

MuGeN User Manual

Mark Hoebeke

INRA - Unité MIG (<http://www-mig.jouy.inra.fr>)

mhoebeke@jouy.inra.fr

MuGeN User Manual

by Mark Hoebeke

Revision History

Revision 1.1 2002-12-12 Revised by: mh

Added sections on preferences file format. Added detailed description of analysis result DTD. Added appendix on secondary dependence

Revision 1.0 2002-06-19 Revised by: mh

Initial release of user manual

Table of Contents

.....	??
What is MuGeN ?	1
Installing MuGeN	??
System Requirements	??
Software dependencies	??
Installation procedure	??
Using MuGeN	2
Using mugen for Interactive Genome Exploration	??
The Map List Window	??
The Map Drawing Window	??
The Information Window	??
Generating Annotated Genome Images.....	??
The MuGeN preferences file	??
Feature data sources.....	??
Display parameters.....	??
Links to external resources	7
Computer Analysis Result formats.....	7
The computer analysis results DTD.....	??
DTD explanation.....	9
MuGeN Option List.....	??
A. Secondary dependencies.....	13
C libraries	??
Perl modules.....	??

List of Tables

1. Modules Needed by MuGeN	1
2. Optional Modules for MuGeN.....	1
3. Options common to mugenb and mugenv	10
4. Options specific to mugenb	11

List of Figures

1. The Map List Window	3
2. The Map Drawing Window	5
3. The Information Window	6

What is MuGeN ?

The *Multi-Genome Navigator*, or MuGeN, is a bioinformatics software package providing tools for exploring multiple annotated genomes along with in silico analysis results. It offers two distinct programs, one for interactive visualization and navigation and another for the generation of images in various formats. Both programs can load annotated sequence data from a local file or retrieve it from databases across the network. Most of the parameters governing the way annotations and analysis results are displayed are customizable, either through the graphical user interface or with command-line parameters. The following sections show how to install and to use MuGeN before describing how to format home-made analysis results in order to integrate them in MuGeN.

Installing MuGeN

System Requirements

MuGeN has been used successfully on Intel/Linux and Sparc/Solaris platforms. It is not intended for use on Windows machines. Disk space requirements are minimal (less than 1 megabyte) but memory footprint can grow as more feature-rich genomes and/or complex analysis results are loaded. For example, the display of two reasonably-sized microbial genomes (about 4 Mb each) and a box plot of the conserved portions between the two of them consumes about 70 Mb of RAM on an Intel/Linux workstation. Line plots having one or more data points per base can be especially memory-consuming.

Software dependencies

MuGeN's programs and modules are all written in Perl. They rely on a set of more or less specialized third-party modules whose list is given in Table 1 (required modules) and Table 2 (optional modules). Notice that these tables only list modules not commonly found in Perl distributions. For a more extensive list of dependencies, see Appendix A. All of these components are freely available, mostly from CPAN.

Table 1. Modules Needed by MuGeN

Component	Release	Available at
bioperl	1.0	bio.perl.org (http://bio.perl.org)
Error	0.15	CPAN (http://www.cpan.org/modules/by-module/Error/)
Gtk	0.7008	CPAN (http://www.cpan.org/modules/by-module/GD/)
libxml	0.07	CPAN (http://www.cpan.org/modules/by-module/XML/)
PodParser	1.18	CPAN (http://www.cpan.org/modules/by-module/Pod/)
Usage	0.10	CPAN (http://www.cpan.org/modules/by-module/Usage/)
XML-Writer	0.4	CPAN (http://www.cpan.org/modules/by-module/XML/)

Table 2. Optional Modules for MuGeN

Component	Release	Available at
<i>EMBL data retrieval</i>		
CORBA-ORBit	0.4.3	CPAN (http://www.cpan.org/modules/by-module/CORBA/)
<i>Micado data retrieval</i>		
DBD-Pg	1.01	CPAN (http://www.cpan.org/modules/by-module/DBD/)
DBI	1.20	CPAN (http://www.cpan.org/modules/by-module/DBI/)
IO-String	1.01	CPAN (http://www.cpan.org/modules/by-module/DBI/)
<i>XEMBL data retrieval</i>		
SOAP-Lite	0.55	CPAN (http://www.cpan.org/modules/by-module/DBI/)

Installation procedure

MuGeN is available as a gzipped archive. Download the archive in an installation directory and expand its contents by issuing:

```
gzip -dc mugen-XXXXXXX.tgz | tar xf -
```

where XXXXXXXX stands for the release number. This will create a subdirectory called `mugen-XXXXXXX` containing all of MuGeN's programs and modules. `cd` in this directory and type:

```
perl install.pl
```

This will check for the required and optional Perl modules and configure MuGeN's scripts for execution. The absence of one or more optional modules will not prevent the program's installation, only a warning will be issued.

The executables **mugenb** and **mugenv** are located in the MuGeN installation directory. By adding this directory to the `$PATH` variable the executables can be run from any directory.

MuGeN relies on a preferences file to fix some display and database connection parameters. By default, it looks for a file named `.mugenrc` in the user's `$HOME` directory. A template preferences file, called `mugenrc_template.xml` and located in the `Data` subdirectory can be used for a start and copied to the `$HOME` directory.

A set of example files, used throughout this document, is available in the `mugen-data-XXXXXXX.tgz` archive. This archive can be extracted anywhere and creates a `mugen-data-XXXXXXX` directory containing several annotated genomes in GenBank format, as well as some computer analysis results.

Using MuGeN

The following sections offer a guided tour of MuGeN's main features and of how to make them work¹. The examples use the data files found in the MuGeN data archive. To run the commands given in these sections, make the `mugen-data-XXXXXXX` directory your current directory, and make sure the **mugenv** and **mugenb** commands are located in a directory accessible through your `$PATH`.

Using mughenv for Interactive Genome Exploration

To start a visual exploration, just specify the name of a file containing annotated sequence data after the **-d** option. This option can be used more than once to load several files. Running **mughenv** to navigate through the complete genomes of *Bacillus subtilis* and *B. halodurans* (contained in the `Bsub.gbk` and `Bhal.gbk` GenBank formatted files) is performed by typing:

```
mughenv -d Bsub.gbk -d Bhal.gbk
```

After a (short) while, the three windows of the graphical user interface pop up.

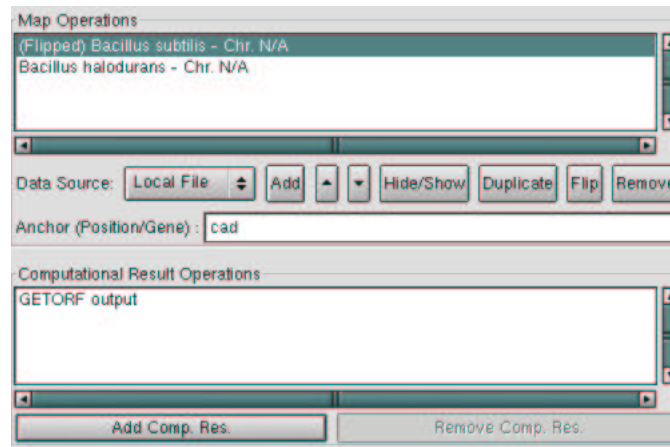
The Map List Window

This window (Figure 1) displays all loaded maps and computer analysis results. It also allows the manipulation of these maps and associated analysis results with the button row located below the map list. To load a new map, select a data source from the available sources in the popup menu, then click on the **Add** button. Depending on the datasource, some additional information will be requested (typically a filename or an access number). It may be useful to work with several copies of the same map (for instance to compare different portions of the same genome). The order in which the maps are displayed can be modified with the two arrow buttons. They shift the currently selected map up or down. A map can be hidden and redisplayed with the **Hide/Show** button. To generate a new copy of a given map, select it in the map list and click on the **Duplicate** button. Any map can be "flipped" with the **Flip** button, meaning that the base positions decrease from left to right, instead of increasing, and that the strands of the features are switched: features on the forward strand move to the reverse strand and vice versa. This feature is useful to compare genome portions which are conserved but whose directions are opposite. Finally a map can be removed using the **Remove** button. Notice that if there is only one map in the list, it cannot be removed.

Below the map operations panel, an **Anchor** textfield can be found. Each map can have its own anchor which "fixes" its relative position. An anchor is either an integer value, representing a base position, or a gene name. In the latter case, the start position of this gene (if it exists in the selected map) will be used as anchor. Moreover, the map will be flipped if the gene is on the reverse strand. Anchors are useful to simultaneously display distant portions of genome maps. For example, after loading two genome maps of closely related organisms, the context of a gene bearing the same name in the two organisms can be examined by selecting each map in turn and entering the common gene name in the anchor textfield. In the case of *B. subtilis* and *B. halodurans*, a possible anchor for both genomes is the *cad* gene.

The remaining part of the map list window contains a list of computer analysis results loaded for the currently selected genome map. Such results can be added (respectively removed) through the **Add Comp. Res.** (resp. **Remove Comp. Res.**) button. A sample analysis result file `Bsub_orfs.xml` contains all ORFs over 300 bp detected by the **getorf** program included in the EMBOSS package.

Figure 1. The Map List Window



The map list window with two genome maps (*Bacillus subtilis* and *Bacillus halodurans*). The selected map (*B. subtilis*) is anchored on the *cad* gene which has caused the map to be flipped. A computer analysis result, *GETORF output* has been added to the *B. subtilis* map.

The Map Drawing Window

This window gives a graphical display of the annotated genome maps along with the computer analysis results. The main area is divided in "strips" or lines. Each strip represents either a portion of an annotated genome or a portion of a computer analysis result. When several annotated maps are loaded, their strips are displayed one above the other (i.e. the first strip of the first map followed by the first strip of the second map followed by the second strip of the first map etc.). In that case, each map will have a different background color, ranging from white to light grey. When computer analysis results exist for a given map, they are displayed either on the map itself, or they are allocated separate strips immediately below the map they belong to.

By default, six lines per strip are used to draw CDSs, one for each reading frame of each strand. Other features are drawn either on the axis, if they are positional features (promoters, terminators, RBSs), or on a separate line below the CDS lines if they extend more than a dozen bp. (different RNAs, miscellaneous features and others). Also by default, CDSs are colored according to the strand they are located on, and filled if they have a known function, and empty otherwise.

Two other view modes are also available:

- a *bird's eye* view: this view is adapted to display large portions of genome maps. It is automatically activated when more than 50 Kb per line are shown. In bird's eye view mode, all features are drawn as simple boxes, or little sticks and are no more reactive.
- a *sequence* view: this is the view mode for lines smaller than 100 b. It shows the nucleotide sequence as well as its translation in the six reading frames.

The majority of display settings can be modified with the user controls at the bottom of the Map Drawing Window or with the menu entries it offers. The topmost row of user controls contains arrow buttons to

move forward or backward along the maps. The row below allows them to be zoomed in or out. Precise starting points, number of lines and bases per line can be set with the text fields below the zoom buttons. Finally, the thresholds for switching between the different view modes can be fixed with the sliders at the bottom of the window. The rightmost slider defines the minimum relative size for features whose names will be displayed. The **Preferences** menu offers several items influencing the map display:

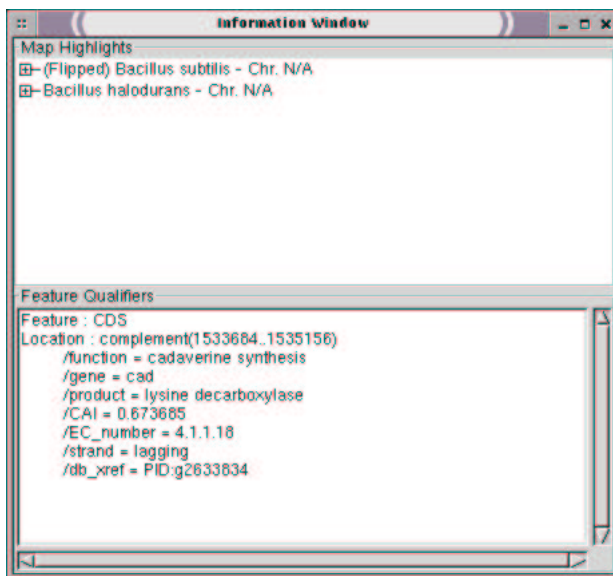
- **Expand Strands:** When checked, features belonging to different strands will be displayed on separate lines. Otherwise they will be displayed on the same line.
- **Show Frames:** When checked, CDSs are displayed on different lines according to their reading frame.
- **Visible Features:** This submenu offers one entry per feature type. Only the checked features are displayed on the map.
- **Map Area Width:** The width in pixels of the area on which the maps are drawn can be selected in this submenu.
- **Save Preferences:** The current settings of the **Preferences** menu are saved in the default preferences file (\$HOME/.mugenrc).

Figure 2. The Map Drawing Window



The Information Window

Figure 3. The Information Window



Generating Annotated Genome Images

The MuGeN preferences file

A major part of the display parameters can be set in MuGeN's preferences file. By default, MuGeN expects to find these settings in the `.mugenrc` file located in the user's `$HOME` directory, but a different file can be used through the `-p` command option. The preferences are stored in XML format and the DTD is specified at the start of the file. The following sections detail the contents of the preferences file.

Feature data sources

Parameters used to connect to remote data sources are specified in the `<featuredatasources>` tags. Commenting out² a given datasource will result in MuGeN ignoring it.

- The `<micado>` tag has attributes specifying how to connect to the Micado database. By default, it is commented out in the preferences template file. To use the Micado database, a valid user name and password are needed.
- The `<embl>` tag points to the location where the BioCorba-0.2 compliant IOR can be found. The specific URL to this location is specified in the `iorurl` attribute. *This data source may be unsupported in future releases now that MuGeN offers access to the same data through the XEMBL data source. The former is not operational on sites shielded by a firewall whereas the latter is.*

Display parameters

These parameters are located between the <mapdisplay> tags.

- The <strands> tag has a `layout` attribute specifying if the CDSs of the two strands should be displayed on separate lines ("expanded") which is the default value, or on a single line ("collapsed").
- The <frames> tag has a `visible` attribute with possible values of "true" (which is the default value) or "false" causing each CDS to be displayed on a separate line according to its reading frame. By combining this tag with the <strands> tag, CDSs can be displayed on one to six separate lines.
- The <feature> tag can occur multiple times, and defines which types of feature will be shown, and which types will remain hidden. By default, all feature types are visible, so <feature> tags are mainly used to hide "obtrusive" features (like *source* or sometimes *gene* features). The `type` attribute denotes the feature type for which the visibility is defined (eg. "CDS", "tRNA", "misc_feature"). The `visible` attribute can be "true" (default value) or "false".
- The <namethresh> tag defines the minimum feature size whose names will be displayed. Its `percent` attribute gives this minimum size relative to the size of a line. For instance, if each line shows 10 kb, and the percentage is set to 10, only features spanning more than 1 kb will have their names displayed.
- The <seqthresh> tag defines the threshold for switching between sequence view mode and default view mode. Its `bp` attribute specifies at which number of base pairs per line the switch will occur (i.e. when the number of base pairs per line drops below the threshold, the sequence view mode will replace the default view mode and vice-versa.).
- The <birdseyethresh> tag defines the threshold for switching between the default view mode and the bird's eye view mode. Its `kbp` attribute specifies at which number of kbps per line the switch will occur.

Links to external resources

MuGeN is capable to communicate with Web browser to visualize resources related with currently displayed features. The following tags define which resources MuGeN should use and how the Web browser will be invoked.

- The <links> tags delimit a set of <links>. Each <link> tag defines an external resource : the `name` attribute defines the name of the resource as it will be displayed in the resources menu, the `id` resource defines the prefix of the `/db_xref` qualifier. For instance, for a `/db_xref="taxon:71421"` qualifier, the prefix is `taxon`. The `url` attribute defines the URL leading to the resource. MuGeN will add the specific suffix when invoking the resource. (For instance, taking the same example, the string 71421 will be appended to the URL leading to the taxon database.
- The <browser> tag defines how to invoke an external browser from within MuGeN. Its `command` attribute specifies the exact command line necessary to launch the chosen browser. Inside the command line the string `_URL_` will be replaced by the actual URL of the external resource when this resource is activated.

Computer Analysis Result formats

MuGeN is capable of loading computer analysis results stored as XML files and conforming to the DTD defined in the `CompAnalResults.dtd` file. Basically, analysis results come in two flavors: lineplots adapted to "curves" having one or more data points per base (typically, pip-plots or HMM state plots), and boxplots to draw boxes extending from a start base to an end base. One file can contain a mix of lineplots and boxplots and provides tags to define specific colors as well as plot highlights which will show up in MuGeN's information window. The DTD is detailed below.

The computer analysis results DTD

```
<!ELEMENT companalresults (colors|lineplots|boxplots)* >
<!ELEMENT colors (color)* >
<!ELEMENT color EMPTY >
<!ATTLIST color
5   name CDATA #REQUIRED
    red  CDATA #REQUIRED
    green CDATA #REQUIRED
    blue CDATA #REQUIRED >
<!ELEMENT highlights (highlight)* >
10 <!ELEMENT highlight EMPTY >
    <!ATTLIST highlight
        label CDATA #REQUIRED
        begin CDATA #REQUIRED
        end   CDATA "REQUIRED >
15 <!ELEMENT lineplots (lineplot|highlights)* >
    <!ATTLIST lineplots
        type (separate|overlay) "separate"
        comment CDATA #REQUIRED
        min   CDATA #IMPLIED
20        max   CDATA #IMPLIED
        smoothing CDATA #IMPLIED >
    <!ELEMENT lineplot (#PCDATA) >
    <!ATTLIST lineplot
        frame (none|all|1|2|3) "none"
25        strand (1|-1) "1"
        color CDATA #IMPLIED
        start CDATA #IMPLIED
        step  CDATA #IMPLIED >
    <!ELEMENT boxplot (box|highlights)* >
30 <!ATTLIST boxplot
        type (separate|overlay) "separate"
        comment CDATA #REQUIRED >
    <!ELEMENT box EMPTY >
    <!ATTLIST box
35        begin CDATA #REQUIRED
        end   CDATA #REQUIRED
        thickness CDATA #IMPLIED
        label CDATA #IMPLIED
        halign (left|middle|right) "middle"
40        valign (above|inside|below) "below"
        labelcolor CDATA #IMPLIED
```

```

        frame (none|all|1|2|3) "none"
        strand (1|-1) "1"
        color CDATA #IMPLIED
45      filled (yes|no) "yes" >

```

DTD explanation

- Color definitions: colors must have a name and three color attributes defining the amount of color in each of the three color channels. These values range from 0 to 1 and conform to the RGB color model. Example:

```
<color name="turquoise" red="0.25" green="0.88" blue="0.8">
```

- Plot highlights: regions of interest can be defined as highlights. Each highlight will have it's own entry in the highlight section of the Information Window. Highlights are defined by a label and begin and end points expressed in bases. Example:

```
<highlight name="putative gene transfer" begin="1695413" end="1878744">
```

- Lineplots: this container tag groups a set of lineplots and their associated highlights. It defines how the lineplots will be positioned wrt. the features through the `type` attribute. If it's value is set to "separate", the lineplots will be drawn on a separate line below the features they are related to; if set to "overlay" they will be mixed with the features. The `comment` attribute is used to set the name of the set of lineplots. This name will be displayed in the Computer Analysis Result panel of the Map List Window. For a given map, there can only be one result with a given name. The `min` and `max` attributes are optional and can be used to specify the extreme values of the plots. By default, these are computed automatically. Finally, the `smoothing` is meant to improve drawing speed by allowing a series of points whose values do not differ by more than the relative amount given, to be drawn as a horizontal line. For instance, if this parameter is set to "0.1" and the plot contains a series of consecutive values in the range 0.9 to 0.99, only the endpoints of the series will drawn and linked with a horizontal segment.
- Lineplot: this tag encloses the actual data values to be plotted. The position of this plot relative to the features is fixed with the `frame` and `strand` attributes. The former allows to position a plot in a specific reading frame (values "1", "2" or "3"), to make a plot span all three reading frames (value "all" or position it below the CDSs with the other types of features (value "none"). Additionally, the `strand` attribute defines over which of the two strands the plot will be drawn. The `color` defines the color of the plot. The `start` attribute sets the position of the base corresponding to the first data point, and the `step` attribute defines the number of bases separating each data point. Examples:

```

<lineplots type="separate" comment="Line plot example 1">
<lineplot frame="all" color="red" start="1000" step="10">
100
0
50
0
100
</lineplot>
</lineplots>

```

This will draw a line plot on a separate line below the map features. The plot starts at base 1000 with value 100, drops to value 0 at base 1010, raises to value 50 at base 1020 etc.

```
<lineplots type="overlay" comment="Line plot example 2">
<lineplot frame="1" strand="1" color="green" start="1000" step="10">
100
0
50
0
100
</lineplot>
</lineplots>
```

This will plot the same curve as above, except that it will be positioned over the CDSs of the leading strand and in reading frame 1.

- **Boxplot:** this container tag encloses a set of box descriptions. It has `type` and `comment` attributes identical to line plots.
- **Box:** this tag describes the precise characteristics of a boxplot component. Its `frame`, `strand` and `color` attributes have the same meaning as for line plots. The `begin` and `end` attributes define the range of base positions the box should cover. The `thickness` attribute defines the vertical width of the box and can take values in [0..1]. The `label` attribute allows the definition of a text string accompanying the box. The position of this string wrt. the box is set with the `halign` and `valign` attributes and its color with the `labelcolor` attribute. Example :

```
<boxplot type="separate" comment="Box plot example">
<box begin="1" end="1000" thickness="0.2" color="red" filled="yes"
label="First Kb" labelcolor="blue" valign="above"/>
<box begin="1001" end="2000" thickness="1" color="black" filled="no"
label="Left of 2nd Kb" labelcolor="green" valign="inside" halign="right"/>
</boxplot>
```

MuGeN Option List

Table 3. Options common to `mugenb` and `mugenv`

Option	Multi ^a	Functionality
-d <i>source:id</i>	Yes	Specifies a resource from which to load annotated genome maps. Each resource consists of two parts, a <i>source</i> and an <i>id</i> . The source can be one of file , genbank , embl , xembl or micado . When no source is specified, file is taken as default. The id points to the specific map in the source. When the latter is a file, the id is simply the filename (in GenBank, EMBL, BSML or fasta format). When the source is a database (genbank , embl , xembl , micado) the id is the access number of the database entry. Maps will be displayed from top to bottom in the order they are entered on the command line. If the <i>id</i> start with a "!" the map will be flipped.

Option	Multi ^a	Functionality
-f <i>firstbase</i>	No	Specifies the starting point of the image to build. In the absence of any reference points, this is the first base of the map that will be located in the upper left corner of the image. If a reference point is given, the upper left corner will be the reference point offset by the amount specified by this option.
-l <i>lastbase</i>	No	Specifies the ending point of the image to build. In the absence of any reference points, this is the last base of the map that will be located in the upper lower right corner of the image. If a reference point is given, the lower right corner will be the reference point offset by the amount specified by this option.
-s <i>step</i>	No	Specifies the number of bases per display line.
-r <i>refpos</i>	Yes	Specifies a <i>reference position</i> or <i>anchor</i> for a genome map. If the reference position is an integer, the start of the displayed image will be computed by adding the value of the -f option to the integer. If the reference position is a string, MuGeN will look for a CDS feature having a gene qualifier whose value equals the given string. If such a CDS is found, it's start base will be used to compute the start of the displayed image as explained above. Moreover, if the gene is on the reverse strand, the map will be flipped. The genome map for which the reference position is defined is determined by the index of the -r option wrt. the -d option (i.e. the first -r option will be applied to the map defined by the first -d option, the second -r applies to the second -d and so on).
-c <i>filename[,index]</i>	Yes	Specifies a computational analysis results file to display with a genome map. If a comma and an <i>index</i> are appended to the filename, the result will be applied to the genome map of the corresponding index. Index 1 is the genome map loaded by the first -d option, index 2 the map corresponding to the second -d and so on.
-e <i>filename</i>	No	Specifies a file containing a color scheme to apply to displayed features.
-w <i>n</i>	No	Specifies the width in pixels of the drawing area
-p <i>filename</i>	No	Specifies the preferences file to load. If no -p option is given, the preferences file will be set to \$ {HOME} / .mugenrc.
Notes:		
a. Multi options are options that can be used several times on the command line.		

Table 4. Options specific to mugenb

Option	Multi	Functionality
-o <i>format</i>	No	Specifies the output format of the image file to be generated. Valid formats are : PNG, IMAP, PS, EPS, XFIG.

Option	Multi	Functionality
-m <i>mediatype</i>	No	Specifies the media type, for PS or EPS output files. Valid types are : a7, a6, a5, a4, a3, a2, a1, a0, b7, b6, b7, b4, b3, b2, b1, b0, letter, legal, executive, ledger.
-u <i>urlprefix</i>	No	Specifies the root URL for client-side image maps in IMAP format. Parameters relative to displayed features will be appended to this root URL. For instance, given a root URL of <code>http://www.somewhere.org/cgi-bin/myscript.pl?myid=xyz&</code> , and an image containing a CDS feature, whose name is abcX positioned from base 1234 to base 5678, the URL generated for it's clickable area will be <code>http://www.somewhere.org/cgi-bin/myscript.pl?myid=xyz&tag=CDS&n</code>

Notes

1. Table 3 details all command line options supported by MuGeN.
2. To comment out a section of an XML file, surround it with the strings `<!--` and `-->`.

Appendix A. Secondary dependencies

C libraries

The following list details a set of libraries upon which the modules used by MuGeN rely. They are part of all major Linux distributions, but may be needed for installing MuGeN on other platforms.

- Glib: Glib is a general purpose C library needed by the Gtk toolkit. It is available at <ftp://ftp.gtk.org/pub/gtk/v1.2/>. (<ftp://ftp.gtk.org/pub/gtk/v1.2/>).
- Gdk: Gdk is a low level graphics library also used by Gtk. It is also available at <ftp://ftp.gtk.org/pub/gtk/v1.2/> (<ftp://ftp.gtk.org/pub/gtk/v1.2/>).
- Gtk: Gtk is the library providing all of the graphics interface widgets and controls. It is available at <ftp://ftp.gtk.org/pub/gtk/v1.2/> (<ftp://ftp.gtk.org/pub/gtk/v1.2/>).
- Gd: GD is the library needed for generating PNG images with MuGeN. It is available at <http://www.boutell.com/gd/>

Perl modules

Below is the list of Perl modules needed by MuGeN. Most of them are part of standard Perl installations on Linux distributions, but again, installing MuGeN on Solaris may require to install some or all of them. They can all be found on CPAN (<http://www.cpan.org>).

- Carp: used for errors, warnings and information messages.
- File::Basename: used to separate a path into its components.
- GetOpt::Long: used to process command-line arguments.
- HTTP::Request: used to retrieve data from EMBL (optional).
- IO::File: used to write preferences with XML::Writer.
- LWP::UserAgent: used to retrieve data from EMBL (optional).