Olex2 Manual (Commands)

Draft: November, 2014

Contents

This document describes some of the commands that are available in Olex2. Many of these commands are also available directly from the Olex2 Graphical User Interface. Most items on the GUI have a small 'info' symbol next to them, where you can find out more about any of these items.

Introduction

There is no special console window in Olex2 – the commands described in this document can be typed where ever you are in Olex2 and the text you type (as well as the program response) will appear in the bottom left hand corner of the main window. The text will then scroll up behind the displayed molecule. The number of lines of text that are visible can be set with the command **lines *n*.** You can also toggle between showing the **molecule only**, showing the **text only** and showing both at the same time (default) using **CTR+T**. You can always examine the text output in your default text editor by typing **text**.

Many commands in Olex2 are modelled on the syntax that may be familiar from SHELX: four letter commands, where the letters often provide a hint about the function of the command. Many commands that are available in XP, for example, can be used in Olex2. Also, all commands of the ShelXL and ShelXS syntax are interpreted by Olex2 and used to construct the internal Olex2 structure model. This model is then used directly to carry out an olex2-refine refinement, whereas a shelx.ins file is generated on the fly if ShelXL/XH is chosen for the refinement.

All commands in Olex2 will **auto-complete** when pressing the TAB key. If the completion is not possible, because there is more than one command starting with the letters that have been typed, a list of these commands will be printed. It is good practice to use the auto-complete feature!

Understanding the Syntax

Selection: If one or more atoms are selected on the screen, then any command that acts on a selection will apply to the ***selected atoms only***. If there is no selection, it will apply to ***all*** atoms. Instead of making a selection on the screen, a list of atom names can also be supplied. *If a command has been successful, the selection will disappear. (Although there are a couple of exceptions to this rule)*

Mode: If Olex2 is in a **Mode**, the chosen action will be applied to all subsequently clicked atoms. The mouse pointer will change from the default arrow symbol to signify that Olex2 is in a mode. To get out of a mode, simply press the **Esc** key.

Syntax used in this document:

{a, b, c}: choice of a, b or c. For example: fix {occu, xyz, Uiso} [atoms] means 'fix occu [atoms]', 'fix xyz [atoms]', 'fix Uiso [atoms]'.

[val=2]: optional parameter. This parameter is not required for the command to work, and if it is not supplied, the default value will be used.

-**k:** This is an option switch.

i: Italic characters are used for variables.

[atoms] means an optional list of atoms. Any atoms that are selected will automatically be present in this list. If there are no selected atoms, **all** atoms will be in this list. Alternatively, the atom names of the atoms that should appear in this list can be typed by hand.

atoms means a compulsory list of atoms. Any atoms that are selected will automatically be present in this list. Alternatively, the atom names of the atoms that should appear in this list can be typed by hand.

Capital Letters are used for commands that will directly affect the structure model in the refinement. These commands will become part of the structure model and will appear in the ShelX input file. Please note that these commands can be typed either in upper or lower case.

Example Commands are represented in this format: `refine 4 20` and can be typed exactly as they are given. In this example, the structure will be refined with 4 refinement cycles and 20 electron density peaks will be returned from the electron density map integration.

Tables of Olex2 Commands

Changing the Model View

matr	[1,2,3 or abc] or [abc a1b1c1] or [x11 x12 x13 y11 y12 y13 z11 z12 z13]	Orients the model along a (1 or 100), b (2 or 010), c (3 or 001) or any other crystallographic direction, like 123, which sets current normal along (1*a+2*b+3*c) vector. Two crystallographic directions (from and to) may be specified align current view normal along the (to-from) vector. Also a full Cartesian matrix can be specified. If the directions are signed or consist of multiple digits all components should be of the same length like in 120101 or -1+1+1 (same as -10101). If no arguments given, prints current Cartesian orientation matrix. **-r**: uses reciprocal lattice instead of the direct **Examples**:

		matr 1 or **matr** *a* or **matr 100** - sets current normal along the crystallographic ***a*** direction **matr 100 011** sets current normal along (011-100) direction (the normal direction changes if from and to are swapped)
rota	[axis angle] or [x y z angle increment]	Changes current view by rotating around given axis (x, y or z) when two arguments are provided and makes a continuous rotation around give axis when 5 arguments are provided. Note that X axis is aligned horizontally, Y - vertically and Z is out of the screen plane. **Examples**: **rota x 90** rotates the structure 90 degrees around X axis **rota 0 0 1 90 1** rotates model in the screen plane (around Z) 90 degrees with 1 degree increment.
direction		The command prints current normal in crystallographic coordinates and tries to match it to a crystallographic direction.
mpln	[atoms] [**-n**] [**-r**][**-rings**]	Finds the best plane through the current selection or given atoms, or out of all visible atoms if none are given. **-n**: if no value is given - sets the view along the normal of the plane, otherwise places the newly created plane to the given group name **-r**: creates a regular plane (normally it is a rectangle), if a number is given, the specified polygon is created **-rings**: creates planes for all rings given by a template like NC5
Line	[atoms or direction] [**-n**][**-f**]	Find the best line through the selection (or for the explicitly given direction expressed by 6 numbers) and either adds the line object or orients current view down to the line direction. -n: same as for the mpln -f [false]: for the explicit direction specifies if the given coordinates are Cartesian or fractional. Examples: **Line 0 0 0 1 0 0 -f**: creates a line along the crystallographic ***a*** direction.
Changing scale/zoom		To get current value of the scene zoom use: **echo zoom()** To set current zoom to a certain value use:

		zoom(eval(Value-zoom())) this can be used to put different structures to the same scale. Note that the value 1 corresponds to the scale when the smallest dimension of the screen view is 1 A To reset zoom to default for current model use: **gl.zoom**

The model can be ***rotated*** using by moving the mouse pointer while holding the left mouse button down (also Shift+arrow keys); ***rotated around Z*** by pressing the **CTRL** key down while rotating; ***zoomed*** using the right mouse button (also Shift+Home/End or Alt key+left mouse button); ***shifted in the viewing plane*** by pressing Ctrl+Shift and holding the right mouse button down or by holding left and right mouse buttons down. The default mouse behaviour can be overridden in some modes (look at mode split) also some objects, like cell, basis and text boxes can override some mouse operations (like zooming on the cell basis) or extend it (moving the basis while holding Shift key down).

Keyboard Shortcuts

CTRL+Q	**ShowQ**	Toggles between three states: show electron density peaks show electron density peaks with bonds hides electron density peaks
CTRL+H	**ShowH**	Toggles between three states: show hydrogen atoms show hydrogens with internal h-bonds hides hydrogen atoms
CTRL+T	**ShowStr**	Toggles between three states: show structure only show show structure and text show text only
CTRL+I	**sel -i**	Inverts the current selection.
CTRL+A	**sel -a**	Selects all atoms currently visible, however if labels are active (i.e. one or more label is selected) then this selects all labels.
CTRL+U	**sel -u**	Deselects all of current selection.
CTRL+G	**mode grow**	Enters mode grow. See also symmetry operations.

CTRL+O	**reap**	Brings up the Open File dialogue.
F2	**swapbg**	Swaps the background between white and coloured.
F3	**labels**	Toggles labels on/off.
F4	**grad -i**	Toggles gradient background on/off.
F5		Go to the **work** menu.
F6		Go to the **view** menu.
F7		Go to the **tools** menu.
F8		Go to the **info** menu.
F11	**Fullscreen(true/false)**	Toggles full screen mode on/off.
Shift+F11	**HtmlPanelVisible**	Toggles html panel on/off.
ESC		Exits current mode (some modes, like mode *match*, can override this), clears current selection and text in the command line
Break		Interrupts the solution/refinement after the current cycle.
Del (Ctrl+Backspace on Mac)	**kill**	Deletes selected object(s)

Fixed/Refined Parameters

fix	{occu, xyz, Uiso} [atoms]	Fixes the specified refinement parameter, ie these parameters will not be refined in subsequent refinement cycles. **occu**: will fix the occupancy **xyz**: will fix the xyz coordinates **Uiso**: will fix the whole ADP **Examples:** **fix occu 0.5**: will set and fix the occupancy of the current selection to 0.5 **fix xyz**: will fix the x, y and z co-ordinates of the currently selected atoms, ie not refine them.
free	{occu, xyz, Uiso} [atoms]	The opposite of fix - makes the specified parameters for the given atoms refineable. Feeing the occupancy is also available

		from the context menu.
mode	fixu	Fixes Uiso or ADP for subsequently clicked atoms.
mode	fixxyz	Fixes coordinates for subsequently clicked atoms.
mode	occu occupancy_to_set	Sets atoms occupancy to the provided value for subsequently clicked atoms.
labels -f show currently fixed atomic parameters, **labels -f -r** show labels for fixed atoms and also the number at which the occupancy of riding atoms is fixed		

Atom Connectivity Table Manipulation

conn	***n*** [*r*] atoms	Sets the maximum number of bonds for the specified atoms to ***n*** and changes the default bond radius for the given atom type to ***r.*** **Examples:** **conn 5 $C** sets the maximum number of bonds all C atoms can have to 5, **conn 1.3 $C** changes the bonding radius for C atoms to 1.3 (the floating point is used to distinguish between ***n*** and ***r*** in this case!), **conn 5 1.3 $C** combines the two commands above
compaq	[**-a**] [**-c**] [**-q**] [**-m**]	Moves all atoms or fragments of the asymmetric unit as close to each other as possible. If no options are provided, all fragments are assembled around the largest one. **-a:** assembles broken fragments **-c:** similar to the default behaviour, but considers atom-to-atom distances and will move all atoms to the closest possible position to the largest fragment in the structure. **-q:** moves the electron density peaks close to the atoms. **-m**: disconnects metals, then does *compaq -a* and the reattaches the metals.
addbond	**A1 A2** *or* atoms	Adds a bond to the connectivity list for the specified atoms. This operation will also be successful if symmetry equivalent atoms are specified.
delbond	**A1 A2** *or* Selected bond(s)	Removes selected bonds from the connectivity list. Use this command to permanently remove bonds from the display too.

sort	[**m**] [**l**] [**p**] [**h**] [**z**] [**n**] [**s**] atoms [**s**] [**h**] [**m**] moiety	The sorting of atoms in the atom list is very powerful, but also quite complex. **-m**: atomic weight -**z**: atomic number **-l**: label, considering numbers **-p**: part, 0 is first followed by all positive parts in ascending order and then negative ones **-h**: to treat hydrogen atoms independent of the pivot atom. **-s**: non-numerical label suffix **-n**: number after the atom symbol Sorting of moieties **-s**: by size **-h**: by heaviest atom **-m**: by molecular weight **Usage:** sort [+atom_sort_type] TBA sort [Atoms] [moiety [+moiety_sort_type] [moiety_atoms]] If just 'moiety' is provided - the atoms will be split into the moieties without sorting. **Examples:** **sort +m1 F2 F1 moiety +s** will sort atoms by atomic **m**ass and **l**abel, put F1 after F2 and form moieties sorted by **s**ize. Note that when sorting atoms, any subsequent sort type operates inside the groups created by the preceding sort types.
name	[selection/atom names] [-c] [-s=]	The command allows changing the atom names. Options c: check if the generated names are unique s: change the suffix only (no value removes the suffix, i.e. the part of label after the element symbol and numerical value) Examples:

		name O1 O2: renames O1 to O2 name 1: (some atoms selected) sequentially names the atoms in the order of the selection by adding 1,2, etc to the element symbol. Note that in this case if any generated name is not unique (and the -c option is not given), a random name will be generated name $q C: changes the element type of Q to C - all the electron density peaks will become carbons name sel -s=a: changes suffix of the selected atoms to 'a', replacing any existing suffix. Note that sel is a required keyword in this case (but may be removed in the future) name Q? C?: change type for all electron density peaks with single number label to carbon atoms preserving the number
mode	name [-**p**] [-**s**] [-**t**] [-**a**=0]	Puts the program into the naming mode. Options: -**p**: label prefix -**s**: label suffix -**t**: element symbol -**a**: autocomplete, off by default. Value 1 switch the autocompleting on with value 2 stopping the procedure when an atom of a different type is encountered on the way, 4 – when an atom with different part is encountered on the way and 6 – combining the cases of 2 and 4. A special value 8 does automatic naming.
FixHL		Updates H-atom labels according to the labels of the bearing atoms
mode	grow -a	The mode to reconstruct the asymmetric unit (look at the mode grow shells option for details).
Olex2 will display the altered connectivity table in the case if structure is grown or packed		

Symmetry Operations

lstsymm		Prints symmetry operations and their codes for current structure.
envi	[*r*=2.7 Å] **A1** *or* selected atom [-**h**] [-**q**] **Note:** if more than one atom is selected the first one is used	Prints a list of those atoms within a sphere of radius ***r*** around the specified atom. -**h**: adds hydrogen atoms to the list -**q**: option adds Q-peaks to the list

mode	**grow** [-**s**] [-***v***] [-**r**] [-**a**] [-**shells**]	Displays the directions in which the molecule can be grown -***s*:** also shows the short interaction directions -***v*:** [2.0 Å] shows directions to the molecules within ***v*** value of the Van der Waals radii of the selected atoms which can be generated by clicking on the direction representations, only unique symmetry operations (producing shortest contacts are displayed). If an atom is selected before entering this mode - the environment of only this atom(s) can be grown. -***r*:** shows directions to all symmetry equivalent atoms of the selected one(s) within 15 Å -***a***: the asymmetric unit reconstruction mode shortcut **CTRL+G** is used to enter the '**mode grow**' **-*shells***: only applicable in '**mode grow -shells**' - allows growing atom by atom, if a 'grow' bond is clicked, only the immediate attached to that bond atom is grow, if the atom with outgoing 'grow' bonds is clicked - atoms for all bonds are grown
mode	**pack**	Displays the position of symmetry equivalent asymmetric units as tetrahedra. These asymmetric units can be generated by clicking on the corresponding tetrahedron.
sgen	atoms The Symmetry operation is represented as 1_555, 1555 or -1+X,Y,Z and atoms as a selection or a names list. As a special case twelve numbers can be provided to specify any matrix operating on the fractional coordinates (e.g. see the match)	Generates symmetry equivalents of the provided (or all atoms, if there is no selection) using the provided symmetry operation. **Note:** For symmetry operations starting with '-' and letter, a leading zero must be added or the expression has to be quoted, for example, **0-x,-y,-z**, otherwise Olex2 confuses this with an option.
pack	**a_from a_to b_from b_to c_from c_to** [atoms]	Packs all or specified atoms within given dimensions **-c:** prevents clearing existing atoms **Example: pack $O** will pack all O atoms with the default of -1.5 to 1.5 cells range.
pack	**from to**	Equivalent to '**pack from to from to from to**', like '**pack 0 1**' is expanded to '**pack 0 1 0 1 0 1**'
pack	**cell**	Shows content of the unit cell. In conjunction with '**grow -w**' allows the creation of views where all asymmetric units contributing to the unit cell are shown.

pack	**wbox**	Packs the volume inside the 3D selection box (the once can be shown by selecting at least 3 atoms and typing 'sel wbox'), to pack multiple boxes - use '-c' option.
pack	***r***	Packs fragments within radius ***r*** of the selected atom(s) or the centre of gravity of the asymmetric unit.
grow	[atoms] [**-w**] [**-s**]	Grows all possible/given atoms; for polymeric structures or structures that require to be grown several times Olex2 will continue grow until the operation results in a symmetry element that has been used previously. **-w:** permits the application of symmetry previously used operations to other fragments of the asymmetric unit **-s**: grows atom shells only **Example:** If the main molecule is grown, but only one solvent molecule is shown, using '**grow -w**' will produce other solvent molecules using symmetry operators used to grow the main molecule
If some atoms are deleted after growing operations, Olex2 will use existing unique atoms as the asymmetric unit atoms; this can be helpful to avoid a sequence of sgen/kill commands. **labels -l -i**: Adds labels only to the 'original' - i.e. not created by symmetry - molecule. In a packed structure: **Right-click on a bond > Graphics > Select the Groups(s)**: Will select all bonds (or atoms) of that type in the grown structure.		

Disorder Modelling: Constraints and Restraints

EXYZ	atom types (to add for the selected atom) [**-EADP**] [**-lo**]	Makes the selected site shared by atoms of several atom types. **-EADP:** adds the equivalent ADPs command for all atoms sharing one site. **-lo:** links the occupancy of the atoms sharing the site through a free variable.
EADP	atoms	Makes the ADP of the specified atoms equivalent.
SADI	atoms *or* bonds [**esd**=0.02]	For selected bonds or atom pairs **SADI** makes the distances specified by selecting bonds or atom pairs similar within the esd. If only **one** atom is selected it is considered to belong to a regular molecule (like PF_6) and adds similarity restraints for P-F and F-F distances. For 2 selected atoms the first is considered as the rotor bearing atoms and the second the pivot defining. The

		command inserts 3 bond similarity restraints to make a regular rotor. Applicable to triflate-like groups. For **three** selected atoms (**A1**, **A2**, **A3**) it creates similarity restraint for **A1-A2** and **A2-A3** distances.
DFIX	***d*** atom pairs or pairwise selection in order [**esd**=0.02]	For selected bonds or atom pairs **DFIX** will generate length fixing restraint with the given esd. If only **one** atom is selected, all outgoing bonds of that atom will be fixed to the given length with provided esd. For **three** selected atoms (**A1**,**A2**,**A3**) the A1-A2 and A2-A3 restraints will be generated.
DANG	***d*** atom pairs *or* pairwise selection in order [**esd**=0.04]	For selected bonds or atom pairs, distance restraints similar to dfix will be generated.
tria	***d1 d2 angle*** [**esd**=0.02]	For given set of bond pairs sharing an atom or atom triplets generates two dfix commands and one dang command. **Example: tria 1 1 180 C1 C2 C3** will generate 'DFIX 1 0.02 C1 C2 C2 C3' and 'DANG 2 0.04 C1 C3' it will calculate the distance for dang from d1 d2 and the angle.
RRings	[**d**=1.39] [**esd**=0.01] **ring_content** or selection	Find rings using the selection or rings content (like C6) and sets DFIX restraint for the bond lengths using the **d** parameter and FLAT with e.s.d. of 0.1 restraint for the ring. It also adds SADI restraints for the 1-3 distances. If **d** is negative, the SADI restraint is used instead.
FLAT	[atoms][**esd**=0.1]	Restrains given fragment to be flat (can be used on the grown structure) within given esd.
CHIV	[atoms][***val***=0] [**esd**=0.1]	Restrains the chiral volume of the provided group to be ***val*** within given esd
SIMU	[***d***=1.7] [**esd12**=0.04] [**esd13**=0.08]	Restrains the ADPs of all 1,2 and 1,3 pairs within the given atoms to be similar with the given esd.
DELU	[**esd12**=0.01] [**esd13**=0.01]	'rigid bond' restraint
ISOR	[**esd**=0.1] [**esd_terminal**=0.2]	Restrains the ADP of the given atom(s) to be approximately isotropic
SAME	N	Splits the selected atoms into the N groups and applies the SAME restraint to them. Olex2 will manage the order of atoms within the ins file, however mixing rigid group constraints and the 'same' instructions might lead to an erroneous instruction file. Note that if only two atoms are selected in two fragments with identical connectivity, Olex2 employs the matching procedure and sets SAME for

		the two fragments to which the atoms belong.
showp	[any]; space separated part number(s)	Shows only the parts requested: **showp 0 1** will show parts 0 and 1, **showp 0** just part 0. **showp** by itself will display all parts.
split	[-**r**={eadp, isor, simu}]	Splits selected atom(s) along the longest ADP axis into two groups and links their occupancy through a free variable. **-r:** adds specific restraints/constraints (EADP, ISOR or SIMU) for the generated atoms
AFIX	shelx afix number{mn} [-**n**]	If no are atoms provided and afix corresponds to a fitted group where n is 6 or 9 (such as 106 or 79), all the rings which satisfy the given afix will be automatically made rigid (this is useful in the case of many PPh3 fragments); alternatively a single ring atom can be selected to make that ring rigid. In other cases, depending on afix either 5,6 or 10 atoms will be expected. Special cases of afix 0, 1 and 2 can be used to remove afix, fix all parameters or leave just the coordinates refinable, all other afix instructions will consider the first atom as a pivot atom and the rest - dependent atom. The AFIX command can also be used to generate missing atoms to complete rings or fragments. For example, the following command generates three missing atoms in positions 4,5 and 6 for the Ph ring when applied to a selection of 3 atoms (assumed to be in positions 1, 2 and 3): AFIX 66 1,2,3 Note that there are no white spaces between the identification of the selected positions. **-n:** consider N-atoms as parts of rings
part	[**part**=new_part] [**occupancy**] [atoms] [-**p**=1] [-**c**=false]	Changes part number/occupancy for given/selected atom; **-lo**: links occupancies of the atoms through a +/-variable or linear equation (SUMP) depending on the -**p**[=1] **-p**: specifies how many parts to create. If -**p**=1, -**lo** is ignored and the given or new part is assigned to the provided atoms. If the number of parts is greater than 2 and –**lo**: option is given, a new SUMP restrain will be automatically added. -**c**: if the atoms are symmetry generated, this command put them into the asymmetric unit.
fvar	[***value***] [atoms]	This command links two or more atoms through a free 'fvar'. If nothing is selected will print current values of the variables. For a selection of even number atoms, will create

		a new variable and link occupancies of the first half of the selection to occupancy the other half of the selection. **Examples**: -'**fvar 0**' - makes occupancy of provided atoms refineable -'**fvar 1**' - fixes occupancy of provided atoms at current value -'**fvar 1 1**' - fixes occupancy of provided atoms at chemical occupancy of 1 -'**fvar 2**' will link occupancy of the given atoms to the value of the 2nd FVAR multiplied by current value of the occupancy of the given atoms, or, if occupancy already linked to a variable - it will replace the variable index. -'**fvar 2 0.5**' will link occupancy of the given atoms to the value of the 2nd FVAR multiplied by 0.5.
sump	[***val***=1] [**esd**=0.01]	Creates a new linear equation. If any of the selected atoms has refinable or fixed occupancy, a new variable is added with value 1/(number of given atoms), otherwise already used variable is used with weight of 1.0. Also look at part command. **Example:** If 3 atoms (A1, A2, A3) are selected this command will generate three free variables (var1, var2 and var3) and inserts the **‘SUMP 1 1 var1 1 var2 1 var3’** instruction (equivalent to 1.0 = 1.0*occu(A1) + 1.0*occu(A2) + 1.0*occu(A3).
mode	split [-**r**={eadp, isor, simu}]	Splits subsequently clicked atoms into parts, or in combination with the Shift key can be used to drag an atom to change its position. While in the mode the newly generated atoms can be selected and moved as a group with Shift down or rotated when dragging the selection. The original and generated atoms will be placed into different parts. -**r**: can be used to generate extra restraints or constraints for original and generated atoms (see also the '***split***' command); values EADP, ISOR or SIMU are allowed
mode	fit [-s]	Allows fitting selected group (moving and rotating in 3D). **-s**: a new group is created at the fitted location and the occupancy of this and the original group is constrained to be 1 This mode is automatically activated by the ImportFrag command.

ImportFrag	[-a] [-d] [-p]	Allows importing an XYZ file into the current model. Mode fit is automatically executed to help with fitting the imported molecule. **-a**: sets give AFIX to the imported molecule **-d**: generates DFIX for 1,2 and 1,3 distance for the imported molecule **-p**: sets given part to the imported molecule
restrain	ADP [Ueq] {Ueq, volume}1 Bond [d atoms] Angle value [atoms][1] Dihedral value [atoms]1	This is a generic macro to generate restraints. The ADP restraint for Ueq can take a number to restrain Ueq of the given atoms to, by default the Ueq/volume similarity restraint is generated.
constrain	U [atoms] Site or xyz [atoms] Same group [n=2 atoms]1	This is a generic macro to generate constraints. Constrain U generates EADP constraint, site or xyz – EXYZ. The same group or non-crystallographic symmetry constraint makes two or more groups identical and linked through a transformation matrix, refined as a shift and 3 Euler angles. If two atoms are given, they must belong to two identical fragments; Olex2 will then try to match the fragments containing the atoms and automatically generate the constraint. In more generic/complex cases the user has to provide the number of groups to generate the constraint for and also the selection which matches atoms in the fragments.
xf.rm.ShareADP1	[atoms]	Generates the shared rotated ADP constraint. For 3 atoms generates an ADP rotated around the bond, like around X-C bond in X-CF_3, for more atoms creates an ADP rotated around the normal of the plane formed by the atoms.

Selection Syntax

sel atoms where xatom.bai.mw > 20	Select all atoms where the atomic mass is larger than 20
sel atoms where xatom.bai.z > 2	Select all atoms where the atomic number greater than 2
sel 1555	Will select all currently shown symmetry generated atoms which were generated by the given symmetry operation

[1] Available only with olex2-refine

sel rings NC5	Will select all NC5 rings in the structure
sel fvar -2	Selects atoms where a parameters is linked to FVAR 2 as (1- value of the FVAR)
sel part 1	Will select part 1 of the structure
sel isot	Selects all isotropic atoms
sel frag C5	selects the whole fragment containing C5
sel $E	Selects all atoms of the given type. E is a chemical element symbol or one of the following: * - all types M - all metals X - all halogens
sel $*,E	Selects all atoms, but of the E type, for example: Sel $*,H Selects all non-H atoms
sel wbox	Shows the 3D selection box constructed for all/selected atoms; this box can be used to pack the structure or to crop the voids display/electron density maps. If the third argument (cell) is provided, the frame gets the size/dimensions of the unit cell rather than being rectangular. The box appearance can be switched to box/sphere by using 'wbox.Type' command provided with the given name (box or sphere). This box can be used for packing and 3D maps trimming/extending.
sel **ofile** N	Selects overlayed file N; if N is 0 – elements of the currently focused file are selected

HKL file Operations

hklstat		Prints detailed information about reflections used in the refinement.
omit	h k l	Inserts 'OMIT h k l' instruction in the ins file. Use 'delIns omit' to remove all the OMITs from the INS file header.
omit	val	Inserts 'OMIT h k l' for all reflections with err > val, where err err = sig(D)(wD^2/<wD^2>)$^{1/2}$, where D=Fc^2-Fo^2 (Shelx) or just D (olex2.refine) .
omit	s 2theta	Inserts 'OMIT s 2theta' instruction in the ins file
hkledit	[h k l]	Brings up a dialogue, where 'bad' reflections from the Shelx lst file and all its constituent symmetry equivalents can be inspected and flagged to be excluded from the refinement. In contrast to the OMIT h k l instruction, which excludes the reflection and *all it equivalents*, this dialogue allows to exclude those equivalents that are actually outliers. If a particular reflection is specified, this particular reflection

		and all its constituent equivalents can be viewed.
hklexclude	-h=h1;h2;.. -k=k1;k2.. -l=l1;l2.. [-c]	This function provides a mechanism to reversibly exclude some reflections from refinement (these reflections will be moved to the end of the hkl file so they appear after the 0 0 0 reflection). **-c:** option controls how the given indices are treated, if not -c option is provided, then any reflection having any of the given h, k or l indices will be excluded, otherwise only reflections with indices within provided h, k and l will be excluded.
hklappend	-h=h1;h2;.. -k=k1;k2.. -l=l1;l2..	Acts in the opposite way to excludehkl
hklView		Shows reflection currently used in the refinement (use Ctrl+T a few times to center on the reflections view).
For more advanced HKL processing, a Python script may be used. A sample hklf5.py script is provided in {Olex2 folder}/etc/scripts. The script can be copied and modified to accommodate any particular twinning law and run inside Olex2. The script allows creating an HKLF 5 file where reflections which belong to different twin components are assigned different batch numbers. To run a python script in Olex2 use the following command to load the script: >>@py -l This command shows a 'File Open' dialog, a python script can be selected. After loading the script can be modified and executed by pressing OK.		

Customising the Olex2 GUI

setfont	{Console, Picture_labels}	Brings up the dialog to choose font for the Console or Labels which end up on the picture. Built in function choosefont([olex2]) to choose system or specially prepared/portable font can be used to specify the font.
EditMaterial	{helpcmd, helptxt, execout, error, exception, any object name available with *lstgo*}	Brings up a dialog to change properties of the specified text section or graphical object. helpcmd - material for the command name in the help window helptxt - material for the body of the help item execout - material for the output text printed in the console of external programs error - material for reporting errors in the console exception - material for reporting exception in the console This command can be used to edit properties of any objects printed by '*lstgo*' macro. An example of that could be editing material of the console text:

		EditMaterial **Console** Note that the object name is case sensitive.
save	{scene, style, view, model} [file_name]	If the file name is not provided, the 'Save as...' dialog will be shown which allows to save current settings to file. The scene save current font names/sizes as well as the materials for the specific console output, like external programs output, error and exception reporting. The style saves information about the appearance of objects in the scene. The view saves current zoom and the scene orientation. The model saves current view including the crystallographic mode and style.
load	{scene, style, view, model, radii} [file_name]	Load one of the previously saved items. If no file name is provided, the 'Open file...' dialog will appear, otherwise if just a file name is provided (the extension will be guessed by Olex2), for styles and scene, the last used folders will be used by default, whereas the current folder will be used for the views and models. Loading radii (vdw, pers, sfil) allows the user to change the radii Olex2 uses for various calculations/display.
grad	[C1 C2 C3 C4] [-**p**]	Choose the colour of the four corners of the graduated background. **-p**: a file name for the picture to be placed at the background
brad	***r*** [hbonds] operates on all or selected bonds	Adjust the bond radii in the display. If the 'hbonds' is provided the second argument, the given radius is applied to all hydrogen bonds.
ads	{elp, sph, ort, std}	A function for drawing styles development. Changes atom draw style for all/selected atoms. elp - represents atoms as ellipsoids (if ellipsoids are available) sph - represents atoms as spheres ort - same as elp, but spheres have one of the quadrants cut out std - a standalone atom (i.e. shown as a cross in wire-frame mode)
arad	{sfil, pers, isot, isoth, bond, vdw}	A function for drawing styles development; applies different radii to all/selected atoms. sfil - sphere packing radii (as in ShelXTL XP) pers - a fixed radii for model viewing isot - each atom has it's own radius depending on the value of

		the Uiso or ADP isoth - same as isot, but the H atoms are also displayed with their real Uiso's bond - all atoms get same radii as default bond radius vdw - the default/loaded Van der Waals radii used in most of the calculations
azoom	% [atoms]	Changes the radii of all/given atoms, the change is given in percents.
group	selection	Groups current selection. -**n**: provides a custom name for the group -**u**: ungroups the selected groups or all groups if nothing is selected

Output: Tables, Reports and Images

pictPS	filename.ps	Generates a post-script file of what is visible in the molecule display. -**atom_outline_color** - the colour of the atom outline, used for extra 3D effect for the intersecting objects [0xFFFFFF] -**atom_outline_oversize** - the size of the outline [5]% -**bond_outline_color** - same as for the atom, can be changed to black to highlight bond boundaries -**bond_outline_oversize** - the size of the outline [10]% -**color_fill:** Fills the ellipses with colour. -**color_bond:** Bonds will be in colour. -**color_line:** Lines representing the ellipses will be in colour. -**div_pie:** number [4] of stripes in the octant -**lw_ellipse:** line width [0.5] of the ellipse -**lw_font:** line width [1] for the vector font -**lw_octant:** line width [0.5] of the octant arcs -**lw_pie:** line width [0.5] of the octant stripes -**p:** perspective -**scale_hb:** scale for H-bonds [0.5]

		The bond width is taken from the display. This can be changed with **brad**
Pict	filename.*ext [**n**=2]* [-**pq**] [-**dpi**]	Generates a bitmap image of what is visible on the molecule display. ***n*** Refers to the size of the output image. If ***n*** is smaller than 10, it refers to a multiple of the current display size, if it is larger than 100, it refers to the width of the image in pixels. ext {png, jpg, bmp}. png is best. -**pq:** print quality -**nbg**: removes the bacground from the picture (making it transparent with the alpha channel) -**dpi**: physical resolution of the image
Picta	filename.*ext* [***n***=1] [-**pq**] [-**dpi**]	A portable version of **pict** with limited resolution (see explanation for ***n*** above), which is OS and graphics card dependent. This function will also use the graphics card settings like antialiasing when producing the picture. -**pq:** print quality -**nbg, -dpi**: as for 'pict'
PictPR	filename	Creates PovRay file for current view
PictS	filename.ext [***n***=1] [-**a**=6] [-**s**=10] [-**h**=n*(screen height)]	Creates a 'stereo' picture with two views taken with the +/- **a** option value rotation around y axis and placed onto one picture separated by s % of a single projection width. -**a**: half of the view angle -**s**: separator width in % -**h**: the height of the output, by default equals to current screen height multiplied by the given resolution
label	**label** [atoms]	Adds labels to all/given/selected atoms and bonds. These labels can be moved by pressing the SHIFT key while holding down the left mouse button and edited by double clicking on them. **-type:** {subscript, brackets, default **-symm**: {[$], #, full} - if an atom is generated by non-identity symmetry operation, it will be added as a superscript. Note that if # is used as the symmetry identifier - every type new atoms generated (in growing or packing) the labels should be recalculated. Olex2 will then print current mapping of the symmetry numbers to the symmetry operators.

PiM		Helps with creating pictures when metal - pi interactions needs special drawing (like in the case of Cp-Me a single bond will be rendered from the ring centroid to the metal vs 5 bonds from every C atom to the metal). Note that currently these stippled bonds do not appear in the PostScript rendered pictures and thus for PS pictures a workaround with creating centroids is needed. To remove all of the lines created by this command - right click on one of them and choose Graphics->Select the group, then hit Del key.

Structure Analysis

There are various tools available for the analysis of structures.		
htab	[minimal angle=150°] [maximum bond length 2.9 Å] [-**t**] [-**g**]	Searches and adds found hydrogen bonds (like HTAB and RTAB in Shelx) into the list for the refinement program to add to the CIF. Equivalent symmetry positions are automatically inserted and merged with the existing ones. The command can be executed several times with different parameter values, only one unique instructions will be added. **-t:** adds extra elements (comma separated like in -t=Se,I) to the donor list. Defaults are [N,O,F,Cl,S,Br] -**g**: if any of the found bonds are generated by symmetry transformations, the structure is grown using those symmetry transformation
PiPi	[centroid-to-centroid distance 4 Å] [centroid-to-centroid shift 3 Å] [-**g**] [-**r**=C6,NC5]	The command analyses the p-p interactions (only stacking interactions for now) for flat regular C6 or NC5 rings and prints information for the ones where the intercentroid distance is smaller than [4] Å and the intercentroid shift is smaller than [3] Å. -**g**: if any of the rings is fully or partially constructed of symmetry generated atoms it grow the structure using those symmetry operators -**r**: ring content, the defaults are C6 and NC5 rings, the rings are tested for being flat and regular
calcvoid	[radii file name] [all atoms/selected atoms] [-**d**=0] [-**p**] [-**r**=0.2Å]	Calculates and displays the structure map. Also calculates the largest channels along crystallographic directions and the packing index. -**d**: extra distance from the surface (added to the atomic radii) -**p**: precise calculation, each map voxel is tested, the default

		quick algorithm, uses the atom masks to find volume occupied by the molecule. The precise calculation is vectorised **-r**: resolution, a resolution of at least 0.1Å and **-p** option is required to get values for publishing **Note:** The radii used in the calculation are currently coming from the CSD website: http://www.ccdc.cam.ac.uk/products/csd/radii However there are several ways how the radii can be changed, one of the ways is to provide a file name with radii ([element radius] a line format), the other one is to load the radii from the same kind of the file using 'load radii vdw' command.
molinfo	[radii file name] [atoms] [-**g**=5] [-**s**=o]	Calculates molecular volume and the surface area for all/selected atoms. -**g**: generation of the triangulation process -**s**: source of the triangles for the sphere triangulation, [*o*]ctahedron or [*t*]etrahedron are available Generation 5 for octahedron approximate sphere by 8192 triangles, for tetrahedron by 4096 triangles, each generation up increases the number of triangles by factor of 4, generation down - decreases it by the same factor.
calcfourier	{-**calc**,- **diff**, -**obs**, -**tomc**} [-**r**=0.25Å] [-**i**] [-**scale**=simple] [-**fcf**]	Calculates Fourier for current model -**r**: the resulting map resolution in angstrems -**i**: integrate the calculated map -**scale**: when Olex2 calculates structure factors, it uses the linear scale as a sum(Fo^2)/sum(Fc^2) by default, however a linear regression scale can be also used (use -**scale**=regression) -**fcf**: Olex2 will use an FCF with LIST 3 structure factors as a source of the structure factors. If this option is not specified, Olex2 will calculate the structure factors using the the reflection used in the refinement (use the 'hklstat' command to see more information on reflections).
calcpatt		Calculates and displays Patterson map
kill	Atom names or selection or 'labels'	Deletes given or selected atoms, bonds or labels. Note that if the bonds are deleted this way - they will reappear the next time the structure connectivity updated, use delbond to remove bonds permanently.
match	[atoms] [-**a**]	This procedure find relation between the connectivity graphs of molecular fragments of loaded structure and aligns the

	[-**w**] [-**i**] [-**n**] [-**u**] [-**esd**] [-**h**] [**-cm**] [-**o**]	fragments. If no arguments are given, the procedure analyses all fragments and in the case fragments with matching connectivity found, aligns Acta A45 (1989), 208 them and prints corresponding root mean square distance (RMSD) in angstroms. If two atoms are provided (explicitly by name or through the selection) the graph relation information - orientation matrix and the matching atoms is printed, use -**a** option to align the fragments. -**a**: align the fragments (used when a pair of atoms is provided) -**w**: specifies weight for the atomic positions - by default the unit weights are used. If this option is given - the atomic position are weighted by the element mass -**i**: try to invert one of the fragments -**n**: transfer labels from one fragment to another (two atoms should be provided for *from* and *to* fragments. If the value a symbol [or set of] this is appended to the label, '$xx' replaces the symbols after the atom type symbol with xx, leaving the ending, '-xx' - changes the ending of the label with xx. Note that if the molecules match with -**i** options, this should also be provided for the label transfer -**u**: restores the coordinates of the matched fragments, this is useful if grown structure is matched -**esd**: if the variance-covariance matrix can be located (after the refinement with the negative MORE option in the xl), the esd on the RMSD can be calculated using this option -**h**: does the final matching and RMSD calculation and without taking H-atoms into account -**o**: when overlaying molecules from different structures, whole lattices (if packed/grown) are overlayed, not only the two fragments. To use – select an atom in a fragment of one lattice and an atom in a matching fragment of the other lattice. When a selection of two atoms is given the command prints the alignment matrix. This matrix alongside the sgen command can be used to generate new atoms. Use the '**-cm**' option to copy the matrix to the clipboard. See the 'How to...?' section for more information.
Notes etc about Structure Analysis		

How to...?

Overlay molecules

Olex2 provides several tools to match/align/overlay fragments or molecules. If the fragments have the same connectivity, the user can just type 'match' for automatic matching of the molecules. Olex2 will search for molecular graphs of the same connectivity* and align them, printing corresponding RMSD in angstroms. If the structure has more than two fragments, one atom of pair of fragments can be selected and the following command issued to match the selected fragments only:

```
>>match
```

The match macro also takes the '-**i**' option. When this option is given, the procedure will try to match the first fragment and the second one with the inverted coordinates. The result will be printed as the RMSD and the transformation matrix, use the '-**a**' option to align the fragments.

Note that if your molecule is disordered or has high symmetry, the automatic matching might fail. Also if there are Q-peaks in the structure, they should be hidden (Ctrl+Q) or deleted using the following commands:

```
>>kill $Q
```

Since graph matching algorithm implementation does not use any pattern recognition for the optimisation, high symmetry may introduce huge number of possible graph permutations (like each CH3 group increases the number of permutations by a factor of 6 (3!)), it is recommended to hide the H atoms (Ctrl+H). If this does not help, a manual change of atom types may be required to break the symmetry.

If automatic alignment fails due to the difference in the connectivity of the fragments, the user might select at least three atoms of one fragment and the same number of atoms in another fragment in matching sequence and type

```
>>match
```

to match fragments using only the selected atoms.

There is also a matching mode, which can be activated by typing:

```
>>mode match
```

This mode enables interactive matching by a maximum of three pairs of atoms. The first pair of atoms are superimposed, the second one causes the rotation to minimize the distance between the atoms of the second pair, the third pair causes rotation around the line formed by the first and second pair to minimize the difference between the atoms of the third pair.

Olex2 also provides a way to load an extra structure on screen. That structure can also be used in all the matching procedures described above. To load an extra structure, type:

```
>>@reap -* [file_name]
```

if the file name is not given, the 'File Open' dialog will apear.

*The fragment connectivity can be adjusted using the AddBond, DelBond and the Conn commands. For example if the compound under the consideration is a metal complex with two or more identical ligands, the ligands can be 'detached' from the metal by selecting the metal atom and typing

```
>>conn 0
```

Copy naming scheme from one fragment to another

Section above describes how to match/overlay fragments. This section describes how to transfer labelling scheme from one fragment to another for consistent labelling. Note that if the '-**i**' option was used for the matching; it also should be used for the naming.

An atom of the fragment with the original naming scheme and an atom of the fragment to which the naming scheme to be transferred should be selected, then the following command have to be typed:

```
>>match -n=mask
```

If the mask starts from '$' or '-' a special action is taken. The '$' character instructs the procedure to replace the give number of characters after the '$' in atom labels with the characters, for example:

```
match -n=$2
```

will replace labels like C101, Cu10, C10a to C201, C20a and Cu20.

The '-' instructs the procedure to replace the ending chars of the labels with the give characters, for example:

```
match -n=-b
```

will replace labels like C101, Cu10, C10a to C10b, C10b and Cu1b.

Any other values of 'mask' are simply added to the labels, like

```
match -n=a
```

will replace labels like C101, Cu10, C10a to C201a, C10a and Cu10a.

Note that the labels may become invalid for the use with some programs and will be trimmed/changed on the next file input/output operation.

Atom name suffix[2] can be changed by the following command:

```
>>name [atoms] -s=[suffix]
```

if no atom names provided, suffix of all atoms will be changed to the provided one or removed (if no value is provide for the '-**s**' option)

Get esd's on geometric measurements

Olex2 can calculate esd's on a variety of geometric measurements, also if a CIF is loaded into Olex2 then esd's available in the CIF (bond lengths and the angles) will be displayed in the tooltip when hovering over the bonds or selection of the bonds; use the 'sel' command to print the values in the console.

To calculate the esd's Olex2 need the variance-covariance matrix. This can be readily produced by shelxl when refining with a negative MORE instruction (like MORE -1). It also will be soon available from the smtbx refinement. Once the matrix file is available, the geometric parameters can be easily calculated by selecting:

[2] The suffix here is assumed the ending of the atom name following the atom symbol or any number, e.g. for C12a, suffix is 'a', for C12 the suffix is empty and for Cz the suffix is 'z'

a bond or two atoms – length two bonds or three atoms – angle four atoms - torsion angle and the tetrahedron volume a plane - RMSD and the centroid coordinates in fractional and Cartesian coordinates a plane and an atom - atom to plane and atom to plane centroid distances a plane and a bond - angle between the plane normal and the bond two planes - angle between the plane normals, plane centroid to plane centroid, plane to centroid and the shift between the plane centroids distances three planes - the angle between the plane centroids

Then type:

```
>>esd
```

to get the measurements with esd's.

Note that the current implementation of the esd calculation procedure does not consider symmetry constraints and gives esd's even for the symmetry related parameters.

Automated parameter recalculation

Olex2 can recalculate some parameters to help with report generation, paper writing and other tasks which may require updating your structure. The parameters are calculated and placed as Olex2 variables. Currently the user can specify bonds, angles, plane normal to bond angle and various plane-to-plane parameters calculation. The following examples show the usage of this feature.

‘DefineVar param_name’:

- for two selected atoms will store the interatomic distance in the ‘olex2.calculated.param_name’
- for three selected bonds it will be the angle stored in ‘olex2.calculated.param_name’
- for the election of a plane and a bond it will be the angle between the plane normal and the bon stored in ‘olex2.calculated.param_namen’
- for two selected planes it will be 5 parameters, centroid-to-centroid distance stored in ‘olex2.calculated.param_namecc’, centroid of the first plane to the second plane distance stored in ‘olex2.calculated.param_namepc’, angle between the plane normals stored in ‘olex2.calculated.param_namea’ and plane centroid shifts stored in ‘olex2.calculated.param_namesa’ and ‘olex2.calculated.param_namesb’

To enforce the variable calculations, type ‘CalcVars’ command which will also print the calculate values.

Automate tables of selected parameters

There is an easy way in Olex2 to automate the generation of the selected tables. You have to put the definitions once and then every time the CIF is recreated it is automatically updated to specify the information required for the selected parameter tables. The key command is ‘sel cif’ followed by the parameter – bond, angle, torsions (or dihedrals) and then followed by the parameter mask. For example the following command will select all metal containing bonds ‘sel cif bonds $M’. If you want to be more specific, you can put ‘sel cif bonds $M $N’ to select bonds only from metal to N atoms. There is also a shortcut for any atom – ‘$*’ which can be used to specify the position of the desired atom. For example this command ‘sel cif angles $* $M $*’ will selec only angles where the metal is in the middle.

Get a stereo view

There are four stereo modes available in Olex2:

Color stereo (resulting in monochrome stereo image)

Anaglyph stereo (resulting in grey-scale color for colors matching with colors of the glasses)

Rendering of two spatially separated projections

Hardware stereo

The first three modes do not put any constraints on the hardware besides that the graphics card should be able to render two image of the scene with convenient frames per second (fsp). These modes work with 'standard' red-green or cyan-blue glasses.

The hardware 3D stereo requires special graphics card and 3D stereo glasses. A test stand we have to demonstrate the hardware 3D stereo is built using the following components:

- ✓ 120Hz refresh rate display (we have Samsung)
- ✓ NVIDIA® 3D active shutter glasses (we have NVIDIA 3D vision kit)
- ✓ NVIDIA® Quadro FX graphics card (we have the low end FX580)

The standard NVIDIA® 3D vision kit is designed for the games and works only with the DirectX® based games (it is Windows® specific too). The Quadro series of the graphics cards provide a generic, OpenGL® based, portable mechanism for the 3D stereo rendering.

To switching stereo mode on/off use either the View tab of the GUI, or the following command:

>>gl.Stereo({*none, cross, color ,anaglyph, hardware, interlace*}, [angle=3]**),**

by providing the angle argument the viewing angle can be changes, the angle is signed and by inverting the sign the left and right projections are swapped.

To change the projection colours use the following command:

>>**gl.StereoColor(**{lef,right}, R,G,B[,A]**)**

R,G,B and A (optional) are the color components, floating point numbers in range of [0,1]. For example to set color of the left (mind the note above about the viewing angle) projection to red and of the right projection to cyan colors:

gl.StereoColor(left,1,0,0)
gl.StereoColor(right,0,1,1)

Pack or grow molecule(s)

There are several commands in Olex2 to grow and pack molecules. First command is 'grow', this command grows all atoms in the asymmetric unit. Grow command will generate the molecule until all newly available growing matrices differ only by translation part, this will create complete set of atoms for discrete molecules and generate quite a large fragment for polymeric structures. If a particular symmetry operator needs to be used, the 'sgen' command might be useful, for example:

```
>>sgen 1556 $N
```

or

```
>>sgen x,y,z+1 $N
```

will generate all nitrogen atoms using the x,y,z+1 symmetry operator (the identity operator, x,y,z is always the first operator in Olex2). If no atoms provided, all atoms will be generated. It is also possible to click on an atom which can be grown, and choose the 'Grow' option from the menu.

To find out the set of operators for loaded structure, use:

```
>>lstsymm
```

There is also growing mode (look in the table for all related options of this mode), which provides visual and metric information about bonds which can be generated. The mode allows using one symmetry operator a time:

```
>>mode grow
```

The growing command generates only fragments which can grow; sometimes it is needed to generate the rest of the asymmetric unit as well, like after growing the main fragment all related solvent molecules need to be generated. For this use the 'grow -w' command, it will grow the rest of the asymmetric unit.

The information about matrices which can grow an atom (if on a special position) or a bond can also be found using the 'envi' command. To find out atoms in special position, use the 'degen' command, which prints the atomic position multiplicity (unless it is 1).

The pack command can be used to pack molecules or particular atoms. For example:

```
>>pack R
```

generates molecules which centre of mass is within sphere of R radius

```
>> pack cell
```

Shows the content of the unit cell. Use 'grow -w' to show all molecules contributing atoms to the unit cell.

```
>>pack $Fe
```

packs only specified atoms, '-c' option can be added to specify that current model should not be cleared.

```
>>mode move [-c]
```

once a fragment atom is selected this mode will copy (if '-c' is provided) or move any other fragment (which is clicked) as close to the selected atom as possible.

```
>>mode pack
```

Displays asymmetric units as a set of tetrahedron, clicking on which generates the asymmetric unit using that particular transformation.

The structure analysis commands like 'pipi' and 'htab' provided with '-g' option can also be used to grow molecules and visualise some particular interactions.

Finally, to show the asymmetric unit, type

```
>>fuse
```

Change P1 space group to P-1

Identify two atoms related by the center of inversion, select them and type

```
>> echo ccrd()
```

to get coordinates of the point between the selected atoms. This will be fractional coordinates of the proposed centre of inversion as 'x,y,z'. The type:

```
>> push -x -y -z
```

to move the content of the asymmetric unit so that the centre of inversion now is at (0,0,0). Then type:

```
>>changesg P-1
```

to change the space group. Olex2 will try to remove the symmetry related atoms, however if the atoms do not overlap within some value, they have to be removed manually or by typing:

>>fuse *r*

command, where *r* is the radius within which atoms of the same type get merged into a single one.

Sometimes the molecule looks 'broken' after this operation and the command 'compaq -a' has to be executed to assemble the molecule.

Change space group settings

Olex2 provides the 'sgs' command to change the space group settings (cell choice and the axis). For a monoclinic space group, type:

```
>>sgs axis_cellchoice  [output HKL file name]
```

For example:

```
>>sgs b1 b1.hkl
```

changes the current space group setting to make the 'b' the principal axis and for the cell choice '1' and creates the b1.hkl file with transformed reflections. It is also possible to enter just the principle axis, like 'a' or 'c'. Please note that the cell esd's will be estimated since no variance-covariance matrix is available for the transformation.

For orthorhombic space groups, at the moment, only 'sgs abc' is valid to transform settings to standard. Olex2 will change the cell the setting accordingly, modify all the atomic coordinates and the ADS's (if any) and, if the output HKL file name is provided, creates the new HKL file according to the transformation. You will have to choose that file for the refinement if needed.

To find out current space group settings, type:

```
>>echo sgs()
```

To control console and graphics visibility

Olex2 has a built in console for typing commands. Sometimes it is desirable to see only the text (output) or the structure. There are several commands and shortcuts to help with this. <Ctrl>+T toggles whether the molecule is displayed or not. So, if your molecule has inexplicably disappeared, it's always worth pressing this key combination...

The program output of Olex2 happens 'behind' the molecule. The wisdom of doing things this way can be debated, but it means that there are fewer windows cluttering your screen. It is possible to adjust the number of lines of output you see by typing:

```
>>lines n
```

(e.g. **lines 5**) to see only 5 lines of the output a time, it may be confusing for procedures producing more that 5 lines (like calcvoid). If you want to see all lines, type:

```
>>lines -1
```

Alternatively, you can type 'text' in the console (or use corresponding GUI links) to view the text output in an external text editor. You can always use PgUp and PgDn keys to scroll the console output. There is a limited buffer to hold the console output, however a full transcript of the Olex2 session is available in the log file, which can be displayed using:

```
>>log
```

command.

Select atoms that became 'too small'

By default, the size of the atom displayed on the screen is proportional to its ADP or Uiso. If the atom type is wrong, and the real element is much heavier, the Ueq will become very small and therefore the displayed sphere will be too small to select by mouse. There are two solutions to this problem:

1) You can select an area that includes the atom you wish to select by drawing a box around the atom with the Shift key and the left mouse button pressed simultaneously.
2) You can switch the view to a Ball and Sticks display (View>Quick Drawing Styles) or type:

```
>>pers
```

Also use (or the GUI):

```
>> telp
```

to switch back to the view when the atoms' proportional Uiso and ADP's are displayed.

Use ShelX programs in Olex2

ShelXL, ShelXLMP, ShelXS and ShelXM executables are available free for academic use from George Sheldrick. If ShelX programs are on the system PATH, then Olex2 will know that ShelX is installed. You could put the ShelX executables alongside the Olex2 executable (on Windows or Linux) or in the olex2.app/Content/MacOS folder (on Mac).

All ShelX commands are – or at least should be – processed by Olex2 correctly. Please let us know immediately if any correct ShelX command does not behave the way you expect it to behave.

Change default programs

To change default text editor, html editor or folder browser, you need to set the Olex2 variables associated with these programs. To keep the changes permanent, create the file 'custom.xld' in the Olex2 installation directory. For example the following construct placed in that file sets programs for KDE:

```
<user_onstartup help="Executes on program start"
  <body <args>
```

```
   <cmd
     <cmd1 "setvar(defeditor,'kate')">
     <cmd2 "setvar(defexplorer,'konqueror')">
     <cmd3 "setvar(defbrowser,'konqueror')">
    >
 >
```

This defines a function which is called when Olex2 starts up, please note if there are several functions with the same name are found in the files - the last one will be used. Please avoid overriding any functions in the macro.xld file as that may cause Olex2 to function incorrectly.

Working with the Maps display

The 2D map can be operated by the following controls:

Expand/Shrink – Ctrl + right mouse button

Level change – Shift + right mouse button (only in the Contour view)

Z change – Ctrl + left mouse button

xgrid.PlaneSize() - prints or sets the plane size. The default value of the plane - is the 128x128 pixels, if you have a high performing graphics card, you can set the value to 256, 512, 1024 or other power of two values. When a map is viewed as a 2D projection, the '*direction*' command prints the location of the center of the map (as well as the information string, normally appearing in the left to corner). Since map depth value is represented by the distance the map cuts on the current view normal, it is not very easy to position it at any particular crystallographic location. However there is a way to change the map center so that it cuts a specific value on any crystallographic direction. This can be achieved by the us of the xgrid.Depth(a,b,c) command. For example, if the one needs to show the contour map which cuts the cell at a/2, the following command can be used:

```
>>matr -r 1
```

to set current view along the reciprocal a axis

```
>>xgrid.Depth(0.5,0,0)
```

to set the map depth to the length a/2 vector cuts on current view normal.

Fix tooltips problems

Functionality:

```
>>GlTooltips(bool)
```

```
>>echo GlTooltips()
```

These two commands allow modifying or querying current tooltip management. By default there are native tooltips for Windows and OpenGL emulated ones fo the other platforms. The reason in the distinction is that native tooltips on Mac are rendered not at the time they were set (at least with wxMac), but after. This leads to displaying previous tooltip instead of current. The case of GTK is still under the investigation, but on older versions of tooltips would not be rendered. This can be overridden using the GlTooltip(true/false) command, the setting will be saved upon normal program termination. Rendering of the native tooltips on platforms, other than Windows will degrade the performance, since all mouse movement events (unless a mouse button is pressed) will cause the scene re-drawing. Using the OpenGl emulation of the tooltip will render the scene only when the mose movement stopped for at least 1/2 of the second.

To control the OpenGl tooltip emulation appearance the use can use the following command:

```
>>editmaterial Tooltip
```

Rotate a group of atoms around a bond

1) Select the group you want to rotate, including the bond you wish to rotate the group around.

2) Type 'mode fit'. The selected group now has changed colour a bit.

3) Right-click on the bond around which you want to rotate once (this 'activates' this bond)

4) Left-click on the bond, then move the mouse while holding down the left mouse button.

You also may wish to experiment with activating (right-click) other atoms, and holding down the SHIFT and CTRL keys while moving the mouse in the last step.

Colour fragments uniformly

You can make a selection by one of the earlier described ways, then you can type ‘group’ in the command line. This will transform the selection into a group, rendered with one colour. The group colour can be changed in the ‘Material Properties’ dialog accessible by right clicking on the group and choosing Graphics->Draw Style from the context menu. By default the groups override the grouped object colour; however some special effects can be achieved by colour blending.

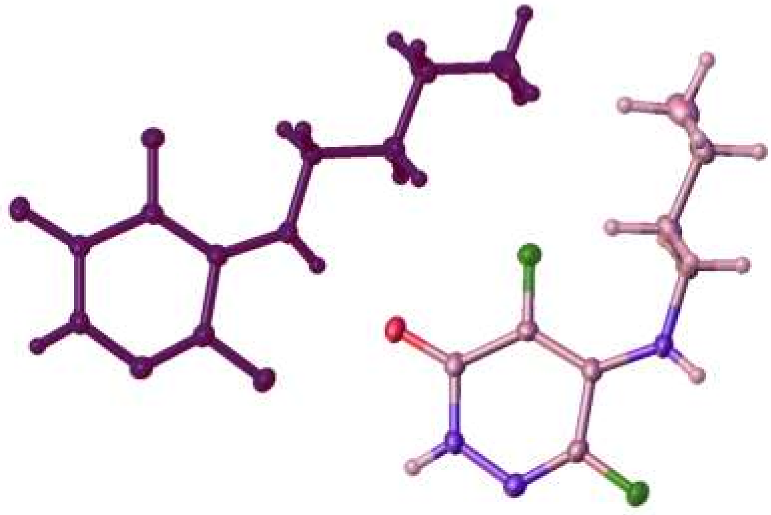
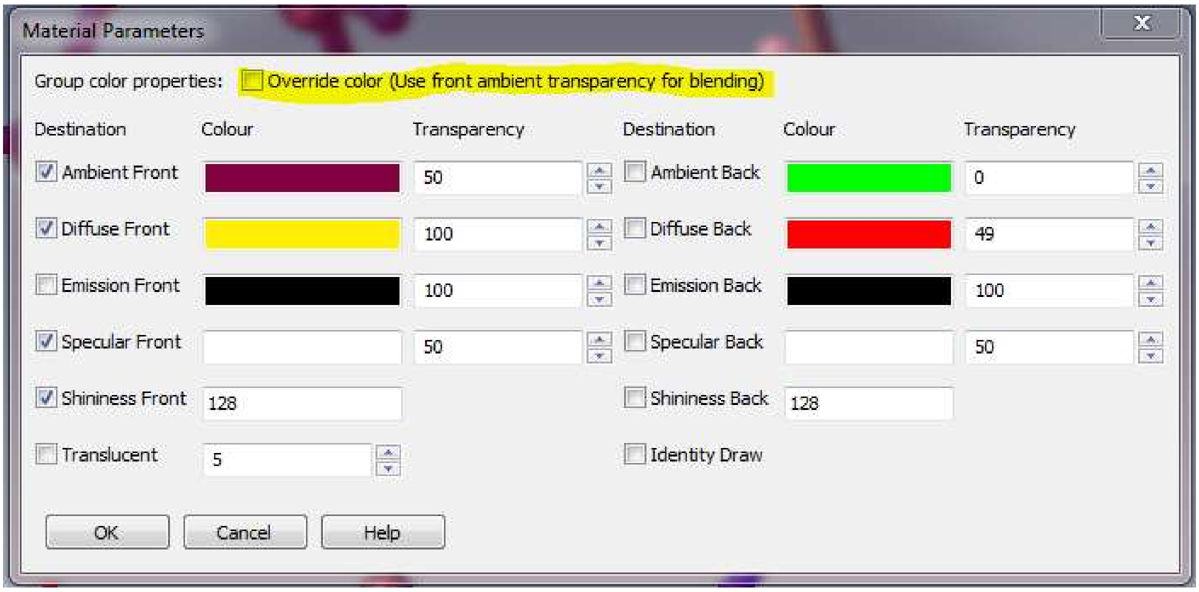


Figure 1. An illustration of the grouping in Olex2. The colour overriding is shown on the left, the colour blending is shown on the right.



Material Parameters

Group color properties: □ Override color (Use front ambient transparency for blending)

Destination	Colour	Transparency	Destination	Colour	Transparency
☑ Ambient Front	🟫	50	□ Ambient Back	🟩	0
☑ Diffuse Front	🟨	100	□ Diffuse Back	🟥	49
□ Emission Front	⬛	100	□ Emission Back	⬛	100
☑ Specular Front	⬜	50	□ Specular Back	⬜	50
☑ Shininess Front	128		□ Shininess Back	128	
□ Translucent	5		□ Identity Draw		

Figure 2. View of the Material Properties dialog for groups.

To enable the colour blending got to the Material properties dialog and un-tick the ‘Override color’ tick-box. After that the colour of the underlying objects will be mixed with the Ambient Front colour of the group, to change the proportion of the group’s colour, change the Transparency of the Ambient Front Colour.

Customise GUI toolbar

You can add your own buttons to the main GUI toolbar there the buttons like EditIns, EditAtom and Text are located. For this purpose first of all locate your configuration directory by typing:

```
>>shell app.ConfigDir()
```

This directory may be located in different places depending on your operating system and configuration. By default it is the same as the DataDir() . Then you will need to create a folder 'gui' inside and a file 'toolbar.htm' in the 'gui' folder. Then you can place the following HTML code into that file:

```
<table border='1'>

<tr>
<td><a href='fuse'>Fuse</a></td>
<td><a href='fmol'>fmol</a></td>
<td><a href="rota 1 0 0 90 3">X</a></td>
</tr>

<tr>
<td><a href='cell'>Cell</a></td>
<td><a href='basis'>Basis</a></td>
<td><a href="rota 0 1 0 90 3">Y</a></td>
</tr>

</table>
```

Then you can update the GUI panel by expanding or collapsing any of the items or by typing the following into the Olex2 command line:

```
>>html.Update
```

Customising Olex2 Shortcuts and main menu bar

Olex2 keyboard shortcuts and main menu can be easily customised and extended. For this you need to open the application configuration directory as above and create custrom_settings.xld file. This file has two sections – shortcuts and menus:

```
<shortcuts
  <shortcut key="Ctrl+Shift+C" macro="exec -o cmd">
>
<menus
    <item title="Help;My item" macro="echo getCompilationInfo(full) -c">
    <item title="My menu;Exit" macro="exit" before="Help">
>
```

Note that to control your menu position in the menubar, you use the 'before' keyword.

Work with idealised and regularised groups and AFIX instructions

There are several commands in Olex2/shelx which allow creating and managing the idealised or regularised groups in Olex2. These groups are typically used when there is a disorder in place or when the group is not behaving due to insufficient or weak data.

There are two ways to use regular group in the refinement: one would be to use restraints and the other - to use pre-computed or average geometries (idealised) which occur in other structures. In the case when there is not enough data available, the use of latter approach is beneficial since it reduces the number of parameters to be refined. There are 6 parameters normally refined for an idealised group - the position (x, y, z) and three angles (AFIX 6). In some cases the number of parameters can be increased to 7 - when the size of the groups is refined as well (AFIX 9). In other cases further constraints can be used and the number of refinable parameters reduced to 4, like in the case of -C-CF_3 group, where the position of the CF_3 group is refined, but only one angle - the rotation around the C-C bond is refined (AFIX 7); in this case the number of parameters can be extended to 5, when the C-F bond lengths are also refined (AFIX 8). In some case only 3 positional parameter can be refined (AFIX 3) which can be used with -X-CH_3 groups in staggered positions and when refining the C-H bond lengths (AFIX 4) the number of parameters increases to 4.

It worth noting that shelx AFIX number consists of two parts - the first one (m, unless 0) is responsible for the re-idealisation of the group and the other one (n) is responsible for how the group is refined. The re-idealisation of the groups happens before every refinement cycle and is required due to the precision being lost during the file output and input operations. Thus the idealised groups which are refined without the re-idealisation tend to get slightly distorted after a series of refinement procedures.

The idealised groups are extensively exploited when refining hydrogen atoms and typically there are no issues (unless when the atom connectivity changes during the refinement) occur. The use of more complex groups, like Ph or Cp requires more attention. One of the potential pitfalls are the intersecting idealised groups - when this happens, the refinement cannot proceed. To avoid this situation, each idealised groups should have its own set of atoms which can be generated in Olex2 using various splitting modes and/or 'split' function.

Here is a list of commands and modes useful for this particular section:

Afix, *HAdd*, *ImportFrag*, *ExportFrag*, *Split, mode fit -s*, *mode split*, *split*, *Fvar*, *Part*. Most of these command are available from the GUI. These commands are suitable for placing restraints to regularise the groups: *Sadi*, *RRings*, *ImportFrag*.

Enable hardware 3D and antialiasing

Since the introduction of antialiasing in Olex2, which became possible in 2.9 and later versions of wxWidgets, where is a change in how antialiasing and hardware 3D (quad buffers) can be enabled, since absence of quad buffers and antialiasing at the same time causes issues with some graphics drivers. If you are sure that you graphics card/drives support quad buffers, to enable it, type '**options**' in the command line and put this line into the opened file:

gl_stereo=true

and restart Olex2. To enable antialiasing, which provides a smoother graphical output but makes the rendering slower (not always noticeable for small models), use the following option:

gl_multisample=true

Modifying atomic radii

There is a way to modify atomic radii used for visualisation and calculations in Olex2. Some procedures (like CalcVoid) can take a file which contains element and it's radius a line, however if you work in an area where atomic radii differ from the ones Olex2 uses, you can modify them permanently by creating a file called radii.xld in DataDir(). The file's structure should be like this:

```
<elements
 <Sr bonding="2.15" vdw=”2” sfil=”1.65” pers=”0.3”>
 <Nd …>
>
```

You need to specify only the elements and radii you want to override. This file will be loaded on Olex2 startup. If you want to modify radii only for one section, you can use similar file and call '*load radii file_name*' or if you have a file with element and it's radius a line, you can specify what radii to load like '*load radii bonding file_name*'.

Customising molecular display

Olex2 provides a number of tools to customise the graphical display for picture production. Some of the customisation is described above the Colour fragments uniformly section. All graphical objects in Olex2 belong to groups. The default atom groups are groups by element type, the next, more detailed group is by atom name and the final group, which Olex2 can create automatically also include the symmetry generating the atom, thus containing a single, unique atom. It is very similar with bonds. Currently existing groups and their usage can be listed using 'lstgo' command. Opening a different file or using 'default' command clears the groups' customisation. Typical use of the groups by atom name is for packing – customisation to atoms or bonds done in the asymmetric unit will be also included into any packing of the molecules.

Any of the Olex2 graphical objects consists of 'primitives'. These can be added or removed to the object thus changing its appearance. Primitives are a part of the group into which the atoms or bond belong and therefore undergo the same convention as the material group's properties. There are a few commands which allow operating on groups, their primitives and primitive material:

- Mask - changes the primitives available in the group. The command can be applied to a selection or a named group. For example 'mask C-C 0' will make C-C bonds invisible; 'mask C-C 256' will render C-C bonds a stippled cone. The numeric value is a bitmask for the primitives to enable.

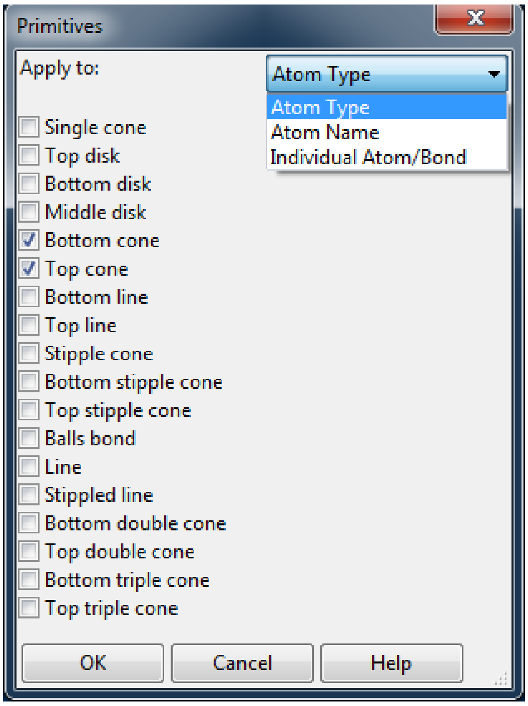
- GetMaterial – returns material properties of a named primitive. For example 'echo GetMaterial(N.Sphere)

- SetMaterial – sets material of a named primitive to the given material string value. For example it can be used to copy material of one primitive onto the other like “SetMaterial ‘destination primitive name’ ‘material string’” where ‘material string’ can be a call to GetMaterial

- Individualise – creates a more specialised group

- Collectivise – joins a specialised group with the parent group (reverts effects of the ‘individualise’). Note that if no argument provided (and the selection is clear) all previous group operations are undone an only default groups remain (same as ‘clear style’).

There are two dialogs available to control groups, their primitives and primitive material. The dialogs are accessible through the object context menu ‘Graphics->Draw style’ and ‘Graphics->Primitives’. The Bond and Atom primitives dialogs are shown in Figure 3. View of the bond primitives. and Figure 4. View of the atom primitives.

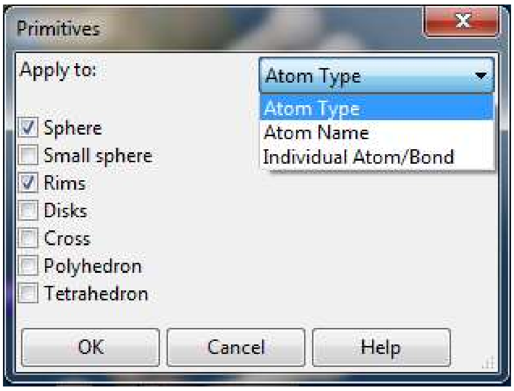


Primitives

Apply to: Atom Type

□ Single cone
□ Top disk
□ Bottom disk
□ Middle disk
☑ Bottom cone
☑ Top cone
□ Bottom line
□ Top line
□ Stipple cone
□ Bottom stipple cone
□ Top stipple cone
□ Balls bond
□ Line
□ Stippled line
□ Bottom double cone
□ Top double cone
□ Bottom triple cone
□ Top triple cone

Figure 3. View of the bond primitives.



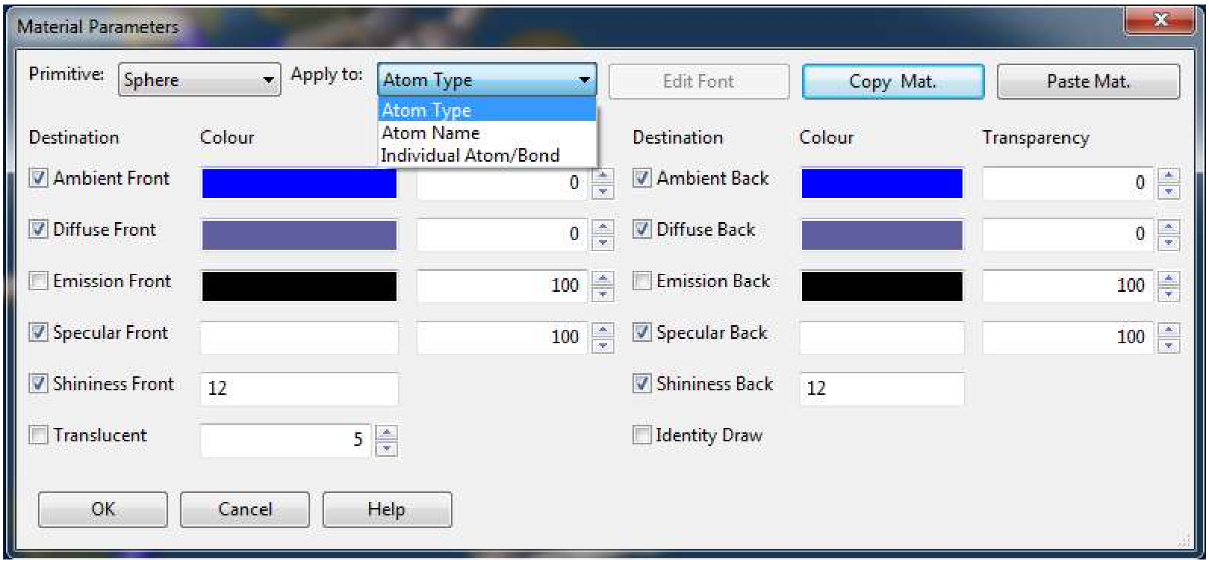
Primitives

Apply to: Atom Type

☑ Sphere
□ Small sphere
☑ Rims
□ Disks
□ Cross
□ Polyhedron
□ Tetrahedron

Figure 4. View of the atom primitives.

The ‘Material properties’ dialog is shown in Figure 5. View of the atom material properties dialog.



Primitive: Sphere

Apply to: Atom Type

Destination	Colour		Destination	Colour	Transparency
☑ Ambient Front	🟦	0	☑ Ambient Back	🟦	0
☑ Diffuse Front	🟪	0	☑ Diffuse Back	🟪	0
☐ Emission Front	⬛	100	☐ Emission Back	⬛	100
☑ Specular Front	⬜	100	☑ Specular Back	⬜	100
☑ Shininess Front	12		☑ Shininess Back	12	
☐ Translucent	5		☐ Identity Draw		

Figure 5. View of the atom material properties dialog.

Appendix

About Versions and Tags

The Olex2 distribution system has undergone many changes since the project was started in 2004. We have always aimed at providing program updates as soon as possible to the Olex2 user community. We think that one of the best ways to encourage bug reports and suggestions is to translate this user feedback as soon as possible into real improvements in the software.

For a while - up to about December 2009 - we have made updates available on a very frequent basis. This has met with a warm welcome from many of our users, but has also caused some problems: Not all updates did *only* do what they were supposed to do! At that point, we have decided to change the policy somewhat, and have come up with the following system for the distribution of Olex2.

There are now distinct versions of Olex2. Before Version 1.0, everything consisted of continuously updated files. At some point, this became no longer supportable, and we decided to introduce proper versions into the Olex2 distribution system. Any new version requires a complete re-install. However different versions of Olex2 can exist next to each other without causing any interference. For each version of Olex2, there are three 'tags', referring to different source repositories. For example, for Version 1.1 there are the following tags:

1.2-alpha

1.2-beta

1.2

Alpha: Whenever we made some changes, we 'make' an *alpha* distribution of Olex2. We use this version for in-house testing (although you are very welcome to use this version too, as long as you are aware of the fact that this version is typically very experimental and will very likely cause some problems. However, if you have suggested a new feature, or reported a bug fix, you may well find that we have implemented your suggestions already!

Beta: Once we've done some testing of this alpha version, we 'promote' it to the *beta* distribution. This version is tested by a wider group of testers - these tend to be those users with whom we have a lot of contact.

Release: Once a distribution has been tested in the *beta* stage, a proper release is made. This can be expected to be stable and if you encounter any problems with release version, please tell us about this! It doesn't matter how small the problem is, we'd like to know.

Installing Olex2

Windows:

Please download the Olex2 installer from the Windows tab and run it. Select the destination folder to which to install Olex2 (typically **C:\Program Files\Olex2**). If you do not have administrator privileges*, please select a folder where you have full access rights.

Make sure you select the latest version of Olex2—Version 1.2—from the download repository.

Click on **Install.** This will install Olex2 on your computer. When it is done, there will be a 'Run' button on the installer form. Click this to run Olex2. The first time Olex2 runs on your computer, it will take some time to start up (up to one minute!). **Unless you used 'Run' in the installer, you should run Olex2 as an administrator for the first time.**

Olex2 should now be opened, there should be no red (error) lines in the main window and there should be a molecule of sucrose displayed on the screen. Olex2 does not require any third party programs to perform structure analyses—Structure Solution as well as Structure Refinement—but, if you have a ShelX installed on your system, you may want to make sure that Olex2 interacts with the ShelXS, ShelXT, ShelXL and ShelXM. If you do not have a ShelX, and would like to obtain one, please go to the ShelX Pages for more information.

Please note that the ShelX executables that are shipped with WinGX do not work with Olex2.

You can either copy your ShelX executables into the Olex2 installation folder, or—better—you can copy your executables into a folder which you then add to the PATH variable of Windows. For example, create a folder **C:\Program Files\Shelx**, then Right-Click on 'My Computer' (XP) or 'Computer' (Vista and 7) and select *Properties*. Then select *Advanced.* There you can add the location of your ShelX executables to the PATH variable.

Linux

Olex2 for 32 and 64 bit Linux is provided in ZIP archives which contains commonly required dependencies. Simply unzip it to a desired location and use the 'start' script located in the Olex2 folder.

Mac

The release versions of Olex2 are provided as DMG images. Other versions are available as ZIP archives. Note that when unzipping Olex2 with Finder, the latter removes the executable flags from files in the archive and these have to be restored in the terminal using chmod command: simply locate olex2.app in Terminal application and use 'chmod +x olex2.app/Contents/MacOS/olex2'

External Programs

SHELX

All programs of the SHELX family can interact seamlessly with Olex2. There is no need for registering any of these programs with Olex2, it is enough if the folder containing the ShelX programs are on the system PATH. This is normally the case if ShelXTL has been installed on a system. Otherwise, you will need to set you system PATH variable to include the folder where you keep your SHELX executables.

Please note that the SHELX executables that are shipped with WinGX do not work with Olex2. These executables have been modified in such a way that they will only work properly with WinGX. Since WinGX puts the folder that contains these executables on the system PATH, you might find that SHELX appears in Olex2 - and then doesn't work. In this case, you will need to get new SHELX executables and put them in the same folder where Olex2 is installed - executables found there will be used by Olex2 preferentially.

Olex2 uses shelxl.exe name as the default ShelXL. If you have new version of ShelXL and still want to use 1997 version occasionally, you may want to rename the latter to shelxl-97.exe – it then will appear as a separate item in the list of the available refinement programs.

Platon

John Warren has provided an interface to PLATON. If Platon is on the system PATH, you will see ‘P’ icon at the top of the GUI. Note that some manufacturers distribute Platon which cannot be used with Olex2. In this case you need to install the original Platon and make sure that it comes first on the system PATH (paths are traversed in order of appearance).

SuperFlip

Originally Arie van der Lee has provided an interface to SuperFlip; this had been superseded by tighter integration of Superflip into Olex2.

About Macros and Scripting in Olex2

Olex2 supports two different types of external scripting: Macros and Python scripts. An example of the internal script is shown in Change default programs. The Python scripts can be either executed externally or attached to the Olex2 console.

List of external packages used in Olex2

Computational Crystallography Toolbox (cctbx)

wxWidgets

Python

Pillow, the ‘friendly’ PIL fork by Alex Clark and Contributors. PIL is the Python Imaging Library by Fredrik Lundh and Contributors. © Copyright 1997-2011 by Secret Labs AB, 1995-2011 by Fredrik Lundh, 2010-2013 Alex Clark.

The Python Cryptography Toolkit (PyCrypto)

Credits

We thank people who stayed with us all these years and provided either code or feedback. Sorry if we missed you in this list!

Andrei Batsanov

John Bacsa

Maxim Borzov

Neil Brooks

Andrei Churakov

Sebastian Cirkel

Jonathan Coome

Ilia Guzei

Simon Harries

Arie Van der Lee

Sergey Lindeman

Pascal Parois

Mike Probert

Radostan Riedel

Helena Shepherd

Richard Staples

John Warren

Zier Yan

Dima Yufit