Visualización (básica) de moléculas con PyMol

PyMOL es un programa open-source de visualización molecular para representación y animación de estructuras moleculares en 3D. Es mantenido y distribuido por Schrödinger.

Instalación

→ Incentive PyMOL

Schrödinger provee una versión paga pre-compilada (Windows, MacOSX y Linux), que incluye algunos plugins preinstalados y algunas facilidades para hacer minimizaciones energéticas, además de soporte técnico. También provee una versión educacional (suelen ser versiones más viejas), previo registro. Descarga http://pymol.org/download

→ Open-source PyMOL, pre-compilados en repositorios de Linux

PyMOL está en el repositorio de varias distribuciones de Linux, sin embargo no suele ser la última versión disponible. En Ubuntu, puede instalarse con:

```
sudo apt-get install pymol
```

→ Open-source PyMOL, instalación desde el código fuente en Linux

Es forma provee acceso a la última versión disponible y es la más recomendada.

-Previamente hay que instalar otras dependencias:

```
sudo apt-get install subversion build-essential python-dev python-pmw
libglew-dev freeglut3-dev libpng-dev libfreetype6-dev libxml2-dev
```

-Obtener la última versión de PyMOL:

Usando subversion:

```
cd /carpeta/donde/se/instalara
svn co svn://svn.code.sf.net/p/pymol/code/trunk/pymol
```

O desde SourceForge:

https://sourceforge.net/projects/pymol/

-Compilar e instalar. El siguiente script instalará PyMOL en la ruta que se especifique en *prefix,* copiarlo a un archivo, darle permisos de ejecución y correrlo.

```
#!/bin/bash -e
prefix=~/pymol1-8
modules=$prefix/modules
# enable c++11
export CPPFLAGS="-std=c++0x"
# If you want to install as root, then split this line up in "build"
# and "install" and run the "install" with "sudo"
python setup.py build install \
    --home=$prefix \
    --install-lib=$modules \
    --install-scripts=$prefix
```

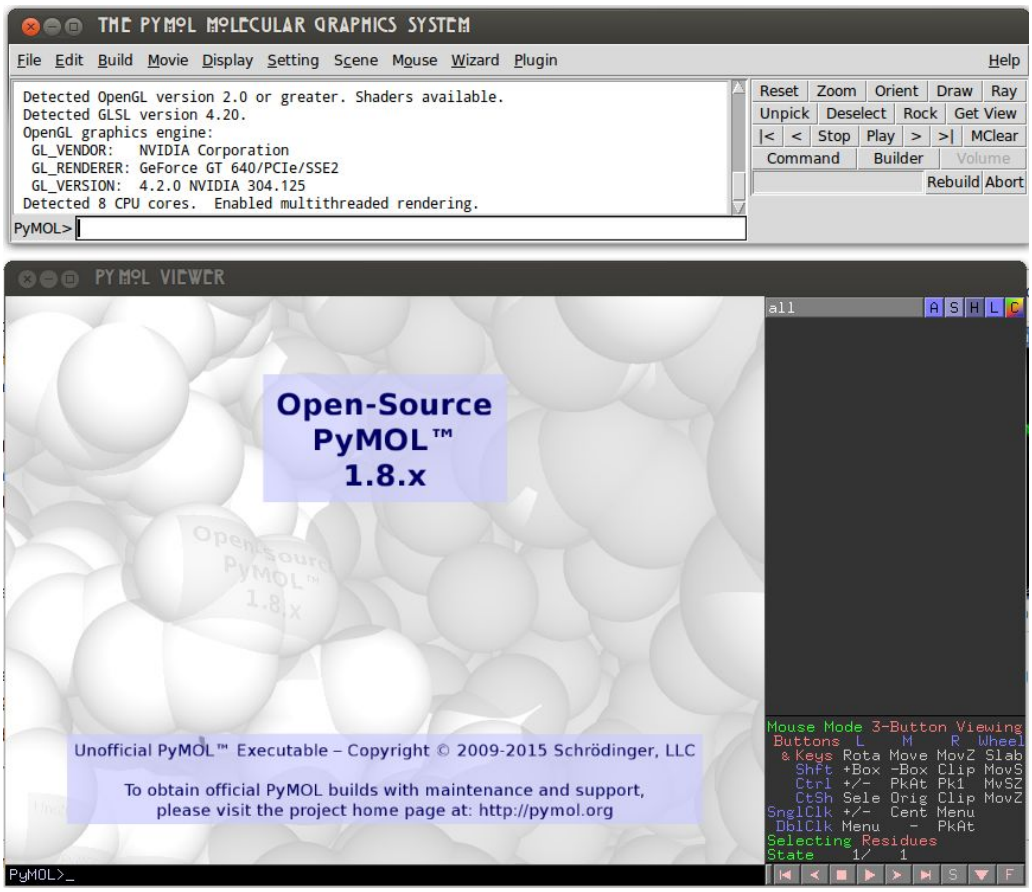
Dentro de la carpeta ~/pymol1-8 se habrá creado un ejecutable, que es el que llamaremos para correr el programa. También se puede poner en el $PATH del sistema la ruta hacia esta carpeta.
Más información:
http://www.pymolwiki.org/index.php/Linux_Install#Open-Source_PyMOL_in_Linux_Distros
Instalación en Windows:
http://www.pymolwiki.org/index.php/Windows_Install

Topicos

1. La interfaz gráfica de PyMOL
2. Explorando una estructura
3. Selectores y otros comandos útiles
4. Alineamiento estructural
5. Modo grid
6. Colorear y representar por B-factor
7. Alterar numeración de cadena de residuos
8. Superficies
9. Tips para generar imágenes de buena calidad

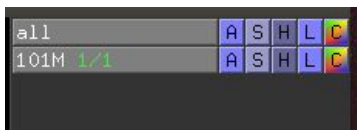
1. *La interfaz gráfica de PyMOL*

Suponiendo que tenemos la ruta de PyMOL en nuestro $PATH, podemos llamarlo desde consola estando parados en cualquier carpeta, simplemente con:
pymol
Esto abrirá dos ventanas, el External GUI (arriba) y el visualizador (abajo):



- En *PyMOL>* se encuentran las consolas para el ingreso de comandos

- Panel de control de objetos:



En el aparecerán los objetos (moléculas, selecciones, y otro tipo de objetos) que estén cargados a la izquierda en el visualizador. Luego del nombre de cada objeto, hay una lista de botones de comandos que controlan a cada objeto, con algunas de las siguientes opciones:

A - *Actions*: renombrar, duplicar, remover, aplicar, hacer algunos cálculos como alineamiento y distancias, hacer zoom, centrar, etc

S - *Show*: muestra el objeto con distintas formas de representación (balls and sticks, cartoon, mesh, surface, etc)

H - *Hide*: Oculta (no elimina) objetos

L - *Label*: Etiqueta átomos, residuos, etc. con distintas nomenclaturas

C - *Color*: cambia los colores del objeto, segun distintas propiedades: tipo de atomo, estructura secundaria, N o C terminal, b-factor, etc
Las acciones de estos botones se pueden realizar con comandos por consola, lo veremos más adelante

- Control de acciones del mouse y demás botones:

Mouse mode, modo de uso del mouse: visualizacion o edicion (haciendo click en este area se cambia de uno a otro), con las acciones que se realiza con cada boton y las techas Shift y Ctrl

Selecting, modo de selección: determina si con el mouse se seleccionarán átomos, residuos, cadenas, moléculas, etc. Haciendo click sobre este área se cambia entre uno y otro

State, estados: si cargamos un PDB con distintos modelos (ej: RMN) o una animacion, aquí nos mostrará que modelo/frame estamos visualizando. Los botones de play, stop, skip que están debajo permiten moverse de un estado a otro.

S (*sequence*): permite mostrar u ocultar la secuencia de los objetos que estén activos, dentro del área de visualización

F (*fit*): ajusta a modo pantalla completa

2. *Primeros pasos, explorando una proteína*

a. Descarga. Descargar de PDB y visualizar la estructura kinasa de EGFR (epidermal growth factor receptor) con el comando:
`fetch 2gs6`
Esto descarga el archivo pdb en la carpeta actual donde ejecutamos PyMOL. Si ya tuviésemos nuestro pdb descargado, podemos abrirlo desde $File \rightarrow Open$ o desde la consola en la misma carpeta donde está el archivo con:
`pymol 2gs6.pdb`
Observar que en el panel de objetos, aparece 2gs6.

b. Movimientos. Identificar las acciones del mouse rotando, trasladando, haciendo zoom y “clipping” en la molécula.

c. Secuencia y Selección. Mostrar la secuencia. Seleccionar residuos en la secuencia y notarlos en la estructura. Explorar la selección en la estructura de átomos, residuos y cadena completa. Probar manteniendo presionada la tecla Shift, seleccionar en el visualizador y en la secuencia.

d. Representación, colores, etiquetas. La representación por defecto de las cadenas peptídicas y de los ligandos (pequeñas moléculas) es en *lines,* y de los O del agua y de los iones, en *nonbonded.*
La estructura 2gs6 incluye una cadena A (dominio kinasa), una cadena B (un péptido de 5 residuos), un ligando (análogo de ATP), un ión de Cl, y los O de moléculas de agua del solvente. Para poder apreciar mejor cada parte, cambiar la representación de la cadena A a *cartoon* y colorear por estructura secundaria, la cadena B y el ligando ("112") en *sticks*, y colorear por tipo de átomo. Seleccionar el ion de Cl y representarlo como *nb-spheres,* colorear de naranja, y etiquetarlo según *atom name*. Ocultar los O de agua. Para todo esto, por ahora, emplear los botones en el Panel de Objetos.

El color de fondo y su transparencia se puede cambiar desde $Display \rightarrow Background$

➔ Hay algunas representaciones pre-establecidas, que pueden aplicarse a cada objeto desde $A \rightarrow Preset$

e. Guardar la sesión. Para guardar todo el trabajo hasta el momento: los objetos tal cual se los ve representados, coloreados, sobre un determinado color de fondo, etc, ir a $File \rightarrow Save\ Session\ As...$. Esto genera un archivo *.pse, que luego podrá ser cargado nuevamente ($File \rightarrow Open$).

3. Empleo de selectores, y otros comandos útiles

Con el comando *select*, se puede seleccionar distintos grupos de átomos, de la forma:
`select nombre, seleccion`
Esto creará un objeto de selección *nombre* en la lista de objetos, que incluirá todo aquello que se especifique, siguiendo ciertas reglas, en *seleccion*. Los atributos de cada objeto selección nuevo que se cree podrán modificarse tanto desde la interfaz gráfica, con los botones antes vistos, como por la línea de comandos
Por ejemplo:
`select cadenas_A, chain A`
Selecciona todas las cadenas A de todos los objetos que estén activos, y crea un objeto cadenas_A que hará referencia a ellas.
Las selecciones pueden hacerse más precisas siguiendo ciertas reglas de propiedades (http://pymolwiki.org/index.php/Property_Selectors) y álgebra de selección con los booleanos *and*, *or*, *not* (http://pymolwiki.org/index.php/Selection_Algebra). Los combinaciones y opciones son muchas, así que veremos algunas más útiles con ejemplos.

Single Word Selector	Short Form	Description
all	*	All atoms currently loaded into PyMOL
none	none	No atoms (empty selection)
hydro	h.	All hydrogen atoms currently loaded into PyMOL
hetatm	het	All atoms loaded from Protein Data Bank HETATM records
visible	v.	All atoms in enabled objects with at least one visible representation
present	pr.	All atoms with defined coordinates in the current state (used in creating movies)

Matching Property Selector	Short Form	Description	Identifier - List of	Example
symbol	e.	*atomic symbols*	1- or 2-letter chemical symbols from the periodic table	select PolarFe, symbol o+n+Fe
name	n.	*PDB Atom Names*	up to 4-letter codes for atoms in proteins or nucleic acids	select carbons, name ca+cb+cg+cd
resn	r.	*Residue Names*	3-letter codes for amino acids	select aas, resn asp+glu+asn+gln orup to 2-letter codes for nucleic acids select bases, resn a+g
resi	i.	*Residue number*	up to 4-digit residue numbers	select mults10, resi 1+10+100+1000 select nterm, resi 1-10
chain	c.	*PDB Chain ID*	single letters or sometimes numbers	select firstch, chain a
segi	s.	*Segment ID*	up to 4 letter identifiers	select ligand, segi lig
ss	ss	*secondary structure*	single letters	select allstrs, ss h+s+l+""

Otros, seleccionando por comparación con valores numéricos

Numeric Selector	Short Form	Description	Argument	Example
b	b	*b-factor-value*	real number	select fuzzy, b > 10
q	q	*occupancy-value*	real number	select lowcharges, q <0.50
formal_charge	fc.	*formal charge-value*	integer	select doubles, fc. = -1
partial_charge	pc.	*partial charge-value*	real number	select hicharges, pc. > 1

Iremos viendo también otros comandos. Cada uno de ellos tiene su manual de uso en PyMOLWiki (http://pymolwiki.org/index.php/Category:Commands). También se admiten comandos como *ls, cd, pwd* como los de Linux

a. Cargar las estructuras 2gs7 y 4g5p, remover (no ocultar!, comando *remove*) los O de agua, mostrar (comando *show*) las cadenas peptídicas como *cartoon,* crear objetos de la selección de cadenas A, B, ligandos y iones y colorear. Pueden ponerse todos los comandos en un script1.pml y ejecutarlo desde $File \rightarrow Run$ o con el comando *run script1.pml*

```
fetch 2gs7 4g5p
#remover átomos cuya propiedad nombre de residuo sea hoh
remove resn HOH
#ocultar todas las representaciones
hide all
#mostrar todo como cartoon (solo se mostrarán las secuencias proteicas)
show cartoon, all
#seleccionar las cadenas A y B por separado, de todas los objetos cargados
select cadenasA, chain A
select cadenasB, chain B
```

```
#colorear por estructura secundaria (ss, s:strand, h:helix, "":regiones de conexion)
color gray15, cadenasA
color density, ss S and cadenasA
color firebrick, ss H and cadenasA
color gray80, cadenasB
color lightblue, ss S and cadenasB
color darksalmon, ss H and cadenasB

#seleccionar átomos cuyo nombre de residuo sea 0WN o ANP (ligandos)
select ligandos, resn 0WN or resn ANP
#mostrarlos en representación sticks
show sticks, ligandos
#colorear por tipo de átomo, pero dejando el CA del color original
util.cnc ligandos

#seleccionar heteroatomos que no sean los ligandos 0WN o ANP
select iones, hetatm and (not (resn 0WN or resn ANP))
#mostrarlos en representación nb-spheres
show nb_spheres, iones
#colorear por tipo de átomo
util.cnc iones

#seleccionar átomos cuyos centros estén dentro de los 5A de los ligandos, sin incluir iones
select vecindades, ligandos around 5 and (not iones)
show lines, vecindades
util.cnc vecindades

#Mostrar en lines dos residuos que se sabe que forman un puente salino, etiquetarlos y medir distancias N-O
select a, resi 762 and( 4g5p and chain A)
select c, resi 745 and( 4g5p and chain A)
show lines, a c
label name CA and (a or c), resn+resi
distance salino, c and name NZ, a and name OE2

#gurdar la sesión
save sesion2.pse
```

➔ El comando *save*, sirve para guardar múltiples salidas, que se reconocen directamente de la extensión del archivo: .pdb, .pqr, .mol, .sdf, .pkl, .pkla, .mmd, .out, .dat, .mmod, .pmo, .pov, .png, .pse, .psw, .aln, .fasta, .obj, .mtl, .wrl, .idtf, or .mol2

4. Alineamiento estructural

Siguiendo con la sesión obtenida en el paso anterior, podemos observar que ambos pdbs (2gs7 y 4g5p) son homodímeros de la misma proteína, es decir estamos en presencia de un total de 4 estructuras proteicas que representan 4 confórmeros de la misma proteína. Cómo hacer para alinearlos? Primero, hay que separar cada cadena en un nuevo objeto, no nos sirven los objetos de selección antes creados. Luego, alineamos todas contra una (2gs7_A), usando *extra_fit*. Hay varias formas de alinear todos contra uno, pero esta permite elegir entre varios algoritmos de alineamiento de pares:

- align: se emplea cuando las estructuras a alinear poseen muy buena similitud de secuencia, recomendado para alinear confórmeros
- super: recomendado cuando hay una decente similitud estructural, pero baja similitud de secuencia
- cealign: se emplea cuando las proteínas tienen muy baja similitud de secuencia

```
#separar las cadenas de todas las moléculas actualmente cargadas, en distintos objetos
split_chains
#alinear todas contra 2gs7_A, empleando CA, algoritmo align y generando el objeto con el nombre all_to_2gs7A_alignCA
extra_fit name CA, 2gs7_A, align, object=all_to_2gs7A_alignCA
#guardar el alineamiento de secuencia generado en el objeto, en formato clustal
save alignment.aln, all_to_2gs7A_alignCA
```

En la ventana del log de los comandos, aparecen listados los RMSD entre pares alineados. Si no se especifica name CA, se realizará un alineamiento all-atom, y no se mostrará RMSD.

El objeto que se genera en cada alineamiento, permite ver el alineamiento de las secuencias (activarlo con el botón S) y sobre las estructuras, líneas amarillas que conectan los átomos alineados.

Cada algoritmo tiene sus propias opciones, y puede correrse directamente cuando se quiere alinear estructuras de a pares, ej:

```
cealign 2gs7_A, 4g5p_A, object=2gs7A_to_4g5pA_cealign
```

En este caso, se realizaría un alineamiento de un par de estructuras, all-atom y con cealign

5. Modo grid

El comando *grid_mode* particiona la pantalla del visualizador en una grilla, y muestra cada objeto en un *slot, al que se le asigna un slot_number*. Cada acción de rotación, zoom, etc, se aplica a toda la grilla. Uno o más objetos se pueden asignar al mismo *slot_number*.
Esto puede ser muy útil para comparar varios homólogos o confórmeros a la vez, previo alineamiento estructural.
La cantidad y organización de los *slots*, dependerá del tamaño de la ventana del visualizador
Siguiendo con la sesión del punto anterior, donde ya tenemos las estructuras alineadas, dejar solo activos los objetos 2gs7_A, 2gs7_B, 4g5p_A y 4g5p_B.

```
#activar el modo grid
set grid_mode,1
```

Trasladar, rotar, hacer zoom, centrar

```
#poner las cadenas de 2gs7 en el primer slot
set grid_slot, 1, 2gs7_A
set grid_slot, 1, 2gs7_B

#desactivar el modo grid
set grid_mode,0
```

6. Colorear y representar por B-factor

El valor de B-factor o factor térmico (última columna de las líneas ATOM y HETATM de un PDB) es un indicador de movimiento térmico de un átomo. Muy brevemente, cuanto mayor sea este valor para un átomo, mayor es la incerteza en la definición de sus coordenadas Cargar dos confórmeros, alinear, representar en modo cartoon y colorear por b-factor:

```
delete all
fetch 2ity 3w33
#alineamiento
align 2ity, 3w33, object=2ity_to_3w33
#colorear estructuras por b-factor, con paleta de colores rainbow de mínimo en azul a máximo en rojo
spectrum b, rainbow
#representación "gusano" con regiones de mayor B-factor con más diámetro
cartoon putty
#volver a representación cartoon normal
cartoon automatic
```

Que regiones parecen tener indicios de ser las más móviles?

➔ El comando *spectrum* tiene muchas otras opciones, y la columna de b-factors puede reemplazarse en los pdbs con cualquier otro valor numérico que represente otros aspectos de interés (ej: accesibilidad al solvente), y de esta forma representar y/o colorear la estructura mostrando la información.

6. Ajustar numeración de cadena de residuos

Muchas veces sucede que la numeración de la cadena en un pdb no coincide con la numeración de la secuencia canónica, que puede encontrarse en UniProt, porque, por ejemplo, la secuencia de interés tiene un péptido señal que se cliva en la maduración de la proteína, o porque se agrega un tag de His par facilitar su purificación. Esto dificulta el análisis de alineamientos, por ejemplo. Se puede resolver fácilmente con:

```
alter (chain A and 2gs7), resv += 10
sort
```

Donde, si la numeración de la secuencia arrancaba en 20, por ejemplo, ahora lo hará en 30. También se puede alterar el nombre de la cadena

```
alter chain A and 4g5p, chain="C"
sort
```

7. Superficies

Si bien una proteína es una cadena de aminoácidos que se ordenan formando estructuras secundarias como las que hemos visto, los átomos de cada aminoácido ocupan un lugar en el espacio, tienen un radio de Van der Waals, y en conjunto determinan un volumen de átomos con una superficie expuesta al solvente. Analizando la superficie proteica podemos identificar, por ejemplo, cavidades en la proteína o bolsillos donde podrían alojarse los sustratos o túneles por donde pueden ingresar los mismos.

Veamos un ejemplo:

```
delete all
fetch 2gs6
select ligando, resn 112
select peptido, chain B and (not hetatm)
select kinasa, chain A and (not hetatm)
select aguas, resn HOH
hide all
show cartoon, kinasa
show surface, kinasa
show sticks, ligando peptido
```

Observar el "hueco" en la superficie que se genera porque no representamos como superficie la cadena del péptido. Para que se vea bien, hay que extraer la selección del péptido en un nuevo objeto

```
extract peptido_ext, peptido
```

Por qué creen que puede estar sucediendo esto solo con el péptido y no con el ligando?

Se puede definir transparencias en las distintas representaciones, ejemplo:

```
#cartoon:
set cartoon_transparency, 0.5, <objeto>
#surface:
set transparency, 0.5
#sticks
set stick_transparency, 0.50, <objeto>
```

Donde los valores de transparencia va de 0 a 1. Si no se especifica objeto (no objeto selección) la transparencia se aplica sobre todo lo que esté representado de ese modo

Opcional: analizar los pockets calculados para 2gs6 con Fpocket (carpeta 2gs6_out)

8. Tips para generar imágenes de buena calidad

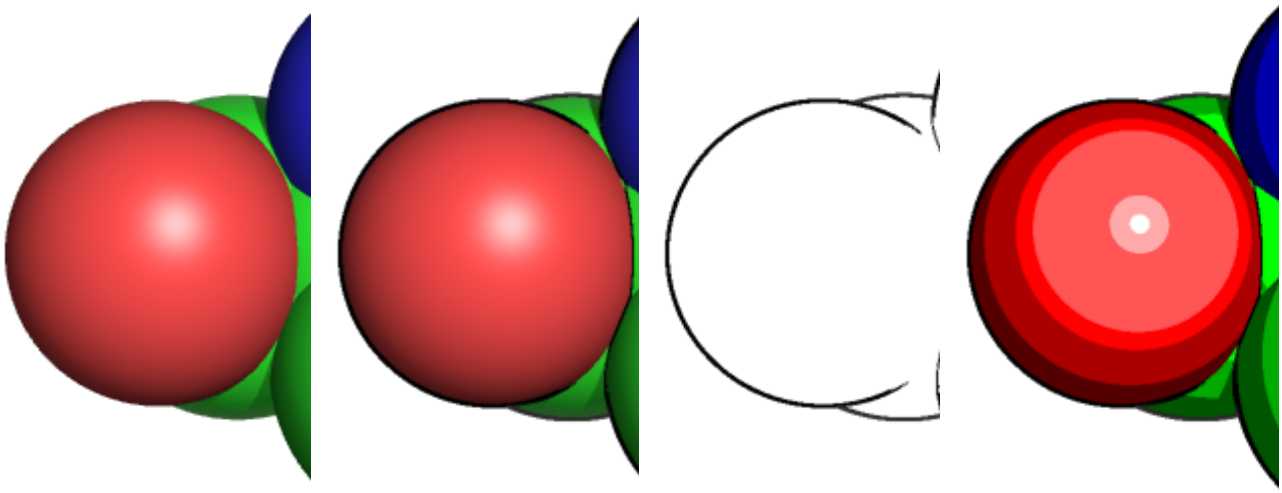
Con el comando *png* podemos guardar la imagen actual del visualizador, en formato png, con varias opciones

png ~/Desktop/prueba.png, width=10cm, dpi=300, ray=1

La imagen tendrá un ancho de 10cm, alto automático, 300dpi y se aplicará renderización (ray) antes de guardarla, para suavizar y definir mejor los bordes de las moléculas

El comando *ray* se puede usar para ir probando distintas configuraciones de delineados, sombras, luces, perspectivas, etc antes de generar las imágenes luego con *pgn*.

Una opción interesante es el comando ray_trace_mode que tiene 4 modos (de 0 a 3)



`set ray_trace_mode, 0` `set ray_trace_mode, 1` `set ray_trace_mode, 2` `set ray_trace_mode, 3`

En esta página pueden encontrarse muy buenos tips para generar distintos efectos:
http://www-cryst.bioc.cam.ac.uk/members/zbyszek/figures_pymol

Y en la galería de PyMOLWiki (http://www.pymolwiki.org/index.php/Gallery) hay una lista de imágenes con efectos y los correspondientes scripts para generarlos. Solo hay que adaptarlos a nuestras moléculas e ir probando el efecto de los comandos.