Envision Software Package for the Concerted Study of Biological Data

Nils Peder Willassen, Ingebrigt Sylte, Rafi Ahmad and Chris Fenton, Molecular Biocomputing Unit, Institute of Medical Biology, University of Tromsø.
Arne O. Smalås and Ronny Helland, The Norwegian Structural Biology Center, University of Tromsø. David Liberales and Inge Jonassen, Computational Biology Unit, University of Bergen.
Werner Van Belle and Kjell Arild Høgda, Norut IT, Tromsø. Tor Johansen The High Performance Computing Group University of Tromsø

Introduction

The study of biological data is difficult, because what may be mathematically or statistically significant is not necessarily biologically significant. Biological data must be interpreted in light of experimental results and methods. Therefore, experimentalists are needed in the analysis of biological data, analyses that often entail the comparison of homologous sequences or structures. However, finding the correct software, using the software intelligently, plus integrating, comparing, disseminating, validating and storing results can be a monumental and confusing task. Questions often asked by experimentalists include the following:

Which software application is appropriate? How do I compare results between applications? Are the results correct? How do I compare and coordinate alignment data, phylogenetic data, structural data, etc? How do I query the data? How can I visualize the results of biological queries? How can I save my results? How can I share my results? Can this process be automated in anyway?

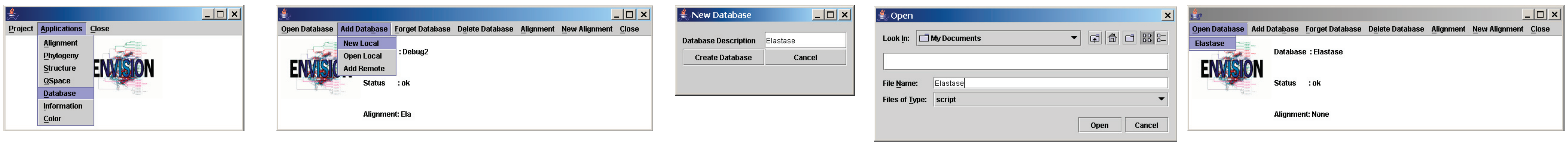
Goals

The envision software package aims to help in the comparison, coordination, visualization, storage, and exploration of biological data. Explicit goals of the envision software package include:

- The ability to compare results of similar applications in the same window.
- Viewers that share and synchronize application events.
- A dynamic self contained database that requires no administration and contains all information
- Local and remote databases for sharing of information.
- An advanced query language that allows experimentalist to form complex, understandable queries without a rigid syntax.
- Viewer window geometry and content is saved in script format for export, demonstrations, automation, and the ability to pick up exactly where you left off.
- Platform independent.
- Most importantly free of charge.

Step 1. Create a local database

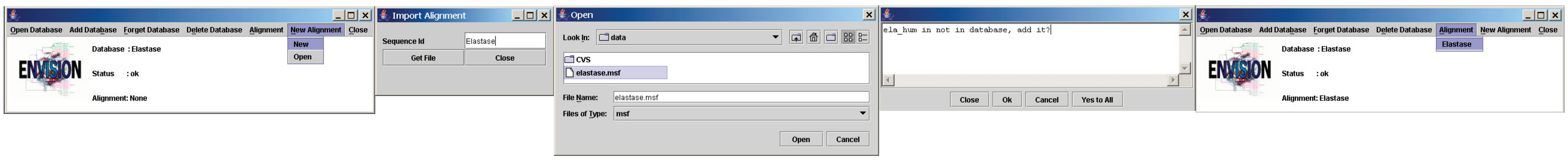
Launch the database manager from the applications menu to display the database manager. Add a new database, name it, and select a file location. The database is now ready to be used.



```
: Debug2
Status      : ok
Alignment: Ela

Database Description  Elastase

File Name:     Elastase
Files of Type: script

Database : Elastase
Status      : ok
Alignment: None
```

Step 2. Load an alignment

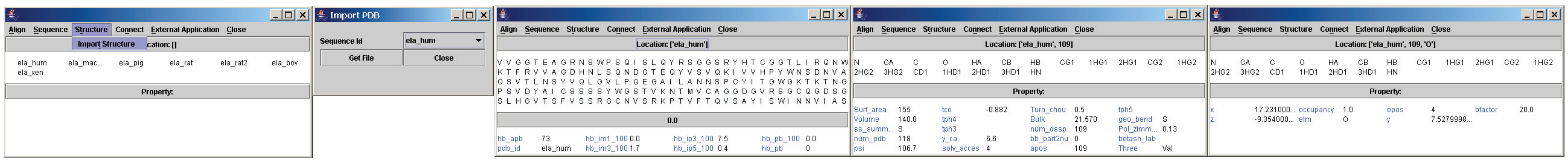
The first file loaded into the database is a multiple alignment. The initial alignment contains the skeletal protein sequences to which all additional information will be mapped.
Secondly, alignments represent a middle layer intercepting events 'ela_hum, 1' and changing to an aligned position 'ela_hum, 2' to coordinate data.



```
Database : Elastase
Status      : ok
Alignment: None

Sequence Id   Elastase

File Name:     elastase.msf
Files of Type: msf

ela_hum in not in database, add it?

Database : Elastase
Status      : ok
Alignment: Elastase
```

Step 3. Map information (PDB structure, DSSP, VADAR, etc.)

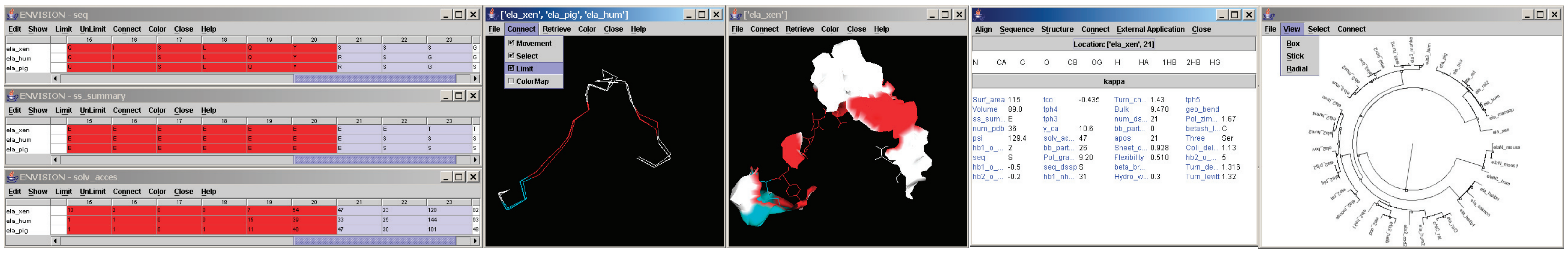
Information such as structure, solvent access, hydrogen bonding, etc can be mapped to the skeletal sequences, and become a protein, amino acid or atom property in the database.
All information is stored in the database (including dynamically generated) allowing for complex queries.



```
cation: []
ela_hum   ela_mac...   ela_pig   ela_rat   ela_rat2   ela_bov
ela_xen

Sequence Id   ela_hum

Location: ['ela_hum']
V V G G T E A G R N S W P S Q I  S L Q Y R S G G S R Y H T C G G T L I  R Q N W
K T F R V V A G D H N L S Q N D G T E Q Y V S V Q K I  V V H P Y W N S D N V A
Q S V T L N S Y V Q L G V L P Q E G A I  L A N N S P C Y I  T G W G K T K T N G
P S V D Y A I  C S S S S Y W G S T V K N T M V C A G G D G V R S G C Q G D S G
S L H G V T S F V S S R G C N V S R K P T V F T Q V S A Y I  S W I  N N V I  A S
0.0
hb_apb   73         hb_im1_100 0.0   hb_ip3_100 7.5   hb_pb_100 0.0
pdb_id   ela_hum    hb_im3_100 1.7   hb_ip5_100 0.4   hb_pb     0

Location: ['ela_hum', 109]
N     CA    C     O     HA    CB    HB    CG1   1HG1  2HG1  CG2   1HG2
2HG2  3HG2  CD1   1HD1  2HD1  3HD1  HN
Property:
Surf_area  155     tco        -0.882   Turn_chou  0.5      tph5
Volume     140.0   tph4                Bulk       21.570   geo_bend   S
ss_summ... S       tph3                num_dssp   109      Pol_zimm...  0.13
num_pdb    118     y_ca       6.6      bb_part2nu 0        betash_lab
psi        106.7   solv_acces 4        apos       109      Three      Val

Location: ['ela_hum', 109, 'O']
N     CA    C     O     HA    CB    HB    CG1   1HG1  2HG1  CG2   1HG2
2HG2  3HG2  CD1   1HD1  2HD1  3HD1  HN
Property:
x    17.231000...  occupancy  1.0   epos   4             bfactor  20.0
z    -9.354000...  elm        O     y      7.5279998...
```

Step 4. Launch and synchronize viewers

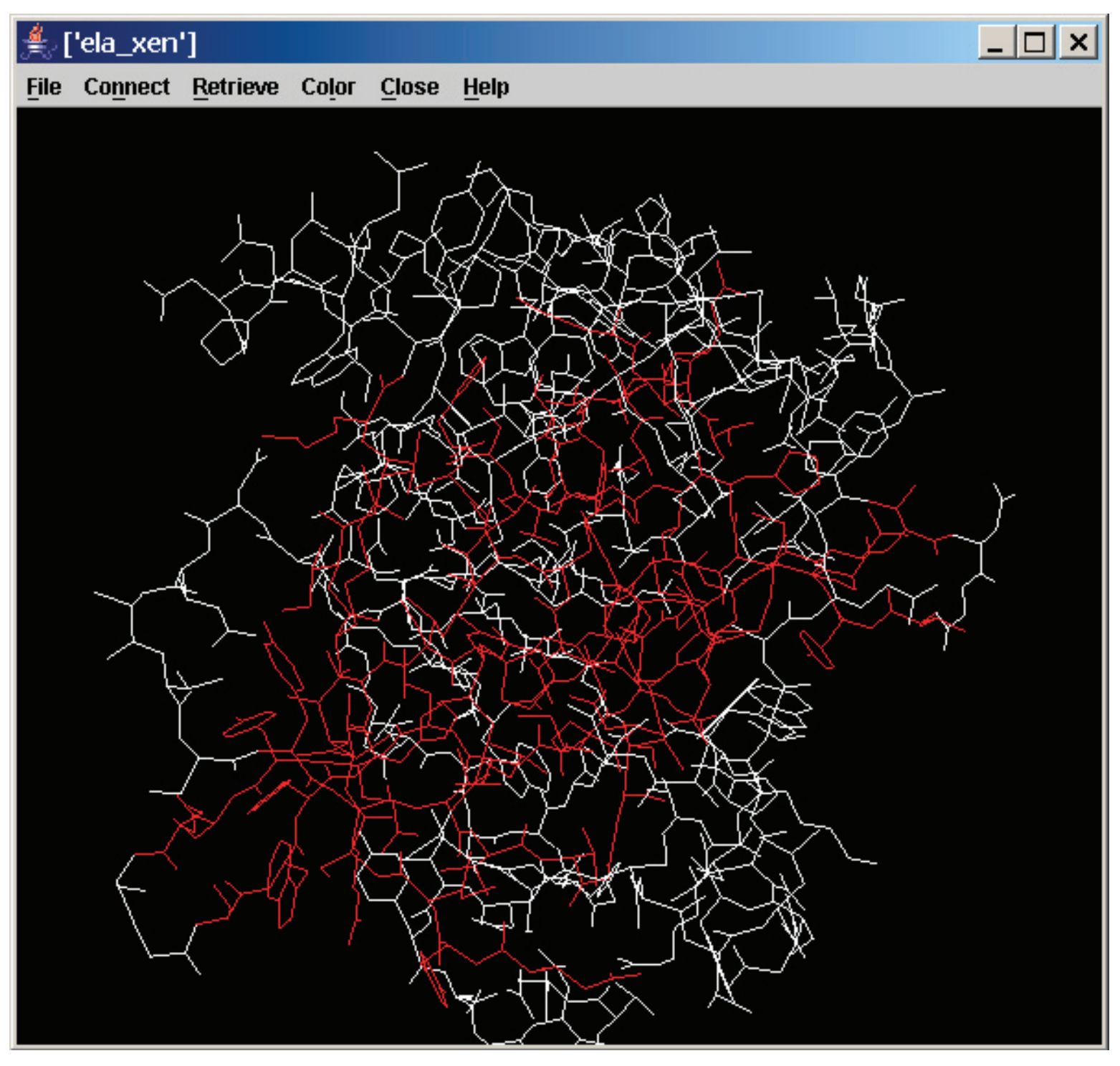
Currently there are five viewers; Alignment, Structure, Phylogeny, Information, and Seqspace. Common events such as Select, Limit, Colormap and viewer specific events such as Vertical (scroll), Movement(3D), etc., are sent to 'Connect'ed viewers. Relevant events are parsed through the alignment middle layer. An example of coordinated events is shown below, seven viewers have received Select, Color, and Limit events.

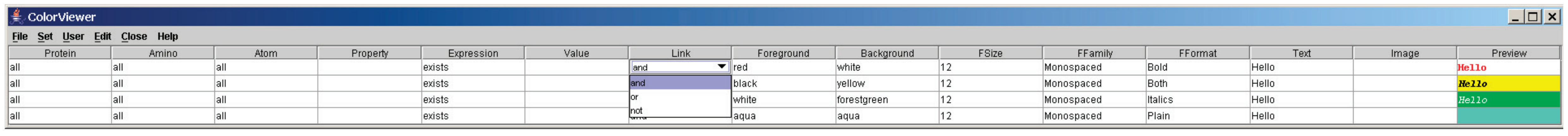


The query language

Querying stored data is often a difficult step requiring the experimentalist to learn a specialized language. The goal of the query language is to simplify the syntax to the point where it is understandable to the experimentalist without limiting available information. The user need not know where or how the information is stored. Additionally, expert users can extend the query language by introducing native functions.

To select all Amino acids for which secondary summary equals sheet. Open the ColorViewer (shown below) pull down the Property -> 'ss_summary', Expression->'=', Value='E' (dssp), and add any desired formatting. The result set can be used to format connnected viewers. The query can be entered as text in the script language as simply 'ss_summary == E', result set used to color viewer (shown right).
A second more complex example query: 'ss_summary == E & dist(AtomX, atom) < 10' which selects all atoms within 10 angstroms of any atom in a sheet. In this case the distance operator was defined by the user and added to the language. Although far from complete the query language represents a significant step in bringing the experimentalist closer to the data.





Protein	Amino	Atom	Property	Expression	Value	Link	Foreground	Background	FSize	FFamily	FFormat	Text	Image	Preview
all	all	all		exists		and	red	white	12	Monospaced	Bold	Hello		Hello
all	all	all		exists		and	black	yellow	12	Monospaced	Both	Hello		Hello
all	all	all		exists			white	forestgreen	12	Monospaced	Italics	Hello		Hello
all	all	all		exists			aqua	aqua	12	Monospaced	Plain	Hello		

Conclusions and remarks

- The envision software project is well underway with a beta version available for testing February, 2005.
- The software has been successfully tested on Windows, Linux, and Macintosh.
- Envision software has been developed by biologists for biologists.
- The graphic user interface has been designed to replicate known biological software to facilitate easy learning.
- Envision software has been successfully tested on an elastase data set of 40 proteins with 20 structures.
- Envision software has been successfully used as a teaching aid.
- Envision software is modular and should be easy to expand.

Current development

- Phylogeny viewer(s) and phylogenetic ancestral sequence work.
- Viewer to show protein specific properties.
- Support for RNA/DNA.
- Further work on the ColorViewer API including union, intersection, etc., of multiple sets of conditions.
- Optimization and testing of query language.
- A graphic script interface for the automated generation of scripts and optimization of the script language.
- Documentation and user groups.
- Simplified linking of external programs.