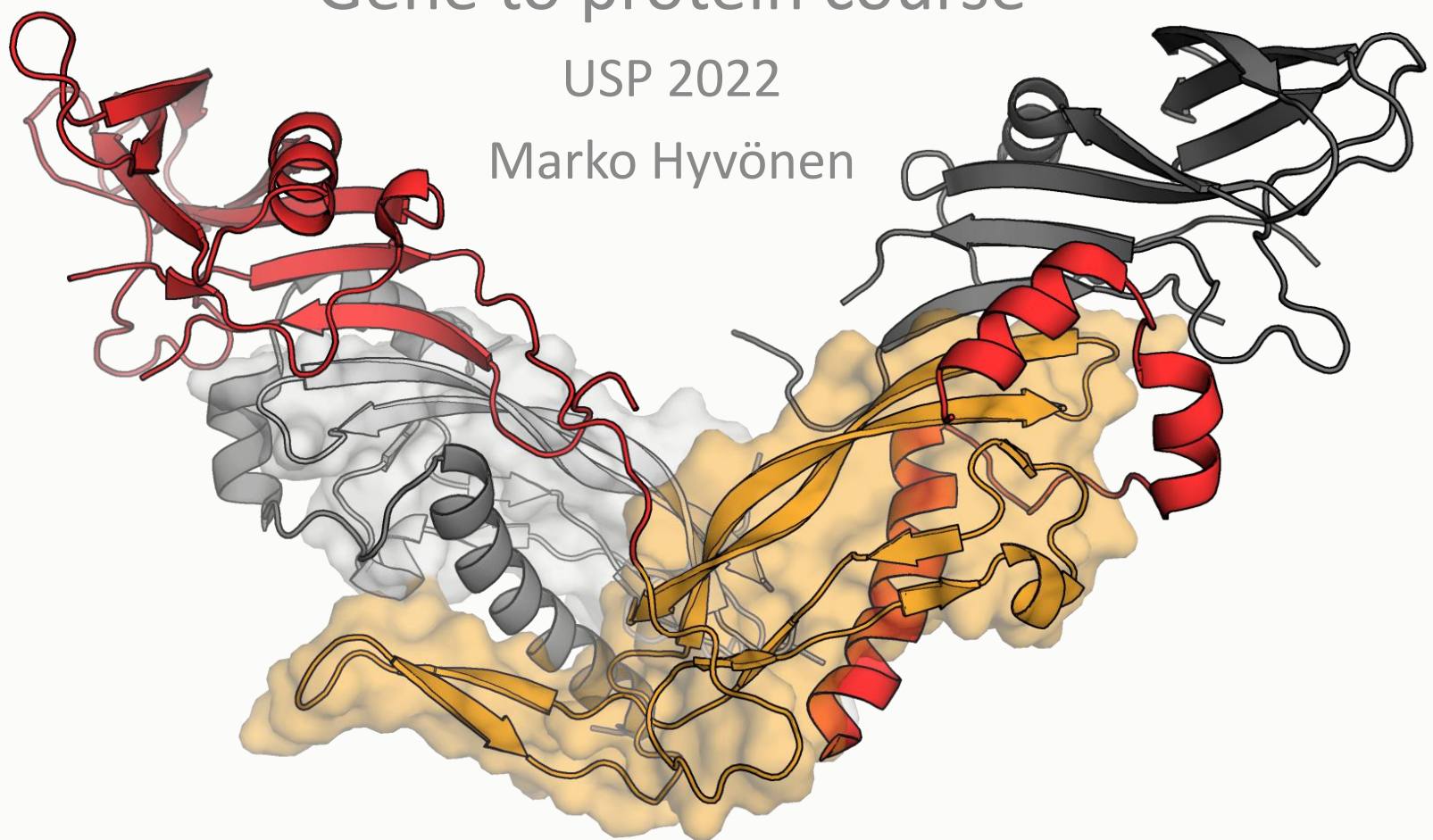


PyMOL tutorial

Gene to protein course

USP 2022

Marko Hyvönen



On these instructions

Commands from top menus are in blue

Scene > Store > F1

Commands from graphics menus are in red

Show > Cartoon

Command line instructions are in **Courier** and shaded

```
select CDK, 2wih & chain A & polymer
```

Commands are always of the syntax:

```
command first_argument, next_argument, next_argument
```

↑
comma

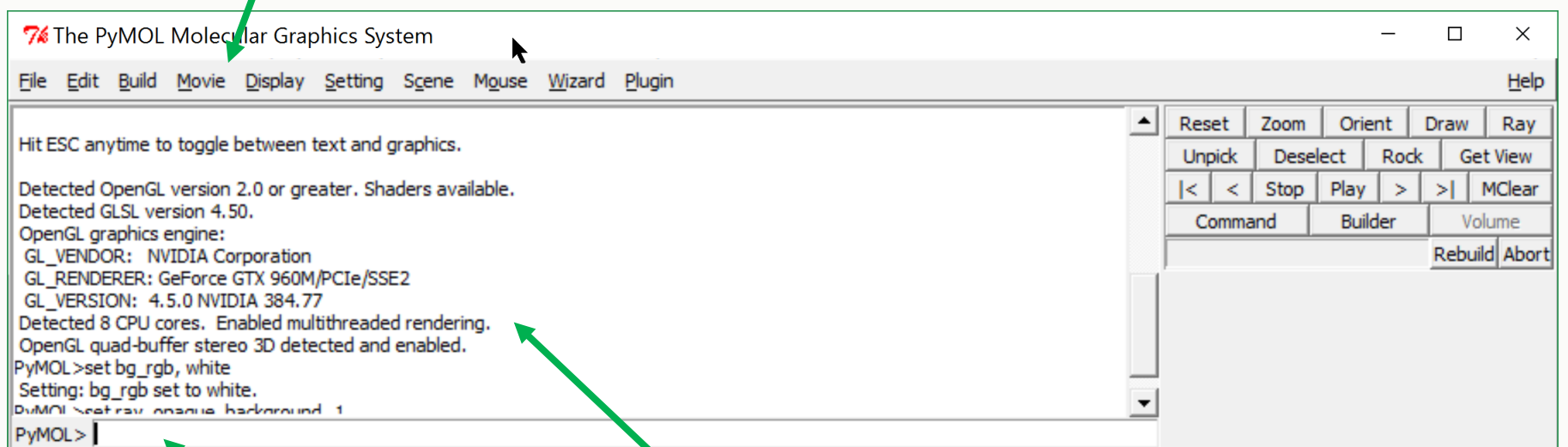
↑
comma

PyMOL tutorial aims

- learn the basics of using PyMOL
 - Retrieve coordinates
 - Familiarise with the menus
 - Use of the mouse
 - Use the command line
- Learn to select specific parts of the molecule
 - Chains, atoms, residues, ligands
- Learn to create and save scenes
- Save sessions (for later use or for sharing)

Command window

Menus for opening and saving files, changing settings etc.

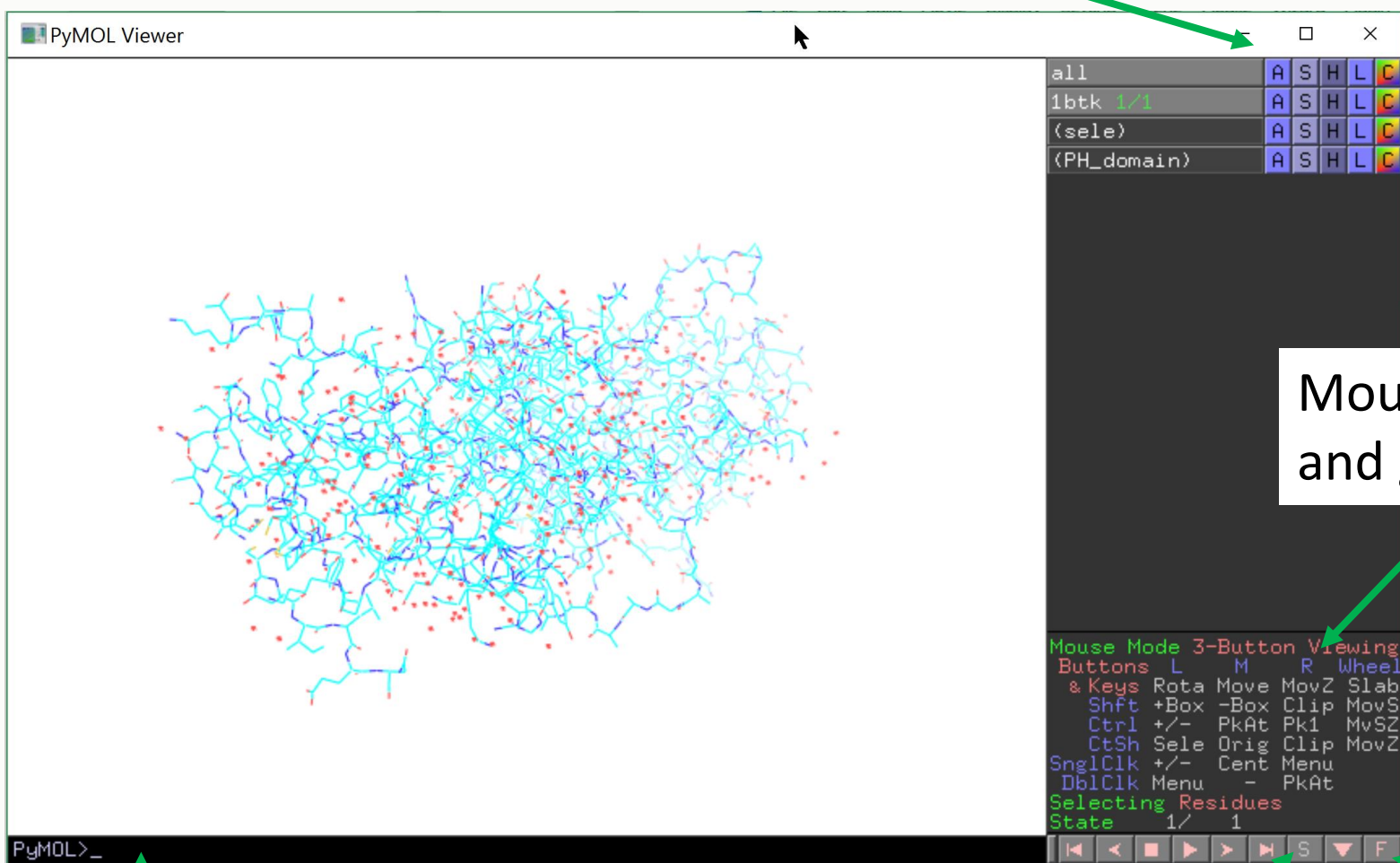


Output of any actions (great for trouble shooting)

Command line for executing commands, be they simple or complex

The Graphics Window

Object menu



Mouse mode and guide

Command line

Sequence on/off

Full screen

The object menu

Action something

Show a representation

Hide representation

Label

Colour

Light gray:
Visible/active

Dark gray:
Hidden/inactive

An object →

Current selection →

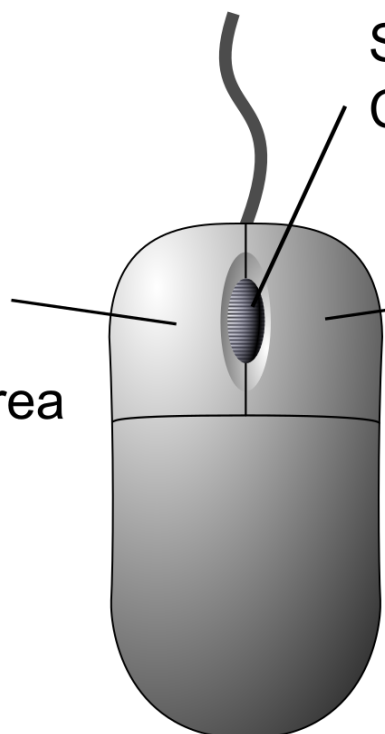
Named selection →

	A	S	H	L	C
all					
1btk 1/1					
(sele)					
(PH_domain)					

Selections are in parenthesis, e.g. (PH_domain)

Mouse control

Press: Rotate
Click: Select
Dbl Click: Menu
Shift-Press: Select area



Press: Move X-Y
Scroll: Slab (Z- thickness)
Click: Centre on atom

Press: Move Z (zoom)
Click: Menu

On canvas:

```
Main Pop-Up  
new  
zoom (vis)  
orient (vis)  
center (vis)  
reset  
movie  
scene  
enable  
disable  
(all)  
(visible)  
ray  
delete all  
reinitialize  
quit
```

On an atom:

```
/2wih/A/A/GLN^287/CA  
drag object matrix  
drag object coords  
atom  
residue  
chain  
segment  
object  
molecule  
fragment  
fragment+joint(s)
```

Setting defaults

If using PyMOL frequently, you might want to have certain settings to be valid all the time.

For now, the main thing to change is the background colour from black to white:

[File > Edit pymolrc](#)

Add this line:

```
bg_color white
```

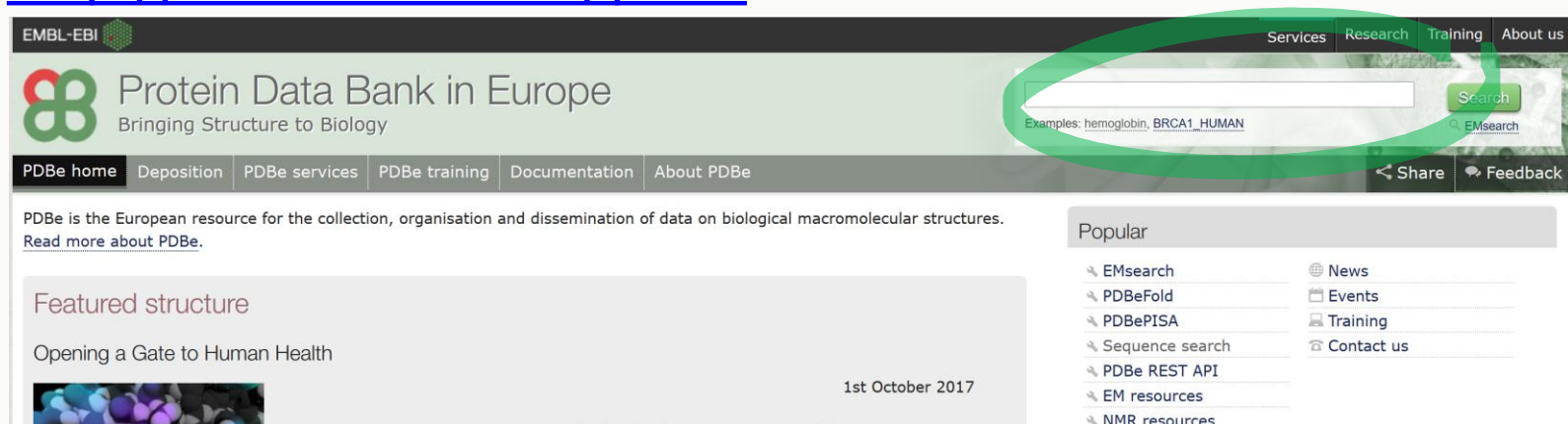
Save and close the file.

Restart PyMOL

(More useful starting settings at the very end of the slide deck)

Searching for coordinates

<http://www.ebi.ac.uk/pdbe>



EMBL-EBI

Services Research Training About us

Protein Data Bank in Europe
Bringing Structure to Biology

Search

Examples: hemoglobin, BRCA1_HUMAN

EMsearch

PDBe home Deposition PDBe services PDBe training Documentation About PDBe

Share Feedback

PDBe is the European resource for the collection, organisation and dissemination of data on biological macromolecular structures.
Read more about PDBe.

Featured structure

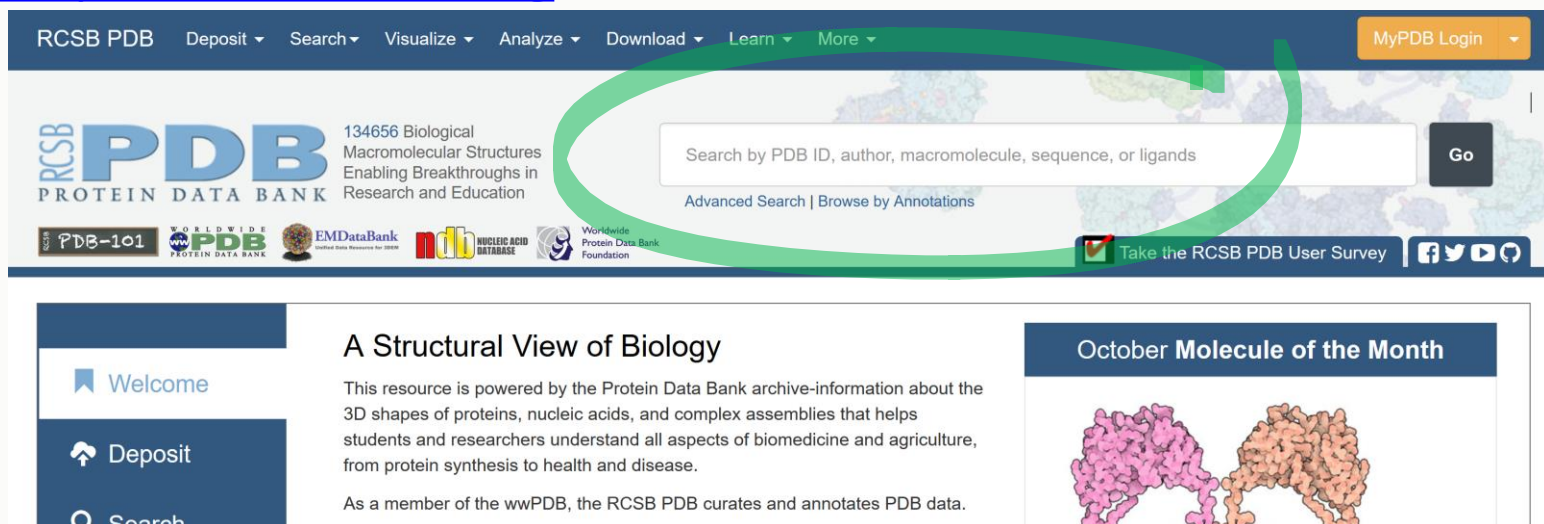
Opening a Gate to Human Health

1st October 2017

Popular

- EMsearch
- PDBeFold
- PDBePISA
- Sequence search
- PDBe REST API
- EM resources
- NMR resources
- News
- Events
- Training
- Contact us

<http://www.rcsb.org>



RCSB PDB Deposit Search Visualize Analyze Download Learn More

MyPDB Login

RCSB PDB PROTEIN DATA BANK

134656 Biological Macromolecular Structures
Enabling Breakthroughs in Research and Education

Search by PDB ID, author, macromolecule, sequence, or ligands

Go

Advanced Search | Browse by Annotations

Take the RCSB PDB User Survey

Welcome

Deposit


Search

A Structural View of Biology

This resource is powered by the Protein Data Bank archive-information about the 3D shapes of proteins, nucleic acids, and complex assemblies that helps students and researchers understand all aspects of biomedicine and agriculture, from protein synthesis to health and disease.

As a member of the wwPDB, the RCSB PDB curates and annotates PDB data.

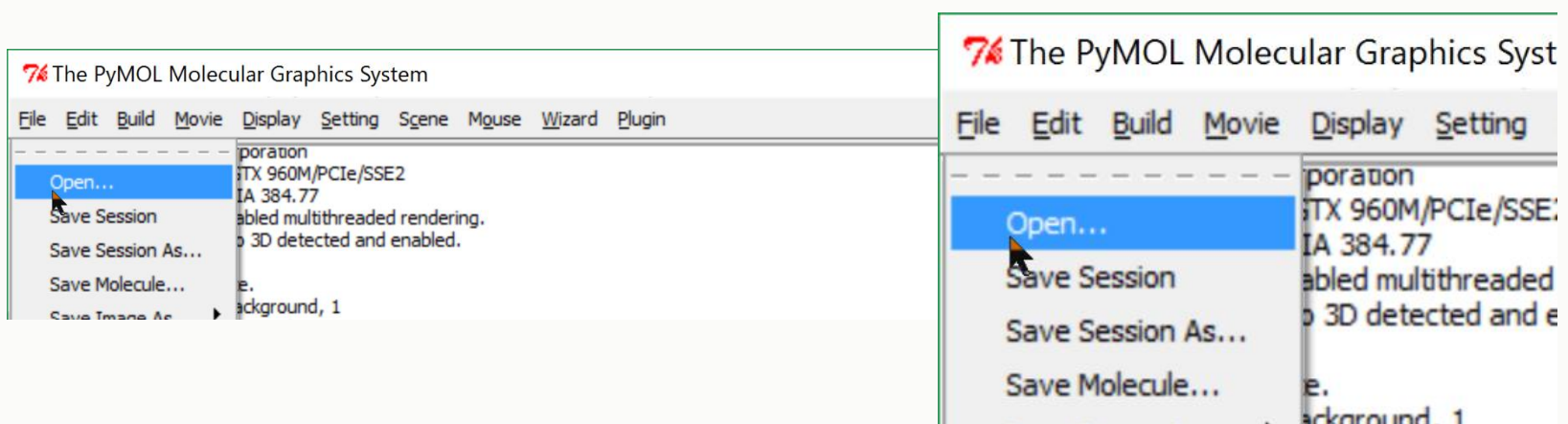
October Molecule of the Month



Opening coordinates

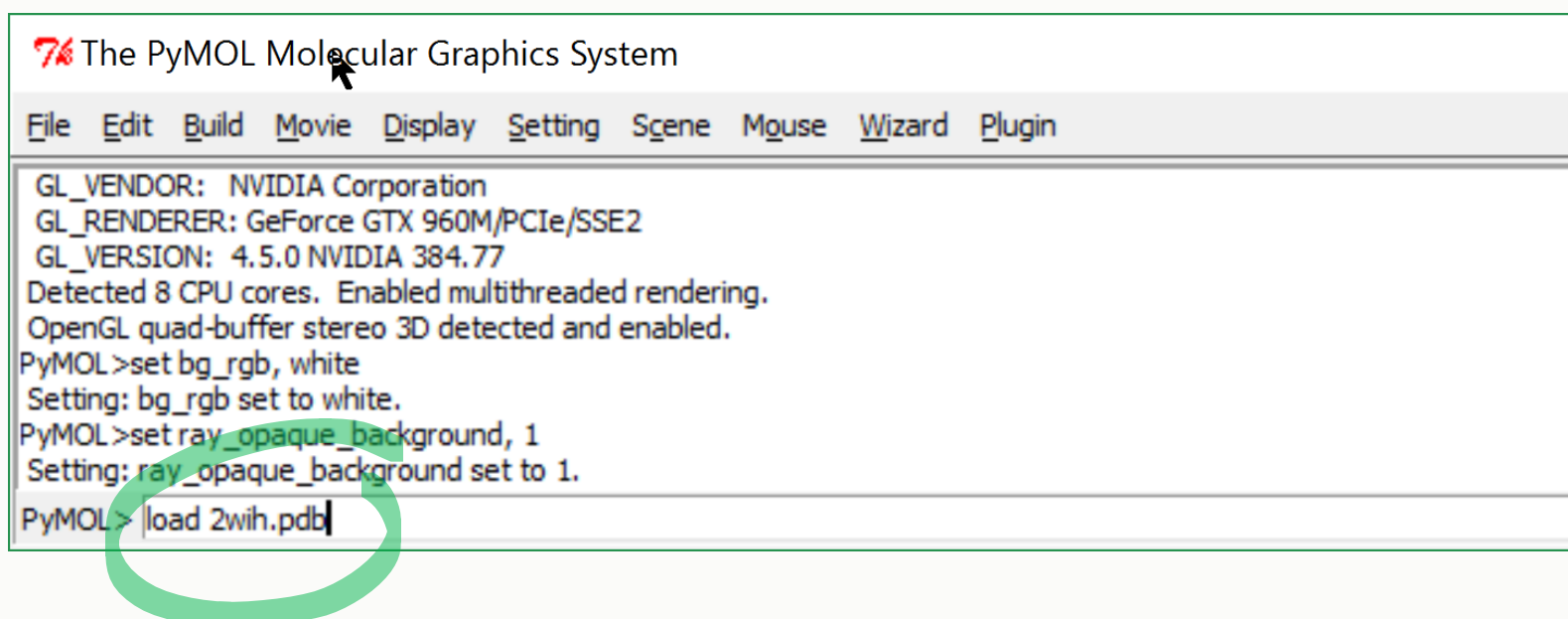
- There are at least four different ways
 - Click on a coordinate file
(assumes you have associated PDB files with PyMOL)
 - Open a file that is in your computer
 - Open coordinates from PDB using a plugin
 - Fetch the coordinates from PDB with a command

Open from your computer



This loads the coordinates (2wih.pdb) into object called 2wih
File > Open...

PDB loading from the command line



```
load 2wih.pdb
```

```
load C:\Users\Marko\Desktop\tutorial\2wih.pdb
```

```
load /Users/Marko/Desktop/tutorial/2wih.pdb
```

```
load /home/Marko/Desktop/tutorial/2wih.pdb
```

Directly from Protein Data Bank

```
fetch 2wih
```

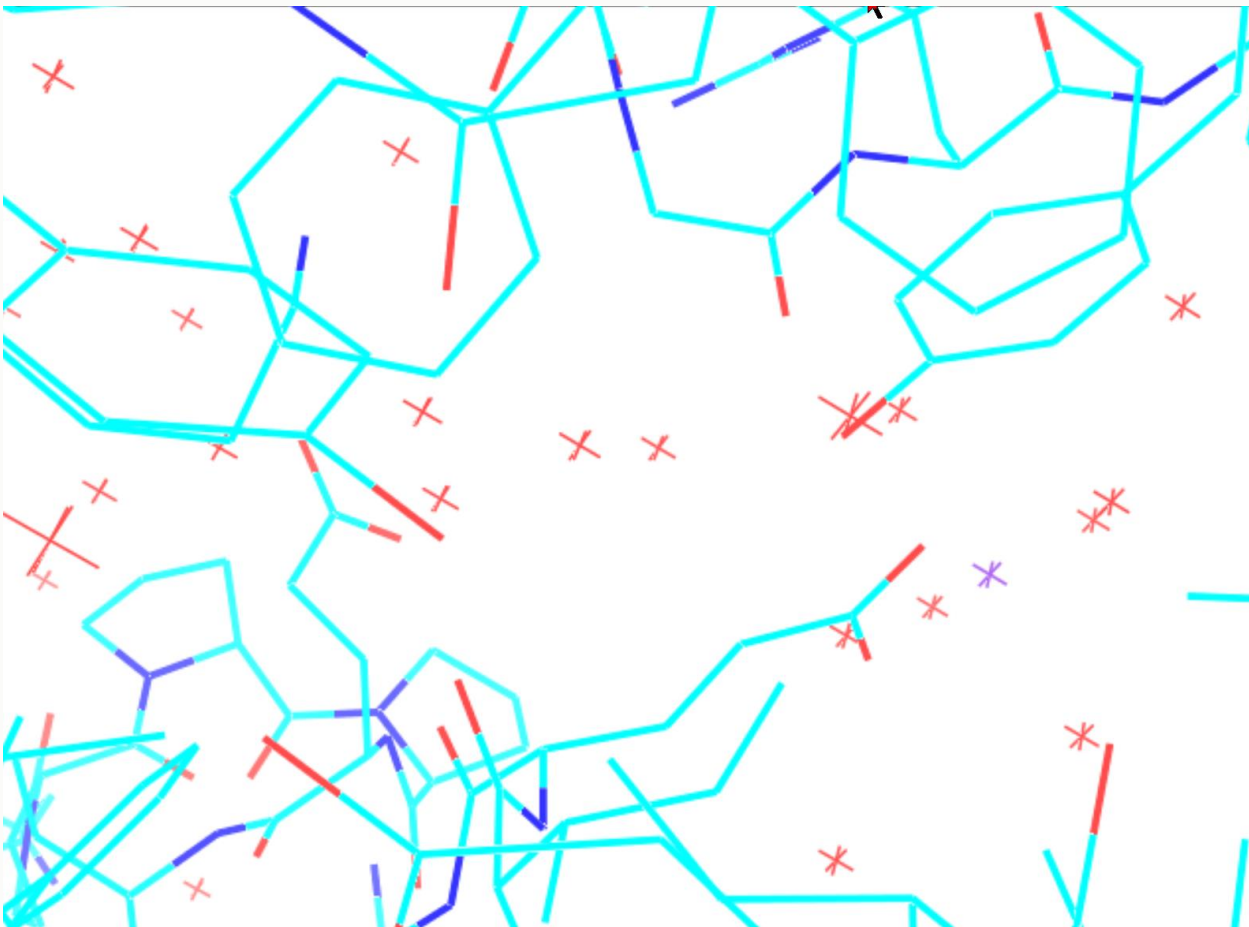
```
fetch 2wih, async=0 (sometime needed)
```

Basic display of coordinates

Default view is lines for protein and nucleic acids and crosses for non-bonded atoms (waters, ions).

Colour for carbons changes for each molecule, the other atoms are coloured by the element

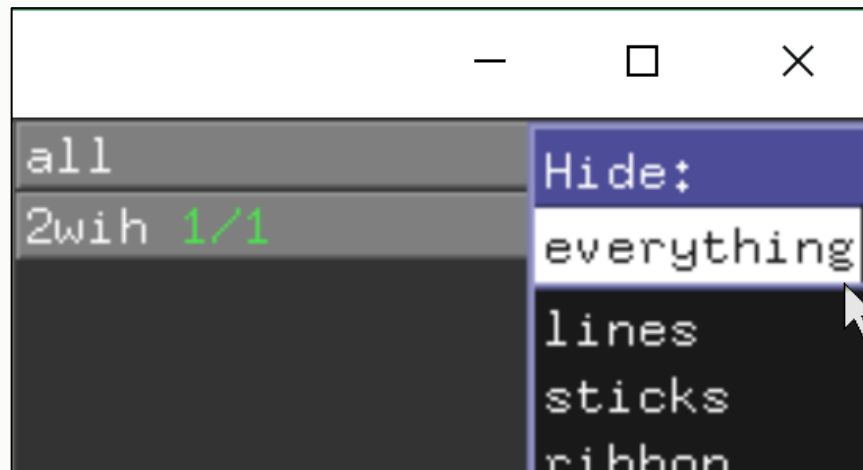
N = blue, O = red, S = yellow, P = orange, H = white



Changing the view

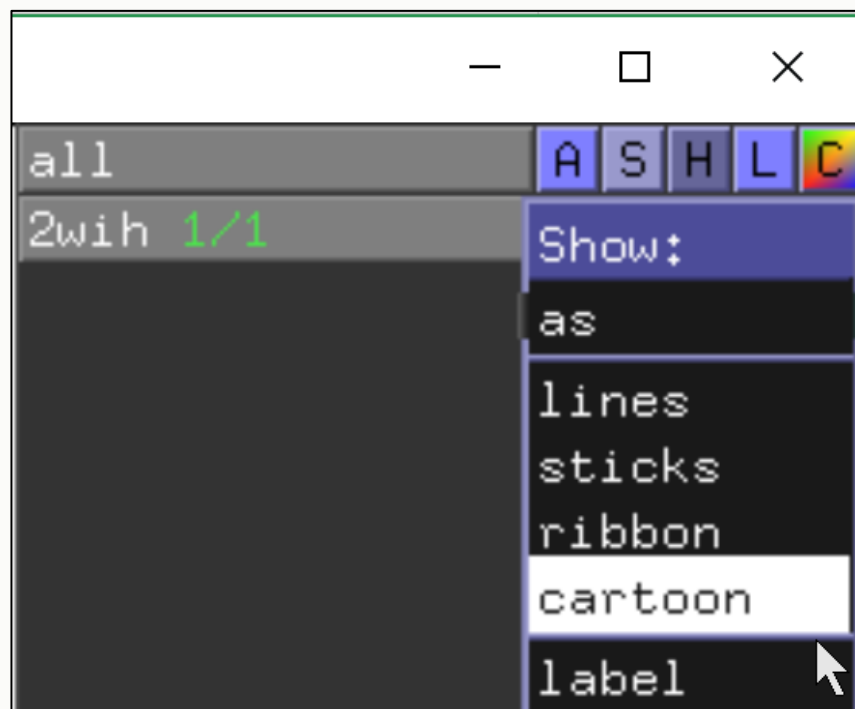
First: hide everything

`all > Hide > everything`



Then, show 2wih as cartoon

`2wih > Show > cartoon`

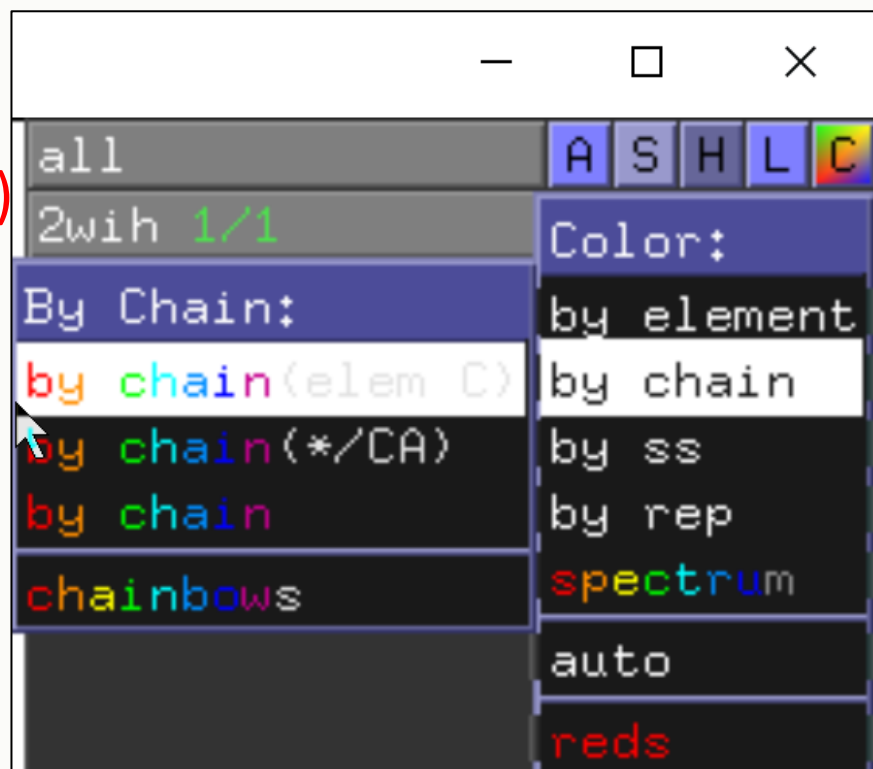


Colour by chain (there are 4 in 2wih):

`2wih > Color > by chain > by chain(elem C)`

Using the command line:

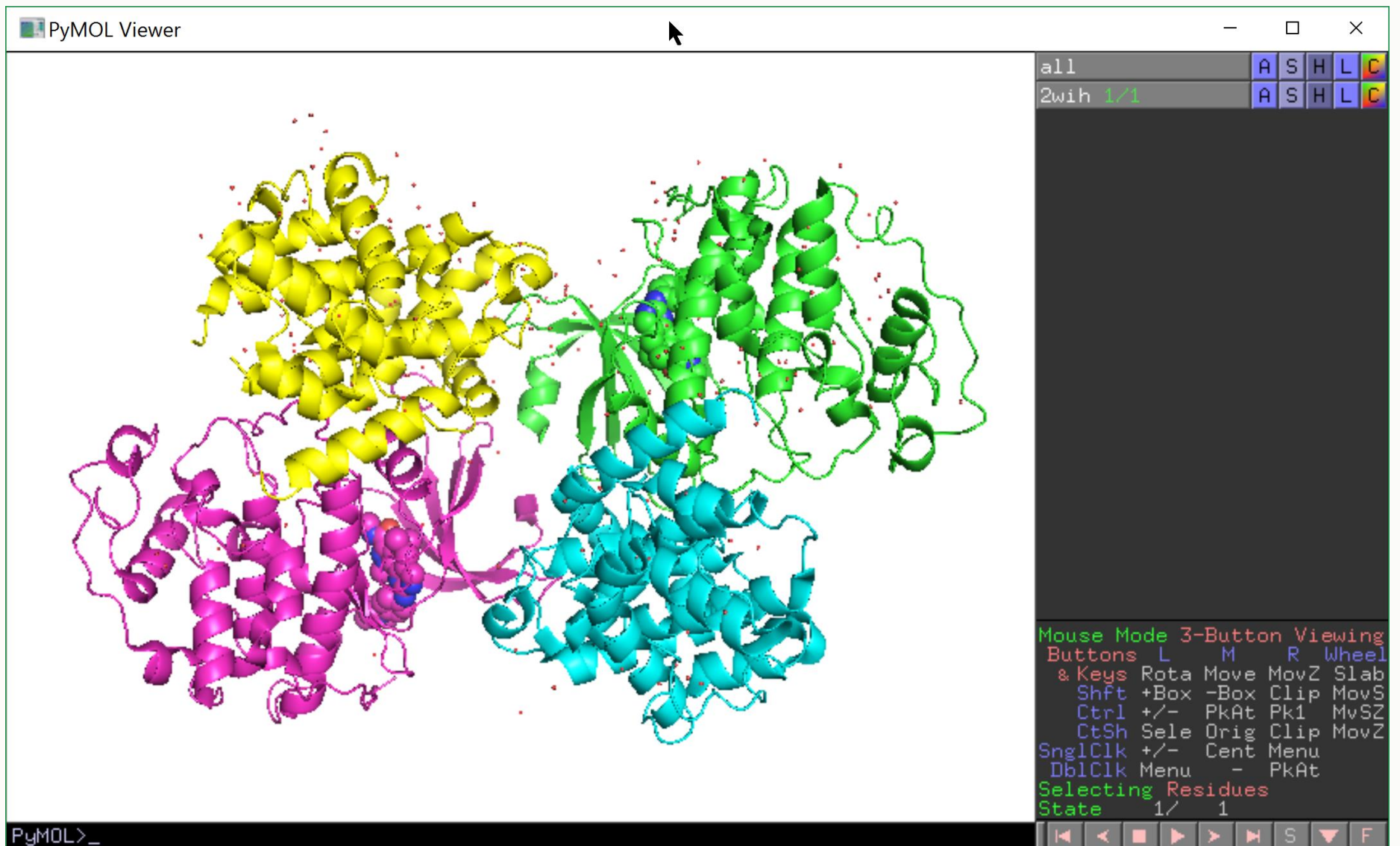
```
hide everything, all
show cartoon, 2wih
colour green, chain A & name C*
colour cyan, chain B & name C*
colour yellow, chain C & name C*
colour magenta, chain D & name C*
```



Getting serious...

2wih > Show > nb_spheres (waters and ions as small spheres)

2wih > Show > organic > spheres (ligands as CPK spheres)



Hierarchical structure of macromolecular coordinates

ATOM: protein/DNA/RNA

HETATM: ligands

	ATOM no.					X coord	Y coord	Z coord	Occupancy		B-factor	Element
ATOM	330	N	ALA	A	141	33.444	27.094	8.380	1.00	8.10		N
ATOM	331	CA	ALA	A	141	33.397	28.479	8.840	1.00	7.81		C
....												
ATOM	1666	N	THR	B	520	23.744	48.961	3.142	1.00	35.07		N
ATOM	1667	CA	THR	B	520	24.098	47.838	4.013	1.00	33.12		C
ATOM	1668	C	THR	B	520	23.952	46.477	3.357	1.00	31.94		C
ATOM	1669	O	THR	B	520	24.698	45.547	3.676	1.00	31.26		O
ATOM	1670	CB	THR	B	520	23.242	47.855	5.296	1.00	33.72		C
ATOM	1671	OG1	THR	B	520	21.871	47.557	4.982	1.00	33.68		O
ATOM	1672	CG2	THR	B	520	23.307	49.243	5.942	1.00	34.95		C

The most important identifiers for selecting parts of a structure for display:

Residue number is unique to each chain

Chain is made of residues (not necessarily covalently linked)

Residues make up a chain (always three letter code)

Atoms are part of a residue (each atom in a residues has a unique name)

In PyMOL when clicking an atom:

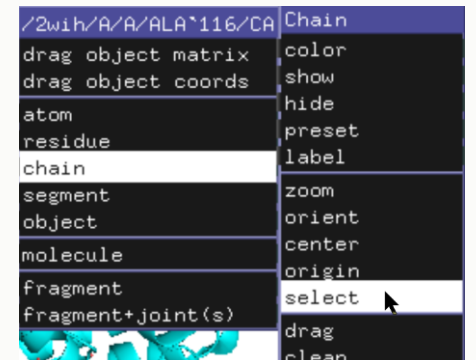
/2wih/A/A/THR`137/CA

/object/chain/conformation/resname`resno/atom

Selecting parts of the molecules

Selections can be done in many ways:

- From the graphics window
 - Left click on an atom selects are residue
 - This will also add to current active selection
 - You can change what is selected from the red “Residues” text in the bottom menu
 - Right click on an atom, follow the menu to atom/residue/chain... and “select”
- From the sequence view
 - To show the sequence, click the little S button at the very bottom



You can select residues from the sequence

- one by one
- by normal text selection method
- great also for locating selected residues in sequence.



The default selection will be called (sele)

To rename that to something you'll remember:

(sele) > Action > rename selection

Selecting parts of the molecules

From the command line

Syntax: `select selection_name, selection`

For selection you can use (among many others):

- object name: `2wih`
- `resn` for residue name: `resn ASP`
- `resi` for residue numbers: `resi 100-200`
- `chain` for chain ID: `chain A+D`
- `name` for atom name: `name C*` (all atoms starting with C : carbon, Ca, Cd,)

And combination of these, linked with “&”

You can negate the selection with a “!” (“no!”)

Example:

select all glycines in residues 1-100 in object 2wih but NOT in chain A and call that section GlyNotA

```
select GlyNotA, resn GLY & resi 1-100 & 2wih & !chain A
```

```
select CDK2, chain A
```

```
select cyclin, chain B
```

```
select inhibitor, resn p48 & chain A
```

Now you can manipulate all those selections independently:

- color
- style
- transparency...

Creating and saving different views

2wih > Hide > everything

(CKD2) > Show > cartoon

(cyclin) > Show > cartoon

Orient the structure as you like

Save this view, or scene:

Scene > store > F1

Now the scene can be recovered by pressing F1

Setting “Buttons” on:

Scene > Buttons

Will show the scene buttons on the screen

Related command line options:

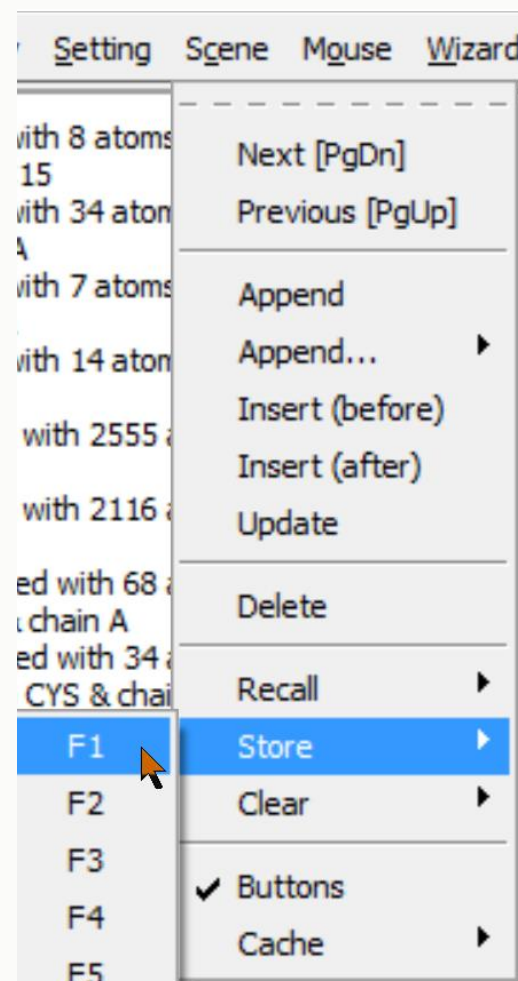
```
scene F1, store
```

```
scene F1, recall
```

```
scene F1, clear
```

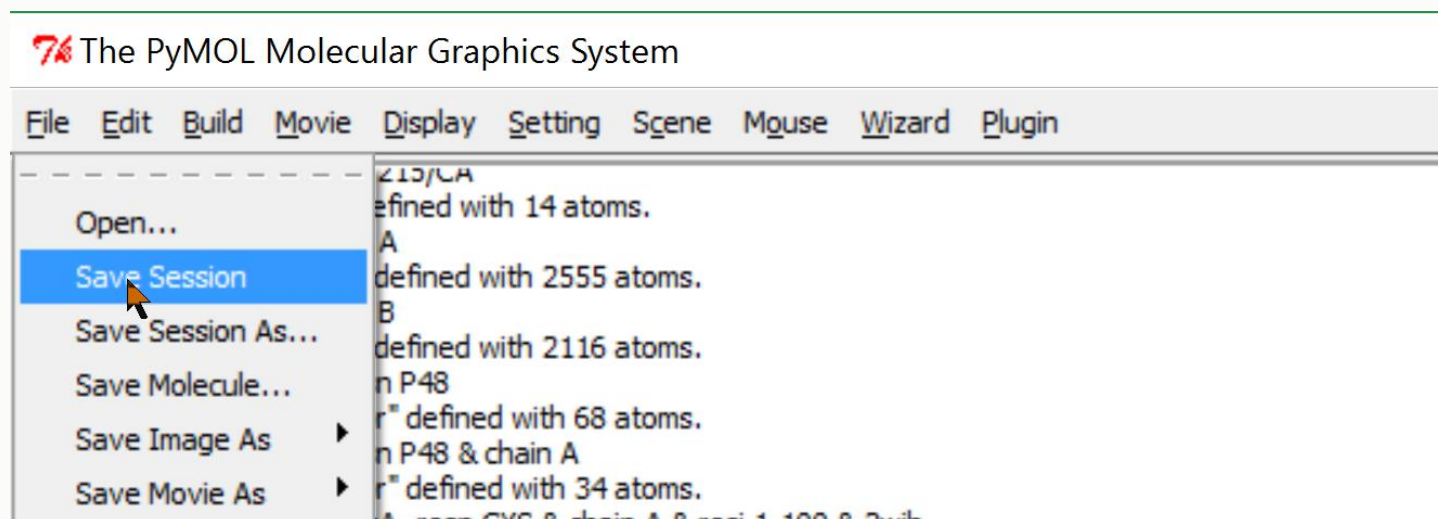
```
set scene_buttons, 1
```

```
scene CDK2_cyclin, store
```



Saving the work

- PyMOL session file (.pse) will preserve all the molecules, selections, scenes, colourings
- Keep saving regularly!



- From the command line:
save my_session.pse

Note: annoyingly, Ctrl-S does not work for this!

Colouring

Command line: `color name_of_the_color, selection`

`color forest, CDK2 & name C*`

Atom type color

The first option does NOT change carbons

Predefined colors for chains

chainbows:
each chain as rainbow

Helices, sheets and loops

All in different colour
(urgh...)

Colour by property

Rainbow: N-term blue to red C-term
B-factor: indicates mobility of the residue

Atoms	Color:
HNOS...	by element
CHNOS...	by chain
CHNOS...	by ss
CHNOS...	by rep
CHNOS...	spectrum
CHNOS...	auto
CHNOS...	reds
CHNOS...	greens
CHNOS...	blues
set 2	yellow
set 3	magenta
set 4	cyan
set 5	orange
set 6/H	tint
	gray

(inhibitor)	Color:
By Chain:	by element
by chain(elem C)	by chain
by chain(*CA)	by ss
by chain	by rep
chainbows	spectrum
	auto
	reds
	greens
	blues
	yellow
	magenta
	cyan
	orange
	tint
	gray

(inhibitor)	Color:
By Secondary Structure:	by element
Helix Sheet Loop	by chain
Helix Sheet Loop	by ss
Helix Sheet Loop	by rep
Helix Sheet Loop	spectrum
	auto
	reds
	greens
	blues
	yellow
	magenta
	cyan
	orange
	tint
	gray

(inhibitor)	Color:
	by element
	by chain
	by ss
	by rep
Spectrum:	spectrum
rainbow(elem C)	auto
rainbow(*CA)	auto
rainbow	reds
b-factors	greens
b-factors(*CA)	blues
area (molecular)	yellow
area (solvent)	magenta
	cyan
	orange
	tint
	gray

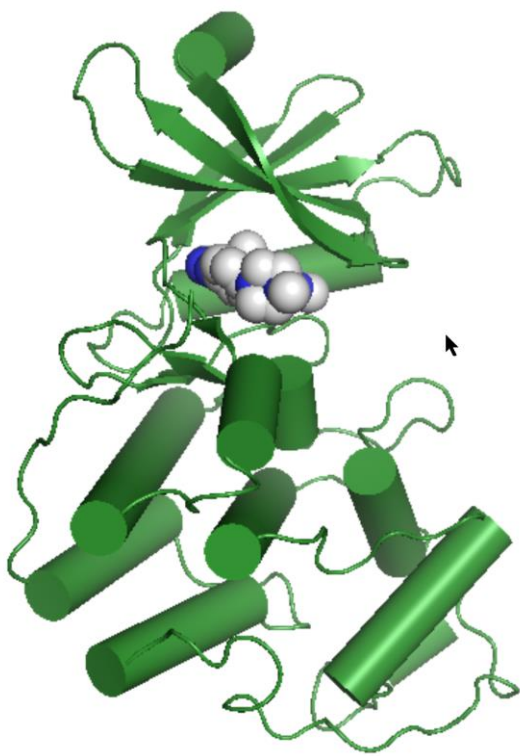
All named colours in PyMOL

https://pymolwiki.org/index.php/Color_Values

Some fancier representations

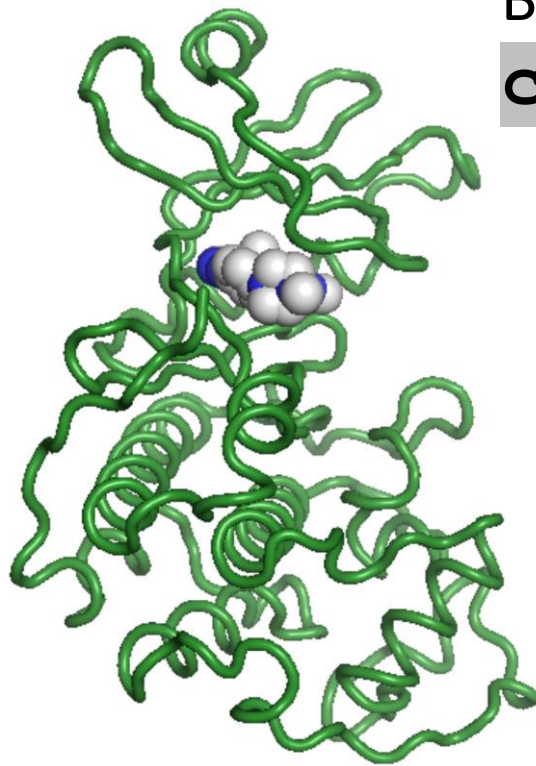
Cylindrical helices:

```
set cartoon_cylindrical_helices, 1  
("1" is a Boolean "on", "0" = "off")
```



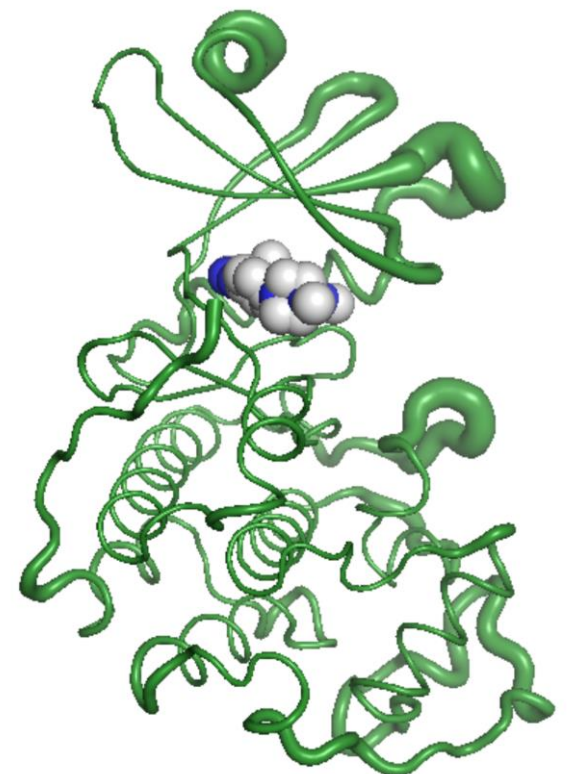
Backbone as tube:

```
cartoon tube
```



Tube with thickness from B-factor:

```
cartoon putty
```



Prefer the "normal" look? Return back to that with:

```
cartoon automatic
```

Surfaces

- Great at showing cavities, channels, binding sites and for analysis of interfaces
- PyMOL draws surface for the whole object
- An issue: surface of an object will cover all of the content of that object, not individual molecules

Solution: create an object from a chain

(CDK2) > Action > copy to object (creates obj01)

(cyclin) > Action > copy to object (creates obj02)

(you might want to rename the new objects)

obj01 > Action > Rename (give whatever name you want)

obj02 > Action > Rename (give whatever name you want)

```
set_name old_name, new_name
```

Draw the surfaces

obj01 > Show > surface

obj02 > Show > surface

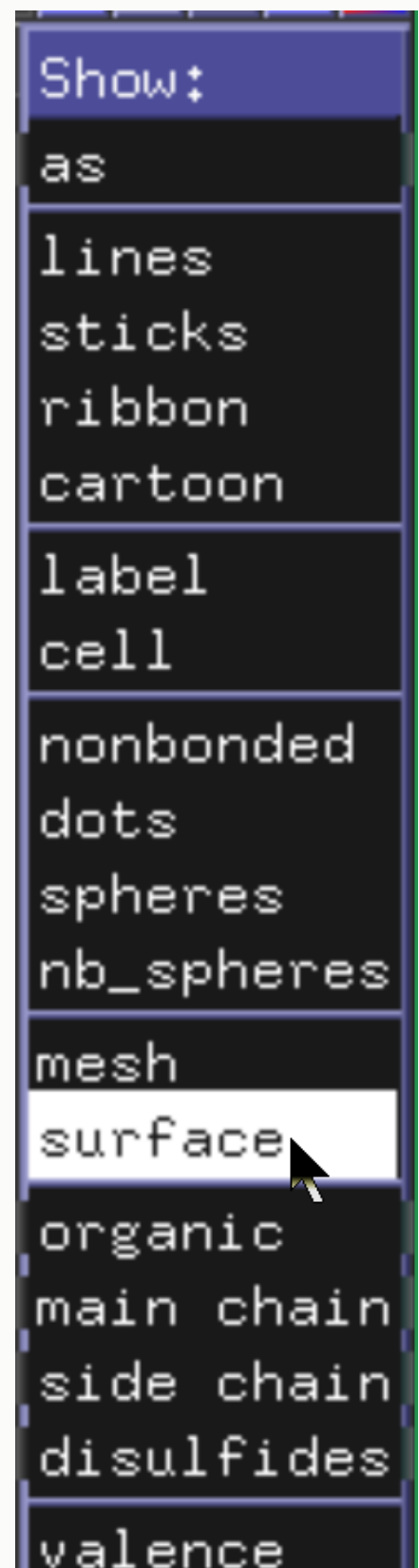
```
show surface, obj01
```

Note also that non-polymer atoms (waters, ligands...) are not covered by surface by default.

This includes also non-natural amino acids like seleno-Methionine

To include this in surface rendering you need to:

```
set surface_mode, 1
```



Surfaces & interfaces

Colour surface differently:

```
set surface_color, white, obj01
```

(to revert to atom colours, `set surface_color, default`)

Colour interface contacts between CDK2 and cyclin

```
select contact1, (obj01 within 4.0 of obj02)
```

```
select contact2, (obj02 within 4.0 of obj01)
```

(these select atoms in each molecules within 4.0 A from the other one)

```
set surface_color, limegreen, contact1
```

```
set surface_color, skyblue, contact2
```

Show water molecules between the two proteins

```
select interface_waters, (2wih within 4 of obj01)  
& (2wih within 4 of obj02) & resn HOH
```

Like that? Best save it:

```
scene interfaces, store
```

More complicated structures: handling symmetry

```
fetch 3ry2, biotinSA
```

(load coordinates and call them biotinSA)

```
hide everything, all
```

(hide all the objects)

```
show cartoon, biotinSA
```

Two protein molecules, but streptavidin is a tetramer. The missing half must come from crystallographic symmetry

Let's generate this by generating symmetry related molecules of 3ry2

```
symexp sym, biotinSA, biotinSA, 4
```

Or use the object menu:

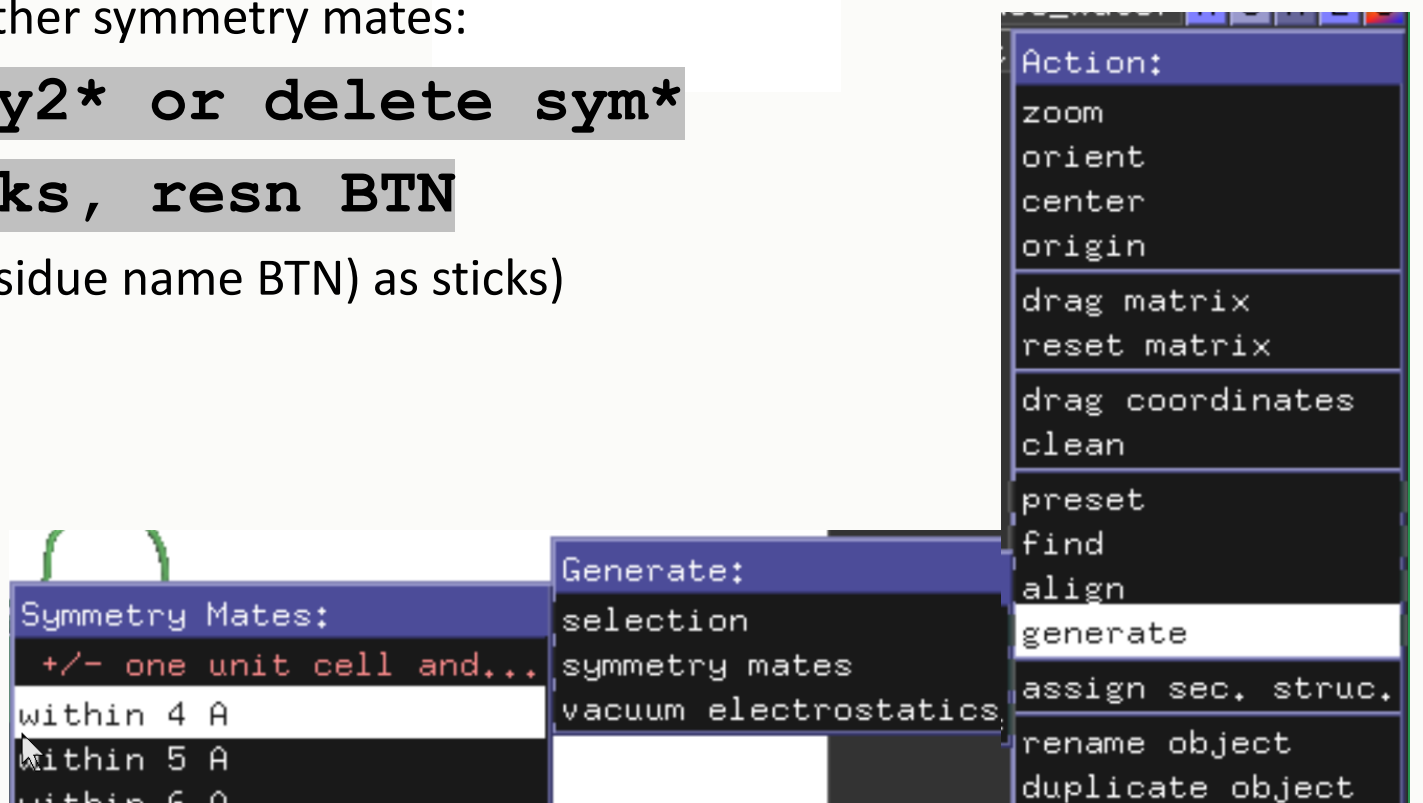
Action > generate > symmetry mates > within 4 Å

- Choose in the graphics window which of the symmetry molecules corresponds to the missing part of the tetramer.
- Rename the other half to biotinSA2
- Delete the other symmetry mates:

```
delete 3ry2* or delete sym*
```

```
show sticks, resn BTN
```

(show biotin (residue name BTN) as sticks)



Aligning structures

Continue with the same PyMOL session:

```
fetch 3ry1, apoSA
```

(load apo coordinates and call them apoSA)

```
hide everything, apoSA
```

(hide everything from the apoSA coords)

```
show cartoon, apoSA
```

(show it as cartoon, and perhaps color differently)

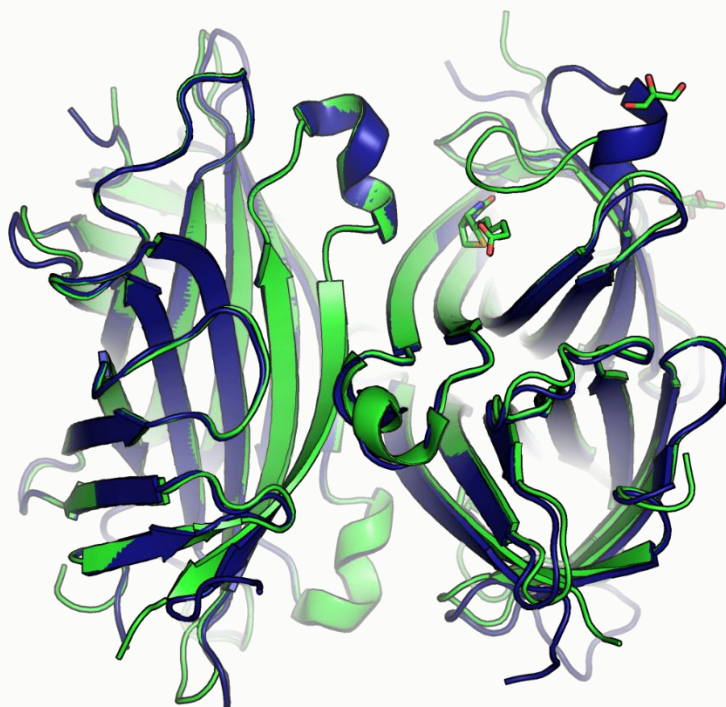
```
align apoSA & chain A, biotinSA
```

(align the apo coords to the complex)

```
colour apoSA & name C*, density
```

(colour apo streptavidin in dark blue, but carbons only)

Can you see what the differences are between biotin bound streptavidin and the apo form?



Electron density

First, load electron density map. Needs to be in so-called ccp4 format. You can get the density file from the Downloads menu at PDB Europe <https://www.ebi.ac.uk/pdbe/> for (almost) any PDB entry.

We'll load map for the biotin-bound streptavidin and show the density for biotin

First we load the map file and call it 3ry2_map:

```
load 3ry2.ccp4, 3ry2_map
```

Then we make selection for biotin in chain A:

```
select biotin, resn BTN & chain A & biotinSA
```

Then we show the map as a mesh at 1.0 sigma/noise level within 1.5 Å of biotin:

```
isomesh 3ry2_mesh, 3ry2_map, 1.0, biotin,  
carve=1.5
```

```
color blue, 3ry2_mesh
```

blue)

(colour the mesh in

```
set mesh_width, 0.5
```

thinner)

(make the mesh a bit

A handy plugin to have is "Isocontour slider"

<https://pymolwiki.org/index.php/Isoslider>

NMR ensembles

- NMR structures typically submitted as ensemble of 20-50 structures that all satisfy experimental constraints.
- By default PyMOL shows only the first molecule of the coordinate set
- To see them all at once:

```
fetch 1mph.pdb, async=1
```

```
set all_states
```

- Or if you want them separated into individual objects:

```
unset all_states (or: set all_states, 0)
```

```
split_states 1mph
```

```
delete 1mph (to remove the original ensemble)
```

- The ensemble is best shown as a C α -trace, aka “ribbon” in PyMOL

```
show ribbon, 1mph*
```

One state or all states

```
set all_states, 0  
set state, 1
```



```
set all_states, 1
```



Making figures

Ray trace for best quality images:

```
ray 2000 (ray trace image as 2000 pixels wide)
```

```
png myimage.png (to save the image as PNG)
```

Educational version of PyMOL does not allow ray tracing, but using:

```
set use_shaders
```

```
png myimage.png, 2000
```

Will result in quite decent images.

If ray tracing, you might want to:

Take the shadows off, as they can make the figures too busy:

[Setting > Rendering > Shadows > none](#)

Set transparent background

```
set opaque_background, 0
```

```
set ray_opaque_background, 0
```

Nice(?) mode with black outlines for the molecule:

```
set ray_trace_mode, 1
```

More PyMOL material

- PyMOL wiki at <https://pymolwiki.org>
- Great PyMOL tutorials:
 - Intro:
<https://www.researchgate.net/publication/313877004>
[4 Introduction to PyMOL](#)
 - Intermediate:
<https://www.researchgate.net/publication/313877009>
[9 Intermediate PyMOL](#)
 - Advanced
<https://www.researchgate.net/publication/313877075>
[5 Advanced PyMOL](#)

Some useful default settings

File > Edit pymolrc

```
bg_color white
set ray_opaque_background, 1
set opaque_background, 1

set mesh_width, 0.5
set mesh_quality, 3
set mesh_colour, blue

set dash_color, grey30
set dash_length, 0.25
set dash_gap, 0.15
set dash_round_ends, off

set ray_shadow, off
set ray_trace_mode, 1
set use_shaders
set pse_export_version, 1.74
```