User manual for Dolmen 1.3 (2014/09/14)

Julien Eychenne

1 Overview

Dolmen is a free, open-source software toolbox for data analysis in linguistics. It offers a user-friendly interface to manage, annotate and query language corpora. It is particularly well suited for dealing with time-aligned data. The main features it offers are:

- Project management: organize files into projects and manage versions.
- Extensible metadata: files can be annotated with properties, which allow you to sort and organize your data.
- Interaction with Praat: Dolmen can read TextGrid files and open files directly in Praat.
- Powerful search engine: build and save complex queries; search patterns across tiers.
- Standard-based: Dolmen files are encoded in XML and Unicode.
- Scripting engine: Dolmen can be extended with plugins written in JavaScript/JSON.

Dolmen runs on all major platforms (Windows, Mac OS X and GNU/Linux) and is freely available under the terms of the GNU General Public License (GPL).

The latest version of Dolmen can be downloaded from www.julieneychenne.info/dolmen. If you encounter any problem or bug, please write to jeychenne@gmail.com.

2 Installation

2.1 Windows

On Windows, Dolmen is provided as a self-contained installer file. Simply double-click on dolmen_setup.exe and follow the instructions on screen. The procedure will install Dolmen in your Program Files directory and will create a shortcut in the start menu and optionally on the desktop.

If you wish to be able to open files in Praat from Dolmen, you will need to install Praat in Dolmen’s installation directory, which should be either C:\Program Files (x86)\Dolmen\Tools or C:\Program Files\Dolmen\Tools, depending on your system. Alternatively, you can modify Praat’s default path with the preference editor (see 9.2 below).

2.2 Mac OS X

On Mac OS X, Dolmen is provided as a standard DMG disk image. Mount the image by double-clicking on it and drag the application Dolmen into your Applications folder. If you wish to be able to open files in Praat from Dolmen, you will need to install Praat separately in the same folder.

Currently, only Mac OS X 10.7 ‘Lion’ and later “officially” supported. It does not work on earlier versions.

2.3 Linux (Debian/Ubuntu)

The official executable that is provided on the website is built on Ubuntu 14.04 and is available 64-bit architectures. The program is available as a dynamically-linked executable.

First make sure that the needed dependencies are installed (asound, libsndfile, speexdsp, Qt 5 and GTK 2). Most of these packages should already be installed, but you can issue the following command in a terminal to make sure they are:

```
sudo apt-get install libasound2 libsndfile1 libspeexdsp1 libgtk2.0-0 qt5-default libqt5sql5-sqlite libqt5script5
```

Next, assuming that you downloaded the archive in your Downloads directory, type the following commands in a terminal (replacing `XX` by the appropriate version number):

```
tar xvjpf dolmen-XX-linux64.tar.bz2

chmod +x dolmen/dolmen

sudo mv dolmen/dolmen /usr/local/bin

rm -r dolmen
```

You can now run Dolmen by simply typing `dolmen` from a terminal window.

3 User interface layout

On start up, Dolmen always displays the main window, which is split into 4 areas:

- the file browser (on the left): it is used to display the files in your current project.
- the application tabs (at the bottom): a number of fixed tabs used to interact with the application.
- the metadata panel (on the right), which is used to display contextual metadata, depending on the task at hand.
- the viewer (which occupies the remainder of the window): it stores a number of 'views' (for instance, the result of a query). Views can be added, removed and moved around. They work very much like tabs on modern web browsers.

Most features in Dolmen are available through context menus: to see what features are available for a given component, simply right-click on it to display its context menu. Important features are also available in the main menu bar.

4 Managing projects

4.1 Overview

Dolmen was designed to deal with large corpora of linguistic data. Therefore, it does not operate on single files but on file collections, which are called "projects". When it starts up, Dolmen opens a default empty project. The project structure is displayed in the file browser, in the left part of the main window. You can add files to the current project via the command Add file(s) to project... available from the context menu in the file browser or from the File menu. To select several files, keep the "control" (ctrl on Windows and Linux) or "command" (cmd on Mac) key pressed and click on each file you want to open. If you want to import a large number of files at once, it is easiest to put them all in one folder (or directory) and use the command Add content of folder to project... This will recursively import all files and sub-folders from the folder you chose into the project.

Currently, Dolmen supports 3 types of files:

- Sound files: Dolmen supports many sound formats including the popular WAV, AIFF and FLAC (Free Lossless Audio Codec). You can see all supported formats by clicking Supported audio formats from the help menu. Note that Dolmen does not support MP3, due to licensing restrictions (if you need a lossy format, you can use OGG instead).
- Annotations: these are time-aligned text files, which are generally Praat TextGrid files or WaveSurfer label files.
- Documents: these plain text files, normally encoded in UTF-8. All files bearing the extension .txt are recognized as documents.

Within the file browser, you can move files around and organize them into folders and sub-folders in any way you want. To create a new folder, select the files you want to group together (maintain ctrl pressed while clicking on the files), right-click on one of the selected files and click on the command Move files to folder...: you are then prompted for a name. Upon validating the name (by pressing Enter or clicking the ok button), the folder is created at the bottom of the file browser, with the selected files in it. You can drag and drop it wherever you want within the file browser and create as many sub-folders as you wish. The hierarchical structure of your project does not affect or depend on the location of the files on your hard-drive; it is stored in the project file directly and is meant to help you organize your files within the application.

To save your project, use the command Save project...: a project ends with the extension .dmpr ("Dolmen project"), which will be added automatically if you omit it. You can also open an existing project using the command Open project... The most recently opened projects are available in the File menu under Recent projects. The last project that was opened during the previous session can be re-opened with the shortcut ctrl+shift+o.

4.2 Managing metadata

4.2.1 Overview

When you click on a file in the file browser[1], its metadata are displayed in the metadata panel. The metadata displayed are the file's name, the file's description, which can be directly edited, the files's properties (see below) and, if the file is an annotation, the sound file it is bound to (if any). To know the full path of a file, simply hover the mouse cursor over the file name in the metadata panel.

4.2.2 Properties

Dolmen does not make any assumptions about the semantics associated with your project. Instead, it offers a simple yet powerful mechanism to add metadata to it, namely "properties". A property is a category/value pair. There is no limit on the number of properties a file may have, but a file may only have one value per category. One way to think of properties is to conceptualize categories as columns in a table and values as rows.

Properties are managed in the property editor. There are two ways to access it. You can either right-click on a file in the file browser and select Edit properties… or you can click on it and

[1] When you hover the mouse cursor over a file in the file browser, its metadata are also displayed in a tool tip. Categories are displayed the format "Category : Label".

click on the Edit properties… button in the metadata panel.

The property editor (see Figure XXX) is very similar to Microsoft Word's, if you have ever used it. It shows the categories currently associated with the file(s) and their value in a table, and lets you add or modify a category by either choosing an already existing category (or value) or creating a new one. Note that if you select a new value for an existing category, the old value will simply be overwritten. This is tantamount to replacing the value of a cell in a spreadsheet program. To remove a category, click on it in the category table and press the remove button.

Categories enable you to organize your data however you see fit. For instance, if you have 12 speakers with 3 tasks each, you may create a "Speaker" category with a different label for each speaker, and a "Task" category with 3 labels corresponding to each task. You could create extra categories for sex, age group, etc.

If you want to tag several files at once, simply select all the files you want in the file browser, right-click on one of them and then click on Edit properties…

Note to PFC/PAC users: if the name of your file follows the PFC/PAC conventions, Dolmen will automatically create properties for the categories "Speaker", "Survey" and "Task", based on the file names.

The value of a tag is generally treated as text, but Dolmen also supports numeric values. Suppose for instance that you need a category for the age of your subjects or for a rating scale; in this case, you could create a category "Age" (or "Rating") and use only numeric values. If all values for a given category are numeric, Dolmen will understand that they must be treated as numbers and will present them differently in the search window (see XXX): instead of showing all values, it will display a value field and will let you choose a mathematical operator ("=", ">", etc.). All files supported by Dolmen can be tagged and you can add as many properties as you wish to a single file.

4.2.3 Metadata for non-native files

Plain text and TextGrid files have no support for metadata. Dolmen circumvents this problem by storing metadata for those files in an SQL database. Since this is all managed transparently, you can tag all your annotations (whether they are DMF or TextGrid files) without having to worry about which format they are stored in. However, be aware that the metadata for non-native files is only available within Dolmen.

The SQL database is stored in the Database sub-directory of your preferences directory.

4.2.4 Importing metadata from a CSV file

When dealing with large quantities of data, tagging metadata manually may be cumbersome. It is possible to partly automate this process by writing metadata to a comma-separated value (CSV) file, and then importing it into Dolmen. To do that, open the project containing the files to bet tagged and choose the CSV file containing the metadata with the command `Import metadata from a CSV file`... from the `File` menu. This will automatically add properties to the project's files.

The CSV file must follow a number of rules. First, the separator must be a comma; other separators such as the semi-colon or the tabulation character will not be recognized. Second, the first line must be a header, and the first field must be named `File`. Other fields in the header represent property categories. Finally, the last line should not be followed by an empty line.

Suppose that we have three files named `ctrl1.TextGrid`, `ctrl2.TextGrid` and `ctrl3.Textgrid` corresponding to three speakers from a control group. We want to automatically add the properties `Subject` and `Group` to these files. The corresponding CSV file will look something like the following:

```
File,Subject,Group
ctrl1.TextGrid,Subject 1,control
ctrl2.TextGrid,Subject 2,control
ctrl3.Textgrid,Subject 3,control
```

Note that Dolmen will tag any file whose full path ends with the string provided in the first field of each row. This means that if several files end in the same string (e.g. `task1-ctrl1.TextGrid` and `task2-ctrl1.TextGrid`), they will all be tagged with the corresponding properties. This can sometimes avoid duplicating properties for each file.

5 File formats

5.1 Annotations

An annotation is a time-aligned text file, typically the transcription and/or labeling of a sound file. The format used by Dolmen is inspired by Praat's TextGrid, but has a number of extensions. The key differences are:

- Annotations support (user-defined) metadata
- an annotation can be bound to a sound file
- Praat's tier intervals are called 'spans' in Dolmen
- Dolmen's items (spans and points) can be mixed within a tier.
- Dolmen's items can have connections to/from any other item in an annotation (they are treated as the vertices of a graph).

Dolmen is able to read TextGrid files and to convert them to its own native format (DMF).

As a convenience, when an annotation file is loaded, Dolmen will automatically try to find a sound file that matches the annotation's name with the extension .wav, .flac or .aiff (in this order). If the sound file exists, the annotation will be bound to it even if it is not part of the project.

5.2 Documents

Documents are plain text files, bearing a .txt extension. They can be annotated with metadata and their content can be searched like annotations.

5.3 Data tables

A data table is a two-dimensional structure which contains rows and columns. Columns represent data types and rows represent records, which contain values for each data type. Data tables are stored in a specific XML-based format named DMT (Dolmen Table). Data tables must bear the extension .dmt.

As currently implemented, data tables are created from a query view: right-click on a query match to display the contextual menu and click “Create table view”. This will convert the result of a query to a data table and open a new table view to display it. Each match of the query represents a record in the table. The table contains a number of fields: the file it comes from, its time stamp(s) (for annotations), the match, the left and right context of the match, and all the properties associated with the file the match comes from. If your query comes from a specific search grammar, the matched string will be broken into fields according to the grammar.

When a new table is created, it is not saved: to save it, click on the save icon in the table view: this will open a standard file dialog that lets you save the table to disk. When the table is created, it is automatically appended at the end of your project.

Dolmen offers two options to export a table, in case you want to analyze your data in a dedicated software package (R, SPSS, Excel, Calc, etc.). Data tables can be exported to CSV (comma-separated value) and to Excel[2]; simply right-click on the table in the file browser, click on the export sub-menu and choose the appropriate command. CSV is a simple, plain text format that which is very often used to store tabular data. Dolmen uses the semi-colon “;” as its field separator and encloses all values between double-quotes.

Note that you can add metadata to data tables: the metadata are stored directly within the DMT file. However, be aware that the metadata is not never exported when you convert a DMT file to a CSV or Excel file.

6 Queries

6.1 The search window

The main search interface is available through the Search button on the left side of the main window. Clicking the button will open a new window (the 'search window').

The file box in the top left corner allows you to select the type of files to search in. Currently, only annotations are supported. The Search box in the top right corner allows you to enter some text or a regular expression to search. Next to the search field, a spin box lets you select the tier you want to search in. The default choice is Any tier which means that Dolmen will try to find your pattern in all tiers of the selected files. You can also restrict the search to a particular tier; in that case, if a file contains less tiers than the tier number you chose (e.g. you try to search in tier 3 of a files that has only 2 tiers), the file is ignored and a warning is written to the output tab. Right below the search field, the "plus" and "minus" buttons let you add and remove search tiers (see cross tier search). Additionally, you can select a search style for your query: valid options are Regular Expression, UNIX Shell Pattern and Plain Text (see search style). A check box allows you to make your query case-sensitive. When the search is case-sensitive, strings like "foo", "Foo" and "FOO" are all treated as different; when it is case-insensitive (the default), they are treated as one and the same string.

Below the file and search boxes is the tag box: its content is generated on the basis of the properties that the current project contains. Each category is displayed as a group box containing a list of all the labels of this category. You can check or uncheck any label in any category (each category also has an "All labels" button check/uncheck all labels at once). The search engine will filter files based on the conditions that you specify in the tag box. Within a category, it uses the Boolean operator OR to find the subset of files that has either label.

[2] Note that the file format that is used is Microsoft Office Excel 2002. This is an XML-based format (not to be confused with the later Office Open XML). This file format can also be read by LibreOffice.

Across categories, it uses the operator AND to find the intersection of all the subsets defined by each category. Once you hit the ok button, the result of your query is presented as a new view in the viewer. You can browse the results with the mouse wheel. The information box on the right-hand side displays information about the selected token.

If an annotation is bound to a sound file, you can play a match by double-clicking on it or by pressing the space bar (you can also interrupt it by pressing Esc). You can also right-click on the selected match and click on the Play selection action from the match context menu. If you have Praat installed and your annotation is a TextGrid, you can use the Open selection in Praat from the match context menu. Dolmen will open the TextGrid (and the sound file if the annotation is bound) in Praat and will display the current match. (You need to have Praat already running for this to work.)

6.2 Cross-tier search

Dolmen is not limited to single-tier search: it is also able to perform "cross-tier" search, that is, it can retrieve results from a tier depending on conditions met on other tiers. Cross-tier search is useful when you have data that is hierarchically organized across several tiers. Typical examples are prosodic hierarchies or syntactic trees. Currently, the hierarchical organization is read off the annotation based on the items' time alignment. The root item on the base tier must be a span (or interval in Praat). On each subordinate tier, items are considered to be dominated by the root item if they are within its boundaries: for instance, if a root item spans from 10" to 15", all items within that range on the dependent tiers will be treated as dependent nodes.

As it is currently implemented, cross-tier search uses the first tier in the search box as the base tier and treats the others as hierarchically dependent on the first one. That is, whenever it finds a match in the base tier, it tries to find a match on each of the other tiers on the items that are within the range of the matching item on the base tier.

To enable cross-tier search, simply add one or several tier(s) in the search box using the "plus" button below the search box. (You can also use the key combination Alt + to add a tier and Alt - to remove the last one). When cross-tier search is enabled, an additional spin box appears above the first search field, which lets you select the tier of which you want to display the text. It can be any tier, and not necessarily one of those you are searching from.

An example will make this clearer: suppose you have 3 tiers in your file: the first one contains spans which denote syllables, the second one contains syllabic constituents ("syll") ("Onset", "Nucleus", "Coda") and the last one individual segments ("p", "a", "t"...). Let's consider a query that looks for "syll" on tier 1, "Coda" on tier 2 and displays text from tier 3. This query will first get all the items that have a "syll" label on the first tier; then, for each of those, it will look for a label "Coda" on tier 2 within the limits of the span on tier 1; for each item that matches both conditions, it will display the concatenated text of the items on tier 3 that are dominated by the matching item on tier 1. Our query will thus print all syllables that end in a coda.

6.3 Exporting queries to a spreadsheet

Dolmen can export query results to a text format (CSV, which historically stands for “comma separated values”) that can be read by spreadsheet programs such Microsoft Excel or LibreOffice Calc. Dolmen uses the CSV file format (rather than say XLS) because it is simple, portable and can be read or parsed by a wide variety of programs.

To export the results of a query to a CSV file, right-click on the results and select "Export" >

"Save as tab-separated file...". To import the CSV file into Excel, click on Data > From text (from the main menu). In LibreOffice, simply open the CSV file like a regular file: its extension will automatically be recognized. Both program will open a new dialog that lets you decide how to import the file. In order to be able to sucessfully load the file, you need to specify the encoding (Unicode or UTF-8), the separator (tabulation character) and the text delimiter (double-quote character). In Excel, if you have numeric codings that start with zero, make sure they are treated as text values when you import them, otherwise the leading zeros will be trimmed off.

CSV files are organized as a table where rows represent query results and columns represent values. The following values are extracted: the file name, the start and end of the item in which the match was found, the left context, the match, the right context, and the project's categories. Each category is represented by a column for all matches: if there are several values associated with a category for a given match (say category Language and values English and French), the values are separated by the character "|" (e.g. English | French). If there are no properties for a category, the field is left empty.

6.4 The 'Queries' tab

The Queries tab (located at the bottom of the main window) stores all the queries you run during a session. Double-clicking on a query will focus the query in the viewer or re-open it if you closed it. To save a query, right-click on it to trigger the context menu and click on Save query...: a standard file dialog will open for you to save your query. Queries bear the extension .dmq (Dolmen Query); the extension is added automatically if you omit it. Note that a query is bound to a particular project: you will not be able to open a DMQ file which doesn't match the current project. The query syntax is currently undocumented (it is loosely based on SQL but is specific to Dolmen). Normal users need not know anything about it.

6.5 PFC and PAC users

When using the "PFC" and "PAC" modes (see application mode), a number of facilities are available to search for schwa and liaison codings.

PFC

If you select tier 2 in the search box, the search field is made "aware" of the fact that you are looking for schwa codings and offers the following conveniences:

you can input a star * to replace any digit in a coding. Thus, the pattern *422 will return all codings in word-final position between two consonants, whether schwa is realized or not.

The pattern **** will return all codings

the symbol % denotes any character but 'e'. This is particularly useful when studying the correlation between spelling and pronunciation, for instance to see whether there is a significant difference between e-ending and consonant-ending words in the realization of schwa.

Similarly, if you look for a pattern in the third tier, the following shortcuts are available:

* represents any digit (** returns all liaison codings)

C represents any liaison consonant.

PAC

If you look for a pattern in tier 2 (r-liaison), you can use a star * to replace any digit in the coding. The pattern *** will return all liaison codings.

7 Search and regular expressions

Dolmen's search engine can make use of regular expressions (sometimes called 'regexp'), which are a special syntax that let users find text patterns. Regular expressions are very powerful, but their syntax can be cumbersome and they can be sometimes be tricky to use properly. Here we give an overview of regular expressions as they are implemented in Dolmen. In what follows, a 'string' is defined as an arbitrary sequence of characters and is show in italics (e.g. the string xyz). A 'pattern' is a sequence of characters that conforms to the syntax of regular expression and is shown in bold face (e.g. the pattern ^.*$). A pattern 'matches' a string if there is a substring in the string that corresponds to the pattern. The match is underlined (e.g. xyz).

7.1 Basics

Regular expressions always try to match a pattern from left to right; in their simplest form, they match a sequence of (non-special) characters. For instance, the pattern the matches the first occurrence of ‘the’ in the string the cat is chasing the mouse. Note that search can be made case-sensitive (it is case-insensitive by default) by ticking the corresponding box in the search window. In Dolmen (as currently implemented), regular expressions are ‘non-greedy’, which means they will match the smallest segment of text that corresponds to the pattern (but see caveats).

The most common symbols are:

. : match any character

^ : match the beginning of a string

$: match the end of a string

\b : match a word boundary

\s : match a white space character

It also possible to define sets of characters, using the bracket notation:

[xyz] : match either of the characters 'x', 'y' or 'z'’

[^xyz] : match any character but 'x', 'y' or 'z'

[a-z] : match any character in the range from 'a' to 'z'

Some useful predefined character sets are:

\d : match a digit character (equivalent to [0-9])

\w : match a word character (including digits and '_' underscore)

In addition, regular expressions offer a number of quantifiers:

E? : match 0 or 1 occurrences of the expression E

E* : match 0 or more occurrences of the expression E

E+ : match 1 or more occurrences of the expression E

E{n} : match exactly n occurrences of the expression E

E{n,m} : match between n and m occurrences of the expression E

E{n,} : match at least n occurrences of the expression E

E{,m} : match at most m occurrences of the expression E (and possibly 0)

In this context, an expression must be understood as either a character (e.g. o{2,} matches the string food) or a sequence of characters enclosed by parentheses (e.g. (do){2} matches the string fais dodo).

Another useful character is '|', which is used to combine expressions (logical OR). For example, the pattern ^(John|Mary) matches the strings John kissed Mary and Mary was kissed by John.

All the characters that are used as part of the syntax of regular expressions ('{', ')', '\', etc.) are treated as special characters by the search engine. As such, if you need to match one of those characters in a string (e.g. the parentheses in the string and (he) she...), you need to escape it with a backslash. For instance, the pattern \(he\) matches the string and (he), I mean she...).

7.2 Extensions

To make things easier, Dolmen recognizes a number of additional symbols which can be useful to linguists (but are not part of the syntax of regular expressions).

# : a word boundary (equivalent to \b)

#* : match a (possibly empty) prefix (equivalent to \b\w*)

# : match a (possibly empty) suffix (equivalent to \w\b)

These symbols offer convenience to look for derived forms. For instance, the pattern #*happ[yi]*# could be used to match forms like happy, happier, unhappy, happiness, unhappiness, etc. Note however that these symbols cannot be used in PFC and PAC mode in the codings tiers.

Additionally, Dolmen defines a few useful variables. Search variables always start with '%' and are capitalized:

%LINE : match a non-empty line (equivalent to ^.+$)

%WORD : match a non-empty word (equivalent to \b\w+\b)

Booby traps

Regular expressions can sometimes be difficult to use, and may not always do what you think they should be doing. Here are some examples:

regular expressions only match characters, they have no "understanding" of linguistic structure. As such, the notion 'word' must be understood in a broad sense: indeed, a string like AR0303BD is a perfectly valid word as far as the regular expression engine is concerned, even though it might be a speaker identifier in your conventions. This is something you may have to take into consideration, for instance if you want to count words in a corpus.

Regular expressions are 'non-greedy', but that doesn't mean that they are 'minimal'. Let's consider the pattern I.*know matched against the string I don't, I don't know. Because search is non-greedy, we might expect that it will match the substring I don't know, but this is not the case: it matches the whole string I don't, I don't know. The reason for this is that the regular expression engine parses a string from left to right and returns the first match without computing all the logical possibilities (for performance reasons). In this case, it matches the first character of the string and continues until the end. 'Minimal' search is not supported by any regular expression engine and is currently not implemented in Dolmen.

7.3 Learn more

If you would like to learn more about the regular expression engine used in Dolmen, you should have a look at Qt's regular expression engine, which is used by Dolmen.

The following website also contains lots of information about regular expressions www.regular-expressions.info.

Another very good resource about regular expressions in general is: Jeffrey E.F. Friedl (1997) *Mastering regular expressions*, O'Reilly.

8 Managing bookmarks

Dolmen lets you bookmark search results so that you can retrieve them easily later on, for instance when you are writing a paper and need to discuss specific cases. A bookmark keeps track of the matched text and the location of the match in the sound file.

To bookmark a search result, simply right-click on it in the query view, and click on Bookmark search result... A new dialog will pop up which will let you assign a title to your bookmark and (optionally) add some notes.

To view your bookmarks, click on the combobox selector in the top-left corner of the window (above the file browser) and select Bookmarks. Your bookmarks will be displayed in the file browser, instead of the project's files. You can view the metadata associated with a bookmark by hovering over it with the mouse cursor. Double-clicking on a bookmark opens the annotation at the location that was bookmarked.

9 Editing preferences

The preference editor is available in the main menu under Edit > Preferences (or Dolmen > Preferences on Mac). The editor contains 2 tabs, which allow you to adjust your settings to your system and/or to your liking.

9.1 The 'General' tab

The application data folder is the folder where Dolmen stores its own files, especially metadata files when they are not stored with the TextGrids. If you need to change this value, click on the choose... button and navigate to the folder that you want to use.

The match context window is the size (in characters) of the context on each side of your match in a query view. Adjust it according to the size of your screen. Note that choosing a longer window will somewhat slow down the search, depending on your hardware and on the size of your corpus. By default, this option is set to 30 characters.

The default search style lets you decide how a pattern is to be searched by default. Options are:

Regular Expressions: this is the most powerful mode. It lets you use Perl-like regular expressions.

UNIX Shell Pattern: easier than regular expressions but much more limited.

Plain Text: no special characters

For more information, have a look at Qt's regular expression engine. Note that your choice in the preferences will only affect the default mode (for each query, you are always able to select an alternate mode in the search window).

The last option lets you decide whether to store the metadata of non-native files (e.g. TextGrids) along with the files. If you select Yes (the default case), the metadata file will be stored in the same folder as the file. If you select No, it will be stored in the Application data folder.

9.2 The 'Advanced' tab

The Praat path option, as you would expect, lets you adjust the path to Praat. This is particularly useful if it is located in a non-standard place (especially on Windows and Linux).

The TextGrid encoding lets you modify which encoding TextGrid files should be read in. By default, Praat uses the ASCII encoding for files that do not use any special characters (e.g. a label file written in English) and UTF-16 for files that do contain special characters (e.g. a transcription in French). In Praat, you can check (and modify) this setting under Praat > Preferences > Text writing preferences... By default, Dolmen assumes that files are encoded in UTF-16, but you can change it to UTF-8 or ASCII if it is not the case. Older formats like Windows ISO 8859-15 and Mac Roman are not supported.

The last option is the Application mode. Most users will use the Default mode. However, users working within the projects “Phonologie du français contemporain” (PFC) and “Phonologie de l’anglais contemporain” (PAC) should select the PFC and PAC mode respectively. These special modes enable a number of features which are specific to those projects (and which were previously available in the PFC/PAC toolbox, which Dolmen supersedes).

10 Plugins

Dolmen can be extended through plugins, which are written in JSON and JavaScript[3]. When it starts up, Dolmen loads all plugins which are located in the system plugin directory or in the user plugin directory. Plugins can be redistributed as ZIP files (the .zip extension is compulsory).

To install a plugin, go to File > Install plugin… and choose the ZIP file. It will be installed in the current user’s plugin directory.

10.1 Structure of a plugin

To be valid, a plugin must adhere to a number of conventions: if they are not respected, Dolmen will silently ignore the plugin. The root directory of the plugin must contain the following:

- a description file, named description.json (compulsory)
- a Scripts sub-directory, which contains all your scripts (optional).
- a Grammars sub-directory, which contains all your
- a Resources sub-directory, which may contain anything (optional).

The description file contains all the information necessary to initialize the plugin. All declarative aspects of the plugin are written in the JSON[4] format and must bear the extension

[3] More pedantically, Dolmen uses the ECMAScript language. See:

http://www.ecma-international.org/publications/standards/Ecma-262.htm

[4] http://www.json.org

.json. Scripts are written in Javascript and must bear the extension .js.

Here is an example of a basic description.json file:

```
{
	"PluginInfo": {
		"Name": "My first plugin",
		"Version": "0.1",
		"MainApplication": "false"
	},

	"Menu": {"Text": "Custom menu", "Actions":
		[
			{"Type": "Action", "Text": "Test script", "Script": "test.js", "Shortcut": "Ctrl+T"}
		]
	}
}
```

The header PluginInfo is the only part that is obligatory. It contains some meta-information about the plugin. The key MainApplication tells whether a plugin takes ownership of Dolmen[5]. The Menu key lets you create a custom menu: each menu entry (called “action”) is associated with a script which must be located in the Scripts sub-directory. When you click on an action in the menu, the corresponding script is executed. It is also possible to assign a shortcut to a given action.

10.2 Defining a search grammar

If you have devised a coding scheme for your data, Dolmen lets you define a “search grammar”. A search grammar is a description of your coding scheme which offers a user-friendly interface for querying your data; it tells Dolmen what to look for and how to present the information to the user. Dolmen will automatically load all valid search grammars that are located in the Grammars sub-directory of your plugin. They are presented as new tabs in the search window.

In a nutshell, a search grammar defines a number of fields which can take on a number of values. The user is presented with a number of checkboxes for each field, and Dolmen converts the query to the corresponding regular expression, as defined by the grammar. Here is a realistic yet simple example, drawn from the PFC project:

```
{
	"Header": {
		"Object": "SearchGrammar",
```

[5] In case several plugins declare themselves as main, the last one that is loaded will take ownership of the application.

```
        "DisplayName": "Schwa",
        "Version": "0.1",
},
"Separator": "",
"FileType": "Annotation",
"Tier": 2,
"FieldsPerRow": 4,

"Fields": [
        {"Name": "Schwa", "MatchAll": "[0-2]",
         "Values": [
                {"Match": "0", "Text": "Absent"},
                {"Match": "1", "Text": "Présent"},
                {"Match": "2", "Text": "Incertain"},
         ]
        },
        {"Name": "Position", "MatchAll": "[1-5]",
         "Values": [
                {"Match": "1",        "Text": "monosyllabe"},
                {"Match": "2",        "Text": "début de polysyllabe"},
                {"Match": "3",        "Text": "syllabe interne"}
                {"Match": "4",        "Text": "fin de polysyllabe"}
                {"Match": "5",        "Text": "métathèse"}
         ]
        },
        {"Name": "Contexte gauche", "MatchAll": "[1-5]",
         "Values": [
                {"Match": "1",        "Text": "voyelle"},
                {"Match": "2",        "Text": "consonne"}
                {"Match": "3",        "Text": "début de groupe intonatif"}
                {"Match": "4",        "Text": "réalisation voc incertaine"}
                {"Match": "5",        "Text": "groupe consonantique simplifié"}
         ]
        },
        {"Name": "Contexte droit", "MatchAll": "[1-4]",
        "Values": [
            {"Match": "1", "Text": "voyelle"},
            {"Match": "2", "Text": "consonne"},
            {"Match": "3", "Text": "frontière intonative forte"},
            {"Match": "4", "Text": "frontière intonative faible"}
        ]
        }
    ]
}
```

The schwa coding is a 4-digit scheme, where each digit is a number. In this case, each digit is treated as field.

In the simple case (like our example), a field may be one digit (or one character), but it does not need to be so. In fact, a field can be of any length, provided that it can be described by a regular expression. Fields may optionally be separated by a separator (for instance the character "_"); in our case, there is no separator so the key Separator has an empty value (the empty string). Alternatively, we could have simply omitted this key.