⟨ n ⟩ *slicesolid*=⟨ n ⟩ *specular*=⟨ x ⟩ *stackline*=⟨ n ⟩
stacksolid=⟨ n ⟩ *transparency*=⟨ x ⟩

Options:

acceptorarrowblue

The arrow color blue component of acceptor markers.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

acceptorarrowgreen

The arrow color green component of acceptor markers.

Valid values:	reals
Default value:	**0.5**

Minimum: 0.0
Maximum: 1.0

acceptorarrowred

The arrow color red component of acceptor markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

acceptorcornradius

The radius of arrow corn of acceptor markers.

Valid values: reals
Default value: **0.35**
Minimum: 0.0

acceptorcylinderheight

The height of arrow cylinder of acceptor markers.

Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

acceptorcylinderradius

The radius of arrow cylinder of acceptor markers.

Valid values: reals
Default value: **0.15**
Minimum: 0.0

acceptorsiteblue

The site color blue component of acceptor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

acceptorsitegreen

The site color green component of acceptor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

acceptorsiteradius

The radius of acceptor site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

acceptorsitered
The site color red component of acceptor markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

aromaticringradius
The radius of the ring of the ring torus, for the ring constructed by 6 or more atoms.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

aromaticringradius5
The radius of the ring of the ring torus, for the ring constructed by 5 or less atoms.

Valid values: reals
Default value: **0.4**
Minimum: 0.0

aromaticsiteblue
The site color blue component of aromatic markers.

Valid values: reals
Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

aromaticsitegreen
The site color green component of aromatic markers.

Valid values: reals
Default value: **0.51**
Minimum: 0.0
Maximum: 1.0

aromaticsitered
The site color red component of aromatic markers.

Valid values: reals
Default value: **0.92**

Minimum: 0.0
Maximum: 1.0

aromatictuberadius
The radius of the tube of the ring torus.
Valid values: reals
Default value: **0.15**
Minimum: 0.0

aromaticustep
The U step domain tolerance.
Valid values: integers
Default value: **5**
Minimum: 2

aromaticvstep
The V step domain tolerance.
Valid values: integers
Default value: **3**
Minimum: 2

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.
Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

donorarrowblue
The arrow color blue component of donor markers.
Valid values: reals
Default value: **0.9**
Minimum: 0.0
Maximum: 1.0

donorarrowgreen
The arrow color green component of donor markers.
Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

donorarrowred
The arrow color red component of donor markers.
Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

donorcornradius

The radius of arrow corn of donor markers.

Valid values: reals
Default value: **0.35**
Minimum: 0.0

donorcylinderheight

The height of arrow cylinder of donor markers.

Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

donorcylinderradius

The radius of arrow cylinder of donor markers.

Valid values: reals
Default value: **0.15**
Minimum: 0.0

donorsiteblue

The site color blue component of donor markers.

Valid values: reals
Default value: **0.9**
Minimum: 0.0
Maximum: 1.0

donorsitegreen

The site color green component of donor markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0
Maximum: 1.0

donorsiteradius

The radius of donor site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

donorsitered

The site color red component of donor markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

drawstyle The styles of rendering feature markers, they are: 1 - solid, and 2 - lines. Default is solid.

Valid values:	solid line
Default value:	**solid**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0
Maximum:	1.0

featurexsiteblue

The site color blue component of featurex markers.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

featurexsitegreen

The site color green component of featurex markers.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

featurexsiteradius

The radius of featurex site markers.

Valid values:	reals
Default value:	**0.8**
Minimum:	0.0

featurexsitered

The site color red component of featurex markers.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

featureysiteblue

The site color blue component of featurey markers.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

featureysitegreen

The site color green component of featurey markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

featureysiteradius

The radius of featurey site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

featureysitered

The site color red component of featurey markers.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

featurezsiteblue

The site color blue component of featurez markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

featurezsitegreen

The site color green component of featurez markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

featurezsiteradius

The radius of featurez site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

featurezsitered

The site color red component of featurez markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

hydrophobicsiteblue
The site color blue component of hydrophobic markers.

Valid values:	reals
Default value:	**0.3**
Minimum:	0.0
Maximum:	1.0

hydrophobicsitegreen
The site color green component of hydrophobic markers.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

hydrophobicsiteradius
The radius of hydrophobic site markers.

Valid values:	reals
Default value:	**0.8**
Minimum:	0.0

hydrophobicsitered
The site color red component of hydrophobic markers.

Valid values:	reals
Default value:	**0.3**
Minimum:	0.0
Maximum:	1.0

labelblue The blue component of feature labels.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

labelgreen The green component of feature labels.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

labelred The red component of feature labels.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

linewidth Set the width of lines in drawing sphere.

Valid values:	integers

Default value: **2**
Minimum: 1

negativesiteblue
The site color blue component of negative markers.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

negativesitegreen
The site color green component of negative markers.

Valid values: reals
Default value: **0.3**
Minimum: 0.0
Maximum: 1.0

negativesiteradius
The radius of negative site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

negativesitered
The site color red component of negative markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positivesiteblue
The site color blue component of positive markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positivesitegreen
The site color green component of positive markers.

Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

positivesiteradius
The radius of positive site markers.

Valid values: reals

Default value: **0.8**
Minimum: 0.0

positivesitered
The site color red component of positive markers.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

projectedsiteblue
The site color blue component of projected markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

projectedsitegreen
The site color green component of projected markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

projectedsiteradius
The radius of projected site markers.

Valid values: reals
Default value: **0.8**
Minimum: 0.0

projectedsitered
The site color red component of projected markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

sitedimpercentage
The percentage to dim Phase sites by.

Valid values: reals
Default value: **0.5**

Minimum: 0.0
Maximum: 1.0

sliceline Set the slices of drawing line sphere.

Valid values: integers
Default value: **10**
Minimum: 2

slicesolid Set the slices of drawing solid sphere.

Valid values: integers
Default value: **36**
Minimum: 2

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

stackline Set the stacks of drawing line sphere.

Valid values: integers
Default value: **8**
Minimum: 2

stacksolid Set the stacks of drawing solid sphere.

Valid values: integers
Default value: **18**
Minimum: 2

transparency
The transparency of rendering feature markers.

Valid values: reals
Default value: **20**
Minimum: 0.0
Maximum: 100.0

phasemarkfeature

Marks the given feature in the Workspace.

Syntax:

`phasemarkfeature` ⟨feature⟩

Operands:

⟨feature⟩

A single letter (A-Z) indicating the feature to mark.

phasemergestereoisomers

Merges stereoisomers for the selected conformers.

Syntax:

`phasemergestereoisomers`

phasemergetitles

Merges the selected conformers by title.

Syntax:

`phasemergetitles`

phaseoptions

Sets the options for Phase.

Syntax:

phaseoptions *convert_hypothesis*=yes | no
include_reference_ligand=yes | no *keep_ligands*=yes | no
prompt_convert=yes | no *random_seed*=⟨ n ⟩
separate_stereoisomers=yes | no *show_unmatched*=yes | no
showhypothesisangles=yes | no
showhypothesisconstraints=yes | no
showhypothesisdistances=yes | no
showhypothesisexcludedvolumes=yes | no
showhypothesislabels=yes | no
showhypothesistolerance=yes | no *split_by_title*=yes | no

Options:

convert_hypothesis
If the prompt_convert option is turned off, then this option controls whether or not hypotheses are converted to the current format as needed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

include_reference_ligand
If true then includes the reference ligand when including hypothesis from DPM panel.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

keep_ligands
Controls whether or not the same ligands are kept in the Workspace when switching hypotheses in the Score Hypotheses and Build QSAR steps.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

prompt_convert
If set, then Find Matches will prompt for whether or not to convert hypotheses as needed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

random_seed
The seed used to compute the random training / test set in the Build QSAR step. Zero means to use a completely random seed–any other value is used explicitly.

Valid values: integers
Default value: **0**
Minimum: 0

separate_stereoisomers
: Whether to separate stereoisomers or combine them into a single confset. Deprecated.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

show_unmatched
: Controls whether or not unmatched ligands are displayed in the alignments table in Score Hypotheses.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

showhypothesisangles
: Controls whether or not angles are displayed when hypotheses are included in the Workspace.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

showhypothesisconstraints
: Controls whether or not hypothesis constraints are displayed in the Workspace.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

showhypothesisdistances
: Controls whether or not inter-site distances are displayed when hypotheses are included in the Workspace.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

showhypothesisexcludedvolumes
: Controls whether or not excluded volumes are displayed when hypotheses are included in the Workspace.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

showhypothesislabels
: Controls whether or not hypothesis labels are displayed in the Workspace.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

showhypothesistolerance
: Controls whether or not hypotheses are shown as tolerance markers.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

split_by_title
: Controls whether or not conformers are split by title.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

phasepatterndelete

Deletes the given pattern.

Syntax:

`phasepatterndelete` *pattern*=⟨ text ⟩ ⟨ feature ⟩

Options:

pattern
: The SMARTS pattern to remove.

  Valid values: text strings
  Default value:

Operands:

⟨ feature ⟩

The feature to remove the pattern from.

phasepatternedit

Updates the given pattern with the new data.

Syntax:

`phasepatternedit` *geometry*=group | point | vector *group*=⟨ text ⟩ *group_all*=yes | no *pattern*=⟨ text ⟩ *point*=⟨ n ⟩ *vector*=⟨ n ⟩ ⟨ feature ⟩

Options:

geometry
: The type of geometry for this pattern.

  Valid values: group
  point
  vector
  Default value: **group**

group The atom numbers for the pattern if it is group geometry.
Valid values: text strings
Default value:

group_all Set to true if the group should include all atom numbers.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pattern The SMARTS pattern to update.
Valid values: text strings
Default value:

point The atom number for the pattern if it is point geometry.
Valid values: integers
Default value: **1**
Minimum: 1

vector The atom number for the pattern if it is vector geometry.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ feature ⟩
The feature to which the pattern is associated with.

phasepatternmovedown

Moves the given pattern down lower in the list for the feature.

Syntax:

`phasepatternmovedown` *pattern*=⟨ text ⟩ ⟨ feature ⟩

Options:

pattern The SMARTS pattern to move.
Valid values: text strings
Default value:

Operands:
⟨ feature ⟩
The feature which contains the pattern.

phasepatternmoveup

Moves the given pattern up higher in the list for the feature.

Syntax:

phasepatternmoveup *pattern*=⟨ text ⟩ ⟨ feature ⟩

Options:

pattern The SMARTS pattern to move.
Valid values: text strings
Default value:

Operands:
⟨ feature ⟩
The feature which contains the pattern.

phasepatternnew

Adds the given pattern.

Syntax:

phasepatternnew *geometry*=group | point | vector *group*=⟨ text ⟩ *group_all*=yes | no *pattern*=⟨ text ⟩ *point*=⟨ n ⟩ *vector*=⟨ n ⟩ ⟨ feature ⟩

Options:

geometry The type of geometry for this pattern.
Valid values: group
point
vector
Default value: **group**

group The atom numbers for the pattern if it is group geometry.
Valid values: text strings
Default value:

group_all Set to true if the group should include all atom numbers.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pattern The SMARTS pattern to add.
Valid values: text strings
Default value:

point The atom number for the pattern if it is point geometry.
Valid values: integers
Default value: **1**
Minimum: 1

vector The atom number for the pattern if it is vector geometry.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ feature ⟩
The feature to add the pattern to.

phasepatternsetoptions

Sets the options for the given pattern.

Syntax:

phasepatternsetoptions *exclude*=yes | no *ignore*=yes | no *mark*=yes | no *pattern*=⟨ text ⟩ ⟨ feature ⟩

Options:

exclude Indicates whether or not to exclude the pattern from the feature.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ignore Indicates whether or not to ignore the pattern in the feature.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

mark Indicates whether or not to mark the pattern in the Workspace.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pattern The SMARTS pattern to update.
Valid values: text strings
Default value:

Operands:
⟨ feature ⟩
The feature which contains the pattern.

phaseplotactivity

Creates a scatter plot of predicted versus actual activities.

Syntax:

`phaseplotactivity` *draw_line*=yes | no *pls_factor*=⟨ n ⟩
⟨ all | selected | training | test ⟩

Options:

draw_line If set, then the plot will include a line at 45 degrees.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **true**

pls_factor This is which set of PLS factors to plot.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ all | selected | training | test ⟩
Which ligands to plot: all, selected, the training set, or the test set.

phasepreview

Launches a Preview job.

Syntax:

`phasepreview`

phaseqsar

Command for controlling Phase Individual QSAR Models

Syntax:

phaseqsar *crossvalidation*=⟨ n ⟩ *eliminatetvalue*=yes | no
grid_spacing=⟨ x ⟩ *max_factors*=⟨ n ⟩ *random*=⟨ n ⟩
random_seed=⟨ n ⟩ *tvalue*=⟨ x ⟩

Options:

crossvalidation

Leave-n-out cross validation.

Valid values: integers
Default value: **1**
Minimum: 1

eliminatetvalue

Inidicates whether to use t-value filter before launching QSAR job.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

grid_spacing

The grid spacing to be for the QSAR model grid. The valid range is 0.5 to 2.0 angstrom.

Valid values: reals
Default value: **1**
Minimum: 0.5
Maximum: 2.0

max_factors

This option value range depends on the ligands in the training set. The valid range is 1 to N/5, where N is the number of ligands in the training set. If N is less than 5, the maximum number of factors is 1, but no QSAR model can be constructed.

Valid values: integers
Default value: **1**
Minimum: 1

random The percentage of the ligands to set as the training set.

Valid values: integers
Default value: **50**
Minimum: 0
Maximum: 100

random_seed

The seed used to compute the random training / test set in the Individual QSAR Model panel. Zero means to use a completely random seed–any other value is used explicitly.

Valid values: integers
Default value: **0**
Minimum: 0

tvalue This option value indicates the values to be used to filter t-value.
Valid values: reals
Default value: **2**
Minimum: 0.01

phaseqsaraddligands

Adds the given ligands to the Individual QSAR Model panel.

Syntax:

`phaseqsaraddligands` *activity*=⟨ text ⟩ *convert_activity*=yes | no *convert_scale*=⟨ x ⟩ *set_property*=⟨ text ⟩ *test_value*=⟨ text ⟩ *training_value*=⟨ text ⟩ ⟨ ESL ⟩

Options:

activity This determines which property (if any) to use as the activity property for the ligands.
Valid values: text strings
Default value:

convert_activity
Set to true if the activity values should be converted from concentration to -log[concentration]
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

convert_scale
A scale factor for conversion.
Valid values: reals
Default value: **1**

set_property
This property is used to read the training / test set information from.
Valid values: text strings
Default value:

test_value The value which is considered part of the test set.
Valid values: text strings
Default value:

training_value
The value which is considered part of the training set.

Valid values: text strings
Default value:

Operands:

⟨ ESL ⟩

The entries to add as ligands.

phaseqsaraddligandsfromfile

Adds the ligands in the given files to the Phase Individual QSAR Model panel.

Syntax:

`phaseqsaraddligandsfromfile` ⟨ file 1 ⟩ ⟨ file 2 ⟩

Operands:

⟨ file 1 ⟩ ⟨ file 2 ⟩

The files to add ligands from.

phaseqsaraddtohypothesis

Adds the current Individual QSAR Model to the given entries, which must be hypotheses.

Syntax:

`phaseqsaraddtohypothesis` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

The entries to add the current QSAR model to.

phaseqsaraddtoproject

Adds the selected ligands from the Individual QSAR Model panel to the Project Table.

Syntax:

```
phaseqsaraddtoproject
```

phaseqsarbuild

Build a QSAR model from the current ligands.

Syntax:

```
phaseqsarbuild
```

phaseqsardelete

Delete the selected ligands.

Syntax:

```
phaseqsardelete
```

phaseqsardeleteall

Deletes all of the ligands from the atom-based QSAR panel.

Syntax:

```
phaseqsardeleteall
```

phaseqsarexport

Exports the QSAR model.

Syntax:

`phaseqsarexport` ⟨file name⟩

Operands:

⟨file name⟩

The name of the file to save the QSAR model to.

phaseqsarexportligandstofile

Exports the selected conformers from the ligands table in the Prepare Ligands or Create Sites steps.

Syntax:

`phaseqsarexportligandstofile` ⟨file name⟩

Operands:

⟨file name⟩

The file name to export the selected conformers to.

phaseqsarextendinclude

Extends the inclusion to the given ligand in the Individual QSAR Model panel.

Syntax:

`phaseqsarextendinclude` ⟨row⟩

Operands:

⟨row⟩

The row number to include.

phaseqsarextendselectligandrow

Extends the selection to the given row in the ligands table on the Individual QSAR Model panel.

Syntax:

`phaseqsarextendselectligandrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend-select.

phaseqsarimport

Imports a QSAR model from the given file.

Syntax:

`phaseqsarimport` ⟨ file ⟩

Operands:
⟨ file ⟩
The QSAR model file to import.

phaseqsarinclude

Includes just the given ligand in the Individual QSAR Model panel in the Workspace.

Syntax:

`phaseqsarinclude` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include.

phaseqsarinvertselection

Invert the selection in the ligands table.

Syntax:

`phaseqsarinvertselection`

phaseqsarplotactivity

Creates a scatter plot of predicted versus actual activities for the Individual QSAR Model.

Syntax:

`phaseqsarplotactivity` *draw_line*=yes | no *pls_factor*=⟨ n ⟩
⟨ all | selected | training | test ⟩

Options:

draw_line If set, then the plot will include a line at 45 degrees.

Valid values:	boolean (true | false; yes | no; y | n; on | off)
Default value:	**true**

pls_factor This is which set of PLS factors to plot.

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:

⟨ all | selected | training | test ⟩

Which ligands to plot: all, selected, the training set, or the test set.

phaseqsarpredict

Predicts activities for the given entries using the Individual QSAR Model.

Syntax:

`phaseqsarpredict` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

The entries to predict activities for.

phaseqsarrandomtraining

Sets the random training set for the Individual QSAR Model panel.

Syntax:

`phaseqsarrandomtraining`

phaseqsarsearch

Transfers the current hypothesis from the Build QSAR step to the Find Matches panel. Also opens the Find Matches panel. Deprecated - use phasesearchformatches instead.

Syntax:

`phaseqsarsearch`

phaseqsarselectall

Selects all of the ligands.

Syntax:

`phaseqsarselectall`

phaseqsarselecthypothesis

Selects the given hypothesis.

Syntax:

`phaseqsarselecthypothesis` ⟨ ID ⟩

Operands:
⟨ ID ⟩
The ID of the hypothesis to select in the Build QSAR step. This will populate the Alignment table. This function is single select. If the operand is blank, the alignments table will be filled in for the currently selected hypothesis, if any.

phaseqsarselectligandrow

Selects the given row in the ligands table on the Individual QSAR Model panel.

Syntax:

`phaseqsarselectligandrow` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number to select.

phaseqsarsetactivity

Sets the activity for the given row in the Individual QSAR Model panel.

Syntax:

`phaseqsarsetactivity` *activity*=⟨ x ⟩ ⟨ row ⟩

Options:

activity	The activity value for the given ligand.	
	Valid values:	reals
	Default value:	**1**

Operands:

⟨ row ⟩

The row number of the ligand to set the activity for.

phaseqsarsettrainingrows

Toggles the Training Set property on or off for the given row in the ligands table on the Individual QSAR Model panel.

Syntax:

`phaseqsarsettrainingrows` *value*=training | test | none ⟨rows⟩

Options:

value The value of the training/test column in the ligands table in the Individual QSAR Model panel.

Valid values:	training
	test
	none
Default value:	**training**

Operands:

⟨rows⟩

The row numbers to toggle, two row numbers should be seprated by , or if the user wants to specify a range then it can be given like 1-5 seperated by a - . eg 1,2,5 or 1,4-7 .

phaseqsarsort

Sort the Individual QSAR Model ligands table based on the data in the specified column

Syntax:

`phaseqsarsort` ⟨column_name⟩

Operands:

⟨column_name⟩

The name of the column to be sorted.

phaseqsartest

Tests a QSAR model on the current ligand test set.

Syntax:

`phaseqsartest`

phaseqsartoggleinclude

Toggles the given ligand in the Individual QSAR Model panel into or out of the Workspace.

Syntax:

`phaseqsartoggleinclude` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle.

phaseqsartoggleligandrow

Toggles the selection for the given row in the ligands table on the Individual QSAR Model panel.

Syntax:

`phaseqsartoggleligandrow` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle-select.

phaseqsartogglerow

Toggles the Training Set property on or off for the given row in the Individual QSAR Model panel.

Syntax:

`phaseqsartogglerow` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle.

phaserandomtraining

Sets the random training set.

Syntax:

```
phaserandomtraining
```

phaserandomtrainingoptions

Holds the options for the random training set.

Syntax:

`phaserandomtrainingoptions` *percentage*=⟨ n ⟩

Options:

percentage The percentage of ligands to use as a training set.

Valid values:	integers
Default value:	**50**
Minimum:	1
Maximum:	100

phaserefreshfrequencytable

Restores the default set of min/max values to the feature frequencies table in the Find Common Pharamcophores step.

Syntax:

```
phaserefreshfrequencytable
```

phaseremoveligands

Removes the given ligands from the table.

Syntax:

`phaseremoveligands` ⟨ESL⟩

Operands:
⟨ESL⟩
The entries to remove as ligands.

phaserenameligand

Changes the name for the given ligand.

Syntax:

`phaserenameligand` *row*=⟨n⟩ ⟨new name⟩

Options:

row	The row number of the ligand to rename.	
	Valid values:	integers
	Default value:	**1**
	Minimum:	1

Operands:
⟨new name⟩
The new name for the given ligand.

phaserescorehypotheses

Rescores all the hypotheses in the hypotheses table in the Score Hypotheses step.

Syntax:

`phaserescorehypotheses`

phaserescoreweighting

Sets the weighting factors for rescoring hypotheses in the Score Hypotheses step.

Syntax:

phaserescoreweighting *activity*=⟨ x ⟩ *energy*=⟨ x ⟩ *inactive*=⟨ x ⟩ *match*=⟨ x ⟩ *selectivity*=⟨ x ⟩ *site*=⟨ x ⟩ *vector*=⟨ x ⟩ *volume*=⟨ x ⟩

Options:

activity The reference ligand activity scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	100.0

energy The reference ligand relative conformational energy scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	100.0

inactive The inactive match scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

match The number of matches scoring factor. The valid range is 1.0 to infinity, but values only slightly above 1.0 might generate huge values for a large number of matched ligands.

Valid values:	reals
Default value:	**1**
Minimum:	1.0

selectivity The selectivity scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	100.0

site The aligned sites scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

vector The vector scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

volume The volume scoring factor. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

phaseresetfeatures

Resets the features to the installation defaults.

Syntax:

`phaseresetfeatures`

phaseruncreate

Creates the run with the given name.

Syntax:

`phaseruncreate` ⟨run name⟩

Operands:
⟨run name⟩
The name of the new run to create.

phaserundelete

Deletes the current run from the project.

Syntax:

`phaserundelete`

phaserunopen

Opens the run with the given name.

Syntax:

`phaserunopen` ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to open.

phaserunrename

Changes the current run's name to the given name.

Syntax:

`phaserunrename` ⟨run name⟩

Operands:
⟨run name⟩
The name to change the current run's name to.

phaserunsaveas

Saves a copy of the current run under the given name.

Syntax:

`phaserunsaveas` ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to save as.

phaserunsetseed

Sets the random seed of the current run based on the current Phase options.

Syntax:

```
phaserunsetseed
```

phaserunsetstereo

Sets the stereoisomer behavior of the current run based on the current Phase options.

Syntax:

```
phaserunsetstereo
```

phasesavematching

Saves the advanced matching options for the given hypothesis.

Syntax:

`phasesavematching` ⟨entry ID⟩

Operands:

⟨entry ID⟩

The ID of the hypothesis to save the advanced matching options for.

phasescorehypotheses

Launches a Score Hypotheses job.

Syntax:

phasescorehypotheses *activityweight*=⟨ x ⟩ *aligncutoff*=⟨ x ⟩ *energyweight*=⟨ x ⟩ *matchreward*=⟨ x ⟩ *maxboxes*=⟨ n ⟩ *minboxes*=⟨ n ⟩ *selectivitybasedalignment*=yes | no *selectivityweight*=⟨ x ⟩ *sitetoppercentage*=⟨ n ⟩ *siteweight*=⟨ x ⟩ *tolerance_a*=⟨ x ⟩ *tolerance_d*=⟨ x ⟩ *tolerance_h*=⟨ x ⟩ *tolerance_n*=⟨ x ⟩ *tolerance_p*=⟨ x ⟩ *tolerance_r*=⟨ x ⟩ *tolerance_x*=⟨ x ⟩ *tolerance_y*=⟨ x ⟩ *tolerance_z*=⟨ x ⟩ *usetolerances*=yes | no *vectorlowercutoff*=⟨ x ⟩ *vectorweight*=⟨ x ⟩ *volumebasedalignment*=yes | no *volumeweight*=⟨ x ⟩

Options:

activityweight

For calculating the Survival score, the weighting factor of the reference ligand activity. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

aligncutoff

In case of score by site-based alignment, keep the hypotheses with RMSD values below the value specified by this option. The valid range is 0.0001 to infinity

Valid values: reals
Default value: **1.2**
Minimum: 0.0001

energyweight

For calculating the Survival score, the weighting factor of the reference ligand relative conformational energy. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

matchreward

For calculating the Survival score, increase the score by this value, raised to the power of the number of matched ligands. The valid range is 1.0 to infinity, but values only slightly above 1.0 might generate huge values for a large number of matched ligands.

Valid values: reals
Default value: **1**

Minimum: 1.0

maxboxes In case of score by site-based alignment, keep at most the number specified by this option. The valid range is 1 to infinity

Valid values: integers
Default value: **50**
Minimum: 1

minboxes In case of score by site-based alignment, keep at least the number specified by this option. The valid range is 1 to infinity

Valid values: integers
Default value: **10**
Minimum: 1

selectivitybasedalignment
For score by selectivity-based alignment, this option is true otherwise false.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

selectivityweight
For calculating the Survival score, the weighting factor of the selectivity score. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

sitetoppercentage
In case of score by site-based alignment, keep the hypotheses in the percentage specified by this option. The valid range is 0 to 100

Valid values: integers
Default value: **10**
Minimum: 0
Maximum: 100

siteweight For calculating the Survival score, the weighting factor given for the site score. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_a
The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_d

The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_h

The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1.5**
Minimum: 0.0
Maximum: 100.0

tolerance_n

The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 100.0

tolerance_p

The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 100.0

tolerance_r

The feature matching tolerance for the aromatic ring feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1.5**
Minimum: 0.0
Maximum: 100.0

tolerance_x

The feature matching tolerance for the custom(X) feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_y

The feature matching tolerance for the custom(Y) feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

tolerance_z

The feature matching tolerance for the custom(Z) feature. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

usetolerances

When using feature matching tolerances, this option is true otherwise false.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

vectorlowercutoff

In case of score by vector-based alignment, keep the hypotheses that score above the value specified by this option. The valid range is 0.0 to 1.0

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

vectorweight

For calculating the Survival score, the weighting factor given for the vector score. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

volumebasedalignment

For score by volume-based alignment, this option is true otherwise false.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

volumeweight
For calculating the Survival score, the weighting factor given for the volume score. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

phasescoreinactives

Launches a Score Inactives job.

Syntax:

`phasescoreinactives` *matchscore*=⟨ x ⟩

Options:

matchscore
For calculating the adjusted survival score, the weighting factor given for the inactive match score. The valid range is 0.0 to 100.0

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 100.0

phasescoreselecthypothesis

Selects the given hypothesis.

Syntax:

`phasescoreselecthypothesis` *alignall*=yes | no ⟨ ID ⟩

Options:

alignall This option is used to control the behavior when filling in hypothesis alignments table. If this is "true" then alignments are

generated for all ligands, including inactives, that match the hypothesis on at least 3 sites. If this is "false" then the table is filled in with only the ligands from the active set that match all of the hypothesis sites.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ ID ⟩

The ID of the hypothesis to select. This will populate the Alignment table. This function is single select. If the operand is blank, the alignments table will be filled in for the currently selected hypothesis, if any.

phasesearchformatches

Copies the selected hypotheses from Develop Common Pharmacophore panel (in either the Score Hypothesis or Build QSAR model steps) to the Project Table, opens the Find Matches panel, and sets the first new entry as the chosen hypothesis entry in that panel.

Syntax:

```
phasesearchformatches
```

phaseselectalltablerows

Selects all rows in the first table in the step.

Syntax:

```
phaseselectalltablerows
```

phaseselectconstraintrow

Selects only this row in the constraints table.

Syntax:

`phaseselectconstraintrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the constraint table.

phaseselectevrow

Selects the given row in the excluded volumes table.

Syntax:

`phaseselectevrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the table.

phaseselectexcludedfeature

Excludes the given feature from Find Common Pharmacophores.

Syntax:

`phaseselectexcludedfeature` ⟨ feature ⟩

Operands:
⟨ feature ⟩
The feature to select for excluding.

phaseselectextendconstraintrow

Extends the selection to this row in the constraint table.

Syntax:

`phaseselectextendconstraintrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend the select to.

phaseselectextendevrow

Extends the selection to this row in the excluded volumes table.

Syntax:

`phaseselectextendevrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend the select to.

phaseselectextendhypothesisrow

Extends the selection to this row in the hypothesis table.

Syntax:

`phaseselectextendhypothesisrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend the select to.

phaseselectextendtablerow

Extends the selection to this row in the table.

Syntax:

`phaseselectextendtablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend the select to.

phaseselecthypothesisrow

Selects the given row in the hypothesis table in the step.

Syntax:

`phaseselecthypothesisrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the hypothesis table.

phaseselectonlyevrow

Selects only this row in the excluded volumes table.

Syntax:

`phaseselectonlyevrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select only in the table row.

phaseselectonlyhypothesisrow

Selects only this row in the hypothesis table.

Syntax:

`phaseselectonlyhypothesisrow` ⟨row⟩

Operands:
⟨row⟩
The row number to select only in the hypothesis table.

phaseselectonlytablerow

Selects only this row in the table.

Syntax:

`phaseselectonlytablerow` ⟨row⟩

Operands:
⟨row⟩
The row number to select only in the table row.

phaseselectsite

Selects the given site in the freestyle hypothesis

Syntax:

`phaseselectsite` ⟨site index⟩

Operands:
⟨site index⟩
The index of the site to select.

phaseselecttablerow

Selects the given row in the first table in the step.

Syntax:

`phaseselecttablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the table.

phaseseparatestereoisomers

Separate stereoisomers for the selected conformers.

Syntax:

`phaseseparatestereoisomers`

phaseseparatetitles

Separate the selected conformers by title.

Syntax:

`phaseseparatetitles`

phasesetactiverows

Toggles the Pharm Set property on or off for the given row in the ligands table in Prepare Ligands or Create Sites.

Syntax:

`phasesetactiverows` *value*=active | inactive | none ⟨ rows ⟩

Options:

value	The value of the Pharm set column in the ligands table in Prepare Ligands or Create Sites.	
	Valid values:	active inactive none
	Default value:	**active**

Operands:

⟨ rows ⟩

The row numbers to toggle, two row numbers should be seprated by , or if the user wants to specify a range then it can be given like 1-5 seperated by a - . eg 1,2,5 or 1,4-7 .

phasesetactivity

Sets the activity for the given row in the current step.

Syntax:

`phasesetactivity` *activity*=⟨ x ⟩ ⟨ row ⟩

Options:

activity	The activity value for the given ligand.	
	Valid values:	reals
	Default value:	**1**

Operands:

⟨ row ⟩

The row number of the ligand to set the activity for.

phasesetactivityproperty

Sets the activity property.

Syntax:

`phasesetactivityproperty` ⟨ property ⟩

Operands:

⟨ property ⟩

The property to get the activity values from.

phasesetactivitythresholds

Sets the activity thresholds of the current run based on the current Phase options.

Syntax:

`phasesetactivitythresholds` *active_threshold*=⟨ text ⟩
inactive_threshold=⟨ text ⟩

Options:

active_threshold
: The value used as the cutoff for assigning a ligand to the active Pharm Set. Can be left blank.
Valid values: text strings
Default value:

inactive_threshold
: The value used as the cutoff for assigning a ligand to the inactive Pharm Set. Can be left blank.
Valid values: text strings
Default value:

phasesetalignmentoptions

Sets whether or not to view non-model ligands and whether or not to have a site mask.

Syntax:

`phasesetalignmentoptions` *alignnonmodel*=yes | no
sitemask=⟨ text ⟩ ⟨ hypothesis ⟩

Options:

alignnonmodel
: Indicates whether or not to align non-model ligands.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sitemask
: The site mask is a set of 0s and 1s indicating which sites to include in the mask. A 1 in a position means to include that site, and a 0 means to ignore it.
Valid values: text strings
Default value:

Operands:
⟨ hypothesis ⟩
The name of the hypothesis to set the options for.

phasesetconstrainttolerance

Sets the tolerance for this row in the constraints table.

Syntax:

`phasesetconstrainttolerance` *tolerance*=⟨ x ⟩ ⟨ row ⟩

Options:

tolerance This is the tolerance to set for the given row.

Valid values:	reals
Default value:	**1**
Minimum:	0.0

Operands:
⟨ row ⟩
The row number to set the tolerance for in the constraint table.

phasesetexcludedvolumes

Sets the value for the given cell.

Syntax:

`phasesetexcludedvolumes` *column*=⟨ n ⟩ *row*=⟨ n ⟩ ⟨ value ⟩

Options:

column The column number of the cell to change.

Valid values:	integers
Default value:	**1**

row The row number of the cell to change.

Valid values:	integers
Default value:	**1**

Operands:
⟨ value ⟩
The value for the given cell.

phasesetfrequency

Sets the minimum or maximum frequency for a given feature.

Syntax:

`phasesetfrequency` *column*=⟨ n ⟩ *value*=⟨ n ⟩ ⟨ row ⟩

Options:

column The table column to set.

Valid values:	integers
Default value:	**2**
Minimum:	1

value The minimum or maximum frequency.

Valid values:	integers
Default value:	**0**
Minimum:	0

Operands:
⟨ row ⟩
The row number of the min / max value to set.

phasesettrainingrows

Toggles the Training Set property on or off for the given row in the Alignments table under Build QSAR step.

Syntax:

`phasesettrainingrows` *value*=training | test | none ⟨ rows ⟩

Options:

value The value of the training/test column in the alignment table under Build QSAR Model step.

Valid values:	training test none
Default value:	**training**

Operands:
⟨ rows ⟩
The row numbers to toggle, two row numbers should be seprated by , or if the user wants to specify a range then it can be given like 1-5 seperated by a - . eg 1,2,5 or 1,4-7 .

phaseshowclusters

Shows hypothesis clusters in the Score Hypotheses step.

Syntax:

`phaseshowclusters` *similarity*=⟨ x ⟩

Options:

similarity How similar hypotheses need to be to be clustered.

Valid values:	reals
Default value:	**0.9**

phasesiteoptions

Sets the site options for a Find Pharmacophores job.

Syntax:

`phasesiteoptions` *match*=all | minimum *minimum*=⟨ n ⟩
minsites=⟨ n ⟩ *numsites*=⟨ n ⟩

Options:

match Indicates whether to match against all or a minimum set of active ligands.

Valid values:	all minimum
Default value:	**all**

minimum The minimum number of active ligands which a pattern has to match against.

Valid values:	integers
Default value:	**2**
Minimum:	2

minsites The minimum number of sites to match.

Valid values:	integers
Default value:	**5**
Minimum:	3
Maximum:	7

numsites The maximum number of sites to match.

Valid values:	integers
Default value:	**5**
Minimum:	3
Maximum:	7

phasesorttable

Resort the given Phase table based on the data in the specified column

Syntax:

`phasesorttable` *table*=⟨ n ⟩ ⟨ column_name ⟩

Options:

table The table to set.

Valid values:	integers
Default value:	**10**

Operands:

⟨ column_name ⟩

The name of the column to be sorted.

phasestepforward

Moves forward to the next Phase step. Deletes any steps after the current step, then creates the next step, using the data from previous steps.

Syntax:

`phasestepforward`

phasestepgoto

Moves to an existing step in the current project.

Syntax:

`phasestepgoto` ⟨step name⟩

Operands:

⟨step name⟩

The name of the step to switch to.

phasetoggleactivetablerow

Toggles the Active property on or off for the given row in the first table in the step.

Syntax:

`phasetoggleactivetablerow` ⟨row⟩

Operands:

⟨row⟩

The row number to toggle.

phasetogglecentroidatom

Adds or removes the given atom from the current centroid list for excluded volumes.

Syntax:

`phasetogglecentroidatom` ⟨atom index⟩

Operands:
⟨atom index⟩
The index of the atom to toggle.

phasetoggleconstraintmark

Toggles the mark for this row in the constraints table.

Syntax:

`phasetoggleconstraintmark` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle the mark for in the constraint table.

phasetoggleexcludedfeature

Toggles the given feature for including / excluding from Find Common Pharmacophores.

Syntax:

`phasetoggleexcludedfeature` ⟨feature⟩

Operands:
⟨feature⟩
The feature to toggle for excluding.

phasetogglerequiredmatch

Toggles the Required Match property for the given row.

Syntax:

`phasetogglerequiredmatch` ⟨row⟩

Operands:
⟨row⟩
The row number of the ligand to toggle the Required Match property for.

phasetoggleselectconstraintrow

Toggles the selection of the given row in the constraints table.

Syntax:

`phasetoggleselectconstraintrow` ⟨row⟩

Operands:
⟨row⟩
The row number to toggle in the constraints table.

phasetoggleselectsite

Toggles the selection of the given site in the freestyle hypothesis

Syntax:

`phasetoggleselectsite` ⟨site index⟩

Operands:
⟨site index⟩
The index of the site to toggle.

phasetoggletrainingrow

Toggles the Training Set property on or off for the given row in the Build QSAR step.

Syntax:

`phasetoggletrainingrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to toggle.

phasetranslateexcludedvolumes

Translate in Angstroms the selected excluded volumes.

Syntax:

`phasetranslateexcludedvolumes` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

x Amount in Angstroms to translate in X
Valid values: reals
Default value: **0**

y Amount in Angstroms to translate in Y
Valid values: reals
Default value: **0**

z Amount in Angstroms to translate in Z
Valid values: reals
Default value: **0**

phasetranslatesite

Translate in Angstroms the selected Phase freestyle hypothesis site.

Syntax:

`phasetranslatesite` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

x Amount in Angstroms to translate in X
Valid values: reals
Default value: **0**

y Amount in Angstroms to translate in Y
Valid values: reals
Default value: **0**

z Amount in Angstroms to translate in Z
Valid values: reals
Default value: **0**

phaseundisplayproperty

Undisplays the given property from the table.

Syntax:

`phaseundisplayproperty` ⟨property name⟩

Operands:
⟨property name⟩
The property to undisplay.

phaseungroupactives

Ungroups a ligand group

Syntax:

`phaseungroupactives`

phaseunmarkfeature

Unmarks the given feature in the Workspace.

Syntax:

`phaseunmarkfeature` ⟨feature⟩

Operands:
⟨feature⟩
A single letter (A-Z) indicating the feature to unmark.

phaseunselectevrow

Unselects the given row in the excluded volumes table.

Syntax:

`phaseunselectevrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the table.

phaseunselecthypothesisrow

Unselects the given row in the hypothesis table in the step.

Syntax:

`phaseunselecthypothesisrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the hypothesis table.

phaseunselecttablerow

Unselects the given row in the first table in the step.

Syntax:

`phaseunselecttablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the table.

picksize

This command allows the user to choose the size of the pick box that is used for picking atoms/bonds/residues etc.

Syntax:

`picksize` ⟨size⟩

Operands:

⟨size⟩

This operand actually defines the size of the pick box. Allowed sizes are 7X7, 10X10 and 15X15 which are specified at command line as 7, 10 & 15 respectively.

place

Place the current fragment on screen at the <x>, <y> and <z> positions given by the operands.

Syntax:

`place` ⟨x⟩ ⟨y⟩ ⟨z⟩

Operands:

⟨x⟩ ⟨y⟩ ⟨z⟩

The operands are three real numbers which are the x, y and z coordinates where the new fragment is to be placed.

plotxyarrangecolumn

Puts all of the plots in a column.

Syntax:

`plotxyarrangecolumn`

plotxyarrangerow

Displays the plots in a row.

Syntax:

`plotxyarrangerow`

plotxyarrangetiled

Tiles all of the displayed plots.

Syntax:

`plotxyarrangetiled`

plotxyaspectratiolock

Displays with/without the aspect ratio maintained for the given plot.

Syntax:

`plotxyaspectratiolock` ⟨ plotname ⟩ yes|no

Operands:
⟨ plotname ⟩ yes|no
The name of the plot to operate on. yes to display the plot with aspect ratio locked or no to display it as it is

plotxyaxis

Creates or modifies an XY axis.

Syntax:

`plotxyaxis` *axis*=x | y *maximum*=⟨ text ⟩ *minimum*=⟨ text ⟩
nummarkers=⟨ n ⟩ *plot*=⟨ text ⟩ ⟨ title ⟩

Options:

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

maximum This option sets the maximum value for the axis.
Valid values: text strings

Default value:

minimum This option sets the minimum value for the axis.

Valid values: text strings
Default value:

nummarkers

This option sets the number of scale markers for the axis.

Valid values: integers
Default value: **10**
Minimum: 0

plot This option is the name of the plot containing the axis.

Valid values: text strings
Default value:

Operands:
⟨ title ⟩
The name of the axis to create or modify.

plotxyaxisautorange

Toggles autorange for the given axis on or off.

Syntax:

`plotxyaxisautorange` *autorange*=yes | no *axis*=x | y
plot=⟨ text ⟩ ⟨ axisname ⟩

Options:

autorange This option indicates whether or not to enable autorange for this axis.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

axis This option indicates whether this axis is an X axis or a Y axis.

Valid values: x
y
Default value: **x**

plot This option is the name of the plot containing the axis.

Valid values: text strings
Default value:

Operands:
⟨ axisname ⟩
The name of the axis to set autorange for.

plotxyaxisdelete

Deletes an XY plot axis.

Syntax:

plotxyaxisdelete *axis*=x | y *plot*=⟨ text ⟩ ⟨ axisname ⟩

Options:

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ axisname ⟩
The name of the axis to delete.

plotxyaxisdisplay

Toggles display of the given axis on or off.

Syntax:

plotxyaxisdisplay *autorange*=yes | no *axis*=x | y *plot*=⟨ text ⟩
⟨ axisname ⟩

Options:

autorange This option indicates whether to display or undisplay the given axis.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ axisname ⟩
The name of the axis to display or undisplay.

plotxyaxisrename

Renames an existing axis.

Syntax:

plotxyaxisrename *axis*=x | y *newname*=⟨ text ⟩ *plot*=⟨ text ⟩ ⟨ title ⟩

Options:

axis This option indicates whether this axis is an X axis or a Y axis.
Valid values: x
y
Default value: **x**

newname This option is the new name for the axis.
Valid values: text strings
Default value:

plot This option is the name of the plot containing the axis.
Valid values: text strings
Default value:

Operands:
⟨ title ⟩
The name of the axis to rename.

plotxycaption

Sets the caption for the given plot.

Syntax:

`plotxycaption` ⟨ plotname ⟩ ⟨ caption ⟩

Operands:
⟨ plotname ⟩ ⟨ caption ⟩
The name of the plot to operate on. The new caption for the plot.

plotxycaptionposition

Sets the position of the caption in the given plot.

Syntax:

`plotxycaptionposition` ⟨ plotname ⟩ top|bottom

Operands:
⟨ plotname ⟩ top|bottom
The name of the plot to operate on. top to display the caption at the top of the plot or bottom to display it at the bottom.

plotxycopy

Copies the selected plots.

Syntax:

`plotxycopy`

plotxydelete

Deletes the selected plots.

Syntax:

`plotxydelete`

plotxydeleteall

Deletes all XY plots from the project.

Syntax:

`plotxydeleteall`

plotxydisplay

Displays the selected plots in addition to any currently displayed plots.

Syntax:

`plotxydisplay`

plotxydisplaycaption

Shows or hides the caption for the given plot.

Syntax:

`plotxydisplaycaption` ⟨ plotname ⟩ yes|no

Operands:

⟨ plotname ⟩ yes|no

The name of the plot to operate on. yes to display the caption or no to not display the caption.

plotxydisplayincluded

Sets the Display Included Markers state for the given plot.

Syntax:

`plotxydisplayincluded` ⟨ plotname ⟩ yes|no

Operands:

⟨ plotname ⟩ yes|no

The name of the plot to operate on. yes to display included markers or no to not display them.

plotxydisplaylegend

Displays or hides the legend for the given plot.

Syntax:

`plotxydisplaylegend` ⟨ plotname ⟩ yes|no [legend_psotion]

Operands:

⟨ plotname ⟩ yes|no [legend_psotion]

The name of the plot to operate on. yes to display the legend or no to not display it. Positions are upper_right, upper_left, lower_left, lower_right, center_left, center_right, lower_center, upper_center.

plotxydisplayname

Shows or hides the name in the caption for the given plot.

Syntax:

`plotxydisplayname` ⟨ plotname ⟩ yes|no

Operands:

⟨ plotname ⟩ yes|no

The name of the plot to operate on. yes to display the plot name in the caption or no to not display the name in the caption.

plotxydisplayonly

Displays only the selected plots.

Syntax:

`plotxydisplayonly`

plotxydisplaypointlabels

Displays or hides the point labels for the given plot.

Syntax:

`plotxydisplaypointlabels` ⟨ plotname ⟩ yes|no

Operands:

⟨ plotname ⟩ yes|no

The name of the plot to operate on. yes to display the point labels or no to not display them.

plotxydisplayselected

Displays or hides the selected entry markers for the given plot.

Syntax:

`plotxydisplayselected` ⟨ plotname ⟩ yes|no

Operands:

⟨ plotname ⟩ yes|no

The name of the plot to operate on. yes to display the selected entry markers or no to not display them.

plotxyequalizeaxisrange

Displays with/without the axis ranges equalized for the given plot.

Syntax:

`plotxyequalizeaxisrange` ⟨ plotname ⟩ yes|no

Operands:

⟨ plotname ⟩ yes|no

The name of the plot to operate on. yes to display the plot with the axis range equalized to be the same.

plotxyhidesidebar

Hides the plotxy side bar.

Syntax:

`plotxyhidesidebar`

plotxyhidetoolbar

Hides the plotxy tool bar.

Syntax:

`plotxyhidetoolbar`

plotxylabel

Turns on labels for the given data point.

Syntax:

`plotxylabel` *entryname*=yes | no *plot*=⟨ text ⟩ *series*=⟨ text ⟩
title=yes | no *xaxis*=yes | no *yaxis*=yes | no ⟨ entry ⟩

Options:

entryname

If this is set to true then the entry name is displayed as part of the label.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

plot This option is the name of the plot containing the series and entry.
Valid values: text strings
Default value:

series This option is the name of the series containing the entry.
Valid values: text strings
Default value:

title If this is set to true then the entry's title is displayed as part of the label.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

xaxis If this is set to true then the X-axis property is displayed as part of the label.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

yaxis If this is set to true then the Y-axis property is displayed as part of the label.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ entry ⟩
The name of the entry to label.

plotxymove

Moves the selected plots.

Syntax:

`plotxymove` ⟨ location ⟩

Operands:
⟨ location ⟩
The location to move the selected plots to. If the location is 0, then the plots are moved to the beginning of the plot list. Otherwise, the plots are moved after the given plot number in the list.

plotxynew

Creates a new XY plot

Syntax:

`plotxynew` ⟨ plotname ⟩

Operands:
⟨ plotname ⟩
The name of the plot to create.

plotxypan

Moves the selected plots.

Syntax:

`plotxypan` ⟨ x ⟩ ⟨ y ⟩

Operands:
⟨ x ⟩ ⟨ y ⟩
This is a percentage of the full range to offset by. This is a percentage of the full range to offset by.

plotxypanplot

Moves the specified plot.

Syntax:

`plotxypanplot` ⟨ plot ⟩ ⟨ x ⟩ ⟨ y ⟩

Operands:
⟨ plot ⟩ ⟨ x ⟩ ⟨ y ⟩
This is the name of the plot to pan. This is a percentage of the full range to offset by. This is a percentage of the full range to offset by.

plotxyrename

Renames the given plot to the new name.

Syntax:

`plotxyrename` ⟨ plotname ⟩ ⟨ newname ⟩

Operands:
⟨ plotname ⟩ ⟨ newname ⟩
The name of the plot to operate on. The new name for the plot.

plotxyresetview

Resets the zoom and pan for the selected plots.

Syntax:

`plotxyresetview`

plotxysaveimage

Capture the current XY plotting window and save it to an image file.

Syntax:

`plotxysaveimage` *format*=tiff | jpeg ⟨ file_name ⟩

Options:

format	Specifies the format of the saved image.	
	Valid values:	tiff jpeg
	Default value:	**tiff**

Operands:
⟨ file_name ⟩
The file where the image will be saved.

plotxyselect

Selects the given plots.

Syntax:

`plotxyselect` ⟨ plotname ⟩

Operands:

⟨ plotname ⟩

Names of the plots to select, or all, or displayed.

plotxyseries

Creates or modifies an XY data series.

Syntax:

`plotxyseries` *color*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *color_by_property*=⟨ text ⟩ *color_map*=autumn | blues | grays | greens | oranges | purples | rainbow | red_blue | red | spring | summer | white_black | winter | yellow_green_blue *line_color*=black | red | green | blue | purple | orange | blue_green | light_green | red_purple | yellow | cyan *marker*=filled_square | square | filled_circle | circle | cross | point | diamond | filled_diamond | none *marker_size*=⟨ n ⟩ *plot*=⟨ text ⟩ *rsquared*=⟨ x ⟩ *show_color_bar*=yes | no *size_by_property*=⟨ text ⟩ *style*=none | solid | dash *unityaspect*=yes | no *width*=⟨ n ⟩ *xaxis*=⟨ text ⟩ *xbuckets*=⟨ n ⟩ *xproperty*=⟨ text ⟩ *yaxis*=⟨ text ⟩ *ybuckets*=⟨ n ⟩ *yproperty*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

color The color of the symbols and lines used on the plot for this data series.

Valid values: black
red
green
blue
purple
orange
blue_green
light_green
red_purple
yellow
cyan
Default value: **black**

color_by_property
This option sets the name of the property the points are to be colored by.

Valid values: text strings
Default value:

color_map The color maps to be used when coloring by properties.

Valid values: autumn
blues
grays
greens
oranges
purples
rainbow
red_blue
red
spring
summer
white_black
winter
yellow_green_blue
Default value: **rainbow**

line_color The color for lines to be used on the plot.

Valid values: black
red
green
blue
purple
orange
blue_green
light_green
red_purple
yellow
cyan

Default value: **black**

marker The marker for this data series.

Valid values: filled_square
square
filled_circle
circle
cross
point
diamond
filled_diamond
none

Default value: **filled_circle**

marker_size

The marker size for this data series.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 8

plot This option is the name of the plot containing the series.

Valid values: text strings
Default value:

rsquared This option sets the correlation coefficient or rsquared value for this data series. This option is for Strike BuildQSAR plots.

Valid values: reals
Default value: **0**

show_color_bar

Settings this option to yes will display a color bar when coloring by a property.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

size_by_property
This option sets the name of the property to be used to size the markers.

Valid values: text strings
Default value:

style The line style for this data series.

Valid values: none
solid
dash
Default value: **none**

unityaspect
Settings this option to yes causes the plot to display with an equal increment on each axis mapping to the same distance on the screen.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

width This option sets the width of the line for this data series.

Valid values: integers
Default value: **1**
Minimum: 1

xaxis This option selects an existing X-axis.

Valid values: text strings
Default value:

xbuckets This option sets the number of buckets for the X-axis.

Valid values: integers
Default value: **100**
Minimum: 0

xproperty This option sets the name of the property for the X-axis.

Valid values: text strings
Default value:

yaxis This option selects an existing Y-axis.

Valid values: text strings
Default value:

ybuckets This option sets the number of buckets for the Y-axis.

Valid values: integers
Default value: **100**
Minimum: 0

yproperty This option sets the name of the property for the Y-axis.

Valid values: text strings
Default value:

Operands:
⟨ seriesname ⟩
The name of the series to create or modify.

plotxyseriesdelete

Deletes the given XY data series.

Syntax:

`plotxyseriesdelete` *plot*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨ seriesname ⟩
The name of the series to delete.

plotxyseriesdisplay

Displays or undisplays the given series.

Syntax:

`plotxyseriesdisplay` ⟨ plotname ⟩ ⟨ series ⟩ yes|no

Operands:
⟨ plotname ⟩ ⟨ series ⟩ yes|no
The name of the plot to operate on. The name of the series. yes to display the given data series or no to not display it.

plotxyseriesdisplay45

Shows or hides a 45-degree reference line for the given series.

Syntax:

plotxyseriesdisplay45 ⟨ plotname ⟩ ⟨ series ⟩ yes|no [slope]

Operands:

⟨ plotname ⟩ ⟨ series ⟩ yes|no [slope]

The name of the plot to operate on, the series to operate on, and a yes or no to display or hide the 45-degree reference line. The slope is optional but should be 1 or -1.

plotxyseriesdisplayregression

Shows or hides the regression line for the given series.

Syntax:

plotxyseriesdisplayregression ⟨ plotname ⟩ ⟨ series ⟩ yes|no

Operands:

⟨ plotname ⟩ ⟨ series ⟩ yes|no

The name of the plot to operate on, the series to operate on, and a yes or no to display or hide the regression line.

plotxyseriesrename

Renames an existing series.

Syntax:

plotxyseriesrename *newname*=⟨ text ⟩ *plot*=⟨ text ⟩ ⟨ title ⟩

Options:

newname	This option is the new name for the series. Valid values: text strings Default value:
plot	This option is the name of the plot containing the series.

Valid values: text strings
Default value:

Operands:
⟨ title ⟩
The name of the series to rename.

plotxyseriesselect

Selects only the entries corresponding to the given series.

Syntax:

`plotxyseriesselect` *plot*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨ seriesname ⟩
The name of the series to select.

plotxyseriesselectadd

Adds the entries corresponding to the given series to the current selection.

Syntax:

`plotxyseriesselectadd` *plot*=⟨ text ⟩ ⟨ seriesname ⟩

Options:

plot This option is the name of the plot containing the series.
Valid values: text strings
Default value:

Operands:
⟨ seriesname ⟩
The name of the series to select.

plotxyseriessetaxesequal

Sets the ranges of the X and Y axes to be equal.

Syntax:

`plotxyseriessetaxesequal` ⟨ plotname ⟩ ⟨ series ⟩

Operands:
⟨ plotname ⟩ ⟨ series ⟩
The name of the plot to operate on. The name of the series.

plotxyshowsidebar

Displays the plotxy side bar.

Syntax:

`plotxyshowsidebar`

plotxyshowtoolbar

Displays the plotxy tool bar.

Syntax:

`plotxyshowtoolbar`

plotxytoggledisplay

Toggles the display of the plot on or off.

Syntax:

`plotxytoggledisplay` ⟨ plotname ⟩

Operands:
⟨ plotname ⟩
The name of the plot to toggle the display of.

plotxyundisplay

Undisplays the selected plots.

Syntax:

`plotxyundisplay`

plotxyunlabel

Turns off the label for the given data point.

Syntax:

`plotxyunlabel` *plot*=⟨ text ⟩ *series*=⟨ text ⟩ ⟨ entry ⟩

Options:

plot This option is the name of the plot containing the series and entry.
Valid values: text strings
Default value:

series This option is the name of the series containing the entry.
Valid values: text strings
Default value:

Operands:
⟨ entry ⟩
The name of the entry to unlabel.

plotxyunselect

Unselects the given plots.

Syntax:

`plotxyunselect` ⟨ plotname ⟩

Operands:
⟨ plotname ⟩
Names of the plots to unselect, or all, or displayed.

plotxyupdate

Updates the selected plots.

Syntax:

`plotxyupdate`

plotxyzoom

Scales the selected plots.

Syntax:

`plotxyzoom` ⟨ x ⟩ ⟨ y ⟩

Operands:
⟨ x ⟩ ⟨ y ⟩
This is the amount of scaling to use in the horizontal direction. This is the amount of scaling to use in the vertical direction.

plotxyzoompan

Scales and offsets the given plot.

Syntax:

`plotxyzoompan` ⟨ plot ⟩ ⟨ zoomx ⟩ ⟨ zoomy ⟩ ⟨ panx ⟩ ⟨ pany ⟩

Operands:
⟨ plot ⟩ ⟨ zoomx ⟩ ⟨ zoomy ⟩ ⟨ panx ⟩ ⟨ pany ⟩
This is the name of the plot to apply the zoom and pan factors to. This is the amount of scaling to use in the horizontal direction. This is the amount of scaling to use in the vertical direction. This is the amount to offset in the horizontal direction as a percentage of the full range. This is the amount to offset in the vertical direction as a percentage of the full range.

potential

Set various options associated with the definition of the potential energy to be used in a MacroModel job.

Syntax:

`potential` *cele*=⟨ x ⟩ *charges*=force_field | structure_file *chnd*=⟨ x ⟩ *cutoff*=normal | extended | user_defined | none *cvdw*=⟨ x ⟩ *debug*=⟨ text ⟩ *dielectric*=⟨ x ⟩ *electrostatics*=field_field | constant | distance_dependant *field*=mm2* | mm3* | amber* | opls* | amber94 | mmff | mmffs | oplsaa | opls2005 | opls2008 *mutualinteractions*=yes | no *solvent*=none | water | chcl3 | octanol *substructure*=yes | no

Options:

cele This option determines what cutoff will be used for the electrostatic part of the energy calculation.

Valid values:	reals
Default value:	**12**
Minimum:	0.0
Maximum:	99999.0

charges This option determines where the charges to be used in the energy calculation will come from.

Valid values:	force_field structure_file
Default value:	**force_field**

chnd This option determines what cutoff will be used for the hydrogen bond part of the energy calculation.

Valid values:	reals
Default value:	**4**

Minimum: 0.0
Maximum: 99999.0

cutoff This option determines what type of non-bonded cutoff will be used in the energy calculation.

Valid values: normal
extended
user_defined
none
Default value: **normal**

cvdw This option determines what VDW cutoff will be used in the energy calculation.

Valid values: reals
Default value: **7**
Minimum: 0.0
Maximum: 99999.0

debug Debug opcodes

Valid values: text strings
Default value:

dielectric The dielectric constant to be used in the electrostatic part of the energy calculation.

Valid values: reals
Default value: **1**
Minimum: 0.9999999999

electrostatics
The electrostatic treatment to be used in the energy calculation.

Valid values: field_field
constant
distance_dependant
Default value: **constant**

field The force field to be used for the energy calculation.

Valid values: mm2*
mm3*
amber*
opls*
amber94
mmff
mmffs
oplsaa
opls2005
opls2008
Default value: **opls2005**

mutualinteractions
Controls whether or not to use constrained-atom mutual interactions

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

solvent The solvent model to be used for the energy calculation

Valid values: none
water
chcl3
octanol
Default value: **water**

substructure
[NOTE: This option is no longer used.]

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pprep

This keyword is used to set various options associated with running protein preparation.

Syntax:

`pprep` *cavity*=default | liaison | no_neutralization
fix_receptor=yes | no *procedure*=both | prepare | refine
rmsd=⟨ x ⟩

Options:

cavity Which preparation procedure to run. Both preparation and refinement by default.

Valid values: default
liaison
no_neutralization
Default value: **no_neutralization**

fix_receptor

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

procedure Which preparation procedure to run. Both preparation and refinement by default.

	Valid values:	both prepare refine
	Default value:	**both**
rmsd	Minimum RMSD.	
	Valid values:	reals
	Default value:	**0.3**
	Minimum:	0.0

pprepreceptorligand

Defines a on-screen molecule or entry to be treated as the ligand for a Pprep calculation

Syntax:

`pprepreceptorligand` ⟨ molecule_number | entry_id ⟩

Operands:

⟨ molecule_number | entry_id ⟩

The molecule number or entry ID to be included as the ligand.

pprepwrite

Write the files required for protein structure preparation.

Syntax:

`pprepwrite`

prefer

Set the global preferences for handling markers (derived graphical objects). Specifically, specify under which conditions markers will be deleted.

Syntax:

prefer *2dbondlinewidth*=⟨ n ⟩ *2dbondspacing*=⟨ n ⟩
2delementlabelmargin=⟨ x ⟩ *2dfontsize*=⟨ x ⟩
2dhashspacing=⟨ n ⟩ *2dlabelallcarbons*=yes | no
2dmaxatoms=⟨ n ⟩ *2dmaxscalefactor*=⟨ x ⟩ *2dscalefactor*=⟨ x ⟩
2dshowimplicithydrogens=yes | no *2dusecolor*=yes | no
adjustmentsfontbold=yes | no *adjustmentsfontitalics*=yes | no
adjustmentsfontname=⟨ text ⟩ *adjustmentsfontsize*=⟨ x ⟩
annotationstoolbarshow=yes | no *autofit*=never | singleentry |
always *avoidhardlinks*=never | afs | always | auto
beep=yes | no *buildbackbonesubjobs*=⟨ n ⟩
buildtoolbarshow=yes | no *buildtoolbarstyle*=icononly |
textonly | textbeside | textunder *changedatanames*=yes | no
checkproteinprepared=yes | no *cleanupusingmmap*=yes | no
clipdistance=⟨ x ⟩ *clipincrement*=⟨ x ⟩ *cmdhistory*=unlimited |
limited *cmdhistorylimit*=⟨ n ⟩ *cmdlogfilename*=⟨ text ⟩
collapsed=⟨ n ⟩ *colorhypothesesbyentry*=yes | no
commandcompletion=yes | no *commandinputshow*=yes | no
deletemarkers=mismatch | missing *displayangleprecision*=⟨ n ⟩
displayatomstoolbarshow=yes | no
displayatomstoolbarstyle=icononly | textonly | textbeside |
textunder *displaydihedralprecision*=⟨ n ⟩
displaydistanceprecision=⟨ n ⟩ *displayeditwarning*=yes | no
displayprecision=⟨ n ⟩ *displaywithinincludesnonpolarh*=yes | no
dockingpanels=yes | no *docklocation*=mainwindow | floating
drawmeasurementborder=yes | no *edittoolbarshow*=yes | no
edittoolbarstyle=icononly | textonly | textbeside | textunder
enablejobdebugoutput=yes | no *entryedittoolbarstyle*=icononly |
textonly | textbeside | textunder *entryfeedbackshow*=yes | no
entryfeedbackshownames=yes | no *entrytoolbarstyle*=icononly |
textonly | textbeside | textunder *eplayertoolbarshow*=yes | no
eplayertoolbarstyle=icononly | textonly | textbeside | textunder
feedbackproperties=⟨ text ⟩ *feedbackproperty*=⟨ text ⟩
feedbackshow=yes | no *filerefreshinterval*=⟨ n ⟩
findtoolbarshow=yes | no *fitenhance*=yes | no
fitenhancedensity=⟨ x ⟩ *fitenhancefar*=⟨ x ⟩ *fitenhancenear*=⟨ x ⟩
fitgrow=yes | no *fitoffset*=⟨ x ⟩ *fluorine*=yes | no
fragmentstoolbarshow=yes | no *fragmentstoolbarstyle*=icononly
| textonly | textbeside | textunder
geometrycleanupmethod=builtin | uff
graphicsmemorycachelimit=⟨ n ⟩
hidewstransformationcmds=yes | no *improvelighting*=yes | no
incorporatejobs=manual | prompt | auto *initworkdir*=⟨ text ⟩
jobnamefilter=⟨ text ⟩ *jobnameretain*=yes | no
jobpanelupdateinterval=⟨ n ⟩ *jobstatusfeedbackshow*=yes | no
jobstatusupdateinterval=⟨ n ⟩ *jobstealth*=yes | no
kbrotationincrement=⟨ n ⟩ *kbtransincrement*=⟨ n ⟩
keepcombiglidejobfiles=yes | no
keepcontrolmultiejobfiles=yes | no
keepphasejobfiles=yes | no *keepprimejobfiles*=yes | no

Options:

2dbondlinewidth

The width for the lines used to draw the bonds.

Valid values:	integers
Default value:	**1**
Minimum:	1
Maximum:	20

2dbondspacing

The spacing between multiple bonds in the 2D structure drawing.

Valid values:	integers
Default value:	**2**
Minimum:	1
Maximum:	20

2delementlabelmargin

The separation between the label and structure. Values less than 1.0 will decrease the space between the label and the structures, values greater than 1.0 will increase it. Note there may be clipping effects for values very different from 1.0.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	2.0

2dfontsize The font size used when drawing the 2D structure

Valid values:	reals
Default value:	**30**
Minimum:	10.0
Maximum:	100.0

2dhashspacing

The spacing used for hashed lines in the 2D structures

Valid values:	integers
Default value:	**40**
Minimum:	1
Maximum:	100

2dlabelallcarbons

Whether all carbons are labeled when drawing the 2D structures.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

2dmaxatoms

This option is used as the maximum number of atoms we will ever attempt to draw a 2D structure for.

Valid values: integers
Default value: **200**
Minimum: 10
Maximum: 5000

2dmaxscalefactor
A scaling factor representing the maximum amount small structures will be scaled up.

Valid values: reals
Default value: **0.15**
Minimum: 0.0
Maximum: 500.0

2dscalefactor
A scaling factor used when drawing the 2D structure

Valid values: reals
Default value: **10**
Minimum: 1.0
Maximum: 500.0

2dshowimplicithydrogens
Whether all hydrogens are shown during the structure drawing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

2dusecolor Whether to use color when peforming the 2D structure drawing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

adjustmentsfontbold
A bool which determines whether bold font is to be used for adjustments markers

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

adjustmentsfontitalics
A bool which determines whether italics font is to be used for adjustments markers

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

adjustmentsfontname
A string which determines the font to be used for adjustments markers

Valid values: text strings
Default value:

adjustmentsfontsize
A double which determines the font size to be used for adjustments markers

Valid values: reals
Default value: **14**
Minimum: 0.0

annotationstoolbarshow
Whether or not to show the Annotations toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

autofit Specifies whether workspace entries fit to Workspace automatically or not. Valid values are "never", "singleentry", and "always". These cause Maestro never fit to Workspace, fit when only one entry in workspace and always fit to Workspace respectively.

Valid values: never
singleentry
always
Default value: **singleentry**

avoidhardlinks
Specifies conditions for which files should be copied rather than hard-linked. Valid values are "never", "afs", "auto", and "always". For "afs", copy the file if the original file appears to be on an AFS mount (having /afs/ in the file path), since hard links are not permitted between files in different directories under AFS ("afs" setting is deprecated, use "auto"). For "auto", try to create a link, if link fails then try to copy the file.

Valid values: never
afs
always
auto
Default value: **auto**

beep Whether or not the system beeps for picking feedback.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

buildbackbonesubjobs
This is the number of templates to run at one time in Build Backbone.

Valid values: integers
Default value: **1**
Minimum: 1

buildtoolbarshow

Whether or not to show the Build toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

buildtoolbarstyle

Deterimines the appearance of the Build toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **icononly**

changedatanames

If set to true, then when a non-standard property is renamed, change the internal data name for the property to match. Otherwise, the data name is not changed when the external (user-visible) property name is changed. A non-standard property is one that entries are not required to have, is added to project only after project is initialized and can be removed from the project.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

checkproteinprepared

Whether or not protein structure, to be used for job input, should be checked for having been prepared using Protein Preparation Wizard.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

cleanupusingmmap

On value uses memory mapped IO (mmap) when running project_cleanup. Off uses traditional read system calls. Performance can vary depending on your file system and on your nfs configuration (if you are using an nfs-mounted disk). Using mmap should give better performance on traditional file systems and where the maestro-project is stored on file systems conforming to POSIX.1b. In some cases such as GPFS you may get better performance by turning mmap off.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

clipdistance

Adjust clipping planes to within the distance when focusing on a substructure.

Valid values: reals
Default value: **4**
Minimum: 0.0

clipincrement
Move the clipping planes by this percentage (of slab width) when moving forward, back, together or apart.

Valid values: reals
Default value: **3**
Minimum: 0.0

cmdhistory
Specifies whether to store limited or unlimited commands in command history. The default option is 'unlimited' and if user specifies 'limited' then 'cmdhistorylimit' option will be used.

Valid values: unlimited
limited
Default value: **limited**

cmdhistorylimit
Specifies number of commands to be stored in command history. This option will be used when 'cmdhistory=limited'.

Valid values: integers
Default value: **500**
Minimum: 0

cmdlogfilename
If MM_OPTION_PREF_WRITE_CMDLOG_ON_EXIT is true, then commands will be written to this file.

Valid values: text strings
Default value: **maestrolog.cmd**

collapsed Whether the status bar is collapsed or not.

Valid values: integers
Default value: **0**

colorhypothesesbyentry
If this option is set to true, then Maestro will color entry-based hypotheses by the entry coloring scheme.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

commandcompletion
If this option is set to true then command completion will be performed in the command input area

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

commandinputshow

Whether the command input area is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

deletemarkers

This option sets the condition for marker deletion. Valid values are "mismatch" or "missing".

Valid values: mismatch
missing
Default value: **mismatch**

displayangleprecision

Default display precision for angle measurement

Valid values: integers
Default value: **1**
Minimum: 1
Maximum: 6

displayatomstoolbarshow

Whether or not to show the Display Atoms toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayatomstoolbarstyle

Deterimines the appearance of the Display Atoms toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **icononly**

displaydihedralprecision

Default display precision for dihedral measurement

Valid values: integers
Default value: **1**
Minimum: 1
Maximum: 6

displaydistanceprecision

Default display precision for distance measurement

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 6

displayeditwarning

If set to true, then a warning message is displayed when user edits value of an entry property in PT.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displayprecision

Default display precision for PT real type columns

Valid values: integers
Default value: **3**
Minimum: 0
Maximum: 15

displaywithinincludesnonpolarh

A bool which determines whether workspace sequence viewer has to be aligned by residue number or not

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dockingpanels

If this option is set to true then most non-application panels will be dockable in the main window.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

docklocation

If set to 0 (mainwindow), dockable panels will be docked within the main window. If it is set to 1 (floating), they will be docked into a separate top-level window.

Valid values: mainwindow
floating
Default value: **mainwindow**

drawmeasurementborder

If this flag is on, a rectangle will be drawn in the background color around measurement text. If this is off, then the text will be drawn with no border.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

edittoolbarshow

Whether or not to show the Edit toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

edittoolbarstyle

Deterimines the appearance of the Edit toolbar

Valid values: iconlyonly
textonly
textbeside
textunder
Default value: **textunder**

enablejobdebugoutput
This option if enabled would print job control diagnostic output to the job log file.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

entryedittoolbarstyle
Deterimines the appearance of the PT Entry Edit toolbar

Valid values: iconlyonly
textonly
textbeside
textunder
Default value: **textunder**

entryfeedbackshow
Whether the workspace entry feedback is activated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entryfeedbackshownames
Whether the workspace entry feedback includes the name of the property along with its value.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entrytoolbarstyle
Deterimines the appearance of the PT Entry toolbar

Valid values: iconlyonly
textonly
textbeside
textunder
Default value: **textunder**

eplayertoolbarshow
Whether or not to show the EPlayer toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

eplayertoolbarstyle
Deterimines the appearance of the ePlayer toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **icononly**

feedbackproperties
These are the entry properties displayed for a single entry in workspace. The property names should be m2io data names with no spaces, separated by spaces in the option string.

Valid values: text strings
Default value: **s_m_title s_pdb_PDB_ID**

feedbackproperty
This is the entry property displayed at the end of the atom feedback string in workspace.

Valid values: text strings
Default value: **Title**

feedbackshow
Whether the workspace feedback is activated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

filerefreshinterval
Refresh interval (in secs) for file dialogs

Valid values: integers
Default value: **10**
Minimum: 0

findtoolbarshow
Whether or not to show the Find toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

fitenhance If set to true, automatically enhance depth cues when we fit to Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fitenhancedensity
The extra density added to fog density for enhance depth cues when we fit to Workspace.

Valid values: reals
Default value: **0.5**
Minimum: 0.0

fitenhancefar
: The far position of structures for enhance depth cues when we fit to Workspace.

  Valid values: reals
  Default value: **1**

fitenhancenear
: The near position of structures for enhance depth cues when we fit to Workspace.

  Valid values: reals
  Default value: **-1**

fitgrow
: If this option is set to true, then Maestro will fit to Workspace after every grow operation in the builder.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

fitoffset
: Offset used when fitting to structures to screen. A non-zero offset will provide more of a buffer around the structure. This value is in Angstroms.

  Valid values: reals
  Default value: **2**
  Minimum: 0.0

fluorine
: This option determines whether Treat organofluorines as H-Bond acceptors.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

fragmentstoolbarshow
: Whether or not to show the Fragments toolbar.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

fragmentstoolbarstyle
: Deterimines the appearance of the Fragments toolbar

  Valid values: icononly
  textonly
  textbeside
  textunder
  Default value: **icononly**

geometrycleanupmethod
: Deterimines how geometry cleanup is performed - using built-in or UFF method.

  Valid values: builtin
  uff

Default value: **builtin**

graphicsmemorycachelimit

Specify number of atoms for which graphics library should cache the memory. If entries are included and later excluded, then graphics library would not release memory until this threshold is reached.

Valid values: integers
Default value: **50000**

hidewstransformationcmds

To hide or show translate, rotate, zoom and tranformation commands in command history

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

improvelighting

If this flag is on, clipping planes settings and color palette will be set closer to PyMOL defaults.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

incorporatejobs

Specifies the conditions under which completed jobs that are incorporatable in the current project are incorporated. This setting does not affect the automatic incorporation of monitored jobs in Maestro. It affects the incorporation of jobs that are found to be incorporatable (incorporatable jobs) when the status of all jobs listed in the monitoring panel is updated. Valid values for this setting are "manual", "prompt", and "auto". For the "manual" setting, incorporatable jobs are incorporated only if they are monitored. For the "prompt" setting, incorporatable jobs are incorporated if user approval is given. If approval is not given to incorporate these jobs, then they can only be incorporated in the current Maestro session if they are monitored. Otherwise, they can be incorporated from a different Maestro session. For the "auto" setting, all incorporatable jobs are incorporated automatically, except for those that the user chose not to incorporate under the "prompt" setting

Valid values: manual
prompt
auto
Default value: **prompt**

initworkdir

The Maestro input/output directory is used for starting jobs and for other file input and output. The i/o directory is always

displayed in the title bar of the Main Application Window. The i/o dir can be set to a number of values which affect Maestro's behavior when reading and writing files and when running jobs. You can specify "currentdir" ("startdir" is a synonym) to set the i/o directory to use Maestro's current working directory (cwd), "project" to change to the project directory, "projectparent" to change to the parent directory which contains the project directory, or "projectjobs" to change to the "jobs" directory within the project directory. Otherwise, specify a directory path to be used. Maestro has a current working directory (cwd), similar to what a Unix shell has. When "currentdir" is in effect job files will be placed in the cwd. This is located at the very top of the window next to the window's border. The cwd is changed whenever a cd (changedirectory) command

Valid values: text strings
Default value: **currentdir**

jobnamefilter

The filter for identifying Maestro job names.

Valid values: text strings
Default value: ***.inp *.in *.com**

jobnameretain

This option is used to record if last used job name should be the default job name for the next job.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

jobpanelupdateinterval

The time interval, in seconds, of how often status gets updated for unfinished jobs listed in the Monitor panel.

Valid values: integers
Default value: **30**

jobstatusfeedbackshow

Whether the workspace job status feedback is activated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

jobstatusupdateinterval

The time interval of how often job status gets updated in the Workspace status bar.

Valid values: integers
Default value: **30**

jobstealth Whether or not job monitoring and incorporation is expected to proceed with minimal disruption to the Workspace and entry selection.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

kbrotationincrement

Uses this value when performing workspace rotation with keyborad.

Valid values: integers
Default value: **5**

kbtransincrement

Uses this value when performing workspace translation with keyborad.

Valid values: integers
Default value: **5**

keepcombiglidejobfiles

If this is set to true, then CombiGlide job files will be preserved after a job completes. Otherwise, the job files will be removed when each wizard-based job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

keepentrydisassociateproperties

If set to true, then copy properties to new entries created by entrydisassociate commands. Normally, individual molecules or chains would not have the same properties as the original structure, but this provides the option to preserve the original structure properties in the new entries.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

keepphasejobfiles

If this is set to true, then Phase job files will be preserved after a job completes. Otherwise, the job files will be removed when each wizard-based job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

keepprimejobfiles

If this is set to true, then Prime job files will be preserved after a job completes. Otherwise, the job files will be removed when each wizard-based job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

labelstoolbarstyle

Deterimines the appearance of the Labels toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

lastphasedb
This records the last Phase 3D Database which was opened.
Valid values: text strings
Default value:

lastphasematchfile
This records the last match file which was created from Phase.
Valid values: text strings
Default value:

lastproject This records the last project which was opened.
Valid values: text strings
Default value:

limitanimationdistance
Whether or not view animation should be suppressed when the distance traversed would be too large.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

macromodelsuffix
The default suffix or extension to be used for exported MacroModel files. The suffix will automatically be appended to a file name if it has no suffix.
Valid values: text strings
Default value: **dat**

maestrocompressedsuffix
The default suffix or extension to be used for exported compressed Maestro files. The suffix will automatically be appended to a file name if it has no suffix.
Valid values: text strings
Default value: **maegz**

maestrosuffix
The default suffix or extension to be used for exported Maestro files. The suffix will automatically be appended to a file name if it has no suffix.
Valid values: text strings
Default value: **mae**

mainmetatoolbarshow
Whether or not to show the Meta toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

mainprojecttoolbarshow
Whether or not to show the Project toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

maxnumcons
Default maximum number of constraints allowed

Valid values: integers
Default value: **5000**
Minimum: 1
Maximum: 20000

measurementsfontbold
A bool which determines whether bold font is to be used for measurements markers

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

measurementsfontitalics
An integer which determines whether italics font is to be used for measurements markers

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

measurementsfontname
A string which determines the font to be used for measurements markers

Valid values: text strings
Default value:

measurementsfontsize
A double which determines the font size to be used for measurements markers

Valid values: reals
Default value: **14**
Minimum: 0.0

mol2suffix The default suffix or extension to be used for exported Mol2 files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **mol2**

monitorjobs

Whether or not jobs launched from maestro are automatically monitored.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

mouseactionset

Sets the current mouse action set.

Valid values: 3buttonsandscrollwheel
3buttonsonly
2buttonsandscrollwheel
2buttonsonly
pymolmode
trackpad
Default value: **3buttonsandscrollwheel**

mruprojectlistlimit

This is the maximum number of most recently used project to be shown in the Project menu.

Valid values: integers
Default value: **5**
Minimum: 0
Maximum: 10

nonlinearanimation

Whether or not nonlinear animation is used for slow-in action near the starting pose and slow-out action near the ending pose.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

nonlineardegree

The degree of a NURBS curve which is used to control the nonlinear animation. matching tolerance for the custom(Z) feature. The valid range is 0.0 to 100.0

Valid values: integers
Default value: **3**
Minimum: 2
Maximum: 4

nonlineareffect

The factor that controls one of slow-in and slow-out effects by changing the shape of a NURBS curve, which is used to control the nonlinear animation.

Valid values: reals
Default value: **10**
Minimum: 0.0
Maximum: 50.0

obfuscatepaths
: If this flag is set, then the postmortem utility will obfuscate paths automatically.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

openlastproject
: If set to true, then when Maestro starts it will re-open the last project that was open when Maestro was closed.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

pdbsuffix The default suffix or extension to be used for exported PDB files. The suffix will automatically be appended to a file name if it has no suffix.

  Valid values: text strings
  Default value: **pdb**

phasedefaultfeaturedefinitions
: This allows the user to override the default pharma_feature.ini file in Phase application panels.

  Valid values: text strings
  Default value:

phasefeedbackproperties
: These are the Phase properties displayed for a single Phase ligand in the workspace. The property names should be m2io data names with no spaces, separated by spaces in the option string.

  Valid values: text strings
  Default value:

phasetolerance_a
: The feature matching tolerance for the hydrogen bond acceptor feature. The valid range is 0.0 to 100.0

  Valid values: reals
  Default value: **1**
  Minimum: 0.0
  Maximum: 100.0

phasetolerance_d
: The feature matching tolerance for the hydrogen bond donor feature. The valid range is 0.0 to 100.0

  Valid values: reals
  Default value: **1**
  Minimum: 0.0
  Maximum: 100.0

phasetolerance_h
: The feature matching tolerance for the hydrophobic feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1.5**
Minimum:	0.0
Maximum:	100.0

phasetolerance_n
: The feature matching tolerance for the negative feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0.75**
Minimum:	0.0
Maximum:	100.0

phasetolerance_p
: The feature matching tolerance for the positive feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**0.75**
Minimum:	0.0
Maximum:	100.0

phasetolerance_r
: The feature matching tolerance for the aromatic ring feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1.5**
Minimum:	0.0
Maximum:	100.0

phasetolerance_x
: The feature matching tolerance for the custom(X) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

phasetolerance_y
: The feature matching tolerance for the custom(Y) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

phasetolerance_z

The feature matching tolerance for the custom(Z) feature. The valid range is 0.0 to 100.0

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	100.0

picktomovetiles

If this flag is on, then tile picking will be turned on when tile mode is entered. This allows tiles to be rearranged without changing the content of the Workspace. If this flag is off, tile picking will be left off when tile mode is entered.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

primextoolbarshow

Whether or not to show the PrimeX toolbar.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

primextoolbarstyle

Deterimines the appearance of the PrimeX toolbar

Valid values:	icononly textonly textbeside textunder
Default value:	**textunder**

prioritizeredraw

If this flag is on, then redraws will ignore pending events. If this flag is off, redraws will be skipped if there are pending events.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

projectbackuplimit

This is deprecated and calls: projectbackup maxbackups=<int>. Please use that and 'projectbackup instead.

Valid values:	integers
Default value:	**0**
Minimum:	0

projectdbsync

Holds the project database synchronous pragma flag value. This synchronous pragma flag will be set to the project database when we open an existing project or create a new project.

Valid values: off
normal
full
Default value: **off**

projectmemorycachelimit

Specifies Maestro project memory cache limit in MBs. If memory usage exceeds from given cache limit, then Maestro dumps entry related data (structure information, volume, and surface) into the disk.

Valid values: integers
Default value: **500**

projectsave

Specifies information to be saved to disk when project is closed. The "small" option saves only the compressed opening state. Because it takes longer to open and close projects when the current state must be expanded or deleted, this option is not recommended unless you are low on disk space. The "medium" option saves only the expanded current project state. This option requires more disk space than the "small" option, but is the fastest because it does not do compression, or save an opening state. There will generally not be an opening state saved for a project when the "medium" option is used, so the ability to revert to the opening state while the project is open is lost. The "large" option saves the compressed opening state and the expanded current project state. This option requires more disk space, but provides faster and easier access (compared to the "small" option) and also the safeguard of redundant project data. Because of the time required to save the opening state,

Valid values: small
medium
large
Default value: **medium**

projectsuffix

The default suffix or extension to be used for project directories. When a project selector is used in Maestro to choose a project, the filter is automatically set to match this suffix. The suffix is automatically appended, if missing, to the returned project path when there is potential for creating a new project (projectnew, projectrename, or projectcopy). However, when a project command is issued without using a project selector (e.g. in a macro, script, or the command input area), the suffix is not automatically appended. Also, in cases where an existing project must be opened (projectopen, projectmerge) the suffix is not appended.

Valid values: text strings
Default value: **.prj**

projectsync

Specifies whether workspace changes to project entries are automatically saved or not. Valid values are "auto", "prompt", and "manual". These cause Maestro to save changes automatically, prompt to save changes, or save changes only when explicitly directed by the user.

Valid values: auto
prompt
manual
Default value: **auto**

projecttabletoolbarstyle

Deterimines the appearance of the Project Table toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

projecttoolbarshow

Whether or not to show the Project Table toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

projecttoolbarstyle

Deterimines the appearance of the Project toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

propertyedittoolbarstyle

Deterimines the appearance of the PT Property Edit toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

propertyselectorssubset

When a new property selector instance is created, it should query this preference. If the subset is available, it should be used to initialize the Subset option menu for that instance.

Valid values: text strings
Default value: **allprimaryproperties**

propertytoolbarstyle
Deterimines the appearance of the PT Property toolbar
Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

pspinterface
Default user interface for Prime Structure Prediction. Can be user , msv or wizard .
Valid values: text strings
Default value: **user**

pspsequenceviewerfontsize
Prime sequence viewer font size.
Valid values: integers
Default value: **10**
Minimum: 6
Maximum: 48

pspsequenceviewerwrap
Sequence wrapping in Prime sequence viewer.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ptcellcolor A string which is a 6-hexadecimal-digit RGB string for the default color of cells in the Project Table.
Valid values: text strings
Default value: **#FFFFFF**

ptcolorcolumnheader
If this flag is on, the column header cells in the Project Table will be colored with a different color.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ptcolornonstandardfixedarea
If this flag is on, the cells in the nonstandard fixed area of the Project Table will be colored with a different color.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ptcolorrownumber
If this flag is on, the row number cells in the Project Table will be colored with a different color.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ptcolorstandardfixedarea
If this flag is on, the cells in the standard fixed area of the Project Table will be colored with a different color.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ptcolumnheadercolor
A string which is a 6-hexadecimal-digit RGB string for the color of colomn header cells in the Project Table.

Valid values: text strings
Default value: **#87CEFA**

ptentryedittoolbarshow
Whether or not to show the Entry Edit toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ptentryselectcolor
A string which is a 6-hexadecimal-digit RGB string for the color of cells for selected entries in the Project Table.

Valid values: text strings
Default value: **#FFFFC0**

ptentrytoolbarshow
Whether or not to show the Entry toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ptfitcolumnbuffer
If set to 0, then no buffer will be added when fitting column width to data, else the given pixels will be added as buffer.

Valid values: integers
Default value: **0**

ptgroupfullselectcolor
A string which is a 6-hexadecimal-digit RGB string for the color of cells for groups with all entries selected in the Project Table.

Valid values: text strings
Default value: **#CDBA96**

ptgrouppartialselectcolor
A string which is a 6-hexadecimal-digit RGB string for the color of cells for groups with some entries selected in the Project Table.

Valid values: text strings
Default value: **#FFE7BA**

pthighlightalternatingrows
: If this flag is on, the cells in the Project Table will be colored to highlight alternating rows.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

ptnonstandardfixedareacolor
: A string which is a 6-hexadecimal-digit RGB string for the color of cells for nonstandard properties in the fixed area in the Project Table.

  Valid values: text strings
  Default value: **#FF8CFF**

ptnumberalignment
: If set to 1, then numeric data in project table will be center aligned. Setting 0 will left align the numbers and 2 will right align.

  Valid values: left
  center
  right
  Default value: **right**

ptpropertyedittoolbarshow
: Whether or not to show the Property Edit toolbar.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

ptpropertytoolbarshow
: Whether or not to show the Property toolbar.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

ptrownumbercolor
: A string which is a 6-hexadecimal-digit RGB string for the color of row number cells in the Project Table.

  Valid values: text strings
  Default value: **#87CEFA**

ptstandardfixedareacolor
: A string which is a 6-hexadecimal-digit RGB string for the color of cells for standard properties in the fixed area in the Project Table.

  Valid values: text strings
  Default value: **#9DB3CE**

pttextalignment

If set to 1, then string property data in project table will be center aligned. Setting 0 will left align the data and 2 will right align the data.

Valid values: left
center
right
Default value: **left**

ptvscrollbar

If set to 0, then vertical scrollbar will be displayed on the left side of project table, else it will be displayed on the right side of project table.

Valid values: left
right
Default value: **left**

refinebackbonesubjobs

This is the number of refinement jobs to run at one time in Refine Backbone.

Valid values: integers
Default value: **1**
Minimum: 1

renamecrystalmates

If this option is set to true, Maestro will assign unique names to generates crystal mate chains.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

representationtoolbarshow

Whether or not to show the Representaion toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

representationtoolbarstyle

Deterimines the appearance of the Representation toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **icononly**

resetclipprefs

When true, resets clipping preferences (adjustclip and zoom_topview) to their default values, and sets itself to false.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

resetfitmarginprefs

When true, resets the fit margin to the default value.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

rightclicktimerperiod

The time delay for workspace menu to appear in milliseconds.

Valid values: integers
Default value: **300**

rotamerfitselectedresidue

If this option is set to true then selected residue will be fit to Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

savedviewstoolbarstyle

Deterimines the appearance of the Saved Views toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **icononly**

savelayoutonexit

If this option is set to true, the panel layout will be saved when you exit Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

savescratchproject

If set to true, then pop up 'Save Scratch Project' dialog to Save/Discard the scratch project. Else the scratch project will be discarded.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

savetempjobdir

This option if enabled would not remove temporary job dir when job completes.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

scriptupdatesws

If this option is set to true, force redraw of main window if redraw was requested by running a command in a script.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sdcompressedsuffix
The default suffix or extension to be used for exported compressed SD files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **sdfgz**

sdsuffix The default suffix or extension to be used for exported SD files. The suffix will automatically be appended to a file name if it has no suffix.

Valid values: text strings
Default value: **sdf**

selectexcludedselected
If this option is set, then expanding the selection using the Workspace Selection menu will exclude the current selection.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

selectexpandresidues
If this option is set, then expanding the selection using the Workspace Selection menu will expand to residues rather than just atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sequenceviewerfontsize
The font size for the Workspace sequence viewer

Valid values: integers
Default value: **10**

sequenceviewerproximity
Cutoff distance for proximity coloring in the sequence viewer

Valid values: reals
Default value: **4**
Minimum: 0.0

sequenceviewershow
Whether the sequence viewer is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

show2dstructures
If set to true, then 2D column is shown in the PT for the new/scratch project otherwise its hidden by default.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

show2dtooltip

If set to true, then hovering the mouse over Project table 'Row' column shows the 2D structure image as the column tooltip otherwise no tooltip is shown for the 'Row' column.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showdialogatcursor

If this option is set to true then the dialogs will be shown at mouse cursor location. If it is false then the dialogs will be displayed at their last shown position.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showdynamictoolbar

Display the dynamic toolbar

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showentryname

If set to true, then Entry Name property column is shown in the PT otherwise its hidden by default.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showjobstatus

Whether or not job status is shown in the Workspace status bar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showmonitorpanel

If this is set to true, and monitorjobs is true, then Monitor panel will be displayed after a job is launched. Otherwise, the Monitor panel will not be displayed on job launch.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

shownewproperties

If set to 0, show all the new properties, for 1 shows only primary properties and for 2 hides new properties

Valid values: all
primaryonly
hidenew
Default value: **primaryonly**

showobjectindex
Whether show the index of the object being picked in atom feedback.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showpanelsontop
If this option is set to true then the panels will always be shown on top of the main window.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showprojecttable
If set to true, then when a project is opened, the project table will automatically be displayed.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showptpropertytree
If set to true, then property tree will be shown when showing the project table.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showstars If set to true, then Starts property column is shown in the PT for new projects otherwise its hidden by default.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

spherequalityreduction
Controls how many steps we can reduce the quality by
Valid values: integers
Default value: **56**
Minimum: 0

statusbarshow
Whether the status bar is show or hide.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

styletoolbarstyle
Deterimines the appearance of the Style toolbar
Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

taskbasedmenu
: If this option is set to true, the Applications and Workflows menus will be replaced by the Tasks menu.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

tbcollapsed
: Whether the tool bar is collapsed or not.
Valid values: integers
Default value: **0**

tblocation
: Whether the tool bar is located left or right.
Valid values: integers
Default value: **1**

tempprojectlocation
: The path to create the Maestro temporary project
Valid values: text strings
Default value:

titlebardirectory
: If this option is set to true, the main window's title bar will include the current working directory path.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

titlebarmmversion
: If this option is set to true, the main window's title bar will include mmshare's version number.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

titlebarprofile
: If this option is set to true, the main window's title bar will include current profile name.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

titlebarversion
: If this option is set to true, the main window's title bar will include Maestro's version number.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

toolbarshow
: Whether the tool bar is show or hide.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

topviewshow

Whether the top view window is show or hide.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

trackpadrotsensitivity

Trackpad rotate sensitivity. The valid range is 1.0 to 100.0

Valid values: reals
Default value: **15**
Minimum: 1.0
Maximum: 100.0

trackpadzsensitivity

Trackpad Pinch gesture sensitivity for zoom. The valid range is 1.0 to 100.0. Larger is more sensitive

Valid values: reals
Default value: **25**
Minimum: 1.0
Maximum: 100.0

turnoffreapplystlewhenclosingproject

If this value is on then the Workspace Style reapply style setting will be turned off when a project is closed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

useclipdistance

Set the flag of using clipping distance or not when focusing on a substructure.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

useremotedriver

If this flag is set, jobs launched on remote hosts will also have their driver script run on the remote host.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

viewanimation

Whether or not view animation is used on actions like fit to Workspace, spot centering, and view reset.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

viewanimationdistance

This value sets the maximum distance for view animation, if it is turned on.

Valid values: reals
Default value: **10**
Minimum: 0.0

viewanimationduration

This value sets the duration of each view animation, if it is turned on. The valid range is 0.0 to 5.0 seconds.

Valid values: reals
Default value: **2**
Minimum: 0.0
Maximum: 5.0

viewname When saving a view, if set to 0, then 'Save View' dialog pops up asking for view name , else view will be stored with default name without prompting for name.

Valid values: prompt
default
Default value: **prompt**

viewstoolbarshow

Whether or not to show the Views toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

viewtoolbarshow

Whether or not to show the View toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

viewtoolbarstyle

Deterimines the appearance of the View toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **icononly**

warnclosescratchproject

This option is used to record if warn users before closing scratch project

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

warnoverwritejobfiles

This option is used to record if warn users before existing job files being overwritten.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

workspacemenushow
Whether the workspace menu should be displayed on right click.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

workspacetoolbarshow
Whether or not to show the Workspace toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

workspacetoolbarstyle
Deterimines the appearance of the Workspace toolbar

Valid values: icononly
textonly
textbeside
textunder
Default value: **textunder**

writecmdlogonexit
If this option is set to true, cmd log will be be saved when you exit Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

wsassistanttoolbarshow
Whether or not to show the Workspace Assistant toolbar.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

wsfeedbackfontbold
A bool which determines whether bold font is to be used for workspace feedback text

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

wsfeedbackfontitalics
A bool which determines whether italics font is to be used for workspace feedback text

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

wsfeedbackfontname
A string which determines the font to be used for workspace feedback text

Valid values: text strings
Default value:

wsfeedbackfontsize
: A double which determines the font size to be used for workspace feedback text

  Valid values: reals
  Default value: **14**
  Minimum: 0.0

wsselectpickstate
: What kind of pick state the Workspace selection contains.

  Valid values: atom
  residue
  molecule
  chain
  entry
  Default value: **atom**

wsseqvieweralignbyresnum
: A bool which determines whether workspace sequence viewer has to be aligned by residue number or not

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

wsseqviewerdisplaynonprotein
: A bool which determines whether to display non-protein molecules in workspace sequence viewer.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

wsseqviewerdisplayssa
: A bool which determines whether display SSA in workspace sequence viewer or not

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

wsseqviewerreorderresidues
: Enables reordering residues by residue number.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

wsseqviewerwrap
: A bool which determines whether workspace sequence viewer has to be wrapped or not

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

zeroorderbondtraversal
: Whether Traverse zero order bonds in molecules.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

previousresidue

If only a single residue is selected in the Workspace, then find the previous one.

Syntax:

`previousresidue`

primexaddwaters

Launches an add waters job.

Syntax:

`primexaddwaters` *bfactor_maximum*=⟨ x ⟩ *cutoff_maximum*=⟨ x ⟩ *cutoff_minimum*=⟨ x ⟩ *hbond_distance*=⟨ x ⟩ *peak_height*=⟨ x ⟩ ⟨ structure file ⟩

Options:

bfactor_maximum
The maximum b-factor.
Valid values: reals
Default value: **100**

cutoff_maximum
The maximum distance cutoff, in angstroms.
Valid values: reals
Default value: **4**
Minimum: 0.0

cutoff_minimum
The minimum distance cutoff, in angstroms.
Valid values: reals
Default value: **2.6**
Minimum: 0.0

hbond_distance
Valid values: reals
Default value: **2.2**

peak_height
: The minimum peak height.

Valid values:	reals
Default value:	**4**
Minimum:	0.0

Operands:
⟨ structure file ⟩
The name of a PDB structure file to use as the starting structure.

primexcalculatedensityfit

Calculates density fit for the given atoms.

Syntax:

`primexcalculatedensityfit`

primexcalculatedensitypeaks

Calculates density peaks.

Syntax:

`primexcalculatedensitypeaks` *peak_threshold*=⟨ x ⟩

Options:

peak_threshold
: Threshold for calculating density peaks. Only peaks that are greater than or equal to this number will be displayed.

Valid values:	reals
Default value:	**3**
Minimum:	1.0
Maximum:	10.0

primexcalculaterfactors

Calculate R-factors for the included structure

Syntax:

```
primexcalculaterfactors
```

primexcreatemap

Create map for Primex.

Syntax:

```
primexcreatemap map_2fofc=yes | no  map_3fo2fc=yes | no
        map_box_xmax=⟨ x ⟩  map_box_xmin=⟨ x ⟩  map_box_ymax=⟨ x ⟩
        map_box_ymin=⟨ x ⟩  map_box_zmax=⟨ x ⟩  map_box_zmin=⟨ x ⟩
        map_extent=model | unitcell | asymmetric | box
        map_extent_box=fractional | orthogonal  map_fo=yes | no
        map_fofc=yes | no  map_grid_size=size033 | size025 | other
        map_grid_size_other=⟨ x ⟩  map_weighting=sigma | unweighted
        maptype=regular | omit | composite  max_displacement=⟨ x ⟩
        method=bhatcohen | kicked  molecule_plus=⟨ x ⟩
        number_averaged=⟨ n ⟩  omit=⟨ x ⟩  scale_map=yes | no
```

Options:

map_2fofc Whether or not to create a 2Fo-Fc map

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

map_3fo2fc

Whether or not to create a 3Fo-2Fc map

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_box_xmax

The maximum X-coordinate for the map extent box.

Valid values: reals
Default value: **0**

map_box_xmin

The minimum X-coordinate for the map extent box.

Valid values: reals
Default value: **0**

map_box_ymax

The maximum Y-coordinate for the map extent box.

Valid values: reals
Default value: **0**

map_box_ymin
The minimum Y-coordinate for the map extent box.
Valid values: reals
Default value: **0**

map_box_zmax
The maximum Z-coordinate for the map extent box.
Valid values: reals
Default value: **0**

map_box_zmin
The minimum Z-coordinate for the map extent box.
Valid values: reals
Default value: **0**

map_extent
The extent of the map.
Valid values: model
unitcell
asymmetric
box
Default value: **model**

map_extent_box
Which space the box extent is in.
Valid values: fractional
orthogonal
Default value: **fractional**

map_fo Whether or not to create an Fo map
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_fofc Whether or not to create an Fo-Fc map
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_grid_size
The grid size for the maps.
Valid values: size033
size025
other
Default value: **size033**

map_grid_size_other
A user-specified grid size.
Valid values: reals
Default value: **0.1**

map_weighting
The weighting for the maps.
Valid values: sigma
unweighted
Default value: **sigma**

maptype The map types of Primex create map.
Valid values: regular
omit
composite
Default value: **regular**

max_displacement
The maximum displacement for kicked method.
Valid values: reals
Default value: **0.1**
Minimum: 0.0

method The methods of Primex create map.
Valid values: bhatcohen
kicked
Default value: **bhatcohen**

molecule_plus
The amount in angstroms to increase the model by when calculating the extent for generating maps.
Valid values: reals
Default value: **5**
Minimum: 0.0

number_averaged
The number averaged for kicked method.
Valid values: integers
Default value: **10**
Minimum: 0

omit The percentage of data to omit.
Valid values: reals
Default value: **5**
Minimum: 0.0

scale_map Indicates whether or not to scale the map to sigma units
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

primexdecreaseisovalue

Decrease isovlaue for currently selected map surface.

Syntax:

`primexdecreaseisovalue` *entry*=⟨ text ⟩ *numsigma*=⟨ x ⟩
volume=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry name of the entry that the surface belongs to.
Valid values: text strings
Default value:

numsigma The amount of decreasing isovalue, e.g. 0.1*sigma.
Valid values: reals
Default value: **0.1**
Minimum: 0.01

volume The name of the volume that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to decrease the isovalue for.

primexdeleteallrigidbodies

Deletes all rows in the rigid bodies table.

Syntax:

`primexdeleteallrigidbodies`

primexdeleteoccupancy

Deletes the partial occupancy data for the given atom

Syntax:

`primexdeleteoccupancy` ⟨atom index⟩

Operands:
⟨atom index⟩
An atom index for the main CT.

primexdeleterigidbodies

Deletes the selected rows in the rigid bodies table.

Syntax:

`primexdeleterigidbodies`

primexentrywscreate

Creates a project entry from atoms in the workspace. The entry name and title are derived from the current WS entry, and all non-structure-specific PrimeX properties are copied to the new entry.

Syntax:

`primexentrywscreate`

primexexport

Export the included structure as a formatted PDB file

Syntax:

`primexexport` ⟨file name⟩

Operands:
⟨file name⟩
The name of the file to export the PrimeX data to.

primexincreaseisovalue

Increase isovlaue for currently selected map surfaces.

Syntax:

`primexincreaseisovalue` *entry*=⟨ text ⟩ *numsigma*=⟨ x ⟩
volume=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry name of the entry that the surface belongs to.

Valid values: text strings
Default value:

numsigma The amount of increasing isovalue, e.g. 0.1*sigma.

Valid values: reals
Default value: **0.1**
Minimum: 0.01

volume The name of the volume that the surface belongs to.

Valid values: text strings
Default value:

Operands:

⟨ surface ⟩

The name of the surface to increase the isovalue for.

primexinputdata

Adds or replaces the PrimeX data with the given data.

Syntax:

primexinputdata *assign_bond_orders*=yes | no *cella*=⟨ x ⟩ *cellalpha*=⟨ x ⟩ *cellb*=⟨ x ⟩ *cellbeta*=⟨ x ⟩ *cellc*=⟨ x ⟩ *cellgamma*=⟨ x ⟩ *constant_bfactor*=⟨ x ⟩ *foproperty*=⟨ text ⟩ *generate_testset*=yes | no *map_file*=⟨ text ⟩ *map_type*=mapfo | mapfofc | map2fofc | map3fo2fc *reflectionfile*=⟨ text ⟩ *sequence_source*=selected_entries | workspace | file *sequencefile*=⟨ text ⟩ *sigmaproperty*=⟨ text ⟩ *spacegroup*=⟨ n ⟩ *spacegroupname*=⟨ text ⟩ *structure_source*=selected_entries | workspace | file *test_percentage*=⟨ x ⟩ *testproperty*=⟨ text ⟩ *use_bfactors*=current | constant ⟨ structure file ⟩

Options:

assign_bond_orders
This option assigns correct bond orders to the input structure.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

cella The unit cell a parameter.
Valid values: reals
Default value: **0**

cellalpha The unit cell alpha parameter.
Valid values: reals
Default value: **90**

cellb The unit cell b parameter.
Valid values: reals
Default value: **0**

cellbeta The unit cell beta parameter.
Valid values: reals
Default value: **90**

cellc The unit cell c parameter.
Valid values: reals
Default value: **0**

cellgamma
The unit cell gamma parameter.
Valid values: reals
Default value: **90**

constant_bfactor
The constant value to set the B-factors to.
Valid values: reals
Default value: **20**

foproperty The Fo property name for mtz reflection data file.

Valid values: text strings
Default value: **FP**

generate_testset

This option indicates whether or not a new random test set should be generated in the reflection file.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_file The map data file.

Valid values: text strings
Default value:

map_type What kind of map the map file contains.

Valid values: mapfo
mapfofc
map2fofc
map3fo2fc
Default value: **mapfo**

reflectionfile

The crystal reflection data file.

Valid values: text strings
Default value:

sequence_source

Whether to get the sequence from an external file or from the contents of the Workspace.

Valid values: selected_entries
workspace
file
Default value: **file**

sequencefile

The sequence file.

Valid values: text strings
Default value:

sigmaproperty

The Sigma(Fo) property name for mtz reflection file.

Valid values: text strings
Default value: **SIGFP**

spacegroup

The space group number.

Valid values: integers
Default value: **0**

spacegroupname
The space group name.

Valid values: text strings
Default value:

structure_source
Whether to get the structure from an external file or from the contents of the Workspace.

Valid values: selected_entries
workspace
file
Default value: **file**

test_percentage
This is the percenatge of reflections to use as the test set.

Valid values: reals
Default value: **5**
Minimum: 0.01
Maximum: 100

testproperty
The test set property name for mtz reflection file.

Valid values: text strings
Default value: **FreeR_flag**

use_bfactors
Whether to keep the existing B-factor values or to replace them with a constant value.

Valid values: current
constant
Default value: **current**

Operands:
⟨ structure file ⟩
The name of a PDB structure file to use as the starting structure.

primexmarkersettings

Set graphical data of PrimeX peak markers.

Syntax:

primexmarkersettings *ambient*=⟨ x ⟩ *diffuse*=⟨ x ⟩ *emission*=⟨ x ⟩ *num_slices*=⟨ n ⟩ *num_stacks*=⟨ n ⟩ *shininess*=⟨ x ⟩ *specular*=⟨ x ⟩ *transparency*=⟨ x ⟩

Options:

ambient Set the ambient material property for the peak markers.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

diffuse Set the diffuse material property for the peak markers.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	1.0

emission Set the emission material property for the peak markers.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0
Maximum:	1.0

num_slices
Set the number of slices for the peak markers.

Valid values:	integers
Default value:	**18**
Minimum:	2

num_stacks
Set the number of stacks for the peak markers.

Valid values:	integers
Default value:	**9**
Minimum:	2

shininess Set the shininess material property for the peak markers.

Valid values:	reals
Default value:	**80**
Minimum:	0.0
Maximum:	128.0

specular Set the specular material property for the peak markers.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

transparency
The transparency percentage of the peak markers.

Valid values:	reals
Default value:	**50**
Minimum:	0.0
Maximum:	100.0

primexmutatemodeltosequence

Mutate the model in the workspace (which must be a single entry) to match the reference sequence specified for PrimeX.

Syntax:

`primexmutatemodeltosequence` *delete_gap_residues*=yes | no

Options:

delete_gap_residues
This options controls whether residues that are aligned with gaps are deleted.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

primexncsaddrow

Adds a row with default settings to the NCS table.

Syntax:

`primexncsaddrow`

primexncsdeleterows

Deletes the selected rows from the NCS tab.

Syntax:

`primexncsdeleterows`

primexncseditcell

Edit a cell in the NCS table

Syntax:

`primexncseditcell` *column*=chains | residues | bfactor | coord
row=⟨ n ⟩ ⟨ value ⟩

Options:

column	The column of the NCS cell to edit.	
	Valid values:	chains residues bfactor coord
	Default value:	**chains**
row	The row of the NCS cell to edit.	
	Valid values:	integers
	Default value:	**1**
	Minimum:	1

Operands:
⟨ value ⟩
The new value for the cell.

primexncsselectextendrow

Extends the selection to this row in the NCS table in PrimeX.

Syntax:

`primexncsselectextendrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend the select to.

primexncsselectonlyrow

Selects only this row in the NCS table in PrimeX.

Syntax:

`primexncsselectonlyrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select only in the table row.

primexncsselectrow

Selects the given row in the NCS table in PrimeX.

Syntax:

`primexncsselectrow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the table.

primexncstogglebbonly

Toggles the BB Only value for the given row in the NCS table in PrimeX.

Syntax:

`primexncstogglebbonly` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to toggle the BB Only value for.

primexncsunselectrow

Unselects the given row in the NCS table in PrimeX.

Syntax:

primexncsunselectrow ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the table.

primexpartialoccupancy

Sets the partial occupancy for selected atoms

Syntax:

primexpartialoccupancy *occupancy*=⟨ x ⟩ ⟨ ASL ⟩

Options:

occupancy	The partial occupancy.	
	Valid values:	reals
	Default value:	**0.5**
	Minimum:	0.0
	Maximum:	1.0

Operands:
⟨ ASL ⟩
An ASL representing the atoms to set the partial occupancy for.

primexplaceligand

Launches a place ligand/solvent job.

Syntax:

primexplaceligand *box_buffer*=⟨ x ⟩ *energy_weight*=⟨ x ⟩
generate_mates=yes | no *ligand*=⟨ text ⟩ *ligandsfile*=⟨ text ⟩
ligandsource=file | entries | solvent *refine_poses*=yes | no
scoring_weight=⟨ x ⟩ *use_ligprep*=yes | no ⟨ job name ⟩

Options:

box_buffer Controls the size of the buffer added to the box which encloses the selected density peaks in order to create the grid box for ligand placement.

Valid values: reals
Default value: **16**

energy_weight
Energy density weight.

Valid values: reals
Default value: **120**
Minimum: 1.0
Maximum: 1000.1

generate_mates
Indicates whether or not to generate symmetry mates near the site.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ligand The entry name of the ligand to use.

Valid values: text strings
Default value:

ligandsfile The file containing one or more ligands.

Valid values: text strings
Default value:

ligandsource
The source of the ligands to be used.

Valid values: file
entries
solvent
Default value: **file**

refine_poses
Set this option to refine the poses that Glide generates using the X-ray data.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

scoring_weight
Scoring density weight.

Valid values: reals
Default value: **50**
Minimum: 1.0
Maximum: 1000.1

use_ligprep
: Indicates whether or not to run LigPrep on the ligand before placing it.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

Operands:

⟨ job name ⟩

The name of the job.

primexpublish

Exports the structure in the Workspace as the final structure for a crystallographic refinement.

Syntax:

`primexpublish` ⟨ file ⟩

Operands:

⟨ file ⟩

The name of a PDB file to save as the final structure.

primexrealspace

Launches a real-space refinement job.

Syntax:

`primexrealspace` *bias_existing*=yes | no *grid_size*=⟨ x ⟩
loop_bfactorsfit=⟨ x ⟩ *loop_chain*=⟨ text ⟩ *loop_end*=⟨ text ⟩
loop_setbfactorsfit=yes | no *loop_start*=⟨ text ⟩
minimize_bfactorsfit=⟨ x ⟩ *minimize_grid_size*=⟨ x ⟩
minimize_setbfactorsfit=yes | no *sidechain_bfactorsfit*=⟨ x ⟩
sidechain_grid_size=⟨ x ⟩ *sidechain_setbfactorsfit*=yes | no
⟨ structure file ⟩

Options:

bias_existing
If set to true, then PrimeX will bias loops towards the existing loop structure. If set to false, then PrimeX will bias loops towards the reflection data.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

grid_size The grid size used for real-space loop refinement jobs.

Valid values: reals
Default value: **1**
Minimum: 0.5
Maximum: 1.0

loop_bfactorsfit
The radius of setting b_factors of atoms to be fit for a real-space loop refinement job.

Valid values: reals
Default value: **5**
Minimum: 0.0

loop_chain This is the chain for loop refinement.

Valid values: text strings
Default value:

loop_end This is the ending residue for loop refinement.

Valid values: text strings
Default value:

loop_setbfactorsfit
An option which determines if the real-space loop refinement job will set b-factors of atoms to be fit.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

loop_start This is the starting residue for loop refinement.

Valid values: text strings
Default value:

minimize_bfactorsfit
The radius of setting b-factors of atoms to be fit for a real-space minimization job.

Valid values: reals
Default value: **20**
Minimum: 0.0

minimize_grid_size
The grid size used for real-space minimization refinement jobs.

Valid values: reals
Default value: **0.5**
Minimum: 0.5
Maximum: 1.0

minimize_setbfactorsfit
An option which determines if the real-space minimization job will set b-factors of atoms to be fit.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

sidechain_bfactorsfit
The radius of setting b-factors of atoms to be fit for a real-space sidechain refinement job.

Valid values: reals
Default value: **20**
Minimum: 0.0

sidechain_grid_size
The grid size used for sidechain refinement jobs.

Valid values: reals
Default value: **0.5**
Minimum: 0.5
Maximum: 1.0

sidechain_setbfactorsfit
An option which determines if the real-space loop refinement job will set b-factors of atoms to be fit.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:

⟨structure file⟩

The name of a PDB structure file to use as the starting structure.

primexreciprocal

Launches a reciprocal-space refinement job.

Syntax:

`primexreciprocal` *bb_angles*=⟨ x ⟩ *bb_bonds*=⟨ x ⟩
bfactor=individual | anisotropic *bfactor_high_limit*=⟨ x ⟩
bfactor_low_limit=⟨ x ⟩ *bfactor_restraint_weighting*=⟨ x ⟩
constant_weight=⟨ x ⟩ *constantvalue*=⟨ x ⟩ *cooling_steps*=⟨ n ⟩
energy_model=approximate | complete *final_steps*=⟨ n ⟩
final_temp=⟨ x ⟩ *heating_steps*=⟨ n ⟩ *high_temp*=⟨ x ⟩
initial_steps=⟨ n ⟩ *initial_temp*=⟨ x ⟩ *map_2fofc*=yes | no
map_3fo2fc=yes | no *map_fo*=yes | no *map_fofc*=yes | no
md_steps=⟨ n ⟩ *md_time_step*=⟨ x ⟩ *method*=minimization |
rigidbodies | simulatedannealing
min_minimizer=truncatednewton | conjugategradient |
quasinewton | optimal *minimization_max_cycles*=⟨ n ⟩
minimization_max_steps=⟨ n ⟩ *minimize*=coordinates | bfactors
| occupancies | groupedbfactors *nonbonded_cutoff*=⟨ x ⟩
rb_minimizer=truncatednewton | conjugategradient |
quasinewton | optimal *rfree*=⟨ x ⟩ *rigidmaxcycles*=⟨ n ⟩
rigidmaxsteps=⟨ n ⟩ *side_angles*=⟨ x ⟩ *side_bonds*=⟨ x ⟩
target=likelihood | leastsquares *ungroupedatoms*=fix | group
usevalues=current | constant *weight_multiplication_factor*=⟨ x ⟩
xray_weight=multiply | constant ⟨ structure file ⟩

Options:

bb_angles The target sigma value for B-factor restraints for the backbone angles.

Valid values: reals
Default value: **2**

bb_bonds The target sigma value for B-factor restraints for the backbone bonds.

Valid values: reals
Default value: **1.5**

bfactor The atomic B-factor for minimization

Valid values: individual
anisotropic
Default value: **individual**

bfactor_high_limit
The high-resolution limit for B-factors.

Valid values: reals
Default value: **100**
Minimum: 0.0

bfactor_low_limit
The low-resolution limit for B-factors.

Valid values: reals
Default value: **2**
Minimum: 0.0

bfactor_restraint_weighting
The B-factor restraint weighting factor

Valid values: reals
Default value: **1**

constant_weight
A constant weight to replace the automatically calculated weight with.

Valid values: reals
Default value: **0.5**

constantvalue
Size of the constant B-factor value to set to.

Valid values: reals
Default value: **20**

cooling_steps
The number of cooling steps.

Valid values: integers
Default value: **1250**
Minimum: 1

energy_model
The energy model for simulated annealing.

Valid values: approximate
complete
Default value: **approximate**

final_steps The number of final minimization steps.

Valid values: integers
Default value: **30**
Minimum: 1

final_temp The final temperature, in Kelvin.

Valid values: reals
Default value: **300**
Minimum: 0

heating_steps
The number of heating steps.

Valid values: integers
Default value: **125**
Minimum: 1

high_temp The high temperature, in Kelvin.

Valid values: reals
Default value: **700**
Minimum: 0

initial_steps

The number of initial minimization steps.

Valid values: integers
Default value: **30**
Minimum: 1

initial_temp

The initial temperature, in Kelvin.

Valid values: reals
Default value: **50**
Minimum: 0

map_2fofc Whether or not to create a 2Fo-Fc map

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_3fo2fc

Whether or not to create a 3Fo-2Fc map

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_fo Whether or not to create an Fo map

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

map_fofc Whether or not to create an Fo-Fc map

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

md_steps The number of MD energy scale estimation steps.

Valid values: integers
Default value: **100**
Minimum: 1

md_time_step

The MD time step.

Valid values: reals
Default value: **0.008**

method The refinement method to use.

Valid values: minimization
rigidbodies
simulatedannealing

Default value: **minimization**

min_minimizer
Which minimizer to use for Minimization.

Valid values: truncatednewton
conjugategradient
quasinewton
optimal
Default value: **optimal**

minimization_max_cycles
The maximum number of minimization cycles for minimization.

Valid values: integers
Default value: **3**
Minimum: 1

minimization_max_steps
The maximum number of minimization steps per cycle for minimization.

Valid values: integers
Default value: **8**
Minimum: 0

minimize The minimize option for Minimization.

Valid values: coordinates
bfactors
occupancies
groupedbfactors
Default value: **coordinates**

nonbonded_cutoff
The non-bonded interactions cutoff, in angstroms.

Valid values: reals
Default value: **9.5**

rb_minimizer
Which minimizer to use for Rigid Bodies.

Valid values: truncatednewton
conjugategradient
quasinewton
optimal
Default value: **optimal**

rfree The R-free percentage.

Valid values: reals
Default value: **0**

rigidmaxcycles
Maximum number of minimization cycles for a rigid body refinement.

Valid values: integers
Default value: **3**
Minimum: 1

rigidmaxsteps
Maximum number of minimization steps per cycle for a rigid body refinement.

Valid values: integers
Default value: **8**
Minimum: 1

side_angles
The target sigma value for B-factor restraints for the side chain angles.

Valid values: reals
Default value: **2.5**

side_bonds
The target sigma value for B-factor restraints for the side chain bonds.

Valid values: reals
Default value: **2**

target The refinement target.

Valid values: likelihood
leastsquares
Default value: **likelihood**

ungroupedatoms
How should ungrouped atoms in a Rigid Body refinement be treated?

Valid values: fix
group
Default value: **fix**

usevalues Whether to use the existing B-factor values or whether to replace them with a constant value.

Valid values: current
constant
Default value: **current**

weight_multiplication_factor
The factor to multiply the automatically calculated weight by.

Valid values: reals

Default value: **1**

xray_weight
Whether to multiply the weight by a factor or replace it with a constant term.
Valid values: multiply
constant
Default value: **constant**

Operands:
⟨structure file⟩
The name of a PDB structure file to use as the starting structure.

primexselectdensityblob

Selects a row in the density blobs table.

Syntax:

```
primexselectdensityblob
```

primexselectdensitypeak

Selects a row in the density peaks table.

Syntax:

```
primexselectdensitypeak
```

primexselectextenddensitypeak

Extends the selection in the density peaks table to the given row.

Syntax:

```
primexselectextenddensitypeak
```

primexselectextendligandsolvent

Extends the selection to this row in the ligand / solvents table.

Syntax:

```
primexselectextendligandsolvent
```

primexselectligandsolvent

Selects a row in the ligand / solvents table.

Syntax:

```
primexselectligandsolvent
```

primexselectonlydensitypeak

Selects only the given row in the density peaks table.

Syntax:

```
primexselectonlydensitypeak
```

primexselectonlyligandsolvent

Selects a row in the ligand / solvents table.

Syntax:

```
primexselectonlyligandsolvent
```

primexselectonlyrigidbody

Selects only the given rigid body row.

Syntax:

`primexselectonlyrigidbody`

primexselectrigidbody

Selects the given rigid body row.

Syntax:

`primexselectrigidbody`

primexsetoccupancy

Sets the partial occupancy data for the given atom

Syntax:

`primexsetoccupancy` *occupancy*=⟨ x ⟩ ⟨ atom index ⟩

Options:

occupancy The partial occupancy value to set.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

Operands:

⟨ atom index ⟩

An atom index for the main CT.

primexsettings

Holds options for PrimeX jobs.

Syntax:

primexsettings *covalent_radius*=⟨ x ⟩ *filterf*=⟨ x ⟩ *filterrms*=⟨ x ⟩
filtersigma=⟨ x ⟩ *high_resolution*=⟨ x ⟩
implicit_solvation=yes | no *include_hatoms*=yes | no
ion_radius=⟨ x ⟩ *low_resolution*=⟨ x ⟩ *maprproperty*=⟨ text ⟩
mapsource=model | file *mapsourcefile*=⟨ text ⟩
max_memory=⟨ x ⟩ *phiproperty*=⟨ text ⟩
planargrouprestraints=low | normal | high *postrefine*=yes | no
rejectreflections=yes | no *resolution_type*=calculate | define
scaling=none | isotropic | anisotropic | both
shrink_factor=⟨ x ⟩ *solventmethod*=mask | babinet | none
weightingfactor=⟨ x ⟩ ⟨ none ⟩

Options:

covalent_radius
The VdW radius for covalent atoms.

Valid values: reals
Default value: **1.4**

filterf Reject reflections with F < this value.

Valid values: reals
Default value: **0**

filterrms Reject reflections with F / rms(F) > this value.

Valid values: reals
Default value: **10000**

filtersigma Reject reflections with F / sigma(F) < this value.

Valid values: reals
Default value: **0**

high_resolution
The high-resolution limit, in angstroms.

Valid values: reals
Default value: **1**
Minimum: 0.2

implicit_solvation
Whether or not to use implicit solvation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

include_hatoms
Whether or not to include H atoms.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ion_radius The radius for ions.

Valid values: reals
Default value: **0.8**

low_resolution

The low-resolution limit, in angstroms.

Valid values: reals
Default value: **30**
Minimum: 2.0

maprproperty

The map F property name for mtz reflection data file.

Valid values: text strings
Default value:

mapsource The map source of Primex create map.

Valid values: model
file
Default value: **model**

mapsourcefile

The input map source file.

Valid values: text strings
Default value:

max_memory

Maximum memory to use for calculations.

Valid values: reals
Default value: **400**
Minimum: 10.0

phiproperty

The phi property name for mtz reflection file.

Valid values: text strings
Default value:

planargrouprestraints

The planar group restraints level (low, normal or high).

Valid values: low
normal
high
Default value: **normal**

postrefine Turns on running a reciprocal-space minimization job after a real-space refinement job.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

rejectreflections
Indicates whether or not to filter reflections.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

resolution_type
Specifies how the low and high resolution limit being defined: 1) calculate from data and cell constants, and 2) use data set by default or by users.
Valid values: calculate
define
Default value: **calculate**

scaling The overall B-factor scaling.
Valid values: none
isotropic
anisotropic
both
Default value: **anisotropic**

shrink_factor
The shrink factor.
Valid values: reals
Default value: **1.4**

solventmethod
The solvent correction method.
Valid values: mask
babinet
none
Default value: **mask**

weightingfactor
The weighting factor.
Valid values: reals
Default value: **1**
Minimum: 0.0

Operands:
⟨ none ⟩

primexswitchaltpositions

Swaps the real and alternate positions (occupancies) for the given atoms.

Syntax:

`primexswitchaltpositions` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The atoms to swap positions.

primextoggleselectrigidbody

Toggles the selection of the given rigid body row.

Syntax:

`primextoggleselectrigidbody`

primexunitcell

This keyword is used to set options associated with the PrimeX unit cell.

Syntax:

`primexunitcell` *showmarker*=yes | no

Options:

showmarker
This determines whether the unit cell is shown or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

primexunselectdensitypeak

Unselects a row in the density peaks table.

Syntax:

`primexunselectdensitypeak`

primexunselectligandsolvent

Unselects a row in the ligand / solvents table.

Syntax:

```
primexunselectligandsolvent
```

primexwriteaddwaters

Write the inputs files for an add waters job.

Syntax:

```
primexwriteaddwaters
```

primexwritecalculaterfactors

Write a calculate R-factors job for the included structure

Syntax:

```
primexwritecalculaterfactors
```

primexwritecreatemap

Write a create map job for Primex.

Syntax:

```
primexwritecreatemap
```

primexwriteexport

Write an export job for the included structure

Syntax:

`primexwriteexport`

primexwriteplaceligand

Writes a place ligand/solvent job.

Syntax:

`primexwriteplaceligand`

primexwriterealspace

Writes the input files for a real-space refinement job.

Syntax:

`primexwriterealspace`

primexwritereciprocal

Write the input files for a reciprocal-space refinement job.

Syntax:

`primexwritereciprocal`

profile

Specifies a profile of Maestro

Syntax:

`profile` *current*=yes | no *default*=yes | no *directory*=⟨ text ⟩
parent=⟨ text ⟩ *select*=yes | no *standard*=yes | no

Options:

current State indicates whether the profile is a current profile or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

default State indicates whether the profile is a default profile or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

directory This is profile directory path.

Valid values: text strings
Default value: **dir**

parent This is the parent of profile.

Valid values: text strings
Default value: **parent**

select State indicates whether the profile is a currently elected or not..

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

standard State indicates whether the profile is a standard profile or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

projectbackup

Perform project backup.See projectbackupprefs for the preferences that control backups. Logging is done to schrodinger.autobackup.log in your home directory on Linux or to your %LOCALAPPDATA% on Windows.

Syntax:

`projectbackup` ⟨comment_string⟩

Operands:
⟨comment_string⟩
The comment to include in the backup

projectbackuppreference

Set project backup preferences. These are persistent settings.

Syntax:

`projectbackuppreference` *auto*=yes | no *dialog*=never | always | errors *emaildebug*=yes | no *emailfrom*=⟨ text ⟩ *emailnotification*=yes | no *emailpassword*=⟨ text ⟩ *emailport*=⟨ n ⟩ *emailsecurity*=none | starttls | ssltls *emailsmtpserver*=⟨ text ⟩ *emailto*=⟨ text ⟩ *limitbackups*=yes | no *maxbackups*=⟨ n ⟩ *time*=⟨ n ⟩

Options:

auto When on perform an automatic backup once a day at the specified time. If off, no automatic backup is performed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dialog When to show a notification dialog. This can be 'never' in which case no dialog is shown, 'always' in which case a dialog is always shown regardless of success or failure, and 'errors' meaning only show the dialog when there are errors. This setting applies to projectbackup. It applies to projectcheck when projectcheck is run from the automated backup. Otherwise, projectcheck always displays a dialog when an error is encountered.

Valid values: never
always
errors
Default value: **errors**

emaildebug

If on, debug output is printed at each step of the EMail process. This value is not persistent and is reset to its default every time Maestro starts.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

emailfrom Full e-mail From address

Valid values: text strings
Default value:

emailnotification

Upon completion of the backup attempt whether or not to send an e-mail notification indicating the status. Only sending email via an smtp server email is supported.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

emailpassword

If sending email needs a password, specify it here.

Valid values: text strings

Default value:

emailport Port to use for SMTP EMail Server

Valid values: integers
Default value: **587**

emailsecurity

Type of security to use for EMail

Valid values: none
starttls
ssltls
Default value: **none**

emailsmtpserver

Full name of the smtp server

Valid values: text strings
Default value:

emailto Full e-mail address to which EMail notifications will be sent

Valid values: text strings
Default value:

limitbackups

If false, the number of backups is unlimited and the maxbackups option is ignored. If enabled, then maxbackups is honored.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

maxbackups

Specifies maximium number of backups allowed for projects. If limitbackups is enabled and this number is exceeded when a new backup is made, then the oldest backups will be deleted to stay within this limit. This option replaces the 'prefer' command's projectbackuplimit option.

Valid values: integers
Default value: **3**
Minimum: 1

time When automatic backup is enabled start the backup some time at this time (may not trigger exactly at this time because it's on a timer but will happen within the hour of the specified time). Time is in 24-hour format.

Valid values: integers
Default value: **2**
Minimum: 0
Maximum: 23

projectcheck

Check a project for consistency. Currently, only prjzip's are supported. Further, it has to be a project saved with the 'medium' (Fast close) projectsave setting, i.e. one which has a .mmproj-admin directory. See the prefer command's projectsave option for further information.

Syntax:

`projectcheck` *checklevel*=none | exists | simple | requiredfiles | count | filesandoffsets ⟨ project path ⟩

Options:

checklevel Specifies how much to check the project Used internally by Maestro to check a backup. Can be used by user, but use values of 3 and greater carefully. 0: no checking, 1: see if prjzip exists, 2: check prjzip to make sure key key files and directories are present, 3: Does 1 and 2 but also checks to make sure that the number of entries and properties matches the currently open project. This is fairly quick and does not require unzipping the full project. 4: Does 1-3 and also ensures that for each entry there is a corresponding structure file and that the starting offset can be reached (is within bounds). This requires unzipping the whole prjzip. So you may need a lot of disk and it may be slow. Note that 1, 2 and 4 can be used on any project that is not open in Maestro. 3, however, uses Maestro's currently opened project to compare against. So make sure that when using 3 the currently opened project and the one you name are correctly paired! This option is persistent across Maestro sessions.

Valid values:	none exists simple requiredfiles count filesandoffsets
Default value:	**simple**

Operands:

⟨ project path ⟩

The project which we are checking. Currently only prjzip files are supported.

projectclose

Close the current project and open a new scratch project.

Syntax:

`projectclose`

projectcopy

Make a copy of the current project.

Syntax:

`projectcopy` *combiglide*=yes | no *entry*=yes | no *job*=yes | no *oldformat*=yes | no *phase*=yes | no *plot*=yes | no *prime*=yes | no *prjzip*=yes | no *run*=yes | no *scenes*=yes | no *snapshot*=yes | no *source*=all | selected | included *table*=yes | no *user*=yes | no ⟨to_dir_path⟩

Options:

combiglide Enable/disable copying of project combiglide data.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entry Enable/disable copying of project entry files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

job Enable/disable copying of project job files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

oldformat Enable/disable copying of project in old format. For any Maestro version, the old format will be the immediate previous version. Say for Suite2011 Maestro, if this option is set to true, then the project will be copied in Suite2011 format.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

phase Enable/disable copying of project phase data.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

plot Enable/disable copying of project plots. If this option is enabled, entries should also be copied.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

prime Enable/disable copying of project prime data.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

prjzip If on, create a .prjzip. If off, create a .prj.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

run Enable/disable copying of project run files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

scenes Enable/disable copying of scenes.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

snapshot Enable/disable copying of project saved state.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

source Selects the source of entries.
Valid values: all
selected
included
Default value: **all**

table Enable/disable copying of project table files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

user Enable/disable copying of project user files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ to_dir_path ⟩
The path (name and location) of the directory to be used to copy the current project. This can either be a new or existing project directory.

projectdelete

Delete the current project and open a new scratch project.

Syntax:

```
projectdelete
```

projectmerge

Merge data from another project into the current Maestro project.

Syntax:

`projectmerge` *entry*=yes | no *job*=yes | no *run*=yes | no
user=yes | no ⟨from_dir_path⟩

Options:

entry Enable/disable merging of project entry files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

job Enable/disable merging of project job files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

run Enable/disable merging of project run files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

user Enable/disable merging of project user files.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨from_dir_path⟩
The path (name and location) of the project directory to be merged into the current project.

projectnew

Create a new project and open it in Maestro.

Syntax:

`projectnew` ⟨ dir_path ⟩

Operands:
⟨ dir_path ⟩
The path (name and location) of the directory to be created as the new project directory.

projectopen

Open an existing project into Maestro.

Syntax:

`projectopen` ⟨ dir_path ⟩

Operands:
⟨ dir_path ⟩
The path (name and location) of the project directory to be opened.

projectprefer

Set project-specific (rather than user-specific) preferences.

Syntax:

`projectprefer` *2dstructureheight*=⟨ n ⟩ *2dstructurewidth*=⟨ n ⟩
entryfeedbackshow=yes | no *entryfeedbackshownames*=yes | no
feedbackproperties=⟨ text ⟩ *feedbackproperty*=⟨ text ⟩
feedbackshow=yes | no *jobstatusfeedbackshow*=yes | no
phasefeedbackproperties=⟨ text ⟩ *show2dstructures*=yes | no

Options:

2dstructureheight
The height of the 2D structure in the table.

Valid values:	integers
Default value:	**200**
Minimum:	16
Maximum:	1000

2dstructurewidth
The width of the 2D structure in the table.

Valid values: integers
Default value: **300**
Minimum: 1
Maximum: 5000

entryfeedbackshow
Whether the workspace entry feedback is activated.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

entryfeedbackshownames
Whether the workspace entry feedback includes the name of the property along with its value.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

feedbackproperties
These are the entry properties displayed for a single entry in workspace. The property names should be m2io data names with no spaces, separated by spaces in the option string.
Valid values: text strings
Default value: **s_m_title s_pdb_PDB_ID**

feedbackproperty
This is the entry property displayed at the end of the atom feedback string in workspace.
Valid values: text strings
Default value: **Title**

feedbackshow
Whether the workspace feedback is activated.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

jobstatusfeedbackshow
Whether the workspace job status feedback is activated.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

phasefeedbackproperties
These are the Phase properties displayed for a single Phase ligand in the workspace. The property names should be m2io data names with no spaces, separated by spaces in the option string.
Valid values: text strings
Default value:

show2dstructures
Whether the 2D structuresare shown in the project table.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

projectpublish

Publish this project to an html file with links to files in a corresponding subdirectory. The files include among other things a .prjzip version of the current project. The html and corresponding directory may then be used to post this project for use on the web.

Syntax:

`projectpublish` *annotation*=yes | no *publishsummary*=⟨ text ⟩
⟨ html_file ⟩

Options:

annotation

Enable/disable inclusion of the project annotation. This text appears in the Annotation section of the Project Summary panel.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

publishsummary

Summary to be included in the published project. The text to use as the project summary. If you save successive projects, you must update this string as the current value will be written the the project you are publishing.

Valid values: text strings
Default value:

Operands:

⟨ html_file ⟩

The path (name and location) of the html file. A corresponding subdirectory using the base name of the html file will be crated.

projectrename

Rename the current project directory and/or move it to a new location.

Syntax:

`projectrename` ⟨to_dir_path⟩

Operands:

⟨to_dir_path⟩

The new path (name and location) for the current project directory.

projectrestore

Replace specified project with a backup created earlier for that project. Does not open the specified project. If the specified project is the current project, it will be closed first. This command is not undoable. This command expects the project to be specified as a .prj directory, not as a .prjzip or .prj.zip file.

Syntax:

`projectrestore` ⟨dir_path⟩ ⟨backup_name⟩

Operands:

⟨dir_path⟩ ⟨backup_name⟩

The path (name and location) of the project .prj directory to be replaced. Both absolute and relative paths are accepted. The name of the backup project file that will be used to replace the project.

projectrevertopen

Revert state of current project to that it had when opened in Maestro.

Syntax:

`projectrevertopen`

projectrevertsnapshot

Revert state of current project to that previously stored. Deprecated in favor of projectrevertstate.

Syntax:

`projectrevertsnapshot`

projectrevertstate

Revert state of current project to that previously stored.

Syntax:

`projectrevertstate`

projectsaveas

Save project and place user into that project

Syntax:

`projectsaveas` ⟨to_dir_path⟩

Operands:

⟨to_dir_path⟩

Name to which the project will be saved. Saves all data from the current project (table, plots, etc.)

projectsceneclose

Close the open scene and return to the master project.

Syntax:

`projectsceneclose`

projectscenedelete

Deletes the selected scenes.

Syntax:

`projectscenedelete`

projectscenedescription

Sets the description for the given row in the scenes table.

Syntax:

`projectscenedescription` *description*=⟨ text ⟩ ⟨ row ⟩

Options:

description
This is the text to set as the description for the given row.
Valid values: text strings
Default value:

Operands:
⟨ row ⟩
The row number of the scene to set the description for.

projectscenedragrows

Drags the selected rows in the scene table to the target row.

Syntax:

`projectscenedragrows` ⟨ target row ⟩

Operands:
⟨ target row ⟩
The index (row number) to drop the selected rows.

projectsceneexportpresentation

Exports the selected scenes as a presentation (project).

Syntax:

`projectsceneexportpresentation` ⟨ project path ⟩

Operands:

⟨ project path ⟩

Project path is the full path to the new presentation, which is stored as a project.

projectsceneexportproject

Exports the current scene as a project.

Syntax:

`projectsceneexportproject` ⟨ project path ⟩

Operands:

⟨ project path ⟩

Project path is the full path to the new project.

projectsceneextendselect

Extends the selection to the given row in the Scenes table.

Syntax:

`projectsceneextendselect` ⟨ row ⟩

Operands:

⟨ row ⟩

The row number of the scene to select.

projectscenenew

Create a new scene.

Syntax:

projectscenenew *description*=⟨ text ⟩ *source*=all | selected | included

Options:

description
A description for the new scene
Valid values: text strings
Default value:

source Selects the source of entries.
Valid values: all
selected
included
Default value: **included**

projectsceneoptions

Holds options for project scenes.

Syntax:

projectsceneoptions *thumbnailheight*=⟨ n ⟩ *thumbnailwidth*=⟨ n ⟩ *warnbeforeclosing*=yes | no

Options:

thumbnailheight
The height for saving thumbnails.
Valid values: integers
Default value: **100**

thumbnailwidth
The width for saving thumbnails.
Valid values: integers
Default value: **100**

warnbeforeclosing
If set, Maestro will put up a warning message before closing a modified scene.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

projectscenereplace

Replaces the current scene with the current Workspace and project contents.

Syntax:

`projectscenereplace`

projectsceneselect

Selects only the given row in the Scenes table.

Syntax:

`projectsceneselect` ⟨row⟩

Operands:
⟨row⟩
The row number of the scene to select.

projectscenetoggleselect

Toggle-selects the given row in the Scenes table.

Syntax:

`projectscenetoggleselect` ⟨row⟩

Operands:
⟨row⟩
The row number of the scene to select.

projectsceneview

View a scene.

Syntax:

`projectsceneview` ⟨scene index⟩

Operands:
⟨scene index⟩
The index (row number) of the scene to view.

projectsceneviewnamed

View a specific scene in the table.

Syntax:

`projectsceneviewnamed` ⟨scene⟩

Operands:
⟨scene⟩
Scene can either be: first, last, previous, or next to view the appropriate scene.

projectstoresnapshot

Save copy of current project state for reversion. Deprecated in favor of projectstorestate.

Syntax:

`projectstoresnapshot`

projectstorestate

Save copy of current project state for reversion.

Syntax:

`projectstorestate`

projectsynchronize

Save changes in workspace to current project.

Syntax:

```
projectsynchronize
```

projecttablefind

Searches for a string in the project table and make that cell editable.

Syntax:

projecttablefind *direction*=up | down *findtext*=⟨ text ⟩
matchcase=yes | no *matchword*=yes | no
searchhiddenrows=yes | no *selectmatchentries*=yes | no
selectproperty=all | selected *selectrow*=all | selected
showtoolbar=yes | no *toolbararea*=top | bottom | none
⟨ find_string ⟩

Options:

direction With this option user can specify the direction of search.
Valid values: up, down
Default value: **down**

findtext Text to find.
Valid values: text strings
Default value:

matchcase With this option user can specify for case sensitive search.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

matchword
With this option user can specify for matching word as a whole.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

searchhiddenrows
With this option user can specify whether to search in collapsed group rows or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

selectmatchentries
With this option user can select entries with matches.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

selectproperty
With this option user can select properties to search

Valid values: all
selected
Default value: **all**

selectrow With this option user can select rows to search

Valid values: all
selected
Default value: **all**

showtoolbar
With this option user can specify for showing the find toolbar or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

toolbararea
With this option user can specify the dock area of the find toolbar.

Valid values: top
bottom
none
Default value: **bottom**

Operands:

⟨ find_string ⟩

Its the string for which the user want to search in the project table.

projecttablereplace

Replace the <findstring> in the project table with the <replacestring> and finds the next <findstring>

Syntax:

`projecttablereplace` ⟨find_string⟩ ⟨replace_string⟩

Operands:
⟨find_string⟩ ⟨replace_string⟩
Its the string to be searched Its the string to be replaced

projecttablereplaceall

Replace all the instances of findstring in the project table with the <replacestring>

Syntax:

`projecttablereplaceall` ⟨find_string⟩ ⟨replace_string⟩

Operands:
⟨find_string⟩ ⟨replace_string⟩
It's the string to be searched It's the string to be replaced

projectupdatecoordinates

Update the coordinates for the included entries in the project. This applies any current transformations in the Workspace to the original coordinates and places the result back into the project, overwriting the original coordinates.

Syntax:

`projectupdatecoordinates`

projectupdateviews

Update any open views on project data.

Syntax:

`projectupdateviews`

propertycalculate

This command calculates the given property for the for the entries that are specified with the ESL.

Syntax:

propertycalculate *buried*=yes | no *gridspacing*=⟨ x ⟩ *hydrophobic*=yes | no *proberadius*=⟨ x ⟩ *propertyname*=⟨ text ⟩ *recalculate*=yes | no *receptor*=⟨ text ⟩ *surfacetype*=vdw | extended | molecular | pbrsp *vdwscaling*=⟨ x ⟩ ⟨ ESL ⟩

Options:

buried This option determines whether to calculate buried surface area.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

gridspacing

This is the grid spacing for generating molecular surface.

Valid values: reals
Default value: **0.6**
Minimum: 0.05

hydrophobic

This option determines whether to calculate hydrophobic surface area.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

proberadius

This is the probe radius for generating extended radius or molecular surface.

Valid values: reals
Default value: **1.4**

propertyname

This option is the name of the property that need to be calculated. The valid values for this option are numatoms , numresidues , nummolecules , molweight , spin , molcharge , secstruct and numheavyatoms , for calculating the properties Number of atoms , Number of residues , Number of molecules , Molecular weight , Spin multiplicity , Molecular charge , Secondary Structure Content and Number of heavy atoms respectively. A new property will be added to the project this <propertyname> and the values will be assigned to those entries which match the ESL.

Valid values: text strings
Default value: **numatoms**

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

receptor The name of the receptor which is used for buried surface area calculation.

Valid values: text strings
Default value:

surfacetype

This is the surface type for generating surface.

Valid values: vdw
extended
molecular
pbrsp
Default value: **extended**

vdwscaling

This is the scaling of VdW radius for generating extended radius or molecular surface.

Valid values: reals
Default value: **1**

Operands:

⟨ ESL ⟩

A valid ESL specification. Calculates specified property for those entries which match the ESL description.

propertycalculatechirality

This command calculates the chiral centers for the entries that are specified with the ESL.

Syntax:

propertycalculatechirality *recalculate*=yes | no ⟨ ESL ⟩

Options:

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which do not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **false**

Operands:
⟨ ESL ⟩
A valid ESL specification. Calculates the chiral centers for those entries which match the ESL description.

propertycalculatemolformula

This command calculates the molecular formula for the entries that are specified with the ESL.

Syntax:

propertycalculatemolformula *maxatoms*=⟨ n ⟩
recalculate=yes | no ⟨ ESL ⟩

Options:

maxatoms This option is used for calculating molecular formula for an entry which has only one molecule. If the entry contains more than the specified maxatom then the molecular formula will not be calculated for that entry.

Valid values: integers
Default value: **100**

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this

option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ ESL ⟩
A valid ESL specification. Calculates molecular formula for those entries which match the ESL description.

propertycalculatepickpka

This command calculates the pickPka atom for the entries that are specified with the ESL.

Syntax:

`propertycalculatepickpka` *atomname*=⟨ text ⟩
recalculate=yes | no ⟨ ESL ⟩

Options:

atomname This option is the name of the atom that is to be set as the pKa atom for the Jaguar calculation.

Valid values: text strings
Default value:

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which does not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ ESL ⟩
A valid ESL specification. Selects the pickPka atom for those entries which match the ESL description.

propertycalculatesubstructs

This command calculates the number of sub-structures for the entries that are specified with the ESL.

Syntax:

propertycalculatesubstructs *name*=⟨ text ⟩
recalculate=yes | no *substructure*=⟨ text ⟩ ⟨ ESL ⟩

Options:

name This option holds the name of substructure. A valid property name is the valid value for this option. A new property of the integer type will be created with this <name>, which holds the count of specified sub-structures in an entry.

Valid values: text strings
Default value:

recalculate The valid values are true and false. With this option the user can specify whether to re-calculate the property data, which has been calculated already. This option will not have any impact on the entries which do not have the property data. If this option value is true, then the property value will be calculated and the new value will be assigned. If the options value is false, then the already existing property values will not be calculated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

substructure

This describes the sub-structure defintion. This value is used for matching substructures in each entry. The count of these sub-structures in each entry is saved as a property in the table.

Valid values: text strings
Default value:

Operands:

⟨ ESL ⟩

A valid ESL specification. Calculates the number of sub-structures for those entries which match the ESL description.

propertyclearvalue

This command clear values of the given properties

Syntax:

`propertyclearvalue` *allentries*=false | true ⟨ propertynames ⟩

Options:

allentries The property value to be cleared for selected entries or all the entries. This has two valid values, true for clearing all property values and false for clearing property values for the selected entries only.

Valid values: false
true
Default value: **false**

Operands:
⟨ propertynames ⟩
The names of properties for which values need to be cleared.

propertycreate

Create a new property for entries in the current project without assigning any entry property values.

Syntax:

`propertycreate` *author*=⟨ text ⟩ *displayprecision*=⟨ n ⟩ *type*=bool | int | double | string ⟨ property_name ⟩

Options:

author The m2io signature of the owner (e.g. person, group, or software module), or authority, that defines the property (meaning and range of values) and generally assigns its values. The author for user-defined properties should normally be user , as these properties can be freely edited in the project table. Be careful if setting author to m , mmod , i , j , qp , sd , or other names reserved for properties generated by existing programs.

Valid values: text strings
Default value: **user**

displayprecision
The Display precision for the real data type variable for display use only.

Valid values: integers

Default value: **4**
Minimum: 0
Maximum: 15

type The manner in which the property values are to be stored and represented, either bool (for Boolean), int (for integer), double (for double precision floating point, real numbers), or string (for text character strings).

Valid values: bool
int
double
string
Default value: **int**

Operands:

⟨ property_name ⟩

The user-assigned name of the property, which propertyrename can alter. The property name must be unique within the current project. If it is, a unique m2io data name will be generated by combining the type, author, and name, possibly modified (such as replacing spaces with underscores and adding a number at the end) to make it valid and unique.

propertydelete

This command deletes the given properties.

Syntax:

`propertydelete` ⟨ propertynames ⟩

Operands:

⟨ propertynames ⟩

The names of properties to delete.

Aliases:

deleteproperty (see [deleteproperty], page 109)

propertygeneratecontacts

This command applies a Contacts measurement as a property to a number of selected entries in a project. It is valid at least Contactset1 has been set.

If onle Contactset1 is defined, then Contactset2 = Contactset1. Normally these two sets are required having been set before this command, otherwise no property will be generated.

Syntax:

`propertygeneratecontacts`

propertygeneratehbond

This command applies an HBond measurement as a property to a number of selected entries in a project. It is valid at least HBondset1 has been set. If onle HBondset1 is defined, then HBondset2 = HBondset1. Normally these two sets are required having been set before this command, otherwise no property will be generated.

Syntax:

`propertygeneratehbond`

propertymeasurementsetting

This command sets whether a measurement is applied as a property to a number of selected entries in a project.

Syntax:

`propertymeasurementsetting` *applytoselectedentries*=yes | no

Options:

applytoselectedentries

This option determines whether to apply a measurement as a property to a number of selected entries in a project.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

propertymove

Moves the property in the first column to the second column.

Syntax:

propertymove ⟨ from ⟩ ⟨ to ⟩ ⟨ table ⟩

Operands:

⟨ from ⟩ ⟨ to ⟩ ⟨ table ⟩

The source column number. The destination column number. The name of the table to use as the source for the column numbers. If the table operand is missing, the current or default table (1) will be used.

propertyprecision

This command sets the precision of the property.

Syntax:

propertyprecision ⟨ property_name ⟩ ⟨ precision ⟩

Operands:

⟨ property_name ⟩ ⟨ precision ⟩

The name of the property. Precision for that property.

propertyrename

This command renames the given property to the given name.

Syntax:

propertyrename ⟨ from ⟩ ⟨ to ⟩

Operands:

⟨ from ⟩ ⟨ to ⟩

The name of the property to rename. The name to rename the property to.

Aliases:

renameproperty (see [renameproperty], page 670)

propertyshowall

This command creates a property subset consisting of all the properties in the project.

Syntax:

```
propertyshowall
```

propertysuperimposesetting

This command sets whether a superimposition is applied as a property to any entry in the project; and if applied, to which entries it is applied i.e. to the selected entries or to the included entries in the project.

Syntax:

`propertysuperimposesetting` *applytoentries*=included | selected
applytoincludedentries=yes | no *createproperty*=yes | no

Options:

applytoentries

This option determines to which entries the superimposition is applied as a property. Valid values are "selected", or "included".

Valid values: included
selected
Default value: **included**

applytoincludedentries

This option determines whether to apply a superimposition as a property to a number of included entries in a project. NOTE: This option is deprecated now onwards and is suppored just for backward compatibility. Internally the option will be interpreted as: true => applytoentries = included createproperty = true false => createproperty = false

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

createproperty

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

protassign

This keyword is used to set various options associated with running protassign jobs.

Syntax:

`protassign` *input_file*=⟨ text ⟩ *structure_source*=selected_entries | workspace | file

Options:

input_file The name of the structure input file.

Valid values: text strings
Default value:

structure_source
Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected_entries
workspace
file
Default value: **workspace**

protassignresidues

Defines a set of atoms for protein assignment.

Syntax:

`protassignresidues` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms that will be used to define the residues for protein assignment.

protassignstart

Start a protein assignment job with the current settings.

Syntax:

`protassignstart`

protassignwrite

Write a protassign input file with the current settings.

Syntax:

`protassignwrite`

proteinsculpting

Define an atom for a protein sculpting operation and/or translate it by X, Y, Z angstroms

Syntax:

`proteinsculpting` ⟨ atom ⟩ [⟨ X ⟩ ⟨ Y ⟩ ⟨ Z ⟩]

Operands:
⟨ atom ⟩ [⟨ X ⟩ ⟨ Y ⟩ ⟨ Z ⟩]
An atom number to be transformed and the X, Y and Z extents for transformation.

psp_helical

Set residues to define a Prime refinement helical constraint.

Syntax:

`psp_helical` *constrain*=yes | no *helixend*=⟨ text ⟩
helixstart=⟨ text ⟩ *loopend*=⟨ text ⟩ *loopresidues*=⟨ text ⟩
loopstart=⟨ text ⟩

Options:

constrain Whether or not to use the constraint in loop prediction jobs.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **false**

helixend The residue where the helix ends.
Valid values: text strings
Default value:

helixstart The residue where the helix starts.
Valid values: text strings
Default value:

loopend The residue where the loop ends.
Valid values: text strings
Default value:

loopresidues
All residue of the loop.
Valid values: text strings
Default value:

loopstart The residue where the loop starts.
Valid values: text strings
Default value:

psp_pairwise

Specifies a pair of atoms to define a Prime refinement pairwise constraint.

Syntax:

`psp_pairwise` *distance*=⟨ x ⟩ *forceconstant*=⟨ n ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

distance The distance from atom1 to atom2 of the pairwise constraint.
Valid values: reals
Default value: **0**
Minimum: 0.0

forceconstant
The force constant for the pairwise constraint.
Valid values: integers
Default value: **350**
Minimum: 0

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩

The two atoms that define a Prime refinement pairwise constraint.

psp_spatial

Specifies a residue to define a Prime refinement spatial constraint.

Syntax:

`psp_spatial` *caatom*=⟨ n ⟩ *constrain*=yes | no *distance*=⟨ x ⟩
xcoord=⟨ x ⟩ *xcoordold*=⟨ x ⟩ *ycoord*=⟨ x ⟩ *ycoordold*=⟨ x ⟩
zcoord=⟨ x ⟩ *zcoordold*=⟨ x ⟩
⟨ chain_name ⟩:⟨ residue_number ⟩⟨ insertion_code ⟩

Options:

caatom The atom number of C-alpha atom of the residue.

Valid values: integers
Default value: **0**
Minimum: 0

constrain Whether or not to use the constraint in loop prediction jobs.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

distance The radius of sphere centered at the C-alpha atom of the residue.

Valid values: reals
Default value: **3**
Minimum: 0.0

xcoord The X coordinate of the C-alpha atom of the residue.

Valid values: reals
Default value: **0**

xcoordold The X coordoldinate of the C-alpha atom of the residue.

Valid values: reals
Default value: **0**

ycoord The Y coordinate of the C-alpha atom of the residue.

Valid values: reals
Default value: **0**

ycoordold The Y coordoldinate of the C-alpha atom of the residue.

Valid values: reals
Default value: **0**

zcoord The Z coordinate of the C-alpha atom of the residue.
Valid values: reals
Default value: **0**

zcoordold The Z coordoldinate of the C-alpha atom of the residue.
Valid values: reals
Default value: **0**

Operands:
⟨ chain_name ⟩:⟨ residue_number ⟩⟨ insertion_code ⟩
The residue that define a Prime refinement spatial constraint.

pspalignaddanchor

Adds an anchor at the given position for Edit Alignment.

Syntax:

`pspalignaddanchor` ⟨ anchor position ⟩

Operands:
⟨ anchor position ⟩
The position at which to add an anchor.

pspalignndeleteanchor

Deletes the anchor at the given position for Edit Alignment.

Syntax:

`pspalignndeleteanchor` ⟨ anchor position ⟩

Operands:
⟨ anchor position ⟩
The position at which to delete an anchor.

pspaligninsertgaps

Inserts gaps into the sequence at the given position.

Syntax:

`pspaligninsertgaps` *number*=⟨ n ⟩ *position*=⟨ n ⟩ ⟨ sequence name ⟩

Options:

number This is the number of gaps to insert.
Valid values: integers
Default value: **1**
Minimum: 1

position This is the position at which to insert a gap.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ sequence name ⟩
The name of the sequence to insert gaps into.

pspalignlockgaps

Locks gaps for all alignments in the Edit Alignment step.

Syntax:

`pspalignlockgaps`

pspalignmoveleft

Moves the alignment left freely starting at the given position. This will shift gaps from the left of the given position to the right of the given position.

Syntax:

`pspalignmoveleft` *number*=⟨ n ⟩ *position*=⟨ n ⟩ ⟨ sequence name ⟩

Options:

number This is the number of spaces to move left.
Valid values: integers
Default value: **1**
Minimum: 1

position This is the position at which to move left.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ sequence name ⟩
The name of the sequence to move left.

pspalignmoveleftblock

Moves the alignment left as a block starting at the given position. This will close up gaps to the left of the given position while preserving them to the right.

Syntax:

`pspalignmoveleftblock` *number*=⟨ n ⟩ *position*=⟨ n ⟩
⟨ sequence name ⟩

Options:

number This is the number of spaces to move left.
Valid values: integers
Default value: **1**
Minimum: 1

position This is the position at which to move left.
Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ sequence name ⟩
The name of the sequence to move left.

pspalignmoveright

Moves the alignment right freely starting at the given position. This will shift gaps from the right of the given position to the left of the given position. If necessary, this will open up gaps to the left of the given position.

Syntax:

`pspalignmoveright` *number*=⟨ n ⟩ *position*=⟨ n ⟩ ⟨ sequence name ⟩

Options:

number This is the number of spaces to move right.

Valid values:	integers
Default value:	**1**
Minimum:	1

position This is the position at which to move right.

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:

⟨ sequence name ⟩

The name of the sequence to move right.

pspalignstructures

Runs 'structalign' to align the selected template structures in the Find Homologs step.

Syntax:

`pspalignstructures`

pspalignunlockgaps

Unloocks gaps for all alignments in the Edit Alignment step.

Syntax:

`pspalignunlockgaps`

pspbstogglehetatom

Toggles the given hetatom between included and excluded.

Syntax:

`pspbstogglehetatom` ⟨hetatom name⟩

Operands:
⟨hetatom name⟩
The name of the hetatom to toggle.

pspbuildbackbone

Runs the build backbone backend.

Syntax:

`pspbuildbackbone`

pspbuildconsensusmodel

Runs the consensus homology modeling backend.

Syntax:

`pspbuildconsensusmodel`

pspbuildheteromultimer

Runs the hetero-multimer building command.

Syntax:

`pspbuildheteromultimer`

pspbuildhomomultimer

Runs the build structure backend to build a homo-multimer model.

Syntax:

`pspbuildhomomultimer`

pspbuildstructure

Runs the build structure backend.

Syntax:

`pspbuildstructure`

pspconstraintmarkersettings

Set graphical data of Phase excluded volume markers.

Syntax:

`pspconstraintmarkersettings` *helicalblue*=⟨ x ⟩ *helicalgreen*=⟨ x ⟩ *helicalred*=⟨ x ⟩ *pairwiseblue*=⟨ x ⟩ *pairwisegreen*=⟨ x ⟩ *pairwisehighlightwidth*=⟨ x ⟩ *pairwisered*=⟨ x ⟩ *spatialambient*=⟨ x ⟩ *spatialblue*=⟨ x ⟩ *spatialdiffuse*=⟨ x ⟩ *spatialemission*=⟨ x ⟩ *spatialgreen*=⟨ x ⟩ *spatialhighlightblue*=⟨ x ⟩ *spatialhighlightgreen*=⟨ x ⟩ *spatialhighlightred*=⟨ x ⟩ *spatiallineblue*=⟨ x ⟩ *spatiallinegreen*=⟨ x ⟩ *spatiallinered*=⟨ x ⟩ *spatiallinewidth*=⟨ x ⟩ *spatialred*=⟨ x ⟩ *spatialshininess*=⟨ x ⟩ *spatialslices*=⟨ n ⟩ *spatialspecular*=⟨ x ⟩ *spatialstacks*=⟨ n ⟩ *spatialtransparency*=⟨ x ⟩

Options:

helicalblue The blue color component of helical markers.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

helicalgreen

The green color component of helical markers.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

helicalred The red color component of helical markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

pairwiseblue
The blue color component of pairwise markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

pairwisegreen
The green color component of pairwise markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

pairwisehighlightwidth
The highlight line width of pairwise markers.

Valid values: reals
Default value: **1**
Minimum: 0.0

pairwisered
The red color component of pairwise markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

spatialambient
Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

spatialblue The blue color component of spatial markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

spatialdiffuse

Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	1.0

spatialemission

Set material property - emission, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0
Maximum:	1.0

spatialgreen

The green color component of spatial markers.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

spatialhighlightblue

The blue color component of spatial markers if the spatial is highlighted.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

spatialhighlightgreen

The green color component of spatial markers if the spatial is highlighted.

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

spatialhighlightred

The red color component of spatial markers if the spatial is highlighted.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

spatiallineblue

The blue color component of spatial line markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

spatiallinegreen

The green color component of spatial line markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

spatiallinered

The red color component of spatial line markers.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

spatiallinewidth

The red color component of spatial line markers.

Valid values: reals
Default value: **1**
Minimum: 0.0

spatialred The red color component of spatial markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

spatialshininess

Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

spatialslices

Set the slices of drawing sphere.

Valid values: integers
Default value: **18**
Minimum: 2

spatialspecular
: Set material property - specular, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.0
Maximum:	1.0

spatialstacks
: Set the stacks of drawing sphere.

Valid values:	integers
Default value:	**9**
Minimum:	2

spatialtransparency
: The transparency of QSAR markers.

Valid values:	reals
Default value:	**20**
Minimum:	0.0
Maximum:	100.0

pspeditquerysequence

Edits query sequence in Input Sequence step.

Syntax:

`pspeditquerysequence`

pspexcludetable1row

Excludes a composite structure from the table shown in the Build Backbone step. Excludes a structure, for the specified row in the input table shown in the current Structure Prediction step, from the Sequence Viewer (and Workspace). This applies to the Refine Backbone step.

Syntax:

`pspexcludetable1row` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩

The row number in the table which is to be excluded.

pspexcludetablerow

Excludes a composite structure from the table shown in the Build Backbone step. Excludes a structure, for the specified row in the table shown in the current Structure Prediction step, from the Sequence Viewer (and Workspace). This applies to the Fold Recognition, Build Backbone, Refine Backbone, and Refine Structure steps (which have structure or template tables, and the ability to specify rows in Workspace independently from selected rows).

Syntax:

`pspexcludetablerow` ⟨ row_number ⟩

Operands:

⟨ row_number ⟩

The row number in the table which is to be excluded.

pspexportalignment

exports alignments selected into a file.

Syntax:

`pspexportalignment` ⟨ filename ⟩

Operands:

⟨ filename ⟩

The name of the file to which the alignments are to be written.

pspfindfamily

Finds the family for the query sequence currently active in the sequence viewer.

Syntax:

`pspfindfamily`

pspfindhomologs

Runs Blast to find homologs for the query sequence currently active in the sequence viewer.

Syntax:

`pspfindhomologs`

pspfoldrecognitionoptions

Sets some options associated with the Fold Recognition step.

Syntax:

`pspfoldrecognitionoptions` *max_results*=⟨ n ⟩

Options:

max_results

Specifies how many results will be shown in the fold recognition results table.

Valid values: integers
Default value: **100**

pspfoldrecognitionsearch

Runs the fold recognition search on the current query sequence.

Syntax:

`pspfoldrecognitionsearch`

pspgetquerysequence

Creates new query sequence from text in Input Sequence step.

Syntax:

`pspgetquerysequence`

pspimportalignment

imports alignments in a file and applies to the templates. since there can be more than one alignment stored in a file, the command gives messages on which templates have been altered.

Syntax:

`pspimportalignment` ⟨ filename ⟩

Operands:

⟨ filename ⟩

The name of the file from which the alignments are to be read.

pspimporthomolog

Imports the sequences from a file as homologs to the current query sequence.

Syntax:

`pspimporthomolog` ⟨ filename ⟩

Operands:

⟨ filename ⟩

The name of the file from which the sequences are to be read.

pspimportssp

Adds the secondary structure prediction from a file to those for the current query sequence.

Syntax:

`pspimportssp` ⟨ filename ⟩

Operands:
⟨ filename ⟩
The name of the file from which the SSP is to be read.

pspincludetable1row

Includes a structure from the input table shown in the current Structure Prediction step into the Sequence Viewer (and Workspace), independent of selection. This applies to the Refine Backbone step.

Syntax:

`pspincludetable1row` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be included.

pspincludetablerow

Includes a structure from the table shown in the current Structure Prediction step into the Sequence Viewer (and Workspace). This applies to the Fold Recognition, Build Backbone, Refine Backbone, and Refine Structure steps (which have structure or template tables, and the ability to specify rows in Workspace independently from selected rows).

Syntax:

`pspincludetablerow` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be included.

pspminimizationresidues

Defines a set of atoms for minimization refinement.

Syntax:

`pspminimizationresidues` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms that will be used to define the residues for minimization refinement.

pspoptimizealignment

optimizes alignment of a template sequence.

Syntax:

`pspoptimizealignment` ⟨ rowindex ⟩

Operands:
⟨ rowindex ⟩
The row index of the sequence which is aligned

pspquickbuild

Runs the build structure backend with fixed options.

Syntax:

`pspquickbuild`

pspquickhomomultimer

Runs the build homo-multimer backend with fixed options.

Syntax:

`pspquickhomomultimer`

psprefinebackbone

Runs the refine backbone backend.

Syntax:

`psprefinebackbone`

psprefinestructure

Runs the structure refinement program with the currently set refinement options on the currently selected structure.

Syntax:

`psprefinestructure`

psprsdefaulthelixloops

Rebuilds the Helices & Loops table for structure refinement with the default loop and helix refinement options for the currently selected structure.

Syntax:

`psprsdefaulthelixloops`

psprshelixlooptogglerefine

Toggle refinement of loop or helix.

Syntax:

`psprshelixlooptogglerefine` ⟨ row_index ⟩

Operands:
⟨ row_index ⟩
The row index for the loop or helix in the Helices & Loops table that is to be toggled for refinement.

psprshelixlooptogglerefineonly

Toggle refinement of loop or helix.

Syntax:

psprshelixlooptogglerefineonly ⟨ row_index ⟩

Operands:

⟨ row_index ⟩

The row index for the loop or helix in the Helices & Loops table that is the only one to be toggled for refinement.

psprshelixoptions

Specifies the refinement options for the currently selected helix in the Helices & Loops table.

Syntax:

psprshelixoptions *roll_range*=⟨ x ⟩ *roll_resolution*=⟨ x ⟩
set_roll=yes | no *set_shift*=yes | no *set_tilt*=yes | no
set_trans=yes | no *shift_range*=⟨ x ⟩ *shift_resolution*=⟨ x ⟩
tilt_range=⟨ x ⟩ *tilt_resolution*=⟨ x ⟩ *trans_range*=⟨ x ⟩
trans_resolution=⟨ x ⟩

Options:

roll_range Range, in degrees, for helix roll.

Valid values: reals
Default value: **20**

roll_resolution

Resolution, in degrees, for helix roll.

Valid values: reals
Default value: **20**

set_roll Set the helix roll range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

set_shift Set the helix shift range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

set_tilt Set the helix tilt range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

set_trans Set the helix translation range and resolution.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

shift_range
Range, in angstroms, for helix shift.

Valid values: reals
Default value: **2**

shift_resolution
Resolution, in angstroms, for helix shift.

Valid values: reals
Default value: **2**

tilt_range Range, in degrees, for helix tilt.

Valid values: reals
Default value: **5**

tilt_resolution
Resolution, in degrees, for helix tilt.

Valid values: reals
Default value: **5**

trans_range
Range, in angstroms, for helix translation.

Valid values: reals
Default value: **2**

trans_resolution
Resolution, in angstroms, for helix translation.

Valid values: reals
Default value: **2**

psprsloopoptions

Specifies the refinement options for the currently selected loop in the Helices & Loops table.

Syntax:

`psprsloopoptions` *ca_max*=⟨ x ⟩ *membrane*=none | inside | outside *overlap_min*=⟨ x ⟩ *restrict_ca*=yes | no *sphere_size*=⟨ x ⟩ *use_sphere*=yes | no

Options:

ca_max Maximum CA atom movement, in angstroms, from initial.

Valid values: reals
Default value: **3**

membrane Membrane constraint type for loop building

Valid values: none
inside
outside
Default value: **none**

overlap_min
Minimum overlap, in angstroms, from initial.

Valid values: reals
Default value: **0.7**
Minimum: 0.1
Maximum: 1.0

restrict_ca Restrict loop CA atom movement.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sphere_size
Distance from loop for sidechain inclusion, in angstroms.

Valid values: reals
Default value: **7.5**

use_sphere
Include nearby sidechains in loop refinement.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

psprsnontemplatehelixloops

Rebuilds the Helices & Loops table for structure refinement with the loops that were not constructed from a template during homology modeling, for the currently selected structure.

Syntax:

`psprsnontemplatehelixloops`

psprunalign

Runs secondary structure prediction (if necessary) and then aligns the query to the active templates in Edit Alignment.

Syntax:

`psprunalign` *gpcr*=yes | no

Options:

gpcr	Specifies whether or not to use GPCR alignment.	
	Valid values:	boolean (true|false; yes|no; y|n; on|off)
	Default value:	**false**

pspruncreate

Creates the run with the given name.

Syntax:

`pspruncreate` ⟨run name⟩

Operands:
⟨run name⟩
The name of the new run to create.

psprundelete

Deletes the current run from the project.

Syntax:

`psprundelete`

psprunopen

Opens the run with the given name.

Syntax:

`psprunopen` ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to open.

psprunquickalign

Runs quick multiple sequence alignment algorithm and to align the query to the active templates in Edit Alignment.

Syntax:

`psprunquickalign`

psprunrename

Changes the current run's name to the given name.

Syntax:

`psprunrename` ⟨run name⟩

Operands:
⟨run name⟩
The name to change the current run's name to.

psprunsaveas

Saves a copy of the current run under the given name.

Syntax:

`psprunsaveas` ⟨ run name ⟩

Operands:

⟨ run name ⟩

The name of the run to save as.

pspsecstructprediction

Runs the available secondary structure prediction programs on the current query sequence.

Syntax:

`pspsecstructprediction`

pspselectextendtable1row

Extend the selection from the selected table row to joing up with an existing selection. This applies to the Refine Backbone step

Syntax:

`pspselectextendtable1row` ⟨ row_number ⟩

Operands:

⟨ row_number ⟩

The row number in the table from which the selection is to begin.

pspselectextendtablerow

Extend the selection from the selected table row to joing up with an existing selection. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

`pspselectextendtablerow` ⟨row_number⟩

Operands:
⟨row_number⟩
The row number in the table from which the selection is to begin.

pspselecthelixlooprow

Select structure for refinement.

Syntax:

`pspselecthelixlooprow` ⟨row_index⟩

Operands:
⟨row_index⟩
The row index for the loop or helix in the Helices & Loops table that is to be selected for refinement option editing.

pspselectonlytable1row

Select a row from the input structure or template table shown in the current Structure Prediction step, deselecting all other rows. This applies to the Refine Backbone step.

Syntax:

`pspselectonlytable1row` ⟨row_number⟩

Operands:
⟨row_number⟩
The row number in the table which is to be selected.

pspselectonlytablerow

Select a row from the structure or template table shown in the current Structure Prediction step, deselecting all other rows. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

pspselectonlytablerow ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be selected.

pspselectrscontext

Specify the context for structure refinement. This command may be executed instead of showing the psp or refinement panels. Be cautious about using this commands while either of these panels is shown, as setting the wrong context will interfere with the execution of the refinement commands issued from the panel.

Syntax:

pspselectrscontext pspstep|standalone

Operands:
pspstep|standalone
The context in which refine structure commands are executed.

pspselectrsrefinement

Selects the current psp structure refinement type.

Syntax:

pspselectrsrefinement ⟨ refine_type ⟩

Operands:
⟨ refine_type ⟩
The type of structure refinement to be performed. Value should be sidechains , helixloops , or minimization .

pspselecttable1row

Selects a row from the input structure or template table shown in the current Structure Prediction step. This applies to the Refine Backbone step.

Syntax:

`pspselecttable1row` ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table which is to be selected.

pspselecttablerow

Selects a row from the structure or template table shown in the current Structure Prediction step. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, Edit Alignment and Refine Structure steps.

Syntax:

`pspselecttablerow` ⟨row_number⟩

Operands:

⟨row_number⟩

The row number in the table which is to be selected.

pspsequenceaddfile

Adds the sequences from the given file to the Select Sequence step in PSP.

Syntax:

`pspsequenceaddfile` ⟨file⟩

Operands:

⟨file⟩

The file name of the sequence file to add.

pspsequenceaddworkspace

Adds the sequences from the Workspace to the Select Sequence step in PSP.

Syntax:

`pspsequenceaddworkspace`

pspsequencecrop

Crops the given sequence to the given residue index, towards the closer end of the sequence.

Syntax:

`pspsequencecrop` *res*=⟨ n ⟩ ⟨ sequence name ⟩ ⟨ residue index ⟩

Options:

res	This is the position to crop to	
	Valid values:	integers
	Default value:	**1**
	Minimum:	1

Operands:
⟨ sequence name ⟩ ⟨ residue index ⟩
The name of the sequence to crop.

pspsequenceselect

Selects the given sequence number as input for the next step in PSP.

Syntax:

`pspsequenceselect` ⟨ sequence number ⟩

Operands:
⟨ sequence number ⟩
The index of the sequence to select.

pspsequenceviewerexport

Exports all of the visible sequences in the sequence viewer to the given file.

Syntax:

pspsequenceviewerexport *sequenceviewer*=⟨ text ⟩ ⟨ file name ⟩

Options:

sequenceviewer
: Specifies whether to save the contents of the Workspace or Prime sequence viewer.
  Valid values: text strings
  Default value: **prime**

Operands:
⟨ file name ⟩
The name of the file to export the sequences to.

pspsethelixloopresidues

Specify residues to define loop or helix feature for structure refinement.

Syntax:

pspsethelixloopresidues ⟨ row_index ⟩ ⟨ R1 ⟩ ⟨ R2 ⟩ ⟨ R3 ⟩ ⟨ R4 ⟩

Operands:
⟨ row_index ⟩ ⟨ R1 ⟩ ⟨ R2 ⟩ ⟨ R3 ⟩ ⟨ R4 ⟩
The row index for the loop or helix in the Helices & Loops table, followed by the strings for the four residue column values, as they are to appear in the table. For a loop, only two end residues (<R1> <R2>) are required, so the <R3> and <R4> values can be missing or left blank. For a helix, <R1> and <R4> specify the outer residues for the loops at the two ends of the helix, while <R2> and <R3> specify the ends of the helix itself. The format for the residue strings is the residue number followed by the insertion code, if any. For example, 12C represent residue number 12, insertion code C.

pspsidechainresidues

Defines a set of atoms for the side chain refinement.

Syntax:

pspsidechainresidues *prediction_algorithm*=default | vectorsampling | backbonesampling ⟨ASL⟩

Options:

prediction_algorithm

Determines which side chain sampling algorithm is used for Prime sidechain prediction jobs.

Valid values:	default vectorsampling backbonesampling
Default value:	**default**

Operands:

⟨ASL⟩

The ASL expression which defines the atoms that will be used to define the residues for side chain refinement.

pspsortbbtable

Resort the Build Backbone table based on the data in the specified column

Syntax:

pspsortbbtable ⟨column_name⟩

Operands:

⟨column_name⟩

The name of the column to be sorted.

pspsortfindhomologstable

Resort the find homologs table based on the data in the specified column

Syntax:

`pspsortfindhomologstable` ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortfoldtable

Resort the fold recognition table based on the data in the specified column

Syntax:

`pspsortfoldtable` ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortrbtable

Resort the Refine Backbone output structure table based on the data in the specified column

Syntax:

`pspsortrbtable` ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortrbtable1

Resort the Refine Backbone composite structure table based on the data in the specified column

Syntax:

`pspsortrbtable1` ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsortrstable

Resort the Refine Structure table based on the data in the specified column

Syntax:

`pspsortrstable` ⟨column_name⟩

Operands:
⟨column_name⟩
The name of the column to be sorted.

pspsspdelete

Deletes the given secondary structure prediction.

Syntax:

`pspsspdelete` ⟨SSP name⟩

Operands:
⟨SSP name⟩
The name of the secondary structure prediction to delete.

pspsspexport

Exports the given secondary structure to the named file.

Syntax:

`pspsspexport` *ssp*=⟨ text ⟩ ⟨ file name ⟩

Options:

ssp This option is the name of the secondary structure prediction to export.

Valid values: text strings
Default value:

Operands:
⟨ file name ⟩
The name of the file to export to.

pspssprevert

Reverts a modified secondary structure prediction back to its original form.

Syntax:

`pspssprevert` ⟨ SSP name ⟩

Operands:
⟨ SSP name ⟩
The name of the secondary structure prediction to revert.

pspsspset

Sets the given range of positions in the secondary structure position to the given code, which must be H , E , or - .

Syntax:

`pspsspset` *code*=⟨ text ⟩ *from*=⟨ n ⟩ *to*=⟨ n ⟩ ⟨ SSP name ⟩

Options:

code This is the code for setting the secondary structure position to. It must be one of H , E , or - .

Valid values: text strings
Default value:

from This is the starting position for setting codes.

Valid values:	integers
Default value:	**1**
Minimum:	1

to This is the ending position for setting codes.

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:

⟨ SSP name ⟩

The name of the step to set data for.

pspstepforward

Moves forward to the next named step. Deletes any steps after the current step and copies data forward to create the specified step.

Syntax:

`pspstepforward` ⟨ step name ⟩

Operands:

⟨ step name ⟩

The name of the step to switch to.

pspstepgoto

Moves to an existing step in the current run.

Syntax:

`pspstepgoto` ⟨ step name ⟩

Operands:

⟨ step name ⟩

The name of the step to switch to.

pspstructureaddentry

Create project entry from selected Prime structure. This operation is only permitted for the Build Structure step.

Syntax:

```
pspstructureaddentry
```

psptemplatesetregion

Sets the region for the given template.

Syntax:

`psptemplatesetregion` *end*=⟨ n ⟩ *start*=⟨ n ⟩ ⟨ template name ⟩

Options:

end This is the position at which the region ends.

Valid values:	integers
Default value:	**1**
Minimum:	1

start This is the position at which to start the region.

Valid values:	integers
Default value:	**1**
Minimum:	1

Operands:

⟨ template name ⟩

The name of the template to set a region for.

psptmplsecstructprediction

Runs the available secondary structure prediction programs on the template sequence identified by row number

Syntax:

`psptmplsecstructprediction` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be selected.

psptranslatespatial

Translate in Angstroms the selected Prime refinement spatial constraint.

Syntax:

`psptranslatespatial` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

x	Amount in Angstroms to translate in X	
	Valid values:	reals
	Default value:	**0**
y	Amount in Angstroms to translate in Y	
	Valid values:	reals
	Default value:	**0**
z	Amount in Angstroms to translate in Z	
	Valid values:	reals
	Default value:	**0**

pspunselecttable1row

Unselects a row from the input table shown in the current Structure Prediction step. This applies to the Refine Backbone step.

Syntax:

`pspunselecttable1row` ⟨ row_number ⟩

Operands:
⟨ row_number ⟩
The row number in the table which is to be unselected.

pspunselecttablerow

Unselects a row from the table shown in the current Structure Prediction step. This applies to the Find Homologs, Fold Recognition, Build Backbone, Refine Backbone, and Edit Alignment steps (which support multiple row selection).

Syntax:

`pspunselecttablerow` ⟨ row_number ⟩

Operands:

⟨ row_number ⟩

The row number in the table which is to be unselected.

pspupdatehetatomlist

Updates list of ligands and cofactors in Build Structure step.

Syntax:

`pspupdatehetatomlist` *showall*=yes | no

Options:

showall Specifies whether or not to list all ligands and waters, or only ligands within 5A from a chain.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pspupdatescores

Updates Scores of all the alignments

Syntax:

`pspupdatescores`

pspwritebuildconsensusmodel

Writes input files for the build structure backend in consensus modeling mode.

Syntax:

```
pspwritebuildconsensusmodel
```

pspwritebuildhomomultimer

Writes input files for the build structure backend in homo-multimer mode.

Syntax:

```
pspwritebuildhomomultimer
```

pspwritebuildstructure

Writes input files for the build structure backend.

Syntax:

```
pspwritebuildstructure
```

pyeval

This is a standard alias for **pythoneval** (see [pythoneval], page 648).

pyimp

This is a standard alias for **pythonimport** (see [pythonimport], page 648).

pyrun

This is a standard alias for **pythonrun** (see [pythonrun], page 648).

pythoneval

Evaluate the python expression using the built-in Python interpreter. KEY_OPERAND_SYNOPOSIS: <python expression>

Syntax:

`pythoneval`
Aliases:
pyeval (see [pyeval], page 647)

pythonimport

Imports the specified python module. If the module has already been imported then it will be reloaded. Under normal usage this command is not needed as pythonrun automatically imports the module specified as its operand. However during development of modules it is useful to be able to reload them using pythonimport KEY_OPERAND_SYNOPOSIS: <module>

Syntax:

`pythonimport`
Aliases:
pyimp (see [pyimp], page 647)

pythonrun

Runs the function in <function name> in module <modulename> KEY_OPERAND_SYNOPOSIS: <function name> [function parameters]

Syntax:

`pythonrun`
Aliases:
pyrun (see [pyrun], page 647)

pythonrunbuiltin

Runs the function in <function name> in module <modulename> KEY_OPERAND_SYNOPOSIS: <function name> [function parameters]

Syntax:

`pythonrunbuiltin`

qikprop

This keyword is used to set various options associated with running QikProp jobs.

Syntax:

`qikprop` *fastmode*=yes | no *input_file*=⟨ text ⟩ *nsim*=⟨ n ⟩ *sim*=yes | no *structure_source*=selected_entries | workspace | file

Options:

fastmode The option to identify the similar molecules

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

input_file The name of the structure input file.

Valid values:	text strings
Default value:	

nsim

Valid values:	integers
Default value:	**5**
Minimum:	1
Maximum:	999

sim The number of similar molecules to identify

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

structure_source

Whether to use the selected entries in the current project or what is in the workspace, or a specified file with multiple structures as input for the job.

Valid values:	selected_entries workspace file
Default value:	**selected_entries**

qsarmarkerdump

Print out the current option values of the QSAR marker command.

Syntax:

`qsarmarkerdump`

qsarmarkersettings

Set graphical data of Phase QSAR markers.

Syntax:

`qsarmarkersettings` *ambient*=⟨ x ⟩ *coefficient_feature*=⟨ text ⟩
combine_effects=yes | no *diffuse*=⟨ x ⟩ *emission*=⟨ x ⟩
negative_blue=⟨ x ⟩ *negative_green*=⟨ x ⟩ *negative_red*=⟨ x ⟩
negativecoefficient=⟨ x ⟩ *numberpls*=⟨ n ⟩ *positive_blue*=⟨ x ⟩
positive_green=⟨ x ⟩ *positive_red*=⟨ x ⟩ *positivecoefficient*=⟨ x ⟩
roundingeffect=⟨ x ⟩ *selectedatomclass*=⟨ text ⟩ *shininess*=⟨ x ⟩
specular=⟨ x ⟩ *step*=⟨ n ⟩ *transparency*=⟨ x ⟩
volumeoccupied=workspacelig | qsarmodel

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.0
Maximum:	1.0

coefficient_feature
Sets which feature the coefficient options apply to. Can also be set to combined .

Valid values:	text strings
Default value:	**combined**

combine_effects
Whether or not to combine the effects from all classes.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

negative_blue
Blue component of negative effects color.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

negative_green
Green component of negative effects color.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

negative_red
Red component negative effects color.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

negativecoefficient
Set the QSAR visualization option of negative coefficient threshold.

Valid values: reals
Default value: **-0.044**
Maximum: 0.0

numberpls Set the QSAR visualization option of number of PLS factors.

Valid values: integers
Default value: **1**
Minimum: 1

positive_blue
Blue component of positive effects color.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

positive_green
Green component of positive effects color.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

positive_red
Red component of positive effects color.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

positivecoefficient
Set the QSAR visualization option of positiv coefficient threshold.

Valid values: reals
Default value: **0.044**
Minimum: 0.0

roundingeffect
This determines the rounding effect of edges of a cell, for front face.

Valid values: reals
Default value: **5**
Minimum: 0.0

selectedatomclass
Set the QSAR visualization option of selected atom class.

Valid values: text strings
Default value: **D**

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals

Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

step The step domain tolerance of cells.

Valid values: integers
Default value: **3**
Minimum: 1

transparency
The transparency of QSAR markers.

Valid values: reals
Default value: **50**
Minimum: 0.0
Maximum: 100.0

volumeoccupied
Set the QSAR visualization option of viewing volume occupied.

Valid values: workspacelig
qsarmodel
Default value: **workspacelig**

qsitebasis

Specifies an atom for QM basis atoms

Syntax:

`qsitebasis` *basis*=⟨ text ⟩ *diaplayname*=⟨ text ⟩ *select*=yes | no ⟨ atom ⟩

Options:

basis The basis set for the atom to be included in the QM basis of an Impact QSite simulation.

Valid values: text strings
Default value: **lacvp***

diaplayname
The display name for the atom to be included in the QM basis of an Impact QSite simulation.

Valid values: text strings
Default value:

select Whether or not the atom is selected.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ atom ⟩

The atom that defines a basis set

qsitehcap

Specifies a pair of atoms to define a hydrogen cap

Syntax:

`qsitehcap` *basis*=⟨ text ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

basis The basis set for the hydrogen cap to be included in the QM region of an Impact QSite simulation.

Valid values: text strings
Default value: **lacvp***

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

The two atoms that define a hydrogen cap

qsitehcapmarkersettings

Set graphical data of QM hydrogen cap markers.

Syntax:

qsitehcapmarkersettings *ambient*=⟨ x ⟩ *blue*=⟨ x ⟩
cornradius=⟨ x ⟩ *cylinderheight*=⟨ x ⟩ *cylinderradius*=⟨ x ⟩
diffuse=⟨ x ⟩ *drawstyle*=solid | line *emission*=⟨ x ⟩ *green*=⟨ x ⟩
linewidth=⟨ x ⟩ *red*=⟨ x ⟩ *selectblue*=⟨ x ⟩ *selectgreen*=⟨ x ⟩
selectred=⟨ x ⟩ *shininess*=⟨ x ⟩ *sliceline*=⟨ n ⟩ *slicesolid*=⟨ n ⟩
specular=⟨ x ⟩ *stackline*=⟨ n ⟩ *stacksolid*=⟨ n ⟩
transparency=⟨ x ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

blue The blue component of attachment markers.

Valid values: reals
Default value: **0.16**
Minimum: 0.0
Maximum: 1.0

cornradius
The radius of corn of attachment markers.

Valid values: reals
Default value: **0.55**
Minimum: 0.0

cylinderheight
The cylinder height ratio of attachment markers.

Valid values: reals
Default value: **0.6**
Minimum: 0.0
Maximum: 1.0

cylinderradius
The radius of cylinder of attachment markers.

Valid values: reals
Default value: **0.26**
Minimum: 0.0

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0

Maximum: 1.0

drawstyle The styles of rendering attachment markers, they are: 1 - solid, and 2 - lines. Default is solid.

Valid values: solid
line
Default value: **solid**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 1.0

green The green component of attachment markers.

Valid values: reals
Default value: **0.16**
Minimum: 0.0
Maximum: 1.0

linewidth Set the width of lines in drawing attachment.

Valid values: reals
Default value: **1.5**
Minimum: 0.0001

red The red component of attachment markers.

Valid values: reals
Default value: **0.65**
Minimum: 0.0
Maximum: 1.0

selectblue The blue component of selected attachment markers.

Valid values: reals
Default value: **1**
Minimum: 0.0
Maximum: 1.0

selectgreen

The green component of selected attachment markers.

Valid values: reals
Default value: **0.9**
Minimum: 0.0
Maximum: 1.0

selectred The red component of selected attachment markers.

Valid values: reals

Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

sliceline Set the slices of drawing line attachment.

Valid values: integers
Default value: **10**
Minimum: 2

slicesolid Set the slices of drawing solid attachment.

Valid values: integers
Default value: **36**
Minimum: 2

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.2**
Minimum: 0.0
Maximum: 1.0

stackline Set the stacks of drawing line attachment.

Valid values: integers
Default value: **8**
Minimum: 2

stacksolid Set the stacks of drawing solid attachment.

Valid values: integers
Default value: **18**
Minimum: 2

transparency
The transparency of rendering attachment markers.

Valid values: reals
Default value: **20**
Minimum: 0.0
Maximum: 100.0

qsiteion

Defines an ion as part of the QM region for an Impact QSite simulation

Syntax:

qsiteion *basis*=⟨ text ⟩ ⟨ molecule_num ⟩

Options:

basis The basis set for the ligand to be included in the QM region of an Impact QSite simulation.
Valid values: text strings
Default value: **lacvp***

Operands:
⟨ molecule_num ⟩
The molecule number of an ion to be included in the QM region in a Impact QSite simulation.

qsiteresidue

Defines a residue as part of the QM region for an Impact QSite simulation

Syntax:

qsiteresidue *backboneresidue*=yes | no *backbonestring*=⟨ text ⟩
basis=⟨ text ⟩ *cuttype*=⟨ n ⟩
⟨ chain ⟩:⟨ molnum ⟩:⟨ resnum ⟩:⟨ inscode ⟩

Options:

backboneresidue
The flag indicates if this residue is a lower or upper bound residue of Free ligand/ion residues.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

backbonestring
The string for all residues of a Free ligand/ion section.
Valid values: text strings
Default value:

basis The basis set for the residue to be included in the QM region of an Impact QSite simulation.

Valid values: text strings
Default value: **lacvp***

cuttype The cut type for the residue to be included in the QM region of an Impact QSite simulation.

Valid values: integers
Default value: **1**
Minimum: 0
Maximum: 5

Operands:

⟨ chain ⟩:⟨ molnum ⟩:⟨ resnum ⟩:⟨ inscode ⟩

The number of a residue to be included in the QM region in a Impact QSite simulation.

qsiteset

Settings associated with QSite simulations in Impact.

Syntax:

`qsiteset` *alie*=yes | no *boxadjustment*=⟨ x ⟩ *charge*=⟨ n ⟩ *electrondensity*=yes | no *enableip472*=yes | no *endalpha*=⟨ n ⟩ *endalphabase*=homominus | lumoplus *endbeta*=⟨ n ⟩ *endbetabase*=homominus | lumoplus *esp*=yes | no *espunits*=kcalmolelectron | ktelectron | hartrees | kcalmol | kt | ev *griddensity*=⟨ x ⟩ *hcap*=none | point | gaussian | gaussgrid *maxiter*=⟨ n ⟩ *method*=dft | hf | lmp2 | pwb6k | m06 | m062x | m06l | m06hf | m05 | m052x | user | rm1 | am1 | pm3 | mndo | mndod *multiplicity*=⟨ n ⟩ *nddo_options*=⟨ text ⟩ *noncovgriddensity*=⟨ x ⟩ *noncovinteraction*=yes | no *numberprocessors*=⟨ n ⟩ *optimization*=singlepoint | minimization | transitionstate *optimize*=yes | no *options*=⟨ text ⟩ *orbitals*=yes | no *pathfraction*=⟨ x ⟩ *product_entry*=⟨ text ⟩ *reactant_entry*=⟨ text ⟩ *spin_unrestricted*=yes | no *spindensity*=yes | no *startalpha*=⟨ n ⟩ *startalphabase*=homominus | lumoplus *startbeta*=⟨ n ⟩ *startbetabase*=homominus | lumoplus *tsguess_entry*=⟨ text ⟩ *tsmethod*=standard | lst | qst *use_guess*=yes | no *use_hessian*=yes | no

Options:

alie Whether to calculate Average local ionization energy
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

boxadjustment
The box size adjustment, in angstroms, per side.
Valid values: reals
Default value: **0**

charge The total charge on the QM part in an Impact QSite simulation.
Valid values: integers
Default value: **0**

electrondensity
Whether to calculate electron density surface.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

enableip472
Whether enable ip472 keyword or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

endalpha The ending alpha orbital number or offset.
Valid values: integers

Default value: **0**
Minimum: 0

endalphabase
The base for the ending alpha orbital number.
Valid values: homominus
lumoplus
Default value: **lumoplus**

endbeta The ending beta orbital number or offset.
Valid values: integers
Default value: **0**

endbetabase
The base for the ending beta orbital number.
Valid values: homominus
lumoplus
Default value: **lumoplus**

esp Whether to calculate electrostatic potential surface.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

espunits Which esp units to use for surfaces (kcal/mol/electron, kT/electron at 298.15K, hartrees, kcal/mol, kT at 298.15K, eV).
Valid values: kcalmolelectron
ktelectron
hartrees
kcalmol
kt
ev
Default value: **kcalmol**

griddensity
The number of grid points per angstrom.
Valid values: reals
Default value: **5**

hcap The QM hydrogen cap electrostatics type in a QSite job in Maestro.
Valid values: none
point
gaussian
gaussgrid
Default value: **gaussgrid**

maxiter The maximun number of iterations for the QM optimization.

Valid values: integers
Default value: **100**

method The QM Method used in a QSite job in Maestro

Valid values: dft
hf
lmp2
pwb6k
m06
m062x
m06l
m06hf
m05
m052x
user
rm1
am1
pm3
mndo
mndod
Default value: **dft**

multiplicity

The multiplicity of the QM part in an Impact QSite simulation.

Valid values: integers
Default value: **1**

nddo_options

Any additional Jaguar options which can be used during the Impact QSite job with NDDO method.

Valid values: text strings
Default value: **mmok=1 noxyz=1 nointer=1 trust=0.5**

noncovgriddensity

Noncovalent grid density for QSite.

Valid values: reals
Default value: **20**

noncovinteraction

Whether to enable noncovalent interactions for QSite.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numberprocessors

The number of processors to be used for the Jaguar Part of the calculation.

Valid values: integers
Default value: **1**

optimization
What type of calculation is to be performed in the QM part.
Valid values: singlepoint
minimization
transitionstate
Default value: **minimization**

optimize [NOTE: This option is no longer used.] An option which determines if the QM part will be geometry optimized during a QSite job.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

options Any additional Jaguar options which can be used during the Impact QSite job.
Valid values: text strings
Default value: **iacc=1 vshift=1.0 maxit=100 trust=0.5**

orbitals Whether to calculate molecular orbital surfaces.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

pathfraction
The fraction of the path between the reactant and product which the TS is along.
Valid values: reals
Default value: **0.5**
Minimum: -0.000000
Maximum: 1.0000001

product_entry
The name of the entry which represents the product in a transition state calculation.
Valid values: text strings
Default value:

reactant_entry
The name of the entry which represents the entry in a transition state calculation.
Valid values: text strings
Default value:

spin_unrestricted
Whether spin is restricted or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

spindensity
Whether to calculate spin density surface.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

startalpha The starting alpha orbital number or offset.

Valid values: integers
Default value: **0**
Minimum: 0

startalphabase
The base for the starting alpha orbital number.

Valid values: homominus
lumoplus
Default value: **homominus**

startbeta The starting beta orbital number or offset.

Valid values: integers
Default value: **0**

startbetabase
The base for the starting beta orbital number.

Valid values: homominus
lumoplus
Default value: **homominus**

tsguess_entry
The name of the entry which represents the transtion state guess in a transition state calculation.

Valid values: text strings
Default value:

tsmethod For a transition state calculation, how that is to be performed.

Valid values: standard
lst
qst
Default value: **standard**

use_guess Whether use wave function in input file or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

use_hessian
Whether use hessian in input file or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

quicktorsion

Adjust the dihedral specified by 4 atoms to the given value. Adjust the marker arrow direction via reverse operand.

Syntax:

`quicktorsion` *contactdisplay*=yes | no *dihedral*=⟨ x ⟩
hbonddisplay=yes | no *reverse*=yes | no movelarger|⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

contactdisplay
This option determines whether Contact markers will be displayed for quick torsion.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

dihedral Value to set the torsion
Valid values: reals
Default value: **0**
Minimum: -180.1
Maximum: 180.1

hbonddisplay
This option determines whether H-bond markers will be displayed for quick torsion.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

reverse To set the torsion marker arrow direction
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
movelarger|⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩
Four atoms which are used to adjust a torsion. The movelarger operand will flip the direction of already defined dihedral marker arrow toward the larger atom structure.

quit

Quit the program. To quit issue just the quit command without any options.

Syntax:

`quit` *confirm*=yes | no

Options:

confirm If this option has been set to "false" then the program will exit without prompting the user in any way. So, for example, the user is not prompted to save any changed macros nor will the Quit panel be displayed. Note that invoking quit with any options only sets the option it does not also try to quit Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

raisepanels

Raise all visible panels

Syntax:

`raisepanels`

read

This is a standard alias for **fileread** (see [fileread], page 195).

readpotential

Read potential settings from a command file.

Syntax:

`readpotential` ⟨file_name⟩

Operands:
⟨file_name⟩

The name of the file from which the potential settings will be read. If no name is specified, the default settings will be used.

reagentprep

Options for Reagent Preparation jobs.

Syntax:

reagentprep *gen_conform*=⟨ n ⟩ *gen_ionization*=yes | no
gen_stereo=⟨ n ⟩ *gen_tautomers*=yes | no
group_name_long=⟨ text ⟩ *group_name_short*=⟨ text ⟩
input_file=⟨ text ⟩ *ph*=⟨ x ⟩ *ph_tolerance*=⟨ x ⟩
sd_title_property=⟨ text ⟩ *sd_title_source*=molecule_name |
property *structure_source*=selected_entries | workspace | file
use_epik=yes | no

Options:

gen_conform
The percentage of generating low energy ring conformations.
Valid values: integers
Default value: **1**
Minimum: 1

gen_ionization
An option which allows generating ionization.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

gen_stereo The percentage of generating stereoisomers.
Valid values: integers
Default value: **10**
Minimum: 1

gen_tautomers
An option which allows generating tautomers.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

group_name_long
The long name of selected functional group for reagent preparation job.
Valid values: text strings
Default value:

group_name_short
The short name of selected functional group for reagent preparation job.

Valid values: text strings
Default value:

input_file The name of the structure input file.

Valid values: text strings
Default value:

ph The reagent ionization pH value.

Valid values: reals
Default value: **7**
Minimum: 0.0
Maximum: 14.0

ph_tolerance
The reagent ionization pH tolerance.

Valid values: reals
Default value: **2**
Minimum: 0.0
Maximum: 7.0

sd_title_property
The property to be used to construct titles for reagents from a SD format file, if sd_title_source is 2 - property.

Valid values: text strings
Default value:

sd_title_source
The source of titles for reagents from a SD format file (1 - molecule_name or 2 - property).

Valid values: molecule_name
property
Default value: **molecule_name**

structure_source
Whether to use the selected entries in the current project, or what is in the workspace, or a specified file with multiple structures as structure input for the job.

Valid values: selected_entries
workspace
file
Default value: **file**

use_epik An option which specifies whether or not Epik should be used for ionization and tautomerization.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

refinestart

Start a Refine input file with the current settings.

Syntax:

```
refinestart
```

refinewrite

Write a Refine input file with the current settings.

Syntax:

```
refinewrite
```

removeatomsfromtable

Removes the selected atoms from the atom-property table.

Syntax:

```
removeatomsfromtable
```

rename

Rename a named object. The object type is the same as the command which is used to create that type of object.
For example to rename a set named “alpha” use: rename set alpha beta .

Syntax:

rename ⟨object_type⟩ ⟨current_object_name⟩ ⟨new_object_name⟩

Operands:
⟨object_type⟩ ⟨current_object_name⟩ ⟨new_object_name⟩
The first operand is the name of the existing object.

renameproperty

This is a standard alias for **propertyrename** (see [propertyrename], page 606).

repall

Set global representation properties

Syntax:

repall *ballhresolution*=⟨n⟩ *balllresolution*=⟨n⟩ *ballsize*=⟨x⟩
bondoutline=yes | no *border*=auto | on | true | yes | off | false | no *borderscale*=⟨x⟩ *bstyle*=split | blend
cpkhresolution=⟨n⟩ *cpklresolution*=⟨n⟩ *cpksize*=⟨x⟩
depthcutoff=⟨n⟩ *depthfactor*=⟨x⟩
drawspheresaspoints=yes | no *enhancedepth*=yes | no
maxwirewidth=⟨x⟩ *minwirewidth*=⟨x⟩ *resolution*=high | low
rstyle=multiple | thick *scalewirewidth*=yes | no
showaltpositions=yes | no *simplifymoving*=yes | no
smooth=yes | no *smooth_sgl*=yes | no *stickradius*=⟨x⟩
stickradiusmulti=⟨x⟩ *thintuberadius*=⟨x⟩
thintuberadiusmulti=⟨x⟩ *tubehresolution*=⟨n⟩
tubelresolution=⟨n⟩ *tuberadius*=⟨x⟩ *tuberadiusmulti*=⟨x⟩
tubestickborder=auto | on | off *usethicklines*=yes | no
wirebondsmooth=auto | on | off *wirethickness*=⟨n⟩
wirewidthpoint=⟨x⟩

Options:

ballhresolution
Set Ball high resolution
Valid values: integers
Default value: **16**

Minimum: 4
Maximum: 53

balllresolution
Set Ball resolution

Valid values: integers
Default value: **4**
Minimum: 4
Maximum: 53

ballsize Percentage to draw balls out at

Valid values: reals
Default value: **16**
Minimum: 4.0
Maximum: 200.0

bondoutline
Enhance appearance of bond overlaps by drawing in the background color a thin border around each bond

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

border Set drawing of bond order to be on, off, or automatically determined by viewing scale.

Valid values: auto
on
true
yes
off
false
no
Default value: **auto**

borderscale
The drawing scale at which bond orders will be displayed

Valid values: reals
Default value: **30**

bstyle Set the bond style

Valid values: split
blend
Default value: **split**

cpkhresolution
Set CPK high resolution

Valid values: integers
Default value: **20**

Minimum: 8
Maximum: 57

cpklresolution
Set CPK low resolution

Valid values: integers
Default value: **8**
Minimum: 8
Maximum: 57

cpksize Percentage to draw CPK spheres out at

Valid values: reals
Default value: **85**
Minimum: 4.0
Maximum: 200.0

depthcutoff
Depth cutoff of ambient occlusion. Used when enhancedepth is on.

Valid values: integers
Default value: **3**
Minimum: 2
Maximum: 10

depthfactor
Depth factor of ambient occlusion. Used when enhancedepth is on.

Valid values: reals
Default value: **0.7**
Minimum: 0.0
Maximum: 1.0

drawspheresaspoints
Draw spheres using point shaders (true) or using triangle shaders (false)

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

enhancedepth
Flag for whether or not enhance depth view.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

maxwirewidth
Maximum thickness of scaled wireframe lines

Valid values: reals
Default value: **3**

Minimum: 2.0
Maximum: 50.0

minwirewidth
Minimum thickness of scaled wireframe lines
Valid values: reals
Default value: **1**
Minimum: 0.1
Maximum: 2.0

resolution Set the overall resolution for drawing molecules. low or high.
Valid values: high
low
Default value: **high**

rstyle Set the bond render style
Valid values: multiple
thick
Default value: **multiple**

scalewirewidth
Change wire width when zooming.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showaltpositions
Show the alternate occupancy positions
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

simplifymoving
Enable/disable use of simplified moving representation
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

smooth Enable/disable line and polygon antialiasing
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

smooth_sgl
Enable/disable linee and polygon antialiasing for workspace transformation operations, if Maestro was started with -SGL option.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

stickradius
Radius of sticks in Angstroms

Valid values: reals
Default value: **0.12**
Minimum: 0.01
Maximum: 1.0

stickradiusmulti
Radius of multiple bonds sticks in Angstroms

Valid values: reals
Default value: **0.08**
Minimum: 0.01
Maximum: 0.5

thintuberadius
Radius of thin tubes in Angstroms

Valid values: reals
Default value: **0.08**
Minimum: 0.01
Maximum: 1.0

thintuberadiusmulti
Radius of multiple bonds thin tubes in Angstroms

Valid values: reals
Default value: **0.05**
Minimum: 0.01
Maximum: 0.5

tubehresolution
Set Tube high resolution

Valid values: integers
Default value: **16**
Minimum: 8
Maximum: 57

tubelresolution
Set Tube resolution

Valid values: integers
Default value: **8**
Minimum: 8
Maximum: 57

tuberadius Radius of tubes in Angstroms

Valid values: reals
Default value: **0.16**
Minimum: 0.01
Maximum: 1.0

tuberadiusmulti
Radius of multiple bonds tubes in Angstroms

Valid values: reals
Default value: **0.08**
Minimum: 0.01
Maximum: 0.5

tubestickborder

Set drawing of tube/stick bond order to be on, off, or automatically determined by viewing scale.

Valid values: auto
on
off
Default value: **auto**

usethicklines

Enables the use of thick lines to replace sticks when rotating.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

wirebondsmooth

Set wire bond smoothing as automatically, on, or off.

Valid values: auto
on
off
Default value: **on**

wirethickness

Thickness of wireframe lines

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 20

wirewidthpoint

The linewidth stage. Points will be displayed at atoms places if current linewidth greater or equal to the stage.

Valid values: reals
Default value: **3.5**
Minimum: 2.0

repatom

Change the representation used to display a group of atoms in the main structure window.

Syntax:

`repatom` *rep*=none | circle | cpk | ballnstick ⟨ ASL ⟩

Options:

rep	Type of atom representation	
	Valid values:	none circle cpk ballnstick
	Default value:	**none**

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this specification will have their representation changed.

repatombonds

Set representation of all of atoms' bonds. Uses bond rep specified in the repbond command.

Syntax:

`repatombonds` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this string will have the representation of all bonds to them changed.

repbond

Change the representation used to draw an on-screen bond.

Syntax:

`repbond` *rep*=none | wire | tube | thin_tube ⟨atom1⟩ ⟨atom2⟩

Options:

rep	Type of bond representation	
	Valid values:	none wire tube thin_tube
	Default value:	**wire**

Operands:

⟨atom1⟩ ⟨atom2⟩

The operands represent the numbers of the two atoms which define the bond to have its representation changed.

repdefault

Set representation of all atoms and bonds

Syntax:

`repdefault` *style*=wire | cpk | ballnstick | tube | thin_tube

Options:

style	Default representation	
	Valid values:	wire cpk ballnstick tube thin_tube
	Default value:	**wire**

replacefromhold

This command replaces the on-screen structure with a copy of the the structure from the specified hold set.

Syntax:

replacefromhold ⟨ hold_name ⟩

Operands:

⟨ hold_name ⟩

The name of the hold. This must be the name which was specified when the hold was created using the "hold" command.

repquick

Set representation of all atoms and bonds

Syntax:

repquick *default*=default | wire | cpk | ballnstick | tube | thin_tube *style*=default | wire | cpk | ballnstick | tube | thin_tube [update]

Options:

default Default representation

Valid values: default, wire, cpk, ballnstick, tube, thin_tube

Default value: **wire**

style Default representation

Valid values: default, wire, cpk, ballnstick, tube, thin_tube

Default value: **default**

Operands:

[update]

When udpate is present the currently set representation will be applied to the on-screen structure.

resetcsearch

Deletes all the variables used in a conformational search.

Syntax:

`resetcsearch`

residuename

Set the residue name for all atoms which match the ASL specification.

Syntax:

`residuename` ⟨ PDBNAME ⟩ ⟨ ASL ⟩

Operands:
⟨ PDBNAME ⟩ ⟨ ASL ⟩
The first operand is the PDB residue name which will be used for all atoms which match the specification. The second operand is a valid ASL string which defines the set of atoms which are to have their residue names changed.

residuenumber

Set the residue number for all atoms which match the ASL specification.

Syntax:

`residuenumber` ⟨ res_num ⟩ ⟨ ASL ⟩

Operands:
⟨ res_num ⟩ ⟨ ASL ⟩
The first operand is an integer, optionally followed by a single-character insertion code, that specifies the residue number for all the atoms which match the ASL specification. If the trailing alphabetic character is omitted, the residue insertion code will be set blank. The second operand must be a valid ASL string which specifies all the atoms to have their residue number changed.

residuerenumber

Renumber the residues, starting with the starting number, for all residues which match the ASL specification.

Syntax:

`residuerenumber` ⟨ starting_res_num ⟩ ⟨ ASL ⟩

Operands:

⟨ starting_res_num ⟩ ⟨ ASL ⟩

The first operand is an integer which represents the starting residue number. The second operand must be a valid ASL string which specifies a set of residues to renumber.

restorepanels

Restores panel locations

Syntax:

`restorepanels`

retype

Change the atom type of the atom number specified by the operand to whatever type or element has previously been made current with the atom command.

Syntax:

`retype` ⟨ atom_num ⟩

Operands:

⟨ atom_num ⟩

An atom number representing the atom which is to have its type changed to the current type.

ribbon

Creates a new Ribbon.

Syntax:

ribbon *ambient*=⟨ x ⟩ *ambientback*=⟨ x ⟩ *backcoloroffset*=⟨ x ⟩ *blend*=yes | no *calphalinewidth*=⟨ n ⟩ *calphatubesteps*=⟨ n ⟩ *calphatubewidth*=⟨ x ⟩ *color*=black | gray | darkblue | blue | lightblue | aquamarine | turquoise | springgreen | darkgreen | green | limegreen | yellowgreen | yellow | orange | maroon | red | pink | plum | purple | bluepurple | white *curvedlinesteps*=⟨ n ⟩ *curvedlinewidth*=⟨ n ⟩ *diffuse*=⟨ x ⟩ *diffuseback*=⟨ x ⟩ *display*=ribbonsonly | atomsonly | both *emission*=⟨ x ⟩ *emissionbackblue*=⟨ x ⟩ *emissionbackgreen*=⟨ x ⟩ *emissionbackred*=⟨ x ⟩ *flatstrand*=yes | no *helixcolor*=onecolor | twocolors *hide*=none | allatoms | backboneatoms *includecalpha*=yes | no *ladderuknot*=⟨ x ⟩ *ladderwidth*=⟨ x ⟩ *lowerresidue*=⟨ n ⟩ *minmolsize*=⟨ n ⟩ *perpendicular*=yes | no *resolution*=high | low *ribbonendweight*=⟨ x ⟩ *ribbonhasthick*=yes | no *ribbonsteps*=⟨ n ⟩ *ribbonthick*=⟨ x ⟩ *ribbonweight*=⟨ x ⟩ *ribbonwidth*=⟨ x ⟩ *scheme*=constant | secondarystructure | chain | calphaatom | residuecharge | residueproperty | residuetype | residueposition | entry *setlowerresidue*=yes | no *setupperresidue*=yes | no *shininess*=⟨ x ⟩ *shininessback*=⟨ x ⟩ *showladder*=yes | no *shownormals*=yes | no *simplifymoving*=yes | no *specular*=⟨ x ⟩ *specularback*=⟨ x ⟩ *spherestacks*=⟨ n ⟩ *strandarrowweight*=⟨ x ⟩ *strandarrowwidth*=⟨ x ⟩ *strandendweight*=⟨ x ⟩ *strandendweight1*=⟨ x ⟩ *strandsteps*=⟨ n ⟩ *strandthick*=⟨ x ⟩ *strandweight*=⟨ x ⟩ *strandwidth*=⟨ x ⟩ *style*=none | cartoon | ribbon | tube | thintube | curvedline | calphaline | calphatube *thintubesteps*=⟨ n ⟩ *thintubeweight*=⟨ x ⟩ *thintubewidth*=⟨ x ⟩ *tubesteps*=⟨ n ⟩ *tubeweight*=⟨ x ⟩ *tubewidth*=⟨ x ⟩ *upperresidue*=⟨ n ⟩ *useshader*=yes | no ⟨ ASL-definition ⟩

Options:

ambient Set material property - ambient, to its red, green, and blue components, for front face.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

ambientback
Set material property - ambient, to its red, green, and blue components, for back face.

Valid values:	reals
Default value:	**0.05**
Minimum:	0.0
Maximum:	1.0

backcoloroffset
The offset for drawing back ribbons with gray color.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0
Maximum:	1.0

blend The falg indicates whether get smooth color changes along the ribbon.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

calphalinewidth
Set linewidth for drawing CA Trace ribbons.

Valid values:	integers
Default value:	**2**
Minimum:	1
Maximum:	40

calphatubesteps
Set the steps of drawing ribbon CA Tube Trace from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values:	integers
Default value:	**3**
Minimum:	1
Maximum:	10

calphatubewidth
Set CA tube width for drawing CA Trace Tube ribbons.

Valid values:	reals
Default value:	**0.4**
Minimum:	0.001

color An option which controls the ribbon constant color.

Valid values: black
gray
darkblue
blue
lightblue
aquamarine
turquoise
springgreen
darkgreen
green
limegreen
yellowgreen
yellow
orange
maroon
red
pink
plum
purple
bluepurple
white

Default value: **green**

curvedlinesteps

Set the steps of drawing ribbon Curved Line from one node point (mapped from one CA atom) to another. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **9**
Minimum: 1
Maximum: 100

curvedlinewidth

Set curve width for drawing Curved Line ribbons.

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 40

diffuse Set material property - diffuse, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.4**
Minimum: 0.0
Maximum: 1.0

diffuseback

Set material property - diffuse, to its red, green, and blue components, for back face.

Valid values: reals
Default value: **0.85**
Minimum: 0.0
Maximum: 1.0

display An option which controls whether the atoms which define the ribbon are to be shown. The three options are: Ribbon Only, Atoms Only, and Both.

Valid values: ribbonsonly
atomsonly
both
Default value: **ribbonsonly**

emission Set material property - emission, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.05**
Minimum: 0.0
Maximum: 1.0

emissionbackblue

Set material property - emission, to its blue component, for back face. Emission R, G, B values are used to control the color of back face. The great the values, the lighter the back face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

emissionbackgreen

Set material property - emission, to its green component, for back face. Emission R, G, B values are used to control the color of back face. The great the values, the lighter the back face.

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

emissionbackred

Set material property - emission, to its red component, for back face. Emission R, G, B values are used to control the color of back face. The great the values, the lighter the back face.

Valid values: reals

Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

flatstrand The falg indicates whether use flat strand arrows.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

helixcolor An option which controls whether ribbons in the helical part are to be colored with a single color or with two colors and the inside of the helix has a contrasting color.

Valid values: onecolor
twocolors
Default value: **twocolors**

hide An option which controls whether the atoms which define the ribbon are to be hide when creating ribbons. The three options are: None, All associated atoms, and associated backbone atoms.

Valid values: none
allatoms
backboneatoms
Default value: **none**

includecalpha

The falg indicates whether ribbons pass through the alpha carbons.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ladderuknot

The value of parameter in the U direction that determines the ladder end of the residue along DNA/RNA backbone ribbon: [0.0, 1.0]. The great the value, the small the radius.

Valid values: reals
Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

ladderwidth

Set tube width for drawing ladders.

Valid values: reals
Default value: **0.4**
Minimum: 0.05
Maximum: 5.0

lowerresidue

The lower residue limit for residue position color scheme.

Valid values: integers
Default value: **1**
Minimum: 1

minmolsize

The minimun size of molecule (number of molecule atoms), which determines if a molecule can be displayed as ribbons.

Valid values: integers
Default value: **20**
Minimum: 1

perpendicular

Set DNA ribbon as perpendicur to base.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

resolution Set the resolution for drawing ribbons.

Valid values: high
low
Default value: **high**

ribbonendweight

This weight value is used to control the radius of the Ribbon ends. The great the value, the small the radius.

Valid values: reals
Default value: **0.6**
Minimum: 0.001

ribbonhasthick

This bool value is used to determine which kind of ribbon will be drawn, a flat sheet or a solid ribbon has thick.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

ribbonsteps

Set the steps of drawing Ribbon ribbon from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 10

ribbonthick

This value is used to control the thick of ribbon, if the ribbon is display in solid ribbon.

Valid values: reals

Default value: **0.1**
Minimum: 0.1
Maximum: 2.0

ribbonweight

This weight value is used to control the radius of the rectangle cross-section of a strand ribbon. The great the value, the small the radius.

Valid values: reals
Default value: **3**
Minimum: 0.001

ribbonwidth

Set ribbon width for drawing Ribbon ribbons.

Valid values: reals
Default value: **2**
Minimum: 0.05
Maximum: 5.0

scheme An option which controls the ribbon coloring scheme.

Valid values: constant
secondarystructure
chain
calphaatom
residuecharge
residueproperty
residuetype
residueposition
entry
Default value: **secondarystructure**

setlowerresidue

The falg indicates whether use lower residue limit for residue position color scheme.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

setupperresidue

The falg indicates whether use upper residue limit for residue position color scheme.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

shininess Set material property - shininess, for front face.

Valid values: reals
Default value: **80**
Minimum: 0.0

Maximum: 128.0

shininessback

Set material property - shininess, for back face.

Valid values: reals
Default value: **80**
Minimum: 0.0
Maximum: 128.0

showladder

Set the falg for showing DNA ladders.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

shownormals

Set the falg for showing normals.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

simplifymoving

Enable/disable use of simplified moving representation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

specular Set material property - specular, to its red, green, and blue components, for front face.

Valid values: reals
Default value: **0.7**
Minimum: 0.0
Maximum: 1.0

specularback

Set material property - specular, to its red, green, and blue components, for back face.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

sphereslices

Set the slices of drawing sphere in the tube style. We use gluSphere() to draw a sphere. The higher the slices, the fine the sphere.

Valid values: integers
Default value: **18**
Minimum: 2

spherestacks

Set the stacks of drawing sphere in the tube style. We use gluSphere() to draw a sphere. The higher the stacks, the fine the sphere.

Valid values: integers
Default value: **18**
Minimum: 2

strandarrowweight

This weight value is used to control the radius of the strand arrow. The great the value, the small the radius.

Valid values: reals
Default value: **10**
Minimum: 0.001

strandarrowwidth

This ratio value is used to control the width of the strand arrow.

Valid values: reals
Default value: **2**
Minimum: 1.0

strandendweight

This weight value is used to control the radius of the strand ends. The great the value, the small the radius.

Valid values: reals
Default value: **0.6**
Minimum: 0.001

strandendweight1

This weight value is used to control the radius of the strand ends in Cartoon. The great the value, the small the radius.

Valid values: reals
Default value: **50**
Minimum: 0.001

strandsteps

Set the steps of drawing ribbon Strand from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

strandthick

Set strand thick for drawing Strand ribbons.

Valid values: reals

Default value: **0.2**
Minimum: 0.1
Maximum: 2.0

strandweight

Ribbons are defined with NURBS curce or surfaces that are defined by 4D homogeneous coordnate (x, y, z, w) control points array or mesh. The weight w can push/pull away/towards the curve/surface part near the control point, by decressing/increasing weight value. This weight value is used to control the radius of the rectangle cross-section of a strand ribbon. The great the value, the small the radius.

Valid values: reals
Default value: **5**
Minimum: 0.001

strandwidth

Set strandwidth for drawing Strand ribbons.

Valid values: reals
Default value: **2**
Minimum: 0.05
Maximum: 5.0

style An option which controls the ribbon style representation.

Valid values: none
cartoon
ribbon
tube
thintube
curvedline
calphaline
calphatube
Default value: **cartoon**

thintubesteps

Set the steps of drawing ribbon Thin Tube from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

thintubeweight

This weight value is used to control the radius of the Thin Tube ends. The great the value, the small the radius.

Valid values: reals
Default value: **1**
Minimum: 0.001

thintubewidth
Set thin tube width for drawing Thin Tube ribbons.

Valid values: reals
Default value: **0.4**
Minimum: 0.05
Maximum: 5.0

tubesteps Set the steps of drawing ribbon Tube from one node point (mapped from one CA atom) to another. In the both U and V directions. It is used when the resolution option is HIGH.

Valid values: integers
Default value: **3**
Minimum: 1
Maximum: 10

tubeweight This weight value is used to control the radius of the Tube ends. The great the value, the small the radius.

Valid values: reals
Default value: **1**
Minimum: 0.001

tubewidth Set tube width for drawing Tube ribbons.

Valid values: reals
Default value: **0.9**
Minimum: 0.05
Maximum: 5.0

upperresidue
The upper residue limit for residue position color scheme.

Valid values: integers
Default value: **1**
Minimum: 1

useshader The falg indicates whether using ribbon specified shaders.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ ASL-definition ⟩
The operand must be a valid string in the atom specification language. It will define which atoms are to have a ribbon drawn.

ribbondump

Print out the current option values of the ribbon command.

Syntax:

`ribbondump`

ringclosure

A command which defines a ring closure to be used during a conformational search.

Syntax:

`ringclosure` *maximum*=⟨ x ⟩ *minimum*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

maximum The maximum distance between the ends of the ring which will be accepted as a candidate for "closure".

Valid values:	reals
Default value:	**2.5**
Minimum:	0.0

minimum The minimum distance between the ends of the ring which will be accepted as a candidate for "closure".

Valid values:	reals
Default value:	**0.5**
Minimum:	0.0

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

The four atom numbers which define a point in a ring which is to be opened while new structures are generated in a conformational search. The actually opening takes place between the second and third atoms specified. Note that specifying a-b-c-d is the same as specifiying d-c-b-a.

ringflip

Conversion of aliphatic ring substituents between axial and equitorial

Syntax:

`ringflip` ⟨ atom_num ⟩

Operands:
⟨ atom_num ⟩
An atom number representing the atom which is a ring atom to have the subsituents flipped.

rotate

Rotate in degrees whatever is specified in the transform set. This is either global (all atoms) or a local grouping defined via an ASL in the transform command.

Syntax:

`rotate` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩ [reset]

Options:

x Amount in degrees to rotate in X
Valid values: reals
Default value: **0**

y Amount in degrees to rotate in Y
Valid values: reals
Default value: **0**

z Amount in degrees to rotate in Z
Valid values: reals
Default value: **0**

Operands:
[reset]
If reset is present the global rotation matrix will be reset.

run

This is a standard alias for **scriptrun** (see [scriptrun], page 698).

saveimage

Capture the current main structure window and save to an image file.

Syntax:

`saveimage` *format*=tiff | jpeg | png *jpeg_quality*=⟨ n ⟩
png_compression=⟨ n ⟩ *png_gamma*=⟨ x ⟩ *showoptions*=yes | no
smooth=yes | no *transparentbg*=yes | no ⟨ file_name ⟩

Options:

format Specifies the format of the saved image.
Valid values: tiff
jpeg
png
Default value: **png**

jpeg_quality
Quality of the JPEG file (1-100)
Valid values: integers
Default value: **95**

png_compression
Compression ratio of png format
Valid values: integers
Default value: **6**

png_gamma
Gamma of the image in png format
Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

showoptions
If enabled, show expanded options when dialog is posted.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

smooth Image saved will have smooth curves in it if enabled.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

transparentbg
Image saved will have transparent background color if enabled.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:
⟨ file_name ⟩
The file where the image will be saved.

saveimageheight

Sets the height of the image saved due to saveimage command

Syntax:

`saveimageheight` ⟨ height ⟩

Operands:
⟨ height ⟩
Height of the image to be saved.

saveimagewidth

Sets the width of the image saved due to saveimage command

Syntax:

`saveimagewidth` ⟨ width ⟩

Operands:
⟨ width ⟩
Width of the image to be saved.

savelayout

Save the size and position of all currently visible panels.

Syntax:

`savelayout`

savemovie

Syntax:

`savemovie` ⟨ filename ⟩

Operands:
⟨ filename ⟩
The file where the movie will be saved.

savemovieoptions

Sets the options for the save movie while eplaying entries

Syntax:

`savemovieoptions` *addframes*=⟨ n ⟩ *duration*=frameduration | movieduration *frameduration*=⟨ x ⟩ *movieduration*=⟨ n ⟩ *quality*=low | medium | high *resolution*=low | medium | high *smooth*=yes | no *speed*=⟨ n ⟩

Options:

addframes This option controls the number of frames to add when 'smooth' is enabled.

Valid values: integers
Default value: **20**

duration This option sets the duration option for the movie

Valid values: frameduration
movieduration
Default value: **frameduration**

frameduration
This option sets the number of seconds that each frame to be displayed for.

Valid values: reals
Default value: **0.04**
Minimum: 0.04
Maximum: 5.0

movieduration
This option sets the number of seconds that the recorded movie should be played for.

Valid values: integers
Default value: **1**
Minimum: 1
Maximum: 300

quality This option sets the quality of movie file.

Valid values: low
medium
high
Default value: **medium**

resolution This option sets the resolution of movie file.

Valid values: low
medium
high
Default value: **medium**

smooth This option controls whether additional interpolated frames are added to the movie or not. Entries should be conformers. This is useful for creating a smoother animation in the movie for things like molecular dynamics (MD) results. Keep in mind this is strictly a linear interpolation of coordinates.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

speed This option sets the number of structures to be displayed per second.

Valid values: integers
Default value: **1**

scanmode

Set the Coordinate Scan mode.

Syntax:

`scanmode` *modetype*=distance | angle | dihedral

Options:

modetype Determines the mode for coordiante scan softlimit=<n> <input_file_name>

Valid values: distance
angle
dihedral

Default value: **distance**

scriptlogfile

Commands are by default logged to a temporary file which is deleted when the program terminates. If the "logfile" command is used to name the logfile then the commands will be logged to that file and that file will not be deleted when the program ends.

Syntax:

`scriptlogfile` *logging*=yes | no ⟨ logfile_name ⟩

Options:

logging If this option is set to "false" then no logging will be done to a file or in memory.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
⟨ logfile_name ⟩
The name of the file to which the commands will be logged. The full name of the file, including any suffix must be included.
Aliases:
logfile (see [logfile], page 338)

scriptrun

Run the script file whose name is given as the operands.

Syntax:

`scriptrun` ⟨ script_name ⟩

Operands:
⟨ script_name ⟩
The name of the script which is to be executed. The full name, including any suffix must be given. If the file is not in the local directory then a full pathname must be given.

Aliases:
run (see [run], page 693)

searchdbconfgen

Defines settings for Find Matches to Hypothesis Generate Conformers job.

Syntax:

`searchdbconfgen` *amidebonds*=vary | retain | trans *eliminate*=atom_deviation | rmsd *field*=mmffs | opls2005 *incorporate*=append | replace | ignore | appendungrouped | workspace *max_rmsd*=⟨ x ⟩ *maxdist*=⟨ x ⟩ *method*=default | mixed *minimizationsteps*=⟨ n ⟩ *numrotatablesteps*=⟨ n ⟩ *numsteps*=⟨ n ⟩ *postmaxiter*=⟨ n ⟩ *postprocessing*=yes | no *postprocessingmethod*=mini | filter | rce *premaxiter*=⟨ n ⟩ *preprocessing*=yes | no *sampling*=standard | rapid | complete | thorough *skipconformergeneration*=⟨ n ⟩ *solvation*=gbsa | distance_dependent *window*=⟨ x ⟩

Options:

amidebonds
: This determines whether to vary amide bond conformation, retain original amide bond conformation, or set amide bond conformation to trans.

  Valid values: vary
  retain
  trans
  Default value: **vary**

eliminate
: The method to use for eliminating redundant conformers: maximum atom deviation or RMSD.

  Valid values: atom_deviation
  rmsd
  Default value: **rmsd**

field
: This determines which force field mmffs|mmff|opls2001 is used. Currently we always use mmffs, so it will have only one option value.

  Valid values: mmffs
  opls2005
  Default value: **opls2005**

incorporate
This option controls the incorporation of the results (replace or append).

Valid values:	append replace ignore appendungrouped workspace
Default value:	**replace**

max_rmsd Maximum RMSD for considering two structures equal.

Valid values:	reals
Default value:	**1**
Minimum:	0.0

maxdist Maximum distance between atoms in equal structures.

Valid values:	reals
Default value:	**2**
Minimum:	0.0

method This determines whether MacroModel uses the ligand torsion search method (default) or the mixed MCMM/LMOD search method (mixed) to generate conformers. Currently Find Matches always uses the default method, so it will have only one option value.

Valid values:	default mixed
Default value:	**default**

minimizationsteps
This option determines the maximum number of minimization steps for Mixed MCMM/LMOD conformer generation.

Valid values:	integers
Default value:	**100**
Minimum:	1

numrotatablesteps
An option which sets the number of steps which will be performed during the ConfGen conformational search.

Valid values:	integers
Default value:	**10**
Minimum:	1

numsteps An option which sets the number of steps which will be performed during the conformational search. This also limits number of conformations generated.

Valid values: integers
Default value: **100**
Minimum: 0

postmaxiter

This option determines the maximum number of iterations for post-minimization of generated structures.

Valid values: integers
Default value: **50**
Minimum: 0
Maximum: 9999999

postprocessing

Indicates whether or not to perform MacroModel postprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

postprocessingmethod

This determines which type of postprocessing method to use (minimization, filtering and redundant conformer elmination, or redundant conformer elmination only).

Valid values: mini
filter
rce
Default value: **rce**

premaxiter

This option determines the maximum number of iterations for pre-minimization of input structures.

Valid values: integers
Default value: **100**
Minimum: 0
Maximum: 9999999

preprocessing

Indicates whether or not to perform MacroModel preprocessing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

sampling This determines whether rapid (standard) or thorough (complete) sampling will be used.

Valid values: standard
rapid
complete
thorough
Default value: **standard**

skipconformergeneration
Skip conformer generation for structures with more than the given number of rotatable bonds.

Valid values: integers
Default value: **15**
Minimum: 0

solvation This determines whether GB/SA Water (gbsa) or Distance Dependent Dielectric (distance_dependent) solvation treatment is used.

Valid values: gbsa
distance_dependent
Default value: **distance_dependent**

window The energy window (in kcal/mol) within which structures will be saved.

Valid values: reals
Default value: **10**
Minimum: 0.0

selectatomsforediting

Selects atoms for editing their properties.

Syntax:

`selectatomsforediting` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The operand is a valid ASL string which specifies which atoms are to be selected.

selecteditatom

Selects a row in the atom-level property table.

Syntax:

`selecteditatom` ⟨row number⟩

Operands:
⟨row number⟩
The operand is the row number to select.

selecteditproperty

Selects an atom-level property from the edit atom-level property list.

Syntax:

`selecteditproperty` ⟨property⟩

Operands:
⟨property⟩
The operand is a property from the edit property list.

sequenceviewersaveimage

Capture the sequence viewer contents and save it to an image file.

Syntax:

`sequenceviewersaveimage` *format*=tiff | jpeg | png
sequenceviewer=⟨text⟩ ⟨file_name⟩

Options:

format Specifies the format of the saved image.

Valid values:	tiff jpeg png
Default value:	**tiff**

sequenceviewer
Specifies whether to save the contents of the Workspace or Prime sequence viewer.

Valid values:	text strings
Default value:	**workspace**

Operands:

⟨ file_name ⟩

The file where the image will be saved.

set

Creates a new named set. The set name must be a single token (or "quoted" if multiple tokens). A set can be redefined by specifying a new definition.

Syntax:

`set` ⟨ set_name ⟩ ⟨ ASL-definition ⟩

Operands:

⟨ set_name ⟩ ⟨ ASL-definition ⟩

The name which will be applied to the set. If the name contains embedded spaces then it must be enclosed in double quotation marks.

setread

Read set definitions from the file whose name is given as the operand. The file name usually has a ".set" suffix.

Syntax:

`setread` ⟨ set_file_name ⟩

Operands:

⟨ set_file_name ⟩

The name of the file from which the set definitions are to be read. The full name of the file (including any .set suffix) must be specified.

setwrite

Write the currently defined sets to the file whose name is given as the operand. The file name usually has a ".set" suffix

Syntax:

`setwrite` ⟨set_file_name⟩

Operands:

⟨set_file_name⟩

The name of the file to which the current set definitions are to be written . The full name of the file (including any .set suffix) must be specified.

showdockablepanels

Show visible dockable panels previously hidden with MM_ID_HIDE_DOCKABLE_PANELS.

Syntax:

`showdockablepanels`

showfirstselectedentry

Scroll to make first selected entry visible in Project Table.

Syntax:

`showfirstselectedentry`

showhwstereosetup

Write out the hardware stereo setup.

Syntax:

`showhwstereosetup`

showmarkers

This command displays the given marker.

Syntax:

`showmarkers` ⟨ marker_name ⟩

Operands:
⟨ marker_name ⟩
The name of the marker.

showpanel

Show the panel whose name is given by the operands.

Syntax:

`showpanel` ⟨ panel_name ⟩ [:⟨ tab_name ⟩]

Operands:
⟨ panel_name ⟩ [:⟨ tab_name ⟩]
The first operand is the name of the panel which is to be displayed. The name must match to all characters. The names of the panels to be used in the "showpanel" command are displayed in parentheses after each item in the main menu bar. The optional second argument is the name of a tab folder within that panel which is to be made the current tab folder. The name of a tab folder to be used in the second optional operand is displayed in the associated panel. The name must match to all characters, but it is not case sensitive.

showpanels

Make visible panels previously hidden with HIDEPANELS

Syntax:

`showpanels`

showpropertiesselectedentries

Displays only those properties for which at least one of the selected entries has a value.

Syntax:

`showpropertiesselectedentries`

showproperty

This command shows the given property by creating a subset of all the properties in show state. This function also switches to the property subset view.

Syntax:

`showproperty` ⟨ propertyname ⟩

Operands:

⟨ propertyname ⟩

The name of the property to show.

showtoolbar

Show the toolbar of given id displayed under given panel.

Syntax:

`showtoolbar` ⟨ panel_name:toolbar_id ⟩

Operands:

⟨ panel_name:toolbar_id ⟩

The first part is the name of the panel under which toolbar would be displayed. The name must match to all characters and it is a case sensitive. The second part is the name of a toolbar id which needs to be displayed. Id must match to all characters and it is a case sensitive. Any toolbar can not be displayed under any panel. There is a fix set of toolbars which can be displayed under given panel.

sleep

This is a standard alias for **energysleep** (see [energysleep], page 134).

sorteditatomcolumn

Sorts the given column in the atom-property table.

Syntax:

`sorteditatomcolumn` ⟨column⟩

Operands:
⟨column⟩
The column to sort.

specifiedname

Set the specified name to that specified for all atoms which match the ASL specification.

Syntax:

`specifiedname` ⟨specified_name⟩ ⟨ASL⟩

Operands:
⟨specified_name⟩ ⟨ASL⟩
The first operand is the specified name which is to be applied to the atom. Only the first 20 characters of the specified name will be used. The second operand is the ASL specification for all the atoms which are to have the specified name applied.

spotcenter

Center the given atom in the Workspace. Make it the center of global rotation.

Syntax:

`spotcenter` ⟨atom_num⟩

Operands:
⟨atom_num⟩
The number of the atom which is to be centered in the Workspace and to become the center of rotation.

spotcenterpoint

Center the given point in the Workspace. Make it the center of global rotation.

Syntax:

`spotcenterpoint` ⟨ x y z ⟩

Operands:
⟨ x y z ⟩
The point is to be centered in the Workspace and to become the center of rotation.

startdiagnosis

This commands collects diagnostic information such as total system memory, available system memory, running maestro process peak virtual memory usage, current virtual memory usage, peak resident memory usage (only Linux), current resident memory usage (Only Linux), total swap disk space, free swap disk space, page fault etc. It also starts timer to collect user, sys, and real time information when explicitly called to stop it. All information are written into the file residing LINUX : $HOME/.schrodinger/maestro_diagnosis_<unique id>.txt Winows : AppData/Schrodinger//maestro_diagnosis_<unique id>.txt Unique file name is generated using second, minute, hour, day, month, and year. This file is created for each session.

Syntax:

`startdiagnosis`

stop

This is a standard alias for **energystop** (see [energystop], page 135).

stopdiagnosis

This commands collects diagnostic information such as total system memory, available system memory, running maestro process peak virtual mem-

ory usage, current virtual memory usage, peak resident memory usage (only Linux), current resident memory usage (Only Linux), total swap disk space, free swap disk space, page fault etc. It also stops timer which is started by MM_ID_START_DIAGNOSIS command and write user, system, and real time information for the tasks performed between start and stop operation. All information are written into the file residing LINUX : $HOME/.schrodinger/maestro_diagnosis_<unique id>.txt Winows : AppData/Schrodinger//maestro_diagnosis_<unique id>.txt Unique file name is generated using second, minute, hour, day, month, and year. This file is created for each session.

Syntax:

```
stopdiagnosis
```

strikebuildqsar

Runs a Build QSAR job.

Syntax:

strikebuildqsar *activity_property*=⟨ text ⟩ *max_pls_factors*=⟨ n ⟩ *method*=pls | pca | mlr *remove_outliers*=yes | no *supyintercept*=yes | no *which_descriptors*=all | subset ⟨ job name ⟩

Options:

activity_property
: The name of the activity property.
  Valid values: text strings
  Default value:

max_pls_factors
: How many factors for Partial Least Squares.
  Valid values: integers
  Default value: **1**
  Minimum: 0

method
: The regression method to use.
  Valid values: pls
  pca
  mlr
  Default value: **pls**

remove_outliers
: Set to true to automatically remove outliers.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

supyintercept
: Set to true to force the y-intercept in MLR fitting (MLRO & MLRS) to be zero.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

which_descriptors
: Which descriptors to use.

  Valid values: all
  subset
  Default value: **all**

Operands:

⟨ job name ⟩

The name of the job to run.

strikedeletemodel

Deletes the given model.

Syntax:

`strikedeletemodel` ⟨ model name ⟩

Operands:

⟨ model name ⟩

The name of the model to delete.

strikeexportmodel

Exports a model from Strike.

Syntax:

`strikeexportmodel` ⟨file⟩

Operands:
⟨file⟩
The file to export the Strike model to.

strikeextendselectdescriptor

Extends the selected descriptors via the given descriptor.

Syntax:

`strikeextendselectdescriptor` ⟨M2IO descriptor name⟩

Operands:
⟨M2IO descriptor name⟩
The M2IO data name of the descriptor to select.

strikeimportmodel

Imports a model to the Strike panels.

Syntax:

`strikeimportmodel` ⟨file⟩

Operands:
⟨file⟩
The file to import the Strike model from.

strikeplotmodel

Plots the given model.

Syntax:

`strikeplotmodel` ⟨model name⟩

Operands:
⟨model name⟩
The name of the model to plot.

strikepredict

Runs a prediction job.

Syntax:

`strikepredict` ⟨job name⟩

Operands:
⟨job name⟩
The name of the job to run.

strikeselectdescriptor

Selects the given descriptor

Syntax:

`strikeselectdescriptor` ⟨M2IO descriptor name⟩

Operands:
⟨M2IO descriptor name⟩
The M2IO data name of the descriptor to select.

strikeselectmodel

Selects only the given model.

Syntax:

`strikeselectmodel` ⟨ model name ⟩

Operands:
⟨ model name ⟩
The name of the model to select.

strikesimilarity

Runs a similarity job.

Syntax:

`strikesimilarity` *job_type*=atompairs | descriptors ⟨ job name ⟩

Options:

job_type	Whether to calculate atom pair or descriptor similarities.	
	Valid values:	atompairs descriptors
	Default value:	**atompairs**

Operands:
⟨ job name ⟩
The name of the job to run.

striketoggleselectdescriptor

Toggles the selection of the given descriptor on or off.

Syntax:

`striketoggleselectdescriptor` ⟨ M2IO descriptor name ⟩

Operands:
⟨ M2IO descriptor name ⟩
The M2IO data name of the descriptor to select.

structalignatoms

Sets the ASL that the next struct align job will operate on.

Syntax:

`structalignatoms` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The residues to align.

structalignatomsalt

Sets the separate ASL that the next struct align job will operate on. If the separate asl is set, the original ASL will be used as reference residues.

Syntax:

`structalignatomsalt` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The residues to align.

structalignstart

Start a struct align job with the current settings.

Syntax:

`structalignstart`

substructure

Specifies a set of atoms to be used as the "substructure" during a substructure minimization or dynamics simulation.

Syntax:

substructure *fillres*=yes | no *radius*=⟨ x ⟩ ⟨ ASL ⟩

Options:

fillres A boolean option that determines if the substructure definition will be expended to complete residue boundaries.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

radius The radius from the basic substructure definition within which atoms will be included in the substructure.

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:

⟨ ASL ⟩

A string in the atom specification language. All atoms which match this will be treated as part of the substructure in a substructure minimization.

substructurefile

This is a standard alias for **substructurefilewrite** (see [substructurefilewrite], page 717).

substructurefileread

Will read a .sbc file and replace the current substructure and constrained/fixed atom shells.

Syntax:

substructurefileread ⟨ sbc_file_name ⟩

Operands:

⟨ sbc_file_name ⟩

The complete name (including the *.sbc suffix) of the file from which the substructure information will be read.

substructurefilewrite

Will write a .sbc file with the current substructure and constrained/fixed atom shells.

Syntax:

substructurefilewrite *absolutecoords*=yes | no
aslformat=yes | no ⟨ sbc_file_name ⟩

Options:

absolutecoords
A boolean option which determines whether the atoms are considered to be at absolute coordinates (fixed at their current workspace locations) or at relative coordinates (at their locations in the input structures).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

aslformat A boolean option which determines whether the atoms specification to be written in ASL format. If user specifies 'aslformat=yes' then absolutecoords option will be ignored.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

⟨ sbc_file_name ⟩

The complete name (including the *.sbc suffix) of the file to which the substructure information will be written.

Aliases:

substructurefile (see [substructurefile], page 716)

substructureshell

Specifies a set of atoms forming a "shell" around the substructure defined by the "subs" command.

Syntax:

`substructureshell` *addatoms*=⟨ text ⟩ *constant*=⟨ x ⟩
fillres=yes | no *frozen*=yes | no *radius*=⟨ x ⟩ ⟨ shell_number ⟩
⟨ ASL-definition ⟩

Options:

addatoms A string in the atom specification language. All atoms which match this will be treated as part of the shell.

Valid values: text strings
Default value:

constant The harmonic force constant to be applied to the constrained shell.

Valid values: reals
Default value: **200**
Minimum: 0.0

fillres A boolean option which determines whether the shell is to be made up of complete residues.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

frozen A boolean option which determines whether the atoms in the shell are to be "frozen" during the substructure calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

radius The radius of a "shell" of atoms around the "substructure" during a substructure energy procedure.

Valid values: reals
Default value: **0**
Minimum: 0.0

Operands:

⟨ shell_number ⟩ ⟨ ASL-definition ⟩

The number of the shell. Shells are usually numbered from 1, but any sequence will work. The shell with the lowest number is defined relative to the atom in the substructure. Subsequent shells are defined relative to next lowest numbered shell.

superimpose

Perform a superposition using previously defined atoms or all corresponding atoms in all onscreen entries if all is used.

Syntax:

`superimpose` *inplace*=yes | no [all]

Options:

inplace Enable superposition to be performed in place (effectively just calculates the RMS and doesn't actually move any entries.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

Operands:

[all]

If all is present then an attempt will be made to superimpose all onscreen entries. Otherwise only the atom pairs defined by superimposeatom commands will be used to perform a superposition.

superimposeatom

Define an atom pair for which will be superimposed in a subsequent superposition operation.

Syntax:

`superimposeatom` ⟨ atom1 ⟩ ⟨ atom2 ⟩

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

Two atom numbers which represent an atom pair to be superimposed by a subsequent superimpose command. The two atoms must be from different entries and all superimposeatom commands must specify entries in the same order.

superimposeset

Use an ASL expression to define superposition atom pairs. The ASL set must define exactly the same number of atoms in each on-screen entry.

Syntax:

`superimposeset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language. This set must define exactly the same number of atoms in each on-screen entry and those atoms will become the basis for superposition.

superimposesmarts

Use a SMARTS expression to define superposition atom pairs. The SMARTS expression must define exactly the same number of atoms in each on-screen entry.

Syntax:

`superimposesmarts` ⟨ SMARTS ⟩

Operands:
⟨ SMARTS ⟩
A SMARTS expression. This must define exactly the same number of atoms in each on-screen entry and those atoms will become the basis for superposition.

surfaceactivegrid

Changes the settings of active grid for the given surface.

Syntax:

`surfaceactivegrid` *centerx*=⟨ x ⟩ *centery*=⟨ x ⟩ *centerz*=⟨ x ⟩
entry=⟨ text ⟩ *mode*=center | size | use | center_point
size=⟨ x ⟩ *surface*=⟨ text ⟩ *use*=yes | no *volume*=⟨ text ⟩
⟨ ASL ⟩

Options:

centerx The X coordinate of active grid center for the surface.
Valid values: reals
Default value: **0**

centery The Y coordinate of active grid center for the surface.

Valid values: reals
Default value: **0**

centerz The Z coordinate of active grid center for the surface.
Valid values: reals
Default value: **0**

entry The entry name of the entry that the surface belongs to.
Valid values: text strings
Default value:

mode Sets the mode of setting active grid for the given surface, mode is center, size, or use.
Valid values: center
size
use
center_point
Default value: **center**

size The size of active grid for the surface.
Valid values: reals
Default value: **10**
Minimum: 0.001

surface The name of the surface.
Valid values: text strings
Default value:

use Whether or not use active grid for eletron density map for the surface .
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

volume The name of the volume that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ ASL ⟩
The ASL defines the center of active grid.

surfacecontours

Control surface contour settings Edges are contoured when the distance along z to the nearest surface point behind it is at least the distance specified in the contour 'depth' value. The contour color is controlled by the red,

green, and blue option values. The contour line's thickness is controlled via the 'thickness' option. The color rendered for the contour is additionally affected by 'intensity' which affects the final color by blending the contour color and the surface color according to 'intensity'.

Syntax:

`surfacecontours` *blue*=⟨ x ⟩ *contours*=yes | no *depth*=⟨ x ⟩
green=⟨ x ⟩ *intensity*=⟨ x ⟩ *red*=⟨ x ⟩ *thickness*=⟨ x ⟩

Options:

blue Sets blue component of contour color

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

contours Display or hide contours on opaque surfaces. For transparent surfaces these are never shown regardless of what value this setting has.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

depth Display contours where the distance to the closest surface point is at least this distance in z.

Valid values:	reals
Default value:	**1.2**
Minimum:	0.0
Maximum:	5.0

green Sets green component of contour color

Valid values:	reals
Default value:	**0**
Minimum:	0.0
Maximum:	1.0

intensity Intensity of the contour. A value of 1.0 is the most intense showing only the contour color. A value of zero is the least intense showing none of the contour color and only the surface color.

Valid values:	reals
Default value:	**0.2**
Minimum:	0.0
Maximum:	1.0

red Sets red component of contour color

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 1.0

thickness Sets thickness of contour

Valid values: reals
Default value: **0.1**
Minimum: 0.0
Maximum: 1.0

surfacedarkencolor

Darkens the color by cavity depth for the given surface.

Syntax:

`surfacedarkencolor` *bycavitydepth*=yes | no *entry*=⟨ text ⟩
⟨ surface ⟩

Options:

bycavitydepth
Darkens the color by surface cavity depth.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

Operands:

⟨ surface ⟩

The name of the surface to set whether to darken colors by cavity depth when drawing the surface.

surfacedelete

Deletes the given surface.

Syntax:

`surfacedelete` *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to delete.

surfacedisplay

Displays or undisplays the given surface

Syntax:

`surfacedisplay` *display*=yes | no *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

display Sets whether or not to display the surface.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to display.

surfaceduplicate

Duplicates the given surface.

Syntax:

surfaceduplicate *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to duplicate.

surfaceextended

Creates a new extended radius surface for the current workspace.

Syntax:

surfaceextended *atom_radius_scaling*=⟨ x ⟩ *context*=entry | molecule | workspace | none | asl *grid_spacing*=⟨ x ⟩ *probe_radius*=⟨ x ⟩ *transparency*=⟨ x ⟩ name of the surface, followed by ASL defining the atoms to be surfaced

Options:

atom_radius_scaling
This is scaling of VdW radii.
Valid values: reals
Default value: **1**
Minimum: 0.5
Maximum: 10.0

context This option sets the context for generating a surface.
Valid values: entry
molecule
workspace
none
asl
Default value: **entry**

grid_spacing
This option sets the grid spacing for the surface.

Valid values: reals
Default value: **0.8**
Minimum: 0.01

probe_radius

This is the probe radius.

Valid values: reals
Default value: **1.4**
Minimum: 1.0

transparency

The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:
name of the surface, followed by ASL defining the atoms to be surfaced
Name of the extended radius surface to be created

surfaceextendedradiuscontext

Defines a set of atoms for which a surface can be clipped against with the surfaceextendedradius command.

Syntax:

`surfaceextendedradiuscontext` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms the surface will be clipped against.

surfaceextendedradiusset

Defines a set of atoms for which a surface can be created for with the surfaceextendedradius command.

Syntax:

`surfaceextendedradiusset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms the surface will be created for.

surfacemolecular

Creates a new molecular surface for the current workspace.

Syntax:

`surfacemolecular` *atom_radius_scaling*=⟨ x ⟩
boolean_operation=none | union | intersection | difference
context=entry | molecule | workspace | none | asl
edgesmoothing=yes | no *grid_spacing*=⟨ x ⟩ *probe_radius*=⟨ x ⟩
transparency=⟨ x ⟩ ⟨ surface name ⟩

Options:

atom_radius_scaling
This is scaling of VdW radii.

Valid values:	reals
Default value:	**1**
Minimum:	0.5
Maximum:	10.0

boolean_operation
This option sets the context for generating a surface.

Valid values:	none union intersection difference
Default value:	**none**

context This option sets the context for generating a surface.

Valid values:	entry molecule workspace none asl
Default value:	**none**

edgesmoothing
: Enables edge smoothing of truncated surfaces.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

grid_spacing
: This option sets the grid spacing for the surface.

  Valid values: reals
  Default value: **0.8**
  Minimum: 0.01

probe_radius
: This is the probe radius.

  Valid values: reals
  Default value: **1.4**
  Minimum: 1.0

transparency
: The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

  Valid values: reals
  Default value: **0**
  Minimum: 0.0
  Maximum: 100.0

Operands:
⟨ surface name ⟩
Name of the molecular surface to be created

surfacemolecularcontext

Defines a set of atoms for which a surface will be clipped against with the surfacemolecular command.

Syntax:

`surfacemolecularcontext` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms the surface will be clipped against.

surfacemolecularset

Defines a set of atoms for which a surface can be created for with the surfacemolecular command.

Syntax:

`surfacemolecularset` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The ASL expression which defines the atoms the surface will be created for.

surfacemolecularset2

Defines a set of atoms for surface boolean operation.

Syntax:

`surfacemolecularset2` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The ASL expression which defines the atoms for surface boolean operation.

surfaceoption

Set surface options.

Syntax:

`surfaceoption` *cavitydepth*=yes | no *cavityiterations*=⟨ n ⟩
defaulttransparency=⟨ n ⟩ *defaulttransparencyback*=⟨ n ⟩
defaulttransparencytogether=yes | no *extentstep*=⟨ x ⟩
fastrender=yes | no *keepcenter*=yes | no
lowqualitytransparency=rotating | always | never
maxmeshwidth=⟨ x ⟩ *meshwidth*=⟨ n ⟩ *minmeshwidth*=⟨ x ⟩
pairedcolor1=black | gray | dark_blue | blue | light_blue |
aquamarine | turquoise | spring_green | dark_green | green |
lime_green | yellow_green | yellow | orange | maroon | red |
pink | plum | magenta | blue_violet | white
pairedcolor2=black | gray | dark_blue | blue | light_blue |
aquamarine | turquoise | spring_green | dark_green | green |
lime_green | yellow_green | yellow | orange | maroon | red |
pink | plum | magenta | blue_violet | white
randomstipple=interlaced | always | never
scalemeshwidth=yes | no *showtable*=yes | no
singlecolor=black | gray | dark_blue | blue | light_blue |
aquamarine | turquoise | spring_green | dark_green | green |
lime_green | yellow_green | yellow | orange | maroon | red |
pink | plum | magenta | blue_violet | white
singlelayertransparency=yes | no *smoothing*=⟨ n ⟩ *style*=solid
| mesh | dots *usecheesy*=yes | no

Options:

cavitydepth

Enables weighting surface colors by cavity depth.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

cavityiterations

Sets number of iterations of cavity depth coloring algorithm.

Valid values:	integers
Default value:	**50**
Minimum:	5
Maximum:	200

defaulttransparency

Sets default front surface transparency.

Valid values:	integers
Default value:	**0**
Minimum:	0
Maximum:	100

defaulttransparencyback

Sets default back surface transparency.

Valid values: integers
Default value: **0**
Minimum: 0
Maximum: 100

defaulttransparencytogether
Set the flag of adjusting default front and back surfaces together.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

extentstep Step size of increasing or decreasing density map extent.

Valid values: reals
Default value: **1**
Minimum: 0.1

fastrender Enables optimized rendering of transparent surfaces.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

keepcenter Set the flag of keeping electron density map center while translation is performed.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

lowqualitytransparency
Specifies drawing quality for transparent surfaces when rotating. Valid values are 'When rotating', 'Always', and 'Never'.

Valid values: rotating
always
never
Default value: **never**

maxmeshwidth
Maximum thickness of scaled mesh lines.

Valid values: reals
Default value: **1**
Minimum: 0.5
Maximum: 10.0

meshwidth Thickness of mesh lines.

Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 20

minmeshwidth
Minimum thickness of scaled mesh lines.

Valid values:	reals
Default value:	**0.5**
Minimum:	0.1
Maximum:	0.5

pairedcolor1

The default first color for a paired surface

Valid values:	black gray dark_blue blue light_blue aquamarine turquoise spring_green dark_green green lime_green yellow_green yellow orange maroon red pink plum magenta blue_violet white
Default value:	**blue**

pairedcolor2

The default second color for a paired surface

Valid values: black
gray
dark_blue
blue
light_blue
aquamarine
turquoise
spring_green
dark_green
green
lime_green
yellow_green
yellow
orange
maroon
red
pink
plum
magenta
blue_violet
white

Default value: **red**

randomstipple

Specifies when to use random stipple mask for lower-quality transparency. The default is to only use the random stipple pattern for interlaced stereo. Valid values are 'interlaced', 'always', and 'never'.

Valid values: interlaced
always
never

Default value: **interlaced**

scalemeshwidth

Change mesh width when zooming.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showtable Set the flag of showing surface manager (table).

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

singlecolor The default color for a standard surface.

Valid values: black
gray
dark_blue
blue
light_blue
aquamarine
turquoise
spring_green
dark_green
green
lime_green
yellow_green
yellow
orange
maroon
red
pink
plum
magenta
blue_violet
white

Default value: **gray**

singlelayertransparency

Enables single layer transparency.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

smoothing Sets number of surface color smoothing iterations.

Valid values: integers
Default value: **3**
Minimum: 0
Maximum: 20

style Sets the drawing style.

Valid values: solid
mesh
dots

Default value: **solid**

usecheesy Off (the default) uses the high-quality but possibly slower-to-draw transparency. On uses the lower-quality cheesy translucency, but in some cases this will be significantly faster to draw. Cheesy translucency mimics transparency by mapping level of transparency to amount of holes in the surface.

Valid values: boolean (true|false; yes|no; y|n; on|off)

Default value: **false**

surfacepbrsp

Creates a new Electrostatic Potential Surface for the current workspace.

Syntax:

`surfacepbrsp` *coarsedepth*=⟨ n ⟩ *coarsespacing*=⟨ x ⟩ *espmapscale*=⟨ x ⟩ *focusdepth*=⟨ n ⟩ *focusspacing*=⟨ x ⟩ *grid_spacing*=⟨ x ⟩ *isosurfaces*=yes | no *isovalue*=isovalue1 | isovalue2 | isovalue3 | isovalue4 *mgextension*=⟨ x ⟩ *pbrspmap*=yes | no *soluteconstant*=⟨ x ⟩ *solventconstant*=⟨ x ⟩ *solventradius*=⟨ x ⟩ *temperature*=⟨ x ⟩ *transparency*=⟨ x ⟩ Name of the surface

Options:

coarsedepth

This is the MG depth (nlev) for PBE coarse grid.

Valid values: integers
Default value: **2**
Minimum: 1

coarsespacing

This option sets the grid spacing for the PBE coarse grid.

Valid values: reals
Default value: **1.2**
Minimum: 0.01

espmapscale

This option sets the scale of ESP mapping range.

Valid values: reals
Default value: **0.25**
Minimum: 0.01
Maximum: 1.0

focusdepth This is the MG depth (nlev) in PBE. for PBE focused grid.

Valid values: integers
Default value: **4**
Minimum: 1

focusspacing

This option sets the grid spacing for the PBE focused grid.

Valid values: reals

Default value: **0.8**
Minimum: 0.01

grid_spacing
This option sets the grid spacing for the surface.

Valid values: reals
Default value: **0.8**
Minimum: 0.01

isosurfaces
This option indicates whether a paired (+ve and -ve isovalue) surfaces will be generated from the PB calculation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

isovalue This is the initial isovalue in PBE.

Valid values: isovalue1
isovalue2
isovalue3
isovalue4
Default value: **isovalue1**

mgextension
This option sets the grid extension for the PBE.

Valid values: reals
Default value: **5**
Minimum: 0.01

pbrspmap This option indicates whether the PBRSP from the PB calculation will be mapped onto a molecular surface.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

soluteconstant
This is the dielectric constant for solute.

Valid values: reals
Default value: **1**
Minimum: 1.0

solventconstant
This is the dielectric constant for solvent.

Valid values: reals
Default value: **80**
Minimum: 1.0

solventradius
This is the solvent radius.

Valid values: reals
Default value: **1.4**
Minimum: 0.01

temperature
This is the temperature in PBE.

Valid values: reals
Default value: **298**
Minimum: 0.01

transparency
The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:

Name of the surface

Name of the Electrostatic Potential Surface to be created

surfacepbrspset

Defines a set of atoms for which a surface can be created for with the surfacepbrsp command.

Syntax:

`surfacepbrspset` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

The ASL expression which defines the atoms the surface will be created for.

surfacerename

Renames the given surface.

Syntax:

surfacerename *entry*=⟨ text ⟩ *newname*=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

newname The new name for the surface.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to rename.

surfaceresolution

Set surface displaying resolution options.

Syntax:

surfaceresolution *lowresstyle*=solid | mesh | dot
resolution=high | low

Options:

lowresstyle
Specifies drawing style for sloid surfaces when rotating. Valid values are 'solid', 'mesh', and 'dot'.
Valid values: solid
mesh
dot
Default value: **solid**

resolution Set the resolution for drawing solid surfaces to low or high .
Valid values: high
low
Default value: **high**

surfacescheme

Sets the color scheme for the given surface.

Syntax:

`surfacescheme` *color*=⟨ text ⟩ *colorramp*=⟨ text ⟩
datarange=on_surface | entire_volume *defaultcolor*=⟨ n ⟩
dotradius=⟨ x ⟩ *entry*=⟨ text ⟩ *espcolorramp*=⟨ text ⟩
espmax=⟨ x ⟩ *espmin*=⟨ x ⟩ *iterations*=⟨ n ⟩ *linewidth*=⟨ x ⟩
mapmax=⟨ x ⟩ *mapmin*=⟨ x ⟩ *negativecolor*=⟨ text ⟩
scheme=⟨ text ⟩ *schemevolume*=⟨ text ⟩ *showlegend*=yes | no
smooth=yes | no ⟨ surface ⟩

Options:

color Sets the color for a constant color.
Valid values: text strings
Default value: **gray**

colorramp Sets the color ramp for the surface when color scheme is Map Values From Volume.
Valid values: text strings
Default value: **redwhiteblue**

datarange Sets the data range to which we apply color ramp.
Valid values: on_surface
entire_volume
Default value: **on_surface**

defaultcolor
Sets default constant color for a new surface.
Valid values: integers
Default value: **2**
Minimum: 1
Maximum: 21

dotradius Sets radius for surface dots, unit is Angstroms.
Valid values: reals
Default value: **0.1**
Minimum: 0.0001

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

espcolorramp
Sets the color ramp for the surface when color scheme is electrostatic potential (ESP).

Valid values: text strings
Default value: **redwhiteblue**

espmax Sets maximum ESP value for ESP color scheme.

Valid values: reals
Default value: **0.3**
Minimum: 0.001

espmin Sets minimum ESP value for ESP color scheme.

Valid values: reals
Default value: **-0.3**
Maximum: -0.001

iterations Specify the number of iterations for Laplacian smoothing.

Valid values: integers
Default value: **35**
Minimum: 0
Maximum: 100

linewidth Sets line width for surface mesh, unit is pixel.

Valid values: reals
Default value: **0.025**
Minimum: 0.0001

mapmax Sets maximum mapping value.

Valid values: reals
Default value: **0**

mapmin Sets minimum mapping value.

Valid values: reals
Default value: **0**

negativecolor
Sets the color for the negative surface.

Valid values: text strings
Default value: **white**

scheme Sets the color scheme.

Valid values: text strings
Default value:

schemevolume
Sets the color scheme volume name.

Valid values: text strings
Default value:

showlegend
Sets whether or not show legends in workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

smooth Turn on/off surface smoothing processing.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:

⟨ surface ⟩

The name of the surface to set the color scheme for.

surfacesetcomment

Changes the comments of a given surface.

Syntax:

`surfacesetcomment` *comment*=⟨ text ⟩ *entry*=⟨ text ⟩ ⟨ surface ⟩

Options:

comment The new comments for the surface.

Valid values: text strings
Default value:

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

Operands:

⟨ surface ⟩

The name of the surface to change the comments for.

surfacesetisovalue

Changes the isovalue of a given surface.

Syntax:

`surfacesetisovalue` *entry*=⟨ text ⟩ *isovalue*=⟨ x ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.

Valid values: text strings
Default value:

isovalue The new isovalue for the surface.

Valid values: reals
Default value: **0.3**

Operands:

⟨ surface ⟩

The name of the surface to change the isovalue for.

surfacesettings

This keyword is used to set various options associated with surface types.

Syntax:

`surfacesettings` *default_color*=black | gray | dark_blue | blue | light_blue | aquamarine | turquoise | spring_green | dark_green | green | lime_green | yellow_green | yellow | orange | maroon | red | pink | plum | magenta | blue_violet | white *default_paired_color*=black | gray | dark_blue | blue | light_blue | aquamarine | turquoise | spring_green | dark_green | green | lime_green | yellow_green | yellow | orange | maroon | red | pink | plum | magenta | blue_violet | white ⟨ model_name ⟩

Options:

default_color
The default color for the surface.

Valid values:	black gray dark_blue blue light_blue aquamarine turquoise spring_green dark_green green lime_green yellow_green yellow orange maroon red pink plum magenta blue_violet white
Default value:	**yellow_green**

default_paired_color
: The default color for the paired surface.

Valid values:	black gray dark_blue blue light_blue aquamarine turquoise spring_green dark_green green lime_green yellow_green yellow orange maroon red pink plum magenta blue_violet white
Default value:	**red**

Operands:
⟨ model_name ⟩
The name of a surface type which has settings like default color and default paired color.

surfacesetviewasl

This command sets a surface to only display the portions of the surface within a given distance of a given ASL.

Syntax:

`surfacesetviewasl` *distance*=⟨ x ⟩ *entry*=⟨ text ⟩ *surface*=⟨ text ⟩
use=yes | no ASL

Options:

distance The distance to the ASL.

Valid values:	reals
Default value:	**5**

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

surface The name of the surface to operate on.
Valid values: text strings
Default value:

use Determines whether or not to use the view by ASL.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

Operands:
ASL
The asl which controls the visible surface.

surfacesplitpair

Split paired surfaces into two separate surfaces.

Syntax:

`surfacesplitpair` *entry*=⟨ text ⟩ *volume*=⟨ text ⟩ ⟨ surface ⟩

Options:

entry The entry name of the entry that the surface belongs to.
Valid values: text strings
Default value:

volume The name of the volume that the surface belongs to.
Valid values: text strings
Default value:

Operands:
⟨ surface ⟩
The name of the surface to be splited.

surfacestyle

Sets the drawing style for the given surface.

Syntax:

surfacestyle *entry*=⟨ text ⟩ *style*=solid | mesh | dots ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

style Sets the drawing style.
Valid values: solid
mesh
dots
Default value: **solid**

Operands:
⟨ surface ⟩
The name of the surface to set the style for.

surfacetransparency

Sets the transparency for the given surface.

Syntax:

surfacetransparency *entry*=⟨ text ⟩ *transparency*=⟨ x ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

transparency
Sets the transparency percentage.
Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:
⟨ surface ⟩

The name of the surface to set transparency on.

surfacetransparencyback

Sets the transparency for the given surface (back).

Syntax:

surfacetransparencyback *entry*=⟨ text ⟩ *transparency*=⟨ x ⟩ ⟨ surface ⟩

Options:

entry The entry that the surface belongs to.
Valid values: text strings
Default value:

transparency
Sets the transparency percentage.
Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:
⟨ surface ⟩
The name of the surface to set transparency on.

surfacevdw

Creates a new vdW surface for the current workspace.

Syntax:

surfacevdw *asl*=⟨ text ⟩ *atom_radius_scaling*=⟨ x ⟩ *context*=entry | molecule | workspace | none | asl *grid_spacing*=⟨ x ⟩ *transparency*=⟨ x ⟩ Name of the surface

Options:

asl This option sets the ASL specification for the atoms which define the surface.

Valid values: text strings
Default value:

atom_radius_scaling
This is scaling of VdW radii.

Valid values: reals
Default value: **1**
Minimum: 0.5
Maximum: 10.0

context This option sets the context for generating a surface.

Valid values: entry
molecule
workspace
none
asl
Default value: **entry**

grid_spacing
This option sets the grid spacing for the surface.

Valid values: reals
Default value: **0.8**
Minimum: 0.01

transparency
The transparency option controls how transparent the surface appears. A value of 100 means 100 percent transparent. 0 means completely opaque.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:

Name of the surface

Name of the vdW surface to be created

surfacevdwcontext

Defines a set of atoms for which a surface will be clipped against with the surfacevdw command.

Syntax:

`surfacevdwcontext` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms the surface will be clipped against.

surfacevdwset

Defines a set of atoms for which a surface can be created for with the surfacevdw command.

Syntax:

`surfacevdwset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms the surface will be created for.

surfaceviewaslset

Defines a set of atoms for the View by ASL property which will be used by the surfacesetviewasl command.

Syntax:

`surfaceviewaslset` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The ASL expression which defines the atoms that will be used for the View by ASL property.

swapptworkspace

Swaps the project table and the Workspace.

Syntax:

`swapptworkspace`

symmetrizeworkspace

Symmetrizes the workspace finding the point groups.

Syntax:

`symmetrizeworkspace` *tolerance*=⟨ x ⟩ analyze|update

Options:

tolerance Tolerance for finding the pointing groups and symmetrizing the workspace

Valid values:	reals
Default value:	**0.04**
Minimum:	0.04
Maximum:	1.00

Operands:

analyze|update

If analyze, finds the point groups for the current tolerance. If update, symmetrize the Workspace with the current tolerance.

system

Execute a command from the system.

Syntax:

`system` ⟨ command ⟩

Operands:

⟨ command ⟩

A command which is to be executed from the current shell.

tablealigncolumn

Set the alignment of the specified column in the current table.

Syntax:

`tablealigncolumn` *alignment*=left | center | right ⟨ columnname ⟩ ⟨ alignment ⟩

Options:

alignment The alignment to be set for the column. Valid values are "left", "center", or "right".

Valid values:	left center right
Default value:	**left**

Operands:

⟨ columnname ⟩ ⟨ alignment ⟩

The name of the column to align. This is the name displayed in the column header.

tablecopyprop

Set property values for selected project entries from a value associated with entry row in the current Project Table, and indirectly associated with the entry. This is similar to entrycopyprop command, but the from property value comes from a table row or from an entry group associated with the table row, not from the entry itself. The to property value will be copied from the from property for the entries selected by the ESL expression. If the to property does not already exist, it will be created. When the data type for the two properties is not the same, a reasonable attempt is made to convert between them. If a selected entry has no value for the from property, or the conversion from a string value fails, the value is generally cleared for the to value for that entry. Entry names are never cleared.

Syntax:

`tablecopyprop` *from*=⟨ text ⟩ *to*=⟨ text ⟩ ⟨ ESL ⟩

Options:

from The user name of the table property which provides the values to be copied. Currently supported table properties are Group Title and Row .

Valid values: text strings
Default value:

to The name of the property to be modified. This can be either the user name or the m2io data name for the property, if the to property exists. If the to property does not exist, it will be created using the data type of the from property, with user as the author. If the property is the entry name (e.g. Entry Name or s_m_entry_name), an entryrename with replace=no will be done. If the property is the 'included in Workspace' property (In or b_m_entry_in_workspace), then entrywsinclude or entrywsexclude will be done, if needed.

Valid values: text strings
Default value:

Operands:

⟨ ESL ⟩

The ESL expression specifies for which entries the property values are to be copied.

tablegroupsort

Reorder the groups in the specified table.

Syntax:

`tablegroupsort` *field*=⟨ text ⟩ *order*=ascending | descending
source=group | firstentry ⟨ table ⟩ all | ⟨ group name list ⟩

Options:

field The name of the group/entry property to be used as sort-key attribute. Use grouptitle to reorder groups based on group's title. Use property name to reorder groups based on first entry's property value.

Valid values: text strings
Default value: **grouptitle**

order This option sets order of sorted values to be either ascending or descending.

Valid values: ascending
descending
Default value: **ascending**

source This option determines whether the 'field' option takes 'group property name' or 'entry property name'. Valid values are group or firstentry .

Valid values: group
firstentry
Default value: **group**

Operands:

⟨ table ⟩ all | ⟨ group name list ⟩

The name of the table to sort. Comma separated list of group names. Or 'all' to sort all groups.

tableresizecolumn

Set the width of the specified column in the current table.

Syntax:

`tableresizecolumn` ⟨ columnname ⟩ ⟨ width ⟩

Operands:

⟨ columnname ⟩ ⟨ width ⟩

The name of the column to resize. This is the name displayed in the column header. The width to set the column to. This is the new width for the column.

tablesort

Sort current entry selection in the specified table.

Syntax:

`tablesort` *field*=⟨ text ⟩ *order*=ascending | descending ⟨ table ⟩

Options:

field The name of the property to be sorted on. This option has been deprecated in favor of the tablesortfields command. This option does nothing now.

Valid values: text strings
Default value: **s_m_entry_name**

order This option sets order of sorted values to be either ascending or descending. This option has been deprecated in favor of the tablesortfields command. This option does nothing now.

Valid values: ascending
descending
Default value: **ascending**

Operands:

⟨ table ⟩

The name of the table to sort.

tablesortall

Sort all rows in the specified table.

Syntax:

`tablesortall` ⟨ table ⟩

Operands:

⟨ table ⟩

The name of the table to sort.

tablesortfields

Sets multiple fields for sorting.

Syntax:

`tablesortfields` *groupbypropertyname*=⟨ text ⟩
groupentriesbeforesort=yes | no *selectoption*=nentries | npercentofentries | nentriesineachgroup | npercentofentriesineachgroup *selecttop*=yes | no *sortentries*=yes | no *sortgroups*=yes | no *topn*=⟨ n ⟩ ⟨ field ⟩ ⟨ ascending | descending ⟩

Options:

groupbypropertyname
If groupentriesbeforesort is true then entries will be grouped based on this property.
Valid values: text strings
Default value: **Title**

groupentriesbeforesort
This option sets if we should group the entries before sorting.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **false**

selectoption
An option which controls the selection of entries after sorting.
Valid values: nentries
npercentofentries
nentriesineachgroup
npercentofentriesineachgroup
Default value: **nentries**

selecttop This option sets if we should select the given number of entries from the top, in the project table, after sorting the rows.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **false**

sortentries
Sorts the entries when tablesort or tablesortall command is executed.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **true**

sortgroups Sorts the groups when tablesort or tablesortall command is executed.
Valid values: boolean (true | false; yes | no; y | n; on | off)
Default value: **false**

topn This option along with selecttop option sets number of entries to be selected from the top, in the project table, after sorting the rows.

Valid values: integers
Default value: **5**

Operands:

⟨ field ⟩ ⟨ ascending | descending ⟩

The fields to sort together with the sort order: ascending or descending.

tablesortgroupfields

Sets multiple fields for sorting groups.

Syntax:

`tablesortgroupfields` *sortgroupoption*=grouptitle | firstentrypropertysortfield | firstentrypropertygroupsortfield

Options:

sortgroupoption

An option which controls the sorting of groups.

Valid values: grouptitle
firstentrypropertysortfield
firstentrypropertygroupsortfield
Default value: **grouptitle**

tableunselectnonsubset

Unselects all entries which are not in the current table's subset.

Syntax:

`tableunselectnonsubset`

targetentry

Designate the target entry for Workspace entry feedback, subject to the entryfeedbackshow and feedbackproperties preferences.

Syntax:

`targetentry` *target*=⟨ text ⟩

Options:

target This option sets the name (ID string) of the target entry. If there is no target entry, this string should be empty. Generally, the target entry should be set by eplayer commands and by entrywsincludeonly commands.

Valid values: text strings
Default value:

tile

Spread the on-screen entries out in a tile-pattern.

Syntax:

`tile`

tileapplyview

Set the view for all tiles from that of the specified Workspace tile.

Syntax:

`tileapplyview` ⟨ tile_name ⟩

Operands:
⟨ tile_name ⟩
The name which identifies the contents of the tile. This name will be the index of an entry group (greater than 0), or 0: followed by the entry ID for un ungrouped entry (or Scratch entry). A tile named 0: (with no entry ID) does not belong to any entry or entry group. In tile by surface mode, the entry ID will be followed by the name of the special surface for that tile.

tileapplyzoom

Set the zoom for all tiles from that of the specified Workspace tile.

Syntax:

`tileapplyzoom` ⟨ tile_name ⟩

Operands:
⟨ tile_name ⟩
The name which identifies the contents of the tile. This name will be the index of an entry group (greater than 0), or 0: followed by the entry ID for un ungrouped entry (or Scratch entry). A tile named 0: (with no entry ID) does not belong to any entry or entry group. In tile by surface mode, the entry ID will be followed by the name of the special surface for that tile.

tilemode

Control over mode which effectively creates separate workspaces for different entries.

Syntax:

`tilemode` *max_columns*=⟨ n ⟩ *mode*=entry | group | surface
tile=yes | no *transformall*=yes | no

Options:

max_columns
Specifies the maximum number of tile columns.

Valid values:	integers
Default value:	**4**
Minimum:	1
Maximum:	10

mode Specifies method used to assign entries to tiles

Valid values:	entry group surface
Default value:	**entry**

tile A boolean option which determines whether to display entries in tile mode.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

transformall
A boolean option which determines whether to apply transforms globally (on) or to tiles individually (off).

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**true**

tilemove

Move the tile identified by the first operand to the location of the tile identified by the second operand.

Syntax:

`tilemove` ⟨move_tile_name⟩ ⟨to_tile_name⟩

Operands:

⟨move_tile_name⟩ ⟨to_tile_name⟩

The names of two Workspace tiles, the first of which is to be moved to the location currently occupied by the second. This name will be the index of an entry group (greater than 0), or 0: followed by the entry ID for un ungrouped entry (or Scratch entry). A tile named 0: (with no entry ID) does not belong to any entry or entry group. In tile by surface mode, the entry ID will be followed by the name of the special surface for that tile.

tilereset

Reset transform for the specified Workspace tile to that which was in effect the last time the current tile mode was entered.

Syntax:

`tilereset` ⟨tile_name⟩

Operands:

⟨tile_name⟩

The name which identifies the contents of the tile. This name will be the index of an entry group (greater than 0), or 0: followed by the entry ID for un ungrouped entry (or Scratch entry). A tile named 0: (with no entry ID) does not belong to any entry or entry group. In tile by surface mode, the entry ID will be followed by the name of the special surface for that tile.

tileresetall

Reset transform for all tiles in Workspace to that which was in effect the last time the current tile mode was entered. This also sets the global transform.

Syntax:

`tileresetall`

timingsetup

Set timing experiment variables.

Syntax:

`timingsetup` *duration*=⟨ x ⟩ *file*=⟨ text ⟩ *period*=⟨ x ⟩

Options:

duration This option sets the maximum duration (in seconds) for the timing experiment. A value of 0.0 means that there is no limit on the time, and that the timing experiment will continue until a timingstop command is done, maestro quits, or a new timingstart command is done.

Valid values: reals
Default value: **0**

file The timing data file.

Valid values: text strings
Default value:

period This option sets the sampling period (in seconds) for accumulating and reporting timing results. A value of 0.0 means that there is no periodic sampling, so that results are accumulated for the entire duration of the timing experiment and reported at the end.

Valid values: reals
Default value: **0**
Minimum: 0.0

timingstart

Start a timing experiment with the settings from the last timingsetup command. This first stops any timing experiment currently in progress (from a previous timingstart command).

Syntax:

`timingstart`

timingstop

Stop any timing experiment currently in progress (from a previous timingstart command). Timing results for a partially completed sampling period are not reported.

Syntax:

`timingstop`

toggleseditatomboolean

Toggles the boolean property for the given row in the atom-property table.

Syntax:

`toggleseditatomboolean` *property*=⟨ text ⟩ ⟨ row number ⟩

Options:

property The M2IO data name of the property to toggle
Valid values: text strings
Default value:

Operands:
⟨ row number ⟩
The operand is the row number to toggle.

toggleselecteditatom

Toggles the selection of the given row in the edit atom-property table.

Syntax:

`toggleselecteditatom` ⟨ row number ⟩

Operands:
⟨ row number ⟩
The operand is the row number to toggle-select.

toggleselecteditproperty

Toggles the selection of the given property in the edit property list.

Syntax:

`toggleselecteditproperty` ⟨ property ⟩

Operands:
⟨ property ⟩
The operand is a property from the edit property list.

torsioncheck

Specifies four atoms which define a torsion to be checked during a conformational search.

Syntax:

`torsioncheck` *maximum*=⟨ x ⟩ *minimum*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩

Options:

maximum The maximum allowed value for the torsion. during the torsion check.

Valid values:	reals
Default value:	**180**
Minimum:	0.0

minimum The minimum allowed value for the torsion check (Degrees)

Valid values:	reals
Default value:	**0**

Operands:
⟨ atom1 ⟩ ⟨ atom2 ⟩ ⟨ atom3 ⟩ ⟨ atom4 ⟩
The numbers of four atoms which define a torsion angle to be checked during the conformational search. Note that specifying a-b-c-d is the same as specifying d-c-b-a.

torsiongroup

Sets the current torsion group and the conformation within that group

Syntax:

`torsiongroup` ⟨ group_name ⟩ ⟨ conformation_name ⟩

Operands:
⟨ group_name ⟩ ⟨ conformation_name ⟩
The first operand must be the name of a torsion group within the current fragment mode. The second operand must be the name of a conformation within that torsion group.

trajectoryexport

Export selected frames from the current project to a file or to PT

Syntax:

`trajectoryexport`

trajectoryexportatoms

Specify the set of atoms to include in each exported frame

Syntax:

`trajectoryexportatoms` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language.

trajectoryexportmovieoptions

Sets the options for the export movie for trajectory player.

Syntax:

`trajectoryexportmovieoptions` *format*=avi | mpeg | quicktime
frames=selected | all

Options:

format This option sets the format of movie file.

Valid values: avi
mpeg
quicktime

Default value: **mpeg**

frames This option sets the number of frames to be exported. The two options are: export selected frames or all frames

Valid values: selected
all

Default value: **all**

trajectoryexportoptions

Sets the options for the export structure for trajectory player.

Syntax:

`trajectoryexportoptions` *exportto*=pt | file *filename*=⟨ text ⟩
frames=selected | currentsingle | currentmultiple

Options:

exportto This option sets the destination of the frame to be written. Valid values are 'pt' or 'file'.

Valid values: pt
file

Default value: **file**

filename If the frames are to be exported to a file, specifies filename to be used for the file to be exported

Valid values: text strings
Default value:

frames This option sets the number of frames to be exported. The three options are: export selected frames or current frame only as single entry or current frame only as multiple entries.

Valid values: selected
currentsingle
currentmultiple

Default value: **currentsingle**

trajectoryplayerdisplayatoms

Specify the set of atoms to be displayed in each frame

Syntax:

`trajectoryplayerdisplayatoms` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language.

trajectoryplayergoto

Go to the specified frame in the ordered sequence of trajectory frames.

Syntax:

`trajectoryplayergoto` ⟨ frame_number ⟩

Operands:

⟨ frame_number ⟩

The frame number within the ordered sequence of trajectory frames, which is to be included in the workspace. This frame is recorded as an option to the trajectoryplayersettings command.

trajectoryplayergotofirst

Go to the start frame in the ordered sequence of trajectory frames.

Syntax:

```
trajectoryplayergotofirst
```

trajectoryplayergotolast

Go to the end frame in the ordered sequence of trajectory frames.

Syntax:

```
trajectoryplayergotolast
```

trajectoryplayersettings

Set trajectoryplayer state variables.

Syntax:

trajectoryplayersettings *arep*=⟨ n ⟩ *brep*=⟨ n ⟩ *crep*=⟨ n ⟩ *drawfaster*=yes | no *end*=⟨ n ⟩ *entry*=⟨ text ⟩ *frame*=⟨ n ⟩ *frameduration*=⟨ x ⟩ *hideclipping*=yes | no *playmode*=loop | reverse | once *position*=none | superimpose | center | workspace *referframe*=⟨ n ⟩ *showaxes*=yes | no *showonlyspecified*=yes | no *showsimbox*=yes | no *showtimes*=yes | no *smoothing*=⟨ n ⟩ *start*=⟨ n ⟩ *step*=⟨ n ⟩ *update_ssa*=yes | no

Options:

arep This option sets the number of replications in the a direction.

Valid values:	integers
Default value:	**1**
Minimum:	1

brep This option sets the number of replications in the b direction.

Valid values:	integers
Default value:	**1**
Minimum:	1

crep This option sets the number of replications in the c direction.

Valid values:	integers
Default value:	**1**
Minimum:	1

drawfaster If true, user lower quality drawing to speed up continuous trajectory play.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

end This option sets the number of the trajectory end frame.

Valid values: integers
Default value: **0**
Minimum: 0

entry This option sets the name (entry ID) of the entry whose trajectory is displayed.

Valid values: text strings
Default value:

frame This option sets the number of the trajectory frame being displayed.

Valid values: integers
Default value: **0**
Minimum: 0

frameduration
This option determines the minimum duration, in seconds, of each displayed (entry) trajectory frame during continuous play. The actual frame duration may be longer than the specified value, due to time required for drawing and screen update.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 10.0

hideclipping
If true, hide clipping planes during continuous trajectory play. TRAJECTORYEXPORTOPTIONS

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

playmode This option sets the mode for continuous play. Valid values are “loop”, “reverse”, or “once”. These cause play to continue, change direction, or stop, respectively, when reaching either end of the current trajectory frame range.

Valid values: loop
reverse
once
Default value: **once**

position This option sets the mode for repositioning frame coordinates. Valid values are "none", "superimpose", "workspace", or "center". For the value "none", do not adjust positions. For the value "superimpose", superimpose current trajectory frame onto specified reference frame, aligning atoms that match the current superimpose ASL expression. For the value "workspace", superimpose current trajectory frame onto specified workspace atoms, aligning atoms that match the current superimpose ASL expression. For the value "center", recenter selected molecules back to simulation box, using the same superimpose ASL expression to select atoms in the molecules of interest.

Valid values: none
superimpose
center
workspace
Default value: **center**

referframe This option sets the number of the trajectory reference frame for superimpose.

Valid values: integers
Default value: **0**
Minimum: 0

showaxes If true, display x, y, z axes for trajectory.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showonlyspecified
If true, display only the atoms specified by trajectoryplayerdisplayatoms ASL (and optionally those within a specified distance) when a trajectory frame is shown. If false, display whichever atoms have currently been made visible in Maestro.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

showsimbox
If true, display simulation box for trajectory.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

showtimes If true, perform timing tests while showing trajectory frames, and display timing results.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

smoothing This option sets the trajectory smoothing.

Valid values: integers
Default value: **1**
Minimum: 1

start This option sets the number of the trajectory start frame.

Valid values: integers
Default value: **0**
Minimum: 0

step This option sets the number of the trajectory frames per step (skipping 1 less than this).

Valid values: integers
Default value: **1**
Minimum: 1

update_ssa

If true, recalculate secondary structure assignment for each frame.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

trajectoryplayerstepahead

Go to the next frame in the ordered sequence of trajectory frames, if there is one after the frame specified in the trajectoryplayersettings command. The step value in the trajectoryplayersettings command determines how many frames to go forward.

Syntax:

```
trajectoryplayerstepahead
```

trajectoryplayerstepback

Go to the previous frame in the ordered sequence of trajectory frames, if there is one before the frame specified in the trajectoryplayersettings command. The step value in the trajectoryplayersettings command determines how many frames to go back.

Syntax:

`trajectoryplayerstepback`

trajectoryplayersuperimposeatoms

Specify the set of atoms used in superimposition

Syntax:

`trajectoryplayersuperimposeatoms` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

A string in the atom specification language.

trajectoryresolvemeasurements

Sets the options for the resolving trajectory measurements

Syntax:

`trajectoryresolvemeasurements` *use*=wsonly | trajonly | both

Options:

use This option sets the preference of the user whether to use trajectory measuremnts or ws measurements or both

Valid values:	wsonly trajonly both
Default value:	**wsonly**

transform

Specify what is to be transformed.

Syntax:

`transform` *centerlocal*=centroid | atom *centerrotation*=yes | no
contactdisplay=yes | no *gui*=none | rotate | translate
hbonddisplay=yes | no *includealternate*=yes | no
rsensitivity=⟨ x ⟩ *scope*=global | local
showrotationcenter=yes | no *smootherrotation*=yes | no
threshhold=⟨ n ⟩ *tsensitivity*=⟨ x ⟩ [reset]

Options:

centerlocal
Whether local center is an atom or centroid of group of atoms
Valid values: centroid
atom
Default value: **centroid**

centerrotation
Set the flag of center rotation.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

contactdisplay
This option determines whether Contact markers will be displayed for local transformations.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

gui Whether rotation, translation or nothing is performed
Valid values: none
rotate
translate
Default value: **rotate**

hbonddisplay
This option determines whether H-bond markers will be displayed for local transformations.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

includealternate
Whether to include alternate positions in local transformations to atom coordinates.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

rsensitivity
Mouse rotation sensitivity. Larger is more sensitive.
Valid values: reals

Default value: **75**
Minimum: 1.0
Maximum: 500.0

scope Whether global or local transformations are done

Valid values: global
local
Default value: **global**

showrotationcenter
Set the flag of displaying rotate center.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

smootherrotation
Set the flag of smoother local rotation.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

threshhold How many pixels a mouse must move from the initial mouse down position before any transformation will occur

Valid values: integers
Default value: **5**
Minimum: 1
Maximum: 30

tsensitivity
[NOTE: This option is no longer used.] Mouse translation sensitivity. Larger is more sensitive.

Valid values: reals
Default value: **3.33**
Minimum: 1.0
Maximum: 500.0

Operands:
[reset]
Specifies that the transformations are to be reset - this must be reset .

translate

Translate in Angstroms whatever is specified in the transform set. This is either global (all atoms) or a local grouping defined useing an ASL expression in the transform command.

Syntax:

`translate` *x*=⟨ x ⟩ *y*=⟨ x ⟩ *z*=⟨ x ⟩

Options:

x Amount in Angstroms to translate in X
Valid values: reals
Default value: **0**

y Amount in Angstroms to translate in Y
Valid values: reals
Default value: **0**

z Amount in Angstroms to translate in Z
Valid values: reals
Default value: **0**

undisplayatom

Undisplay atoms in the set described by the ASL.

Syntax:

`undisplayatom` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
A string in the atom specification language which describes the set of atoms which are to be undisplayed.

undo

Undo the effect of the last change on the on-screen structure.

Syntax:

`undo`

ungroupentries

Ungroups the entries that match the given ESL expression.

Syntax:

`ungroupentries` ⟨ ESL ⟩

Operands:

⟨ ESL ⟩

<ESL> A valid ESL expression to specify which entries are to be ungrouped. Ungroups the entries that match the given ESL expression, and move them all to the end of ungrouped section. In project table, all the ungrouped entries (if any) will be present at the top i.e. before all the groups.

unhookimport

Restores normal Import behavior.

Syntax:

`unhookimport`

uniquename

With the set of specified atoms, make the atom name unique. This is done by adding a "~N" to each duplicate name where the "N" is a digit which represents how often this name is repeated in the set.

Syntax:

`uniquename`

uniquepdb

Set unique PDB atom names (by residue) for all atoms that match the ASL specification.

Syntax:

`uniquepdb` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
The operand is a valid ASL string that defines the set of atoms that are to have their PDB atom names changed.

update

This is a standard alias for **energyupdate** (see [energyupdate], page 136).

updateribbons

Update existing ribbons with current ribbon style and color scheme.

Syntax:

`updateribbons`

varymolecule

A command which defines a rotatable/translatable molecule during a conformational search. The molecule is defined by specifying any atom which belongs to it.

Syntax:

`varymolecule` *rmax*=⟨ x ⟩ *rmin*=⟨ x ⟩ *tmax*=⟨ x ⟩ *tmin*=⟨ x ⟩
⟨ atom_number ⟩

Options:

rmax The maximum value for the molecule rotation.
Valid values: reals
Default value: **180**
Minimum: 0.0

rmin The minimum value for the molecule rotation.
Valid values: reals
Default value: **0**
Minimum: 0.0

tmax The maximum value for the molecule translation

Valid values:	reals
Default value:	**1**
Minimum:	0.0

tmin The minimum value for the molecule translation

Valid values:	reals
Default value:	**0**
Minimum:	0.0

Operands:

⟨ atom_number ⟩

The number of an atom which is a member of the molecule which is to be translated or rotated.

varytorsion

Defines a rotatable bond to be used in a conformational search.

Syntax:

varytorsion *maximum*=⟨ x ⟩ *minimum*=⟨ x ⟩ ⟨ atom1 ⟩ ⟨ atom2 ⟩

Options:

maximum The maximum value for the torsional rotation.

Valid values:	reals
Default value:	**180**
Minimum:	0.0

minimum The minimum value for the torsional rotation.

Valid values:	reals
Default value:	**0**
Minimum:	0.0

Operands:

⟨ atom1 ⟩ ⟨ atom2 ⟩

The two atom numbers which define a bond to be rotated in a conformational search. These two atoms must have a bond (usually single) between them. Note that specifying a-b is the same as specifying b-a.

vcsaddattachment

Adds an attachment to the core molecule using the given atoms.

Syntax:

`vcsaddattachment` *atom1*=⟨ n ⟩ *atom2*=⟨ n ⟩ ⟨ attachment name ⟩

Options:

atom1 The atom number of the atom in the original core to set as an attachment point. This is the atom which will be kept.

Valid values: integers
Default value: **1**
Minimum: 1

atom2 The atom number of the atom in the original core to set as an attachment point. This is the atom which will be removed.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:
⟨ attachment name ⟩
The name of the attachment.

vcsaddcorefromproject

Adds the given entry as a core pose in CombiGlide.

Syntax:

`vcsaddcorefromproject` ⟨ entry name ⟩

Operands:
⟨ entry name ⟩
The entry name.

vcsaddmincapcore

Adds the minimally capped core to the poses for Define Core Poses in CombiGlide.

Syntax:

`vcsaddmincapcore`

vcsaddoriginalcore

Adds the original core to the poses for Define Core Poses in CombiGlide.

Syntax:

`vcsaddoriginalcore`

vcscanceldockjob

Cancels the currently running Dock Library job associated results.

Syntax:

`vcscanceldockjob`

vcsclearreagentfile

Clears the reagent file for the selected rows.

Syntax:

`vcsclearreagentfile`

vcscombiexportdockingfile

Exports the combinatorial docking results from CombiGlide to the given file.

Syntax:

`vcscombiexportdockingfile` ⟨file name⟩

Operands:
⟨file name⟩
The file name.

vcscombiexportdockingproject

Exports the combinatorial docking results from CombiGlide.

Syntax:

```
vcscombiexportdockingproject
```

vcscombiexportoptions

Holds the options for exporting combinatorial docking results from CombiGlide.

Syntax:

`vcscombiexportoptions` *includereceptor*=yes | no
numreagents=⟨ n ⟩

Options:

includereceptor
Indicates whether or not to include the receptor in the exported results

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numreagents
The number of reagents to export

Valid values: integers
Default value: **100**
Minimum: 1

vcsconfiguredocking

Allows the settings of some values that determine how the overall VCS job runs.

Syntax:

vcsconfiguredocking *bondrotation*=⟨ text ⟩ *gridfilename*=⟨ text ⟩ *gridfiletype*=⟨ text ⟩ *inputring*=yes | no *lig_ccut*=⟨ x ⟩ *lig_vscale*=⟨ x ⟩ *ligandwithmetal*=charged | either | neutral *maxatom*=⟨ n ⟩ *maxrotbonds*=⟨ n ⟩ *ninvert*=yes | no *numreqgroup1*=⟨ n ⟩ *numreqgroup2*=⟨ n ⟩ *numreqgroup3*=⟨ n ⟩ *numreqgroup4*=⟨ n ⟩ *penalizeamidebondrotations*=yes | no *reqmodegroup1*=all | atleast *reqmodegroup2*=all | atleast *reqmodegroup3*=all | atleast *reqmodegroup4*=all | atleast *ringconf*=yes | no

Options:

bondrotation
The option of amide bond rotation.
Valid values: text strings
Default value: **penal**

gridfilename
The base name for the file which the receptor grid is to be written to or read from.
Valid values: text strings
Default value:

gridfiletype
The type for the file which the receptor grid is to be written to or read from. The type is either zip or grd .
Valid values: text strings
Default value: **zip**

inputring An option which allows input ring conformation
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

lig_ccut The partial atomic charge below which ligand atoms are considered to be non-polar and will have their VDW radii scaled.
Valid values: reals
Default value: **0.15**
Minimum: 0.00000001

lig_vscale The scaling factor for the VDW radii of non-polar ligand atoms.
Valid values: reals
Default value: **0.8**
Minimum: 0.00000001

ligandwithmetal
Controls which ligand atoms can interact with metal sites.

Valid values: charged
either
neutral
Default value: **charged**

maxatom Any ligands in the input with more than this number of atoms will be skipped.

Valid values: integers
Default value: **300**
Minimum: 1
Maximum: 300

maxrotbonds
Any ligands in the input with more than this number of rotatable bonds will be skipped.

Valid values: integers
Default value: **50**
Minimum: 1

ninvert An option which allows nitrogen inversions

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

numreqgroup1
Number of constraints to be required for group 1 in docking.

Valid values: integers
Default value: **1**
Minimum: 0
Maximum: 4

numreqgroup2
Number of constraints to be required for group 2 in docking.

Valid values: integers
Default value: **1**
Minimum: 0
Maximum: 4

numreqgroup3
Number of constraints to be required for group 3 in docking.

Valid values: integers
Default value: **1**
Minimum: 0
Maximum: 4

numreqgroup4
Number of constraints to be required for group 4 in docking.

Valid values: integers

Default value: **1**
Minimum: 0
Maximum: 4

penalizeamidebondrotations
An option that penalizes twisted (non-planar) amide bonds.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

reqmodegroup1
The mode determines how to set the number of required constraints for group 1 in docking.
Valid values: all
atleast
Default value: **atleast**

reqmodegroup2
The mode determines how to set the number of required constraints for group 2 in docking.
Valid values: all
atleast
Default value: **atleast**

reqmodegroup3
The mode determines how to set the number of required constraints for group 3 in docking.
Valid values: all
atleast
Default value: **atleast**

reqmodegroup4
The mode determines how to set the number of required constraints for group 4 in docking.
Valid values: all
atleast
Default value: **atleast**

ringconf An option which allows ring flips
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

vcscoreoptions

Sets the options for Define Core Poses.

Syntax:

`vcscoreoptions` *centroidx*=⟨ x ⟩ *centroidy*=⟨ x ⟩ *centroidz*=⟨ x ⟩
constrainradius=⟨ x ⟩ *maxglidermsd*=⟨ x ⟩ *maxrmsd*=⟨ x ⟩
poseconstraint=box | sphere | glidecore

Options:

centroidx The X-coordinate of the location to constrain the core center of mass to.

Valid values: reals
Default value: **0**

centroidy The Y-coordinate of the location to constrain the core center of mass to.

Valid values: reals
Default value: **0**

centroidz The Z-coordinate of the location to constrain the core center of mass to.

Valid values: reals
Default value: **0**

constrainradius
The radius of the constraint for the core's center of mass.

Valid values: reals
Default value: **5**
Minimum: 1.0

maxglidermsd
The maximum RMSD that the Glide core can move.

Valid values: reals
Default value: **1**
Minimum: 0.0

maxrmsd The maximum RMSD that the core can move.

Valid values: reals
Default value: **2**
Minimum: 0.0

poseconstraint
Controls the allowed placement for core structures in CombiGlide docking.

Valid values: box
sphere
glidecore
Default value: **box**

vcscreatedockedlibrary

Start the creation and docking of the library in the Analyze Library step of CombiGlide. This is similar to vcsrunenumerateddocking, except that it uses only the selected reagents at each position and does not remove or replace contents of the Combinatorial Screening run.

Syntax:

`vcscreatedockedlibrary`

vcsdeleteattachment

Deletes all of the selected attachments.

Syntax:

`vcsdeleteattachment`

vcsdeletecore

Deletes the selected cores in the Define Core Poses step in CombiGlide.

Syntax:

`vcsdeletecore`

vcsdeleteresults

Deletes the given results.

Syntax:

`vcsdeleteresults` ⟨ name ⟩

Operands:
⟨ name ⟩
The name of the results file.

vcsdisplayreceptor

This function displays the receptor for the current CombiGlide run in the Workspace.

Syntax:

```
vcsdisplayreceptor
```

vcsdocking

Sets docking job options for CombiGlide.

Syntax:

`vcsdocking` *applystrain*=yes | no *postdock*=yes | no
task=combination | single | combinatorial | enumerated

Options:

applystrain
: An option which determines if the CombiGlide docking job will apply strain correction terms.
  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

postdock
: An option which determines if the CombiGlide docking job will do post-docking minimization.
  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **true**

task
: Determines what kind of docking task will be used in CombiGlide dicking, 1) user-capped core ane 2) minimally capped core.
  Valid values: combination
  single
  combinatorial
  enumerated
  Default value: **combination**

vcsenumeratedockoptions

Holds the options for enumerate and dock for CombiGlide.

Syntax:

`vcsenumeratedockoptions` *mode*=⟨ text ⟩

Options:

mode An option controlling what type of docking will be done after combinatorial enumeration. The allowed values are combi , xp , sp , and htvs .
Valid values: text strings
Default value: **combi**

vcsexcludetablerow

Excludes the given row in the structure table in the step from the Workspace.

Syntax:

`vcsexcludetablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to exclude in the Workspace.

vcsexportdefinition

Stores the current core molecule and attachments in a file.

Syntax:

`vcsexportdefinition` ⟨ file name ⟩

Operands:
⟨ file name ⟩
The name of the file to store the core definition in.

vcsexportresults

Stores the settings from the Filter and Select dialog, and all the results, in a human-readable text file.

Syntax:

`vcsexportresults` ⟨ file name ⟩

Operands:
⟨ file name ⟩
The name of the file to store the settings and results in.

vcsexportspreadsheet

Export actives found by libselector to a file for use in a spreadsheet. This can be comma-separated value (.csv) format or tab-delimited format.

Syntax:

`vcsexportspreadsheet` *delimiter*=⟨ text ⟩ ⟨ filename ⟩

Options:

delimiter This option sets the delimiter to use to separate columns.
Valid values: text strings
Default value: **,**

Operands:
⟨ filename ⟩
The name of the file to which actives will be written. If no name is specified, then no export will be done.

vcsimportdefinition

Reads a core molecule and attachments from the given file.

Syntax:

`vcsimportdefinition` ⟨ file name ⟩

Operands:
⟨ file name ⟩
The name of the file to read the core definition from.

vcsimportrun

Incorporates the combiglide results into the Maestro project for the specified run

Syntax:

`vcsimportrun` ⟨ directory name ⟩ ⟨ run name ⟩

Operands:
⟨ directory name ⟩ ⟨ run name ⟩
The first operand is the name of the directory to read the combiglide output files. The second operand is the run name to be imported into maestro

vcsincludeextendtablerow

Extends the rows included in the workspace to include this one.

Syntax:

`vcsincludeextendtablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include in the Workspace.

vcsincludeonlytablerow

Includes only the given row in the structure table in the step into the Workspace.

Syntax:

`vcsincludeonlytablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include in the Workspace.

vcsincludetablerow

Includes the given row in the structure table in the step into the Workspace.

Syntax:

`vcsincludetablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to include in the Workspace.

vcsinverttableselection

Inverts the row selection in the first table in the step.

Syntax:

`vcsinverttableselection`

vcsoptions

This command holds general options for CombiGlide.

Syntax:

`vcsoptions` *mode*=⟨ text ⟩ *sidechainnode*=⟨ n ⟩ *untangle*=yes | no

Options:

mode An option controlling what type of job CombiGlide will run.
Valid values: text strings
Default value: **sidechain**

sidechainnode
Which sidechain node we are working on.
Valid values: integers
Default value: **1**
Minimum: 1

untangle An option which allows post-combgen minimization (for Create Library, enumeration only).
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

vcsrefreshstructure

This function refreshes the structure in the Workspace from the current core structure in CombiGlide.

Syntax:

`vcsrefreshstructure` *viewcappedcore*=yes | no

Options:

viewcappedcore

An option which allows viewing of the minimally capped core, rather than the original core, in the Workspace.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

vcsrenameattachment

Renames the attachment in CombiGlide to the new name.

Syntax:

`vcsrenameattachment` *row*=⟨ n ⟩ ⟨ new name ⟩

Options:

row The row to rename.

Valid values: integers
Default value: **1**
Minimum: 1

Operands:

⟨ new name ⟩

The new name for the attachment.

vcsrestoreresults

Restores the filter and selection settings, and all associated results.

Syntax:

`vcsrestoreresults` ⟨name⟩

Operands:
⟨name⟩
The name of the results file.

vcsruncombinationdocking

Runs a both single-position and combinatorial docking job for CombiGlide.

Syntax:

`vcsruncombinationdocking`

vcsruncombinatorialdocking

Runs a combinatorial docking job for CombiGlide.

Syntax:

`vcsruncombinatorialdocking` *maxresults*=⟨n⟩ *mode*=sp | xp
⟨job name⟩

Options:

maxresults
The number of combinatorial results to return.

Valid values:	integers
Default value:	**1000**
Minimum:	0

mode Determines what kind of docking will be taken, 1) SP docking (faster), and 2) CombiGlide XP docking.

Valid values:	sp
	xp
Default value:	**sp**

Operands:
⟨job name⟩

The job name.

vcsruncombinatorialselection

Runs a combinatorial selection job for CombiGlide.

Syntax:

`vcsruncombinatorialselection` *cmdargs*=⟨ text ⟩

Options:

cmdargs The arguments for the reagent selection command.
Valid values: text strings
Default value:

vcsruncreate

Creates the run with the given name.

Syntax:

`vcsruncreate` ⟨ run name ⟩

Operands:
⟨ run name ⟩
The name of the new run to create.

vcsrundelete

Deletes the current run from the project.

Syntax:

`vcsrundelete`

vcsrunenumerateddocking

Runs an enumerated docking job in the Dock Library step of CombiGlide.

Syntax:

`vcsrunenumerateddocking`

vcsrunopen

Opens the run with the given name.

Syntax:

`vcsrunopen` ⟨run name⟩

Operands:
⟨run name⟩
The name of the run to open.

vcsrunrename

Changes the current run's name to the given name.

Syntax:

`vcsrunrename` ⟨run name⟩

Operands:
⟨run name⟩
The name to change the current run's name to.

vcsrunsaveas

Saves a copy of the current run under the given name.

Syntax:

`vcsrunsaveas` ⟨ run name ⟩

Operands:
⟨ run name ⟩
The name of the run to save as.

vcsrunsingledocking

Runs a single position docking job for CombiGlide.

Syntax:

`vcsrunsingledocking` *usecore*=usercapped | minimallycapped

Options:

usecore Determines what kind of core will be used in single-position docking, 1) user-capped core, 2) minimally capped core.
Valid values: usercapped
minimallycapped
Default value: **usercapped**

vcsrunsingleselection

Runs a single position selection job for CombiGlide.

Syntax:

`vcsrunsingleselection` *cmdargs*=⟨ text ⟩

Options:

cmdargs The arguments for the reagent selection command.
Valid values: text strings
Default value:

vcssaveresults

Saves the current filter and selection settings, and all associated results.

Syntax:

`vcssaveresults` ⟨ name ⟩

Operands:
⟨ name ⟩
The name of the results file.

vcsselectalltablerows

Selects all rows in the first table in the step.

Syntax:

`vcsselectalltablerows`

vcsselectextendtablerow

Extends the selection to this row in the table.

Syntax:

`vcsselectextendtablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to extend the select to.

vcsselectonlytablerow

Selects only this row in the table.

Syntax:

`vcsselectonlytablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select only in the table row.

vcsselecttablerow

Selects the given row in the first table in the step.

Syntax:

`vcsselecttablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to select in the table.

vcssetattachmentfile

Sets the reagent file for the given attachment.

Syntax:

`vcssetattachmentfile` *file*=⟨ text ⟩ ⟨ attachment name ⟩

Options:

file The file name of the reagent file to add to the given attachment.
Valid values: text strings
Default value:

Operands:
⟨ attachment name ⟩
The name of the attachment.

vcssetmolecule

Sets the core molecule for the current CombiGlide run to the molecule containing the given atom.

Syntax:

`vcssetmolecule` *title*=⟨ text ⟩ ⟨ atom number ⟩

Options:

title This option sets the title for the core molecule.

Valid values:	text strings
Default value:	**core**

Operands:

⟨ atom number ⟩

The atom number of the molecule.

vcssetreagentfile

Sets the reagent file for the selected rows.

Syntax:

`vcssetreagentfile` ⟨ reagent name ⟩

Operands:

⟨ reagent name ⟩

The name of the reagent file.

vcssingleexportdockingfile

Exports the single-position docking results from CombiGlide to the given file.

Syntax:

`vcssingleexportdockingfile` ⟨ file name ⟩

Operands:

⟨ file name ⟩

The file name.

vcssingleexportdockingproject

Exports the single-position docking results from CombiGlide.

Syntax:

`vcssingleexportdockingproject`

vcssingleexportoptions

Holds the options for exporting single-position docking results from CombiGlide.

Syntax:

`vcssingleexportoptions` *includereceptor*=yes | no
numreagents=⟨ n ⟩

Options:

includereceptor
Indicates whether or not to include the receptor in the exported results

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

numreagents
The number of reagents to export

Valid values: integers
Default value: **100**
Minimum: 1

vcssorttable

Resort the given CombiGlide table based on the data in the specified column

Syntax:

`vcssorttable` *table*=⟨ n ⟩ ⟨ column_name ⟩

Options:

table The table to set.

Valid values: integers
Default value: **10**

Operands:
⟨ column_name ⟩
The name of the column to be sorted.

vcsstepforward

Moves forward to the next VCS step. Deletes any steps after the current step, then creates the next step, using the data from previous steps.

Syntax:

`vcsstepforward`

vcsstepgoto

Moves to an existing step in the current project.

Syntax:

`vcsstepgoto` ⟨ step name ⟩

Operands:
⟨ step name ⟩
The name of the step to switch to.

vcsundisplayreceptor

This function undisplays the receptor for the current CombiGlide run in the Workspace.

Syntax:

`vcsundisplayreceptor`

vcsunselecttablerow

Unselects the given row in the first table in the step.

Syntax:

`vcsunselecttablerow` ⟨ row ⟩

Operands:
⟨ row ⟩
The row number to unselect in the table.

vcswritecombinationdocking

Writes a both single-position and combinatorial docking job input file for CombiGlide.

Syntax:

`vcswritecombinationdocking`

vcswritecombinatorialdocking

Writes a combinatorial docking job input file for CombiGlide.

Syntax:

`vcswritecombinatorialdocking`

vcswriteenumerateddocking

Writes an enumerated docking job input file in the Dock Library step of CombiGlide.

Syntax:

`vcswriteenumerateddocking`

vcswritesingledocking

Writes a single position docking job input file for CombiGlide.

Syntax:

`vcswritesingledocking`

viewcrystalmates

Set the options for removing and adding crystal mates in Workspace

Syntax:

`viewcrystalmates` *groupradius*=⟨ x ⟩ *renamechains*=yes | no *viewallcrystalmates*=yes | no *viewcrystalmateswithin*=⟨ x ⟩ *viewmates*=yes | no

Options:

groupradius

Controls the group radius used in the crystal mates calculation. This controls the radius within which a symmetric unit is considered to be in conact with the AsU. This is different to the viewcrystalmateswithin value which is only control the display and not the generation of the mates.

Valid values: reals
Default value: **14**
Minimum: 0.0

renamechains

Enabling this option will automatically rename crystal mates as they are generated.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

viewallcrystalmates

Indicates whether or not to view all crystal mates.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

viewcrystalmateswithin

This feature displays crystal mates within the given distance. This only takes affect if viewallcrystalmates is set to false.

Valid values: reals
Default value: **10**
Minimum: 0.0

viewmates Indicates whether or not to include crystal mates in Workspace for included project entries that have the required crystal settings.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

viewdelete

Deletes transform data and view data corresponding to the given view name.

Syntax:

`viewdelete` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewdeleteselected

Delete selected views in the current views table.

Syntax:

`viewdeleteselected`

viewdragselection

Move the current view selection to the specified table row.

Syntax:

`viewdragselection` ⟨ row ⟩

Operands:
⟨ row ⟩
A valid destination row number in the views table.

viewexport

Exports views from the current views table to given file. If non of the views are selected then exports all the views, else exports only selected views.

Syntax:

`viewexport` ⟨ file ⟩

Operands:
⟨ file ⟩
The views file name to export.

viewextendselect

Extend current view selection to encompass the specified view in the table.

Syntax:

`viewextendselect` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table. A range selection will be done for the views between the currently selected view and the specified view name.

viewextendselectrow

Extend current view selection to encompass the specified view table row.

Syntax:

`viewextendselectrow` ⟨ row ⟩

Operands:
⟨ row ⟩
A row number between 1 and the total number of rows in the views table. A range selection will be done for the views between currently selected view rows and the specified view row number.

viewimport

Imports views from the given file.

Syntax:

`viewimport` ⟨ file ⟩

Operands:

⟨ file ⟩

The views file name to import.

viewmatrix

Sets workspace rotation matrix to the matrix supplied When 'inverse' is supplied, it sets the inverse roation matrix and when 'nocenter', no center rotation matrix. And in the absence of both the keywords, it sets the rotation matrix. The transpose of this matrix is used in a call to glMultMatrixd to set the user rotation. Points in molecule coordinate space (x, y, z, 1) are pre-multiplied by this matrix to generate rotated coordinates for display, subject to additional view transforms for perspective or stereo rotation, and mapping into normalized view space and to window or screen pixel space. The viewmatrix command takes, as operands, 16 floating point numbers, to define a 4-by-4 viewing matrix in row major order, plus an optional matrix identifier string (inverse or nocenter). Maestro reads each line from the wsview file and processes it as a separate command, silently ignoring any errors. The viewmatrix settings are all inter-dependent. If one is set, all three should be set.

Syntax:

`viewmatrix`

viewplayergotofirst

Goto the first view in the current views table, for which play state is set.

Syntax:

```
viewplayergotofirst
```

viewplayergotolast

Goto the last view in the current views table, for which play state is set.

Syntax:

```
viewplayergotolast
```

viewplayerplaybackward

Start playing views backward from the recently restored view in the ordered sequence of play toggled on views in the current views table.

Syntax:

```
viewplayerplaybackward
```

viewplayerplayforward

Start playing views forward from the recently restored view in the ordered sequence of play toggled on views in the current views table.

Syntax:

```
viewplayerplayforward
```

viewplayersettings

Set viewplayer state variables.

Syntax:

`viewplayersettings` *playmode*=loop | reverse | once

Options:

playmode This option sets the mode for continuous play. Valid values are "loop", "reverse", or "once". These cause play to continue, change direction, or stop, respectively, when reaching either end of the current play toggle on views.

Valid values: loop
reverse
once

Default value: **once**

viewplayerstepahead

Goto the next view in the ordered sequence of play toggled on view in the current views table.

Syntax:

`viewplayerstepahead`

viewplayerstepback

Goto the previous view in the ordered sequence of play toggled on views in the current views table.

Syntax:

`viewplayerstepback`

viewplayerstop

Stop playing views/saving views movie.

Syntax:

`viewplayerstop`

viewrecordmovie

Save a movie with views for which Play is set.

Syntax:

`viewrecordmovie` ⟨file⟩

Operands:
⟨file⟩
The file where the views movie will be saved.

viewrename

Renames the given view name.

Syntax:

`viewrename` ⟨curr_name⟩ ⟨new_name⟩

Operands:
⟨curr_name⟩ ⟨new_name⟩
A valid view name existing in the current views table. A valid new view name.

viewreset

Resets the viewing transform to original.

Syntax:

`viewreset`

viewresetnofit

Resets the viewing transform to original but not fit to window.

Syntax:

`viewresetnofit`

viewrestore

Restores the viewing transform corresponding to the view name.

Syntax:

`viewrestore` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewrestorerow

Restore the transform of the view corresponding to the given row.

Syntax:

`viewrestorerow` ⟨ row ⟩

Operands:
⟨ row ⟩
A valid row in the views table.

viewsave

Saves the current viewing transform.

Syntax:

`viewsave` *pause*=⟨ n ⟩ *play*=yes | no *select*=yes | no ⟨ view_name ⟩

Options:

pause	Pause value of the view.	
	Valid values:	integers
	Default value:	**1**
	Minimum:	0

	Maximum:	5
play	Play state of the view.	
	Valid values:	boolean (true|false; yes|no; y|n; on|off)
	Default value:	**true**
select	Selection state of the view.	
	Valid values:	boolean (true|false; yes|no; y|n; on|off)
	Default value:	**true**

Operands:
⟨ view_name ⟩
A valid view name with which the current transform will be saved.

viewselect

Select the given view in the current views table.

Syntax:

`viewselect` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewselectonly

Select only the given view in the current views table.

Syntax:

`viewselectonly` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewselectonlyrow

Select only the view corresponding to the row in the current views table and unselct remaining views.

Syntax:

`viewselectonlyrow` ⟨ row ⟩

Operands:
⟨ row ⟩
A valid row in the views table.

viewselectrow

Select the view corresponding to the given row in the current views table.

Syntax:

`viewselectrow` ⟨ row ⟩

Operands:
⟨ row ⟩
A valid row in the views table.

viewsetpause

Sets the pause value to the given view.

Syntax:

`viewsetpause` ⟨ view_name ⟩ ⟨ pause_value ⟩

Operands:
⟨ view_name ⟩ ⟨ pause_value ⟩
A valid view name existing in the current views table. A valid pause value greater than -1.

viewsetplay

Set the play state of the given view.

Syntax:

`viewsetplay` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewsetrowpause

Set the given pause value to the view corresponding to the row in the current views table.

Syntax:

`viewsetrowpause` ⟨ row ⟩ ⟨ pause ⟩

Operands:
⟨ row ⟩ ⟨ pause ⟩
A row number between 1 and the total number of rows in the views table. A valid pause value greater than -1.

viewtrajectory

Attempt to view the Chorus trajectory associated with the specified entry.

Syntax:

`viewtrajectory` ⟨ entry_name ⟩

Operands:
⟨ entry_name ⟩
The name (entry ID) of the entry, whose trajectory data is to be viewed.

viewunselect

Unselect the given view in the current views table.

Syntax:

`viewunselect` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewunselectrow

Unselect the view corresponding to the given row in the current views table.

Syntax:

`viewunselectrow` ⟨ row ⟩

Operands:
⟨ row ⟩
A valid row in the views table.

viewunsetplay

Unset the play state of the given view.

Syntax:

`viewunsetplay` ⟨ view_name ⟩

Operands:
⟨ view_name ⟩
A valid view name existing in the current views table.

viewvolume

Sets workspace view bounding box to the supplied values

Syntax:

`viewvolume` *bottom*=⟨ x ⟩ *far*=⟨ x ⟩ *left*=⟨ x ⟩ *near*=⟨ x ⟩ *right*=⟨ x ⟩ *top*=⟨ x ⟩

Options:

bottom Bottom co-ordinate of the bounding box
Valid values: reals
Default value: **-5**

far Far co-ordinate of the bounding box
Valid values: reals
Default value: **-10**

left Left co-ordinate of the bounding box
Valid values: reals
Default value: **-5**

near Near co-ordinate of the bounding box
Valid values: reals
Default value: **10**

right Right co-ordinate of the bounding box
Valid values: reals
Default value: **5**

top Top co-ordinate of the bounding box
Valid values: reals
Default value: **5**

visexport

Export a volume to the given file.

Syntax:

`visexport` *entry*=⟨ text ⟩ *format*=cnsunformatted | cnsformatted | cnsccp4 *surface*=⟨ text ⟩ *surfacetype*=⟨ text ⟩ *surfacetypecomment*=⟨ text ⟩ *volume*=⟨ text ⟩ ⟨ file name ⟩

Options:

entry The entry name of the entry that the surface belongs to.
Valid values: text strings
Default value:

format The format of the file that will be exported.

Valid values: cnsunformatted
cnsformatted
cnsccp4
Default value: **cnsformatted**

surface The name of the surface.

Valid values: text strings
Default value:

surfacetype
The surface type of the surface that will be exported.

Valid values: text strings
Default value:

surfacetypecomment
The surface type comment of the surface that will be exported.

Valid values: text strings
Default value:

volume The name of the volume that the surface belongs to.

Valid values: text strings
Default value:

Operands:
⟨ file name ⟩
The name of the visio file to export.

visimport

Creates all volumes and surfaces in the given file.

Syntax:

`visimport` *entry*=⟨ text ⟩ *isovalue*=⟨ x ⟩ *transparency*=⟨ x ⟩
⟨ file.vis ⟩ [:⟨ surface name ⟩]

Options:

entry This is the entry to associate the objects with.

Valid values: text strings
Default value:

isovalue If any of the volumes do not have associated surfaces or suggested isovalues, then this isovalue will be used to isosurface the volumes.

Valid values: reals
Default value: **0.03**

transparency
This option sets the transparency of any surfaces.

Valid values: reals
Default value: **0**
Minimum: 0.0
Maximum: 100.0

Operands:
⟨ file.vis ⟩ [:⟨ surface name ⟩]
The first operand is the name of the visio or Jaguar plot file to load. The optional second operand is the specified surface name.

volumedelete

Deletes the given volume.

Syntax:

`volumedelete` *entry*=⟨ text ⟩ ⟨ volume ⟩

Options:

entry The entry that the volume belongs to.

Valid values: text strings
Default value:

Operands:
⟨ volume ⟩
The name of the volume to delete.

volumedisplay

Displays or undisplays the given volume

Syntax:

`volumedisplay` *display*=yes | no *entry*=⟨ text ⟩ ⟨ volume ⟩

Options:

display	Sets whether or not to display the volume.	
	Valid values:	boolean (true|false; yes|no; y|n; on|off)
	Default value:	**true**
entry	The entry that the volume belongs to.	
	Valid values:	text strings
	Default value:	

Operands:
⟨ volume ⟩
The name of the volume to display.

volumeisosurface

Isosurfaces the given volume.

Syntax:

`volumeisosurface` *entry*=⟨ text ⟩ *isovalue*=⟨ x ⟩ *surface*=⟨ text ⟩ ⟨ volume ⟩

Options:

entry	The entry that the volume belongs to.	
	Valid values:	text strings
	Default value:	
isovalue	The isovalue to use for the isosurfacing.	
	Valid values:	reals
	Default value:	**0.1**
surface	The name for the new surface.	
	Valid values:	text strings
	Default value:	

Operands:
⟨ volume ⟩

The name of the volume to isosurface.

volumerename

Renames the given volume.

Syntax:

`volumerename` *entry*=⟨ text ⟩ *newname*=⟨ text ⟩ ⟨ volume ⟩

Options:

entry The entry that the volume belongs to.
Valid values: text strings
Default value:

newname The new name for the volume.
Valid values: text strings
Default value:

Operands:

⟨ volume ⟩

The name of the volume to rename.

volumesetcolors

Changes color map for the specifed volume. The color points has to be specified after the volume name as five floating point numbers. The numbers correspond to data value, alpha value (opacity), R, G, B color components.

Syntax:

`volumesetcolors` *entry*=⟨ text ⟩ ⟨ volume ⟩

Options:

entry The entry that the volume belongs to.
Valid values: text strings
Default value:

Operands:
⟨ volume ⟩
The name of the volume to change colors.

wake

This is a standard alias for **energywake** (see [energywake], page 137).

workspacecopy

Currently this is only used by Edit->Copy to determine if it generates in the current working directory a number of files corresponding the data it puts on the clipboard. It does not actually cause the copy to execute. This has to be done graphically.

Syntax:

`workspacecopy` *writefiles*=yes | no

Options:

writefiles If true, write out files corresponding to the clipboard data. If false, do not write out the files.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

workspacefontdecrease

Decrease the size of the Workspace fonts

Syntax:

`workspacefontdecrease`

workspacefontincrease

Increase the size of the Workspace fonts

Syntax:

```
workspacefontincrease
```

workspacepaste

If there is a mime type of application/x-qt-windows-mime;value=SMILES on the clipboard, a plain text SMILES string in the global mouse selection, a plain text SMILES string on the clipboard, or an ISIS/Draw MDLCT, then use that as the basis for pasting into center of the Workspace a 2D structure. The structure is created as a project entry and included into the contents of the Workspace. If atoms were cut or copied within Maestro, then the pasted atoms are placed in the Workspace but not added as an entry to the Project Table.

Syntax:

```
workspacepaste
```

workspacepastesettings

Settings that affect workspacepaste

Syntax:

`workspacepastesettings` *enableselect*=yes | no

Options:

enableselect

Enable this to allow pasting of selected text on Linux. This only applies to systems running X Windows (such as Linux). If true, allows user to do text selection in windows (such as an xterm) and does not require copying the text to the clipboard (via a text editor's Copy, for example). There have been cases where text selection in python-based tkinter panels can be extremely slow when this feature is enabled. As of Suite 2011 most, if not all, GUI panels done in python use PyQt. So this should not be an issue.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

workspacescopeoff

Turn off the Workspace scope

Syntax:

```
workspacescopeoff
```

workspacescopeon

Turn on the Workspace scope

Syntax:

```
workspacescopeon
```

workspacescopesettings

Settings for the Workspace scope

Syntax:

`workspacescopesettings` *alpha*=⟨ x ⟩ *blendcolor*=background | user *blendfactor*=⟨ x ⟩ *blue*=⟨ x ⟩ *green*=⟨ x ⟩ *radius*=⟨ x ⟩ *red*=⟨ x ⟩ *track*=yes | no *x*=⟨ x ⟩ *y*=⟨ x ⟩

Options:

alpha Alpha component. 0 is invisible. 1 is opaque.

Valid values:	reals
Default value:	**1**
Minimum:	0.0
Maximum:	1.0

blendcolor For pixels outside of the scope the color source to use.

Valid values:	background user
Default value:	**background**

blendfactor

How much the pixels outside of the scope should be blended with the background color. 1.0 means 100%.

Valid values:	reals

Default value: **0.75**
Minimum: 0.0
Maximum: 1.0

blue Blue componentwhen blendcolor is 'user'.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

green Green component when blendcolor is 'user'.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

radius Radius in pixels

Valid values: reals
Default value: **100**
Minimum: 0.0
Maximum: 1000.0

red Red component when blendcolor is 'user'.

Valid values: reals
Default value: **0.5**
Minimum: 0.0
Maximum: 1.0

track Enable or disable tracking of the cursor. This is only use if the workspace scope is on.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

x X position

Valid values: reals
Default value: **0**

y Y position

Valid values: reals
Default value: **0**

workspacescopeview

Determine which things are visible inside and outside of the scope. Only used when the scope is enabled.

Syntax:

`workspacescopeview` *molecules*=nowhere | inside | outside | everywhere *ribbons*=nowhere | inside | outside | everywhere *surfaces*=nowhere | inside | outside | everywhere *volumes*=nowhere | inside | outside | everywhere

Options:

molecules View molecules nowhere, inside the scope, outside the scope or everywhere (inside and outside).

Valid values: nowhere, inside, outside, everywhere

Default value: **everywhere**

ribbons View ribbons nowhere, inside the scope, outside the scope or everywhere (inside and outside).

Valid values: nowhere, inside, outside, everywhere

Default value: **everywhere**

surfaces View surfaces nowhere, inside the scope, outside the scope or everywhere (inside and outside).

Valid values: nowhere, inside, outside, everywhere

Default value: **everywhere**

volumes View volumes nowhere, inside the scope, outside the scope or everywhere (inside and outside).

Valid values: nowhere, inside, outside, everywhere

Default value: **everywhere**

workspaceselectionadd

Extends workspace selection with the atoms confirming to the supplied ASL.

Syntax:

`workspaceselectionadd` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
ASL representing the set of atoms

workspaceselectionclear

Resets current workspace selection to none.

Syntax:

`workspaceselectionclear`

workspaceselectioninvert

Flips the state of an atom in the workspace selection. i.e., removes if the atom already exists or adds if it doesn't.

Syntax:

`workspaceselectioninvert` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩
ASL representing the set of atoms

workspaceselectionreplace

Overwrites the workspace selection with the supplied set of atoms.

Syntax:

`workspaceselectionreplace` ⟨ ASL ⟩

Operands:
⟨ ASL ⟩

ASL representing the set of atoms

workspaceselectionsubtract

Removes atoms confirming to the supplied ASL from the workspace selection.

Syntax:

`workspaceselectionsubtract` ⟨ ASL ⟩

Operands:

⟨ ASL ⟩

ASL representing the set of atoms

workspacetile

The workspacetile command specifies contents for a single tile in workspace tile mode.

Syntax:

`workspacetile` ⟨ tile_name ⟩

Operands:

⟨ tile_name ⟩

The name which identifies the contents of the tile. This name will be the index of an entry group (greater than 0), or 0: followed by the entry ID for un ungrouped entry (or Scratch entry). A tile named 0: (with no entry ID) does not belong to any entry or entry group. In tile by surface mode, the entry ID will be followed by the name of the special surface for that tile.

write

This is a standard alias for **filewrite** (see [filewrite], page 197).

writeangle

Write the angles in the angle table to a file.

Syntax:

`writeangle` *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.

Valid values:	entry molecule workspace
Default value:	**entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values:	comma tab userdefined
Default value:	**comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values:	text strings
Default value:	

Operands:

⟨ file_name ⟩

The name of the file to which the angles will be written.

writecontact

Write Contacts information to a file.

Syntax:

`writecontact` *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.
Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.
Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.
Valid values: text strings
Default value:

Operands:
⟨ file_name ⟩
The name of the file to which Contacts information will be written.

writecoupling

Write the coupling constants in the coupling table to a file.

Syntax:

`writecoupling` *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩
The name of the file to which the coupling constants will be written.

writedihedral

Write the dihedrals in the dihedral table to a file.

Syntax:

`writedihedral` *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber
This option has been deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:
⟨ file_name ⟩
The name of the file to which the dihedrals will be written.

writedistance

Write the distances in the distance table to a file.

Syntax:

`writedistance` *atomnumber*=entry | molecule | workspace *delimiter*=comma | tab | userdefined *userdelimiter*=⟨ text ⟩ ⟨ file_name ⟩

Options:

atomnumber
This option is deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter
The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩

The name of the file to which the distances will be written.

writehbond

Write H-Bonds information to a file.

Syntax:

`writehbond` *atomnumber*=entry | molecule | workspace
delimiter=comma | tab | userdefined *userdelimiter*=⟨ text ⟩
⟨ file_name ⟩

Options:

atomnumber

This option is deprecated.

Valid values: entry
molecule
workspace
Default value: **entry**

delimiter This option sets what delimiter will be used in the output file: comma=1, tab=2, and userdefined=3.

Valid values: comma
tab
userdefined
Default value: **comma**

userdelimiter

The delimiter string defined by users. It is valid only when option delimiter is userdefined.

Valid values: text strings
Default value:

Operands:

⟨ file_name ⟩

The name of the file to which H-Bonds information will be written.

wsassistantapply

Command that applies workspace style settings.

Syntax:

`wsassistantapply`

wsassistantapplycontacthbond

Command that applies contacts and hbonds options of workspace assistant.

Syntax:

`wsassistantapplycontacthbond`

wsassistantapplyfocus

Command that applies workspace style settings and sets focus on ligand.

Syntax:

`wsassistantapplyfocus`

wsassistantbindingsurface

Command that handles workspace assistant binding site surface options.

Syntax:

`wsassistantbindingsurface` *colorscheme*=constant | partialcharge | atomtype | chainname | element | molnum | molnumcarbon | residuecharge | residueproperty | residueposition | residuetype | entry | atomcolor | elepotential | pbelepotential *createligandsurfaces*=yes | no *createreceptorsurfaces*=yes | no *radius*=⟨ x ⟩ *spacing*=⟨ x ⟩ *style*=solid | mesh | dot *transparency*=⟨ n ⟩

Options:

colorscheme
: What kind of color scheme for the binding surface.

Valid values: constant
partialcharge
atomtype
chainname
element
molnum
molnumcarbon
residuecharge
residueproperty
residueposition
residuetype
entry
atomcolor
elepotential
pbelepotential
Default value: **constant**

createligandsurfaces
Create ligand surfaces or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

createreceptorsurfaces
Create receptor surfaces or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

radius The radius used to truncate receptor surface.

Valid values: reals
Default value: **5**
Minimum: 0.01
Maximum: 20.0

spacing The surface grid spacing value.

Valid values: reals
Default value: **0.6**
Minimum: 0.01
Maximum: 10.0

style What kind of style for the binding surface: solid (0), Mesh (1), or Dot (2).

Valid values: solid
mesh
dot
Default value: **solid**

transparency
The percentage of surface transparency.

Valid values:	integers
Default value:	**70**
Minimum:	0
Maximum:	100

wsassistantcontacthbond

Command that handles workspace assistant contacts and hbonds options.

Syntax:

`wsassistantcontacthbond` *contacttype*=badandugly | ugly | bad | good *intraligandcontacts*=yes | no *receptorligandcontacts*=yes | no *receptorligandhbonds*=yes | no

Options:

contacttype
What kind of contact type the workspace assistant contains.

Valid values:	badandugly ugly bad good
Default value:	**badandugly**

intraligandcontacts
Show intra-ligand contacts or not.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

receptorligandcontacts
Show receptor-ligand contacts or not.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

receptorligandhbonds
Show reseptor-ligand hbonds or not.

Valid values:	boolean (true|false; yes|no; y|n; on|off)
Default value:	**false**

wsassistantcreatesurfaces

Command that creates binding site surfaces.

Syntax:

```
wsassistantcreatesurfaces
```

wsassistantdefaultstyle

Command that sets default workspace style.

Syntax:

```
wsassistantdefaultstyle
```

wsassistantfixreceptor

Command to fix the receptor in the Workspace.

Syntax:

```
wsassistantfixreceptor
```

wsassistanthideallsurfaces

Command that Hides all surfaces.

Syntax:

```
wsassistanthideallsurfaces
```

wsassistantliganddetection

Command that handles workspace assistant ligand detection options.

Syntax:

`wsassistantliganddetection` *allowaminoacids*=yes | no *allowsmallions*=yes | no *includedresiduecode*=⟨ text ⟩ *includedresiduecodes*=⟨ text ⟩ *maximumatomcount*=⟨ n ⟩ *minimumatomcount*=⟨ n ⟩ *residuecode*=⟨ text ⟩ *residuecodes*=⟨ text ⟩

Options:

allowaminoacids
: Allow molecules containing only amino acids or not.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

allowsmallions
: Allow molecules that are small ions or not.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

includedresiduecode
: The 3-letter residue code for included residues.

  Valid values: text strings
  Default value:

includedresiduecodes
: All 3-letter residue codes for included residues.

  Valid values: text strings
  Default value:

maximumatomcount
: The maximum atom count of ligand molecule.

  Valid values: integers
  Default value: **130**
  Minimum: 2

minimumatomcount
: The minimum atom count of ligand molecule.

  Valid values: integers
  Default value: **5**
  Minimum: 1

residuecode
: The 3-letter residue code for excluded residues.

  Valid values: text strings
  Default value:

residuecodes
: All 3-letter residue codes for excluded residues.

Valid values: text strings
Default value:

wsassistantmaterial

Command to set the the appearance to one of a set of preset definitions. These control the lighting and material properties used in the Workspace.

Syntax:

`wsassistantmaterial`

wsassistantprimarystyle

Command that handles workspace assistant primary style options.

Syntax:

`wsassistantprimarystyle` *colorscheme*=element | elementgreen | elementlight | elemententry | bfactor | densityfit | moleculesize | residuetype | entry *distalribbonscheme*=residueposition | secondarystructure | chain | residuetype | residueproperty | entry | calphaatom *distalribbonstyle*=none | cartoon | ribbon | tube | thintube | curvedline | calphaline | calphatube *fit*=yes | no *pickstate*=atom | residue | molecule | chain | entry *proximalribbonscheme*=residueposition | secondarystructure | chain | residuetype | residueproperty | entry | calphaatom *proximalribbonstyle*=none | cartoon | ribbon | tube | thintube | curvedline | calphaline | calphatube *representation*=default | wire | cpk | ballbstick | tube

Options:

colorscheme
What kind of color scheme the primary style contains.

Valid values: element
elementgreen
elementlight
elemententry
bfactor
densityfit
moleculesize
residuetype
entry
Default value: **element**

distalribbonscheme
What kind of distal ribbon scheme the primary style contains.

Valid values: residueposition
secondarystructure
chain
residuetype
residueproperty
entry
calphaatom
Default value: **residueposition**

distalribbonstyle
What kind of distal ribbon style the primary style contains.

Valid values: none
cartoon
ribbon
tube
thintube
curvedline
calphaline
calphatube
Default value: **calphatube**

fit Fit Workspace to picked entity or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

pickstate What kind of pick state the primary style contains.

Valid values: atom
residue
molecule
chain
entry
Default value: **residue**

proximalribbonscheme

What kind of proximal ribbon scheme the primary style contains.

Valid values: residueposition
secondarystructure
chain
residuetype
residueproperty
entry
calphaatom

Default value: **residueposition**

proximalribbonstyle

What kind of proxiaml ribbon style the primary style contains.

Valid values: none
cartoon
ribbon
tube
thintube
curvedline
calphaline
calphatube

Default value: **cartoon**

representation

What kind of representation the primary style contains.

Valid values: default
wire
cpk
ballbstick
tube

Default value: **default**

wsassistantrendering

Command to set the quality of workspace rendering. This allows a tradeoff between quality and speed.

Syntax:

```
wsassistantrendering
```

wsassistantshowallsurfaces

Command that shows all surfaces.

Syntax:

wsassistantwsstyle

Command that handles workspace assistant workspace style options.

`wsassistantshowallsurfaces`

Syntax:

`wsassistantwsstyle` *applycolorschemes*=yes | no
applyrepresentations=yes | no *applysettings*=yes | no
atomradius=⟨ x ⟩ *ballpercentage*=⟨ n ⟩
createligandsurfaces=yes | no *createreceptorsurfaces*=yes | no
defaultrepresentation=wire | cpk | ballnstick | tube |
thin_tube *defaultscheme*=element | elementgreen |
elementlight | elemententry | bfactor | densityfit | moleculesize
| residuetype | entry *displayperspective*=yes | no
displayribbons=yes | no *displaywithin*=yes | no
fittoligand=yes | no *focusligands*=yes | no
ionrepresentation=wire | cpk | ballnstick | tube | thin_tube
ionscheme=element | elementgreen | elementlight |
elemententry | bfactor | densityfit | moleculesize | residuetype
| entry *ligandhydrogen*=none | polaronly | all
ligandrepresentation=wire | cpk | ballnstick | tube | thin_tube
ligandscheme=element | elementgreen | elementlight |
elemententry | bfactor | densityfit | moleculesize | residuetype
| entry *materialeffect*=default | aluminum | sundown |
dramatic | gloss *multibondradius*=⟨ x ⟩ *reapply*=yes | no
reapplyfocus=yes | no *reapplyligand*=yes | no
reapplyto=workspace | ligand *receptorhydrogen*=none |
polaronly | all *ribbonlimit*=yes | no *ribbonradius*=⟨ x ⟩
ribbonscheme=residueposition | secondarystructure | chain |
residuetype | residueproperty | entry | calphaatom
ribbonstyle=none | cartoon | ribbon | tube | thintube |
curvedline | calphaline | calphatube *singlebondradius*=⟨ x ⟩
stickradius=⟨ x ⟩ *suppressdisplay*=yes | no *waterdisplay*=none
| all | beyond3a | beyond5a | beyond10a
waterrepresentation=wire | cpk | ballnstick | tube | thin_tube
waterscheme=element | elementgreen | elementlight |
elemententry | bfactor | densityfit | moleculesize | residuetype
| entry *within*=⟨ x ⟩

Options:

applycolorschemes
Apply color schemes or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

applyrepresentations
Apply representations or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

applysettings
: When Workspace changes applying Workspace style settings or not.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

atomradius
: The radius of atom in representation.

  Valid values: reals
  Default value: **5**
  Minimum: 0.0
  Maximum: 1000.0

ballpercentage
: The percentage of ball radius in representation.

  Valid values: integers
  Default value: **16**
  Minimum: 4
  Maximum: 200

createligandsurfaces
: Create ligand surface or not when reapply.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

createreceptorsurfaces
: Create receptor surface or not when reapply.

  Valid values: boolean (true|false; yes|no; y|n; on|off)
  Default value: **false**

defaultrepresentation
: What kind of default representation the workspace style contains.

  Valid values: wire
  cpk
  ballnstick
  tube
  thin_tube
  Default value: **wire**

defaultscheme
: What kind of default color scheme the workspace style contains.

Valid values: element
elementgreen
elementlight
elemententry
bfactor
densityfit
moleculesize
residuetype
entry
Default value: **element**

displayperspective
Display in perspective or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

displayribbons
Display ribbons or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

displaywithin
Display residues within a distance or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

fittoligand When Workspace changes Fitting to ligand or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

focusligands
Focus display on lignds or not.
Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ionrepresentation
What kind of ion representation the workspace style contains.
Valid values: wire
cpk
ballnstick
tube
thin_tube
Default value: **cpk**

ionscheme What kind of ion color scheme the workspace style contains.

Valid values: element
elementgreen
elementlight
elemententry
bfactor
densityfit
moleculesize
residuetype
entry
Default value: **element**

ligandhydrogen
Options for displaying hydrogens of ligand: None (0), Polar only (1), or All (2).

Valid values: none
polaronly
all
Default value: **all**

ligandrepresentation
What kind of ligand representation the workspace style contains.

Valid values: wire
cpk
ballnstick
tube
thin_tube
Default value: **ballnstick**

ligandscheme
What kind of ligand color scheme the workspace style contains.

Valid values: element
elementgreen
elementlight
elemententry
bfactor
densityfit
moleculesize
residuetype
entry
Default value: **elemententry**

materialeffect
What kind of lighting/material effect the workspace style contains.

Valid values: default
aluminum
sundown
dramatic
gloss
Default value: **default**

multibondradius
The radius of multiple tube bond in representation.

Valid values: reals
Default value: **0.05**
Minimum: 0.01
Maximum: 1.0

reapply Reapply workspace style settings for receptor or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

reapplyfocus
Reapply workspace style focus or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

reapplyligand
Reapply workspace style settings for ligand or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

reapplyto Reapply workspace style to which structures in the Workspace, entire Workspace (0) or ligand only (1).

Valid values: workspace
ligand
Default value: **workspace**

receptorhydrogen
Options for displaying hydrogens of receptor: None (0), Polar only (1), or All (2).

Valid values: none
polaronly
all
Default value: **polaronly**

ribbonlimit
Display ribbons within a range or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

ribbonradius

The additional radius of ribbon in representation.

Valid values:	reals
Default value:	**5**
Minimum:	0.0
Maximum:	1000.0

ribbonscheme

What kind of ribbon scheme the workspace style contains.

Valid values:	residueposition secondarystructure chain residuetype residueproperty entry calphaatom
Default value:	**residueposition**

ribbonstyle

What kind of ribbon style the primary style contains.

Valid values:	none cartoon ribbon tube thintube curvedline calphaline calphatube
Default value:	**cartoon**

singlebondradius

The radius of single tube bond in representation.

Valid values:	reals
Default value:	**0.1**
Minimum:	0.01
Maximum:	1.0

stickradius

The radius of stick bond in representation.

Valid values:	reals
Default value:	**0.12**
Minimum:	0.01
Maximum:	1.0

suppressdisplay

Suppress nonpolar-hydrogen display or not.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

waterdisplay
What kind of water display suppression the workspace style contains.

Valid values: none
all
beyond3a
beyond5a
beyond10a
Default value: **none**

waterrepresentation
What kind of water representation the workspace style contains.

Valid values: wire
cpk
ballnstick
tube
thin_tube
Default value: **ballnstick**

waterscheme
What kind of water color scheme the workspace style contains.

Valid values: element
elementgreen
elementlight
elemententry
bfactor
densityfit
moleculesize
residuetype
entry
Default value: **element**

within The radius within which structures will be displayed.

Valid values: reals
Default value: **8**
Minimum: 0.0
Maximum: 999.0

xcluster

This keyword is used to set various options associated with starting XCluster jobs from Maestro.

Syntax:

`xcluster` *cluster_by*=atom | torsion *compare_enantiomers*=yes | no
input_file=⟨ text ⟩ *job*=⟨ text ⟩ *rms_in_place*=yes | no
structure_source=selected_entries | workspace | file

Options:

cluster_by Whether to use atomic RMS or torsional RMS differences for clustering analysis.

Valid values: atom
torsion
Default value: **atom**

compare_enantiomers
A boolean which controls whether XCluster will perform superposition in such a way so as to compare enantiomers.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **true**

input_file The name of the structure input file.

Valid values: text strings
Default value:

job The name for the XCluster job.

Valid values: text strings
Default value: **xclustmp**

rms_in_place
A boolean which controls whether XCluster will calculate RMS differences without first doing a superposition.

Valid values: boolean (true|false; yes|no; y|n; on|off)
Default value: **false**

structure_source
Whether to use the selected entries in the current project, or a specified file with multiple structures as structure input for the job.

Valid values: selected_entries
workspace
file
Default value: **selected_entries**

xclusterstart

Start a XCluster job with the current settings.

Syntax:

`xclusterstart`

xclusterwrite

Write a XCluster input file with the current settings.

Syntax:

`xclusterwrite`

zoom

Zoom in or out (i.e. scale up or down)

Syntax:

`zoom` *factor*=⟨ x ⟩ in | out

Options:

factor A zooming (scaling) factor which determines by how much the on-screen structures is scaled. 1.0 is 1 percent, 99.0 is 99 percent, etc. Values are used relative to current size

Valid values:	reals
Default value:	**10**
Minimum:	1.0
Maximum:	99.0

Operands:
in | out
Must specify whether to zoom in and make larger or out and make smaller.

Command Index

1

2

A

B

C

D

E

F

G

H

I

J

L

M

N

O

P

Q

R

S

T

U

V

W

X

Z

Table of Contents